XML API for Mitel SIP Phones

2024

58015045-00

DEVELOPMENT GUIDE



The information conveyed in this document is confidential and proprietary to Mitel® and is intended solely for Mitel employees and members of Mitel's reseller channel who specifically have a need to know this information. If you are not a Mitel employee or a Mitel authorizedPARTNER, you are not the intended recipient of this information. Please delete or return any related material. Mitel will enforce its right to protect its confidential and proprietary informationand failure to comply with the foregoing may result in legal action against you or your company.

TABLE OF CONTENTS

1 Introduction 15

1.1 MITEL XML API 15

1.2 Revision History 15

1.2.1 Version 5.0.0 (6863i / 6865i / 6867i / 6869i / 6873i / 6920 / 6930 / 6940) 15

1.2.2 Version 4.3.0-SP1 (6863i / 6865i / 6867i / 6869i / 6873i) 16

1.2.3 Version 4.3.0 (6863i / 6865i / 6867i / 6869i / 6873i) 16

1.2.4 Version 4.2.0-SP1 (6863i / 6865i / 6867i / 6869i/ 6873i) 17

1.2.5 Version 4.2.0 (6863i / 6865i / 6867i / 6869i/ 6873i) 17

1.2.6 Version 4.1.0 (6863i / 6865i / 6867i / 6869i) 17

1.2.7 Version 4.0.0 SP2 (6863i / 6865i / 6867i / 6869i) 17

1.2.8 Version 4.0.0 SP1 (6863i / 6865i / 6867i / 6869i) 17

1.2.9 Version 4.0.0 (6863i / 6865i / 6867i / 6869i) 18

1.2.10 Version 3.3.1 SP4 (9143i / 9480i / 9480iCT / 6730i / 6731i / 6739i / 53i / 55i / 57i / 57iCT / 6863i / 6865i / 6867i) 18

1.2.11 Version 3.3.1 SP3 (9143i / 9480i / 9480iCT / 6730i / 6731i / 6739i / 53i / 55i / 57i / 57iCT / 6863i / 6865i / 6867i) 18

1.2.12 Version 3.3.1 SP 2(9143i/9480i/9480iCT/6730i/6731i/6739i/53i/55i/57i/57iCT) 18

1.2.13 Version 3.3.1 (9143i/9480i/9480iCT/6730i/6731i/6739i/53i/55i/57i/57iCT) 18

1.2.14 Version 3.3.0 (9143i/9480i/9480iCT/6730i/6731i/6739i/53i/55i/57i/57iCT) 18

1.2.15 Version 3.2.2 (9143i/9480i/9480iCT/6730i/6731i/6735i/6737i/6739i/53i/55i/57i/57iCT) 19

1.2.16 Version 3.2.2 (9143i/9480i/9480iCT/6730i/6731i/6739i/53i/55i/57i/57iCT) 19

1.2.17 Version 3.2.1 (9143i/9480i/9480iCT/6730i/6731i/6739i/53i/55i/57i/57iCT) 19

1.2.18 Version 3.2.0 (9143i/9480i/9480iCT/6730i/6731i/6739i/53i/55i/57i/57iCT) 19

1.2.19 Version 3.0.1 (6739i) 20

1.2.20 Version 3.0.0 (6739i) 20

1.2.21 Version 2.6.0 (9143i/9480i/9480iCT/6730i/6731i/51i/53i/55i/57i/57i CT) 20

1.2.22 Version 2.5.3 (9143i/9480i/9480iCT/6730i/6731i/51i/53i/55i/57i/57iCT) 20

1.2.23 Version 2.5.2 (9143i/9480i/9480iCT/6730i/6731i/51i/53i/55i/57i/57i CT) 21

1.2.24 Version 2.4.1 (9143i/9480i/9480iCT/6730i/6731i/51i/53i/55i/57i/57i CT) 21

1.2.25 Version 2.4.0 (9143i/9480i/9480iCT/51i/53i/55i/57i/57i CT) 21

1.2.26 Version 2.3.1 (9143i/9480i/9480iCT/51i/53i/55i/57i/57i CT) 21

1.2.27 Version 2.3.0 (9143i/9480i/9480iCT/51i/53i/55i/57i/57i CT) 21

1.2.28 Version 2.2.1 (9143i/9480i/9480iCT/51i/53i/55i/57i/57i CT) 22

1.2.29 Version 2.2.0 (51i/53i/55i/57i/57i CT) 22

1.2.30 Version 2.1.1 (51i/53i/55i/57i/57i CT) 23

1.2.31 Version 2.1.0 (53i/55i/57i/57i CT) 23

1.2.32 Version 2.0.2 (53i/55i/57i/57i CT) 23

1.2.33 Version 2.0.1 (53i/55i/57i/57i CT) 23

1.2.34 Version 1.4.2 (9112i/9133i/480i/480i CT) 24
1.2.35 Version 1.4.1 (9112i/9133i/480i/480i CT) 24
1.2.36 Version 1.3.1 (9112i/9133i/480i/480i CT) 25
1.2.37 Version 1.3.0 (9112i/9133i/480i/480i CT) 25

2 XML and the Mitel SIP Phones 26

2.1 What is XML? 26
2.2 Functionality 26
2.3 How does it work? 27
2.3.1 Phone initiated application 27
2.3.2 Server initiated application 27
2.4 System Architecture 28
2.4.1 Corporate applications 28
2.4.2 Telephony applications 29
2.5 Development environment 29
2.5.1 Typical software architecture 29
2.5.2 Web server 30
2.5.3 Scripts/Languages 30
2.5.4 XML Validation tools 30
2.6 XML format 30
2.7 HTTP format 31
2.8 XML display control and keys 31
2.8.1 Mitel 6863i 32
2.8.2 Mitel 6865i 33
2.8.3 Mitel 6867i / Mitel 6920 34
2.8.4 Mitel 6869i / Mitel 6930 35
2.8.5 Mitel 6873i / Mitel 6940 35

3 Mitel IP Phone XML Objects 37

3.1 TextMenu Object (All models) 38
3.1.1 Implementation (softkey and non softkey phones) 38
3.1.2 Implementation (6873i/6940) 39
3.1.3 XML Description 41
3.1.4 Examples 47
3.2 IconMenu Object (6867i/6869i and 6873i) 52
3.2.1 Implementation (6867i) 53
3.2.2 Implementation (6869i) 54
3.2.3 Implementation (6873i) 55
3.2.4 XML Description 56
3.2.5 Examples 61
3.3 ImageMenu Object (6867i/6869i/6873i/6920/6930/6940) 65

3.3.1 Implementation (6867i/6920) 65
3.3.2 Implementation (6869i/6930) 66
3.3.3 Implementation (6873i/6940) 67
3.3.4 XML Description 68
3.3.5 Examples 71
3.4 TextScreen Object (all models) *72*
3.4.1 Implementation (6863i/6865i) 72
3.4.2 Implementation (6867i/6869i) 72
3.4.3 Implementation (6873i/6940) 73
3.4.4 XML Description 74
3.4.5 Examples 78
3.5 FormattedTextScreen Object (all models) *80*
3.5.1 Implementation (6863i/6865i) 80
3.5.2 Implementation (6867i/6869i/6920/6930) 81
3.5.3 Implementation (6873i/6940) 82
3.5.4 XML Description 84
3.5.5 Examples 90
3.6 ImageScreen Object (6867i/6869i/6873i/6920/6930/6940) *93*
3.6.1 Implementation (6867i/6869i/6920/6930) 93
3.6.2 Implementation (6873i/6940) 94
3.6.3 XML Description 95
3.6.4 Examples 99
3.7 InputScreen Object – Single Input field (all models) *99*
3.7.1 Implementation (6863i/6865i) 100
3.7.2 Implementation (6867i/6869i/6920/6930) 100
3.7.3 Implementation (6873i/6940) 101
3.7.4 XML Description 102
3.7.5 Input Type: IP 106
3.7.6 Input Type: Number 107
3.7.7 Input Type: String or StringN 109
3.7.8 Input Type: timeUS 112
3.7.9 Input Type: timeInt 114
3.7.10 Input Type: dateUS 116
3.7.11 Input Type: dateInt 118
3.8 InputScreen Object – Multiple Input fields (6867i / 6869i / 6873i / 6920 / 6930 / 6940) *121*
3.8.1 Implementation (6867i/6869i/6920/6930) 121
3.8.2 Implementation (6873i/6940) 122
3.8.3 XML Description 123
3.8.4 Examples 129
3.9 PhoneStatus Object *133*

3.9.1 Implementation (6863i/6865i) ... 133
3.9.2 Implementation (6867i/6869i/6873i/6920/6930/6940) ... 133
3.9.3 XML Description ... 134
3.9.4 Examples ... 137
3.10 PhoneExecute Object (all models) ... 140
3.10.1 Implementation ... 140
3.10.2 XML Description ... 141
3.10.3 Examples ... 141
3.11 PhoneConfiguration Object (all models) ... 143
3.11.1 Implementation ... 143
3.11.2 XML Description ... 144
3.11.3 Examples ... 145
3.12 PhoneSoftkey Object (6867i/6869i/6873i/6920/6930/6940) ... 147
3.12.1 Implementation ... 147
3.12.2 XML Description ... 151
3.12.3 Examples ... 151
4 XML extensions ... 155
4.1 Customizable Softkeys ((6867i/6869i/6873i/6920/6930/6940)) ... 155
4.2 Graphics (6867i/6869i/6873i/6920/6930/6940) ... 158
4.2.1 Images ... 158
4.2.2 Icons ... 159
4.2.3 Colors ... 164
4.3 LED control ... 165
4.3.1 Implementation (6867i/6869i/6920/6930) ... 166
4.3.2 Implementation (6873i/6940) ... 168
4.3.3 XML Examples ... 170
4.4 RTP streaming ... 171
4.4.1 RTPRx ... 172
4.4.2 RTPTx ... 173
4.4.3 RTPMRx ... 175
4.4.4 RTPMTx ... 176
4.4.5 Interaction with action uris ... 177
4.5 Keypress emulation ... 178
4.6 Wav file streaming ... 180
4.6.1 XML Commands ... 180
4.6.2 File format ... 181
4.6.3 Interaction with action uris ... 181
4.7 Wav file loop playback (melody) ... 183
4.7.1 XML Commands ... 183

4.7.2 File format 184
4.8 Dial and DialLine URIs 184
4.9 Crash and configuration files retrieval 184
4.10 Special attributes 185
4.10.1 Beep 185
4.10.2 Timeout 185
4.10.3 LockIn 185
4.10.4 CallProtection 186
4.10.5 triggerDestroyOnExit 186
4.11 TextMenu or IconMenu user selection (6867i, 6869i, 6873i, 6920, 6930, 6940) 186
4.12 "Select" and "Dial" in the same TextMenu or IconMenu object 187
4.13 "Select", "Dial" and "Dial2" behavior in a TextMenu or IconMenu object 187
4.14 Refresh of an XML page 188
4.15 HTTP headers format for a phone HTTP GET 188
4.15.1 User-Agent 188
4.15.2 X-Aaastra-ExpModi 189
4.15.3 Accept-Language 190
4.16 Some development guidelines 190

5 URL Format and Variables 191
5.1 URL format 191
5.2 URL Variables 191
5.2.1 Variables related to the active line 192
5.2.2 Variables related to the current call 193
5.2.3 Usage with action uris 193
5.2.4 Phone State 194
5.3 HTTPS 195
5.3.1 User Certificates 196
5.3.2 Configuration 197
5.4 XML Objects Pushed to the Phone 198

6 Action URIs 201
6.1 Configuration 201
6.2 Action uri detailed behavior 202
6.2.1 Action uri offhook 202
6.2.2 Action uri onhook 202
6.2.3 Action uri incoming 202
6.2.4 Action uri outgoing 202
6.2.5 Action uri connected 203
6.2.6 Action uri disconnected 203
6.2.7 Action uri startup 203

6.2.8 Action uri registered 203
6.2.9 Action uri registration event 204
6.2.10 Action uri poll 204
6.2.11 Action uri blf 204
6.2.12 Action uri xml sip notify 205
6.3 "Answer" and "Ignore" keys for action uri incoming 206
6.4 "Drop", "Xfer" and "Conf" keys for XML apps in connected state 206
6.5 Applications 207

7 XML Configuration 208
7.1 Configuring a Soft or Programmable Key using the phone Web UI 208
7.2 Configuring the XML Push Server List using the phone Web UI 208
7.3 Configuring the Action URIs using the phone Web UI 209
7.4 Configuring the XML Beep Support using the phone Web UI 209
7.5 Configuring the Status Scroll Delay using the phone Web UI 210
7.6 Configuring the XML SIP Notify using the phone Web UI 211
7.7 XML Configuration using the Configuration Files 211
7.7.1 General XML parameters 211
7.7.2 Action uri parameters 218
7.7.3 Programmable and Soft keys 223
7.7.4 Examples 223

8 Troubleshooting XML Applications 224
8.1 Introduction 224
8.2 Troubleshooting tools 225
8.3 Configuring the Syslog Server using the WebUI 225
8.4 Configuring the Syslog server using the configuration files 226
8.5 Parsing error debug example 226

9 Why XML Applications for an IP Phone? 228
9.1 Telephony applications 228
9.1.1 Directory 228
9.1.2 Call Processing 228
9.1.3 Voice-Mail 228
9.1.4 Conference Bridge 228
9.1.5 Contact Center 228
9.2 Media and information 228
9.3 Vertical applications 229
9.3.1 Human Resources 229
9.3.2 Travel/Hotel 229
9.3.3 Health Care 230

9.3.4 Education 230
9.3.5 Law Enforcement 230

10 Phone Self-Configuration using XML 231

10.1 Introduction 231
10.2 Message flow 231
10.3 Auto-configuration policy 232
10.4 Architecture 233

11 Sample XML applications 234

11.1 Access from the Internet 234
11.2 Local server using XAMPP 234
11.2.1 XAMPP installation 235
11.2.2 XML scripts installation 235
11.2.3 XAMPP start and stop 235
11.2.4 Test your installation 237
11.2.5 Troubleshooting Apache 237
11.3 Applications 237
11.3.1 Area code lookup (US) 237
11.3.2 Biorhythms 238
11.3.3 CNN News 239
11.3.4 Currency Converter 239
11.3.5 ESPN News 240
11.3.6 FOX News 241
11.3.7 Horoscope 241
11.3.8 IP Geolocation 242
11.3.9 Movies 242
11.3.10 Netflix 243
11.3.11 Stock Quotes 243
11.3.12 Today… 244
11.3.13 Local Weather 244
11.3.14 Yahtzee 245
11.3.15 Global menu 245

12 Appendix A: XSL Model 247

13 Appendix B: Object Oriented PHP Classes 248

13.1 AastraIPPhoneCallLog() 248
13.2 AastraIPPhoneConfiguration() 250
13.3 AastraIPPhoneExecute() 251
13.4 AastraIPPhoneFormattedTextScreen() 251
13.5 AastraIPPhoneIconMenu() 253

13.6 AastraIPPhoneImageMenu() 255
13.7 AastraIPPhoneImageScreen() 258
13.8 AastraIPPhoneInputScreen() – Single Input field 261
13.9 AastraIPPhoneInputScreen() – Multiple Input fields 263
13.10 AastraIPPhoneSoftkey() 266
13.11 AastraIPPhoneStatus() 266
13.12 AastraIPPhoneTextMenu() 267
13.13 AastraIPPhoneTextScreen() 270
13.14 AastraIPPhoneScrollableTextMenu() 272
13.15 AastraIPPhoneScrollableDirectory() 273
13.16 Examples provided with the PHP API 275

14 Appendix C: Dynamic Parameters 277

15 Appendix D: Localized input character set 282
15.1 English 282
15.2 French/Français 282
15.3 Spanish/Español 282
15.4 German/Deutsch 283
15.5 Italian/Italiano 283
15.6 Portuguese/Portugês 284

16 Appendix E: XML Self-Configuration 285

17 Appendix F: CSV based Directory 298
17.1 Introduction 298
17.2 Phone compatibility 298
17.3 Installation 298
17.4 XML key configuration 298
17.5 CSV file format 299
17.6 Configuration files 299
17.6.1 server.conf 299
17.6.2 directory.conf 299
17.7 Application Files 300

18 Appendix G: LDAP Directory 302
18.1 Introduction 302
18.2 Installation 302
18.3 Network Requirements 303
18.3.1 Phone -> XML Proxy Server 303
18.3.2 XML Proxy Server -> LDAP-Server 303
18.4 Active Directory / LDAP-Server Requirements 303
18.5 Configuration files 303

18.5.1 server.conf 303
18.5.2 ldap_directory.conf 304

19 Appendix H: PictureCallerID (6867i, 6869i, 6873i, 6920, 6930 and 6940) 307

19.1 Introduction 307
19.2 Application implementation 307
19.3 Requirements/Compatibility 308
19.3.1 HTTP Server 308
19.3.2 Compatibility 308
19.4 Installation 308
19.4.1 Introduction 308
19.4.2 HTTP Server 308
19.4.3 Package installation 308
19.4.4 Test your installation 309
19.4.5 Troubleshooting 309
19.4.6 Windows PC using XAMPP Lite 309
19.5 Configuration 309
19.5.1 Phone Configuration 309
19.5.2 Application configuration 310
19.5.3 Number matching examples 313
19.5.4 Pattern matching examples 313
19.6 Files 315

20 LAST PAGE OF THE DOCUMENT 317

TABLE OF FIGURES

Figure 1: Basic XML document 26
Figure 2: Mitel IP Phone acting as a client 27
Figure 3: Mitel IP Phone acting as a server 28
Figure 4: Access to an internal application 28
Figure 5: Access to an Internet application 29
Figure 6: Access to a telephony application 29
Figure 7: Typical software architecture of an XML application 30
Figure 8: XML conversion table 31
Figure 9: Mitel 6863i XML display and keys 32
Figure 10: Mitel 6865i XML display and keys 33
Figure 11: Mitel 6867i XML display and keys 34
Figure 12: Mitel 6869i XML display and keys 35
Figure 13: Mitel 6873i/6940 XML display 36
Figure 14: TextMenu Implementation on 6873i/6940 39
Figure 15: TextMenu touchLaunch tag usage 40
Figure 16: fontMono tag usage 40
Figure 17: icon positions in MenuItem 41
Figure 18: TextMenu Example 1 (6863i/6865i) 47
Figure 19: TextMenu Example 1 (6869i/6930) 48
Figure 20: TextMenu Example 1 (6873i/6940) 48
Figure 21: TextMenu Example 2 (6869i/6930) 49
Figure 22: TextMenu Example 3 (6873i/6940) 51
Figure 23: IconMenu cell layout 52
Figure 24: IconMenu cell rendering in portrait mode 52
Figure 25: IconMenu cell rendering in landscape mode 53
Figure 26: IconMenu screen modes “regular” and “extended” 53
Figure 27: 6867i - Layout 1 and 2 mode “regular” 54
Figure 28: 6869i - Layout 1 and 2 in “regular” mode 55
Figure 29: 6873i - Layout 1 and 2 in “regular” mode 56
Figure 30: IconMenu Example 1 (6869i) 61
Figure 31: IconMenu Example 2 (6869i) 63
Figure 32: IconMenu Example 2 (6873i) 64
Figure 33: ImageMenu 6867i/6920 implementation (mode regular/extended and full screen) 66
Figure 34: ImageMenu 6869i/6930 implementation (mode regular/extended and full screen) 67
Figure 35: ImageMenu 6873i/6940 implementation (mode regular/extended and full screen) 68
Figure 36: ImageMenu Example (6869i/6930i) 71
Figure 37: TextScreen implementation on 6867i/6920 73

Figure 38: TextScreen implementation on 6873i/6940 74
Figure 39: TextScreen Example (6863i/6865i) 78
Figure 40: TextScreen Example (6869i/6930) 78
Figure 41: TextScreen Example (6869i/6930) 79
Figure 42: FormattedTextScreen layout 80
Figure 43: FormattedTextScreen layout 81
Figure 44: FormattedTextScreen implementation on 6867i/6920 82
Figure 45: FormattedTextScreen layout on 6873i/6940 83
Figure 46: FormattedTextScreen implementation on 6873i/6940 83
Figure 47: FormattedTextScreen Example 1 (6863i/6865i) 90
Figure 48: FormattedTextScreen Example 2 (6869i/6930) 91
Figure 49: FormattedTextScreen Example 3 (6869i/6930) 92
Figure 50: ImageScreen 6867i/6920 implementation (mode regular/extended and full screen) 94
Figure 51: ImageScreen 6873i/6940 implementation (mode regular/extended and full screen) 95
Figure 52: ImageScreen Example (6739i) 99
Figure 53: InputScreen implementation on 6869i/6930 101
Figure 54: InputScreen implementation on 6873i/6940 101
Figure 55: InputScreen “IP” Example (6863i/6865i) 106
Figure 56: InputScreen “IP” Example (6869i/6930) 107
Figure 57: InputScreen “IP” Example (6873i/6940) 107
Figure 58: InputScreen “Number” Example (6863i/6865i) 108
Figure 59: InputScreen “Number” Example (6869i/6930) 108
Figure 60: InputScreen “Number” Example (6873i/6940) 109
Figure 61: InputScreen “String” Example (6863i/6865i) 110
Figure 62: InputScreen “String” Example (6869i/6930) 111
Figure 63: InputScreen “String” Example (6873i/6940) 111
Figure 64: InputScreen “TimeUS” Example (6863i/6865i) 112
Figure 65: InputScreen “TimeUS” Example (6869i/6930) 113
Figure 66: InputScreen “TimeUS” Example (6873i/6940) 113
Figure 67: InputScreen “TimeInt” Example (6863i/6865i) 114
Figure 68: InputScreen “TimeInt” Example (6869i/6930) 115
Figure 69: InputScreen “TimeInt” Example (6873i/6940) 115
Figure 70: InputScreen “DateUS” Example (6863i/6865i) 116
Figure 71: InputScreen “DateUS” Example (6869i/6930) 117
Figure 72: InputScreen “DateUS” Example (6873i/6940) 117
Figure 73: InputScreen “DateInt” Example (non softkey phone) 118
Figure 74: InputScreen “DateInt” Example (6869i/6930) 119
Figure 75: InputScreen “DateInt” Example (6873i/6940) 119
Figure 76: InputScreen implementation on 6869i/6930 “normal” mode 121
Figure 77: InputScreen implementation on 6869i/6930 “condensed” mode 122

Figure 78: InputScreen implementation on 6873i/6940 “normal” mode 123
Figure 79: InputScreen implementation on 6873i/6940 “condensed” mode 123
Figure 80: InputScreen multiple inputs “condensed” (6869i/6930) 130
Figure 81: InputScreen multiple inputs “condensed” (6873i/6940) 130
Figure 82: InputScreen multiple inputs “normal” (6869i/6930) 131
Figure 83: InputScreen multiple inputs “normal” (6873i/6940) 132
Figure 84: PhoneStatus Example (6863i/6865i) 137
Figure 85: PhoneStatus Example (6869i/6930) 137
Figure 86: PhoneStatus icons example 138
Figure 87: PhoneStatus icons example 138
Figure 88: PhoneStatus icons example 139
Figure 89: PhoneStatus toaster example 139
Figure 90: “xmladvanced” key attributes 147
Figure 91: XML colors (6867i/6869i/6873i/6920/6930/6940) 165
Figure 92: HTTP flow for XML applications 224
Figure 93: Auto-configuration message flow 232
Figure 94: Self-configuration implementation architecture 233
Figure 95: LDAP directory message flow 302

1 INTRODUCTION

1.1 MITEL XML API

Mitel SIP phones support an XML API since firmware release 1.3.0. This XML API allows external applications to control the display of the phone as well as its configuration.

The list of potential XML applications is endless, see chapter 9 for some examples of potential XML applications.

This document details the XML objects supported by the Mitel SIP phones using firmware version 5.0.0 and how to implement them.

1.2 REVISION HISTORY

1.2.1 VERSION 5.0.0 (6863I / 6865I / 6867I / 6869I / 6873I / 6920 / 6930 / 6940)

- Support for Mitel 6920/6930/6940 SIP Phones (warning HTTP User Agent is Mitel69xx for these phones)
- New background color tag (“bgColor”) for all UI XML objects (not available for 6863i and 6865i)
- New notification type for PhoneStatus object to allow access to the “toaster” notification mechanism (not available for 6863i and 6865i)
- On the 6873i/6940 the pseudo-LED now supports slow and fast blinking
- New key for keypress emulation to simulate the Page keys on the M685i and M695 expansion modules
- New “CallProtection” tag for all UI XML objects to avoid the XML screen to be destroyed on an incoming call.
- PhoneSoftkey icons can now be pictures or avatars downloaded from the image server.
- PhoneSoftkey now supports a “status left” and “status right” attributes to display a status icon over the icon.
- LED control for softkeys now supports “active/inactive” colors to match new softkey UI not available for 6863i and 6865i)
- New XML command "Command ClearMobile" is added to clear the Mobile directory and Mobile device.
- New predefined icons (not available for 6863i and 6865i)
    - "Icon:CallFwdActive"
    - "Icon:CallFwdInactive"
    - "Icon:DndActive"
    - “Icon:DndInactive"
    - "Icon:MakeBusyActive"
    - "Icon:MakeBusyInactive"
    - "Icon:NightServiceActive"
    - "Icon:NightServiceInactive"
    - "Icon:PhoneLockActive"
    - "Icon:PhoneLockInactive"
    - “Icon:RecordingOn”
    - “Icon:RecordingOff”
    - "Icon:StateActive"
    - "Icon:StateActive2"
    - "Icon:StateInactive"

1.2.2 VERSION 4.3.0-SP1 (6863I / 6865I / 6867I / 6869I / 6873I)

- New AastraIPPhoneExecute key press “key:NavEnter” to emulate the cluster center key.

1.2.3 VERSION 4.3.0 (6863I / 6865I / 6867I / 6869I / 6873I)

- New predefined icons for 6867i, 6869i and 6873i
    - “Icon:Alarm”,
    - “Icon:AlarmFilled”,
    - “Icon:Reset”,
    - “Icon:CallTransfer”,
    - “Icon:Park”,
    - “Icon:PhoneConnected”,
    - “Icon:PhoneDiverted”,
    - “Icon:PhoneDivertedA”,
    - “Icon:PhoneDivertedB”,
    - “Icon:PhoneRecall”,
    - “Icon:PhoneRingingA”,
    - “Icon:PhoneRingingB”
    - “Icon:GroupRoomOccupied”
    - “Icon:GroupRoomVacant”
    - “Icon:RoomOccupied”
    - “Icon:RoomVacant”
    - “Icon:RoomNotInspected”
    - “Icon:RoomInspected”
    - “Icon:RoomNotClean”
    - “Icon:WakeUpExpired”
    - “Icon:WakeUpPending”
- New icon positions (up to 4) for AastraIPPhoneTextMenu object (6867i, 6869i and 6873i)
- New XML object “AastraIPPhoneSoftkey” to control the layout and behavior of top softkeys when they are typed as “xmladvanced” (6867i, 6869i and 6873i)
- New LED state “active” for AastraIPPhoneExecute command to match BLF state via XML (6867i, 6869i and 6873i)
- New XML variables to be used in URIs
    - $$LINEINDEX$$ for the line Index of focused line (SIP registration line number of the focused line)
    - $$REMOTEURI$$ for the Remote party SIP URI (incoming or outgoing)
    - $$DIVERSIONURI$$ for the full URI of the Diversion information
    - $$DIVERSIONREASON$$ for the Diversion Reason information
- Updated XML colors to be more consistent with phone UI (6867i, 6869i and 6873i)
- New “wrap” tag for individual lines in AastraIPPhoneFormattedTextScreen to allow text wrapping for individual lines (6867i, 6869i and 6873i).

- New “blink” tag for individual lines in AastraIPPhoneFormattedTextScreen to allow slow or fast blinking of the line (6867i, 6869i and 6873i).
- New XML object “AastraIPPhoneIconMenu” which is a graphical equivalent of AastraIPPhoneTextMenu (6867i, 6869i and 6873i).
- Modification of the RTPRx and RTPMRx commands available via AastraIPPhoneExecute which now reconnects to an existing audio stream after user dropped the paging call if same port/address is used. A new command “RTPRx:Resume” and “RTPMRx:Resume” performs the same reconnection using the configuration of the last connected stream.

1.2.4 VERSION 4.2.0-SP1 (6863I / 6865I / 6867I / 6869I/ 6873I)

- New AastraIPPhoneExecute commands to allow the XML launch of the internal applications:
  - LaunchDirectory
  - LaunchCallersList
  - LaunchRedialList
  - LaunchServices
- New *action blf uri* to override the BLF and BLF/Xfer key regular behavior.

1.2.5 VERSION 4.2.0 (6863I / 6865I / 6867I / 6869I/ 6873I)

- Support for 6873i touchscreen phone
- New user interface for 6867i and 6869i
- New “touchLaunch” root tag for the AastraIPPhoneTextMenu object (6873i only) to allow user to select and launch an item in the menu.
- New “fontMono” root tag for the AastraIPPhoneTextMenu and AastraIPPhoneFormattedTextScreen objects (6867i, 6869i and 6873i only) to allow user to indicate if a monotype or proportional font is displayed.
- New “title” tag in ExecuteItem for AastraIPPhoneExecute object to be used as a title on the Wav.Play screen (6867i, 6869i and 6873i only).
- Support for dynamic icons to complement native icons (6867i, 6869i and 6873i only).

1.2.6 VERSION 4.1.0 (6863I / 6865I / 6867I / 6869I)

- Bug fixes

1.2.7 VERSION 4.0.0 SP2 (6863I / 6865I / 6867I / 6869I)

- Bug fixes

1.2.8 VERSION 4.0.0 SP1 (6863I / 6865I / 6867I / 6869I)

- New Dial tag added to XML objects AastraIPPhoneTextScreen and for AastraIPPhoneFormattedTextScreen to allow dialing from these object going off-hook or via the custom softkey Softkey::Dial2.
- LED control of the XML hardkeys via AastraIPPhoneExecute.
- Alert status without timeout can now be removed sending an empty alarm using AastraIPPhoneStatus
- New input type for AastraIPPhoneInputScreen, “stringN” which behaves like a regular “string” input but the inputstarts with a number instead of a letter (toggle key starts with “123>” instead of “ABC>” on 67i/69i, pressing “2” on the keypad browses “2ABCabc” instead of “ABC2abc” on 63i/65i)
- 6867i/6869i only
  - Simulated LED control of the softkeys via AastraIPPhoneExecute.

- AastraIPPhoneInputScreen, exiting the object with parameter values is no longer limited to the Softkey::Submit custom softkey. The parameter values and selection are now added to any URI custom softkey.
- AastraIPPhoneStatus now supports custom icons to be displayed on the idle screen status bar, similar to 6739i

1.2.9 VERSION 4.0.0 (6863I / 6865I / 6867I / 6869I)

- Support for 6869i

1.2.10 VERSION 3.3.1 SP4 (9143I / 9480I / 9480ICT / 6730I / 6731I / 6739I / 53I / 55I / 57I / 57ICT / 6863I / 6865I / 6867I)

- 6867i only
  - Icon support in TopTitle tag for UI XML objects
  - Color tag for AastraIPPhoneStatus

1.2.11 VERSION 3.3.1 SP3 (9143I / 9480I / 9480ICT / 6730I / 6731I / 6739I / 53I / 55I / 57I / 57ICT / 6863I / 6865I / 6867I)

- Firmware not available for 6735i/6737i
- Added support for 6863i, 6865i and 6867i

1.2.12 VERSION 3.3.1 SP 2(9143I/9480I/9480ICT/6730I/6731I/6739I/53I/55I/57I/57ICT)

- Firmware not available for 6735i/6737i
- Wav.Repeat command removed.
- New Melody.Play and Melody.Stop commands to repeatedly play a wav file.

1.2.13 VERSION 3.3.1 (9143I/9480I/9480ICT/6730I/6731I/6739I/53I/55I/57I/57ICT)

- Firmware not available for 6735i/6737i
- AastraIPPhoneConfiguration now allows locking/unlocking softkeys and programmable hardkeys using the “softkeyN locked: 0/1” parameter.
- New wave file streaming command Wav.Repeat to repeat the wave streaming until it is stopped by a Wav.Stop command.
- Volume control tag added to the wave streaming commands Wav.Play and Wav.Repeat.

1.2.14 VERSION 3.3.0 (9143I/9480I/9480ICT/6730I/6731I/6739I/53I/55I/57I/57ICT)

- Firmware not available for 6735i/6737i
- AastraIPPhoneConfiguration now supports range settings in order for instance to reset a list of softkeys in a single command.
- New XML command to control all the softkey attached LEDs in a single command.
- New GoodbyeLockinURI parameter attaching an action to the Goodbye key when displayed XML UI object is in LockedIn mode.
- New phone variable $$ACTIVEPROXY$$ reflecting current active proxy and providing XML server redundancy information.
- Mitel 6739i only
  - AastraIPPhoneStatus can be used to display icons on the top line
  - Custom icons can now be downloaded from the server as a png file
  - Label color in every UI object can now be configured

- New Call log XML object to provide call logs or redial applications equivalent to the native 6739i UI.
- New line wrapping control mechanism in AastraIPPhoneTextMenu items using the split tag
- Top line title and icon can now be customized by the XML UI objects using TopTitle tag
- AastraIPPhoneInputScreen now supports immediate edition mode using the defaultFocus tag

1.2.15 VERSION 3.2.2 (9143I/9480I/9480ICT/6730I/6731I/6735I/6737I/6739I/53I/55I/57I/57ICT)

- Support for Mitel 6735i and Mitel 6737i SIP phones

1.2.16 VERSION 3.2.2 (9143I/9480I/9480ICT/6730I/6731I/6739I/53I/55I/57I/57ICT)

- Some bug fixes, see firmware release notes for more details
- Mitel 6739i only
  - Default softkey “Done” for all UI objects
  - Idle screen now supports a background picture.

1.2.17 VERSION 3.2.1 (9143I/9480I/9480ICT/6730I/6731I/6739I/53I/55I/57I/57ICT)

- Mitel 6739i only
  - AastraIPPhoneImageScreen and AastraIPPhoneImageMenu now support jpeg images on top of png images
  - AastraIPPhoneImageScreen and AastraIPPhoneImageMenu now support 3 image size
    - 380x340 pixels
    - 640x340 pixels
    - 640x480 pixels
  - AastraIPPhoneTextMenu now supports a new tag “unitScroll” to change the behavior of the scrolling and mimic the behavior of a 57i (the arrow keys move the selected item one by one)
  - AastraIPPhoneStatus now supports custom icons to be displayed along with the alert/message text and also an URI to be triggered when the user actually presses the message.
  - AastraIPPhoneInputScreen now supports the inputLanguage tag
  - Wave file streaming commands now supported
- All phones
  - AastraIPPhoneTextMenu now supports new tags “scrollUp” and “scrollDown” which are triggered when the scrolling reaches the top or the bottom of the menu items
  - Feedback on XML errors as well as ongoing operations is now provided to the user.

1.2.18 VERSION 3.2.0 (9143I/9480I/9480ICT/6730I/6731I/6739I/53I/55I/57I/57ICT)

- Mitel 6739i now supports
  - AastraIPPhoneImageScreen loading a remote png file
  - AastraIPPhoneImageMenu loading a remote png file
  - Custom XML menu under the “Services” key
  - Beep in the root tag
  - Icons for Textmenu items and custom softkeys
  - RTP streaming
  - Lock/Unlock and ClearCallersList commands
- LED control on expansion modules

- XML object AastraIPPhoneDirectory no longer supported
- Mitel 6751i no longer supported
- Play Wav command command is supported on all phones but the 6739i.

1.2.19 VERSION 3.0.1 (6739I)

- Support for Keypress emulation but limited to hard keys.
- AastraIPPhoneInputScreen now supports multiple input fields.
- AastraIPPhoneFormattedTextScreen is now supported with 1 enhanced tag and 1 new tag to leverage the Mitel 6739i color display
- Size now supports “small” and “large” on top of “double”
- A Color tag is now supported
- AastraIPPhoneStatus is now supported.
- Also the capability to override a ‘telephony’ key by an XML script is now supported using the following configuration parameters:
    - services script
    - callers list script
    - directory script
    - redial script
    - xfer script
    - conf script
    - icom script
    - voicemail script
    - options script
- UTF-8 encoding is now supported and is the default mode.

1.2.20 VERSION 3.0.0 (6739I)

- First version of the XML API SDK dedicated to the 6739i running firmware 3.0.0

1.2.21 VERSION 2.6.0 (9143I/9480I/9480ICT/6730I/6731I/51I/53I/55I/57I/57I CT)

- AastraIPPhoneInputScreen now supports up to 10 fields in normal mode.
- New action uri connected which is triggered when the phone enters in the “connected” state.
- New line parameter for the Dial tag in AastraIPPhoneTextMenu to indicate which SIP line to use when the dial command is performed
- New DialLine URI equivalent to the Dial command but adds the SIP line to use.
- New AastraIPPhoneExecute command “UploadSystemInfo” for crash file and configuration files retrieval

1.2.22 VERSION 2.5.3 (9143I/9480I/9480ICT/6730I/6731I/51I/53I/55I/57I/57ICT)

- New scrollConstrain tag for AastraIPPhoneTextMenu to control the wrap-around in the list. When enabled, scrolling down on the last entry of the list does not wrap to the first item.
- New numberLaunch tag for AastraIPPhoneTextMenu to allow a user to launch the following menu by typing its number on the keypad, of course it is limited to menu 1-9.
- New scrollUp, scrollDown, scrollLeft and scrollRight tags for
    - AastraIPPhoneTextScreen,

- AastraIPPhoneFormattedTextScreen
- AastraIPPhoneImageScreen

which allows an override of the navigation keys default behaviour.

- "Please Wait..." is now displayed instead of "Loading Page..." when the phone is waiting for an answer of the XML server.
- AastraIPPhoneExecute, when using a regular URI the command now supports phone variables.
- Localized input mapping has been modified to allow the input of the '+' character by pressing '0' twice.
- XML objects now properly support UTF-8 for the encoding and the charset.

1.2.23 VERSION 2.5.2 (9143I/9480I/9480ICT/6730I/6731I/51I/53I/55I/57I/57I CT)

- Documentation fixes.
- Object oriented php classes updates.
- Updated sample csv directory application allowing search by company.
- Updated media applications.

1.2.24 VERSION 2.4.1 (9143I/9480I/9480ICT/6730I/6731I/51I/53I/55I/57I/57I CT)

- Support for the new Mitel 6730i and 6731i SIP phones.

1.2.25 VERSION 2.4.0 (9143I/9480I/9480ICT/51I/53I/55I/57I/57I CT)

- New action uri onRegistrationEvent which is triggered each time the registration status of the phone changes.
- New setType tag for AastraIPPhoneConfiguration which allows to set configuration parameter at a different precedence than just the server provided configuration.
- New contextual softkeys 'Drop', 'Conf' and 'Xfer' available in the connected state, these new softkeys are similar to the 'Answer' and 'Ignore' softkeys for incoming call state. For tnhe non softkey phones, new 'allowDrop', 'allowXfer' and 'allowConf' have been added.
- action uri disconnected now triggered at the end of a wav file stream.

1.2.26 VERSION 2.3.1 (9143I/9480I/9480ICT/51I/53I/55I/57I/57I CT)

- New configuration parameter webapps uri to
  - Remove the WebApps option from the 51i, 53i, and 9143i Service Menus, and disable the WebApps key for all other phone models.
  - Specify a customized URI for the WebApps key instead of using the default URI of http://xml.myaastra.com/?localip=$$LOCALIP$$

1.2.27 VERSION 2.3.0 (9143I/9480I/9480ICT/51I/53I/55I/57I/57I CT)

- New XML configuration
- The following telephony keys can now be assigned to call an XML script instead of the regular telephony feature:
  - Redial
  - Transfer
  - Conference
  - Intercom
  - VoiceMail
- Phone 'Options' Key can now be redirected to an XML script.

- On a normal key press of the options key the XML application set in the URI is displayed
- On a long key press the normal/local options menu is displayed
- New configuration auto offhook to prevent the phone from going into the offhook/dialing state when the handset is off hook and the call ends. This parameter is very useful for XML applications such as Visual Voicemail to stay in the application at the end of the message.
- New PhoneExecute commands
    - Play/stop a wav file hosted on a HTTP or TFTP server.
    - Simulate keypress on the phone
    - Reset local Callers list
    - Reset local Directory
    - Reset local Redial list
    - Delete the local configuration and reboot
- New attributes
    - New root tag attribute allowDTMF in some interactive UI objects to allow DTMF passthrough when the phone is in the connected mode.
    - AastraIPPhoneTextScreen,
    - AastraIPPhoneFormattedTextScreen,
    - AastraIPPhoneImageScreen
- New action uri(s)
    - action uri disconnected is triggered when the phone transitions from any active (non idle) call state to idle.
- New URL variables
    - $$LOCALIP$$ Local IP address of the phone
    - $$CALLDURATION$$ Current/last Call duration
    - $$CALLDIRECTION$$ Current/last call direction
    - $$LINESTATE$$ Line state when the 'disconnect' event happens.
- Others
    - Custom SSL certificates for HTTPS can now be loaded on the phones.
    - Pre-programmed XML Service Key called WebApps pointing to an Mitel hosted service located at http://xml.myaastra.com/?localip=$$LOCALIP$$ that you can also access from your Web browser at www.myaastra.com
    - Enhancements/fixes
    - XML calls are now non-blocking for action uri and pushed XML objects
    - Action uri GET calls no longer display an error message if failure to load the page.
    - Unicast and Multicast mixed RTP streaming, the phone can now send ear+mouth and not just mouth or receive an RTP stream over an existing one.
    - HTTPS URI now supported in the AastraIPPhoneExecute object.

1.2.28 VERSION 2.2.1 (9143I/9480I/9480ICT/51I/53I/55I/57I/57I CT)

- Support of the 9143i, 9480i and 9480iCT SIP Phones

1.2.29 VERSION 2.2.0 (51I/53I/55I/57I/57I CT)

- New wrapList root tag for the AastraIPPhoneTextMenu object to display menu items on 2 lines.

- Number of items extended to 30 (instead of 15) for the AastraIPPhoneTextMenu object.
- Action uri polling with a customizable interval to have the phone call an XML URI on a regular basis
- SIP notify to trigger a XML GET, to have the phone making a XML call when an authorized SIP Notify is sent via the SIP proxy server.
- The XML browser is now available when the phone Web UI is disabled.
- Support of Unicast and Multicast RTP streaming triggered by an XML call.
- Improved error handling, the phone now displays the HTTP error message when available instead of "Cannot display".
- XML GET are now non blocking, the phone keeps processing events when waiting for an answer to an HTTP GET.

1.2.30 VERSION 2.1.1 (51I/53I/55I/57I/57I CT)

- Support for the new 51i SIP Phone
- Lock and Unlock command for the AastraIPPhoneExecute object to lock or unlock the phone.
- inputLanguage root tag for the AastraIPPhoneInputScreen XML object. This tag allows access to language localized input characters.
- New configuration parameter xml lock override added in order to allow a XML Push when the phone is locked.

1.2.31 VERSION 2.1.0 (53I/55I/57I/57I CT)

- `doneAction` root tag for the `AastraIPPhoneTextScreen` and `AastraIPPhoneFormattedTextScreen` XML objects. This tag allows redirecting the user to a specified URI after a "Done" key press; this can be very useful for non softkey phones such as the Mitel 53i.
- New custom softkeys "Ignore" and "Answer" and "allowAnswer" tag to answer a call when an XML page is called upon the incoming call.
- New custom softkey (List) for the AastraIPPhoneInputScreen to enter a list of configured symbols. This new softkey allows for instance an easy email address input using the "@." List of symbols.
- New configuration parameter xml get timeout added in order to control the server answer delay.
- New configuration parameters services script, directory script and callers list script to override internal applications and link the features to an XML script.

1.2.32 VERSION 2.0.2 (53I/55I/57I/57I CT)

- New command "FastReboot" for the PhoneExecute object to trigger a fast reboot of the phone (no firmware check and limited language package check).
- New universal URI type "Led:" support to control the LED state of the phone keys when they are typed as XML.
- Extension of the AastraIPPhoneInputScreen XML object to support multiple input fields (55i/57i and 57i CT only)

1.2.33 VERSION 2.0.1 (53I/55I/57I/57I CT)

- New XML objects
    - AastraIPPhoneConfiguration
    - AastraIPPhoneImageScreen (55i/57i/57i CT)
    - AastraIPPhoneImageMenu (55i/57i/57i CT)
    - AastraIPPhoneFormattedScreen
- Other

- HTTPS support for XML calls.
- Custom port support for http(s) XML calls
- “Dial:XXXX” universal URI support
- New “style” root tag for AastraIPPhoneTextMenu
- Optional “Title” tag for all UI objects
- “Title” wrapped on 2 lines (“wrap” tag).
- Cancel remap for all UI XML objects
- Timeout attribute common to all UI XML objects
- LockIn attribute common to all UI XML objects
- triggerDestroyOnExit root tag for non UI objects
- Icons in AastraIPPhoneTextMenu (55i/57i/57i CT)
- Icons in customizable softkeys (55i/57i/57i CT)
- New HTTP header to indicate the presence of expansion modules
- New input types for AastraIPPhoneInputScreen
  - timeUS, timeInt
  - dateUS, dateInt

1.2.34 VERSION 1.4.2 (9112I/9133I/480I/480I CT)

- New attributes
- New “style” tag for AastraIPPhoneTextMenu

1.2.35 VERSION 1.4.1 (9112I/9133I/480I/480I CT)

- New XML objects
  - AastraIPPhoneStatus
  - AastraIPPhoneExecute
- Action URIs
  - End of the boot sequence action uri startup
  - Successful registration action uri registered
  - On-hook action uri onhook
  - Off-hook action uri offhook
  - Incoming call action uri incoming
  - Outgoing call action uri outgoing
- URI System variables
  - $$SIPUSERNAME$$ line user name
  - $$DISPLAYNAME$$ the display name of the focused line
  - $$SIPAUTHNAME$$ the SIP auth name of the focused line
  - $$PROXYURL$$ the SIP proxy of the focused line
  - $$INCOMINGNAME$$ returns the Caller-ID of the incoming call
  - $$REMOTENUMBER$$ returns the number of the remote
- Other
  - No more need of URL encoding for pushed pages

- Beep attribute common to all XML objects

1.2.36 VERSION 1.3.1 (9112I/9133I/480I/480I CT)

- Customizable softkeys (480i/480i CT)
  - Select
  - Exit
  - Dial
  - Submit
  - BackSpace
  - NextSpace
  - Dot
  - ChangeMode
- Other
  - XML softkeys (480i/480i CT)
  - XML programmable keys (9112i/9133i)
  - destroyOnExit attribute common to all XML objects

1.2.37 VERSION 1.3.0 (9112I/9133I/480I/480I CT)

- Fist revision of this document
- New XML objects
  - AastraIPPhoneTextScreen
  - AastraIPPhoneTextMenu
  - AastraIPPhoneInputScreen
  - AastraIPPhoneDirectory (480i/480i CT)

2 XML AND THE MITEL SIP PHONES

2.1 WHAT IS XML?

XML stands for **eXtensible Markup Language**. It is a markup language much like HTML. HTML was designed to display data and to focus on how data looks. XML was designed to describe data and to focus on what data is.

The following are characteristics of XML:

- XML tags are not predefined. You must define your own tags
- XML uses a Document Type Definition (DTD) or an XML Schema to describe the data
- XML with a DTD or XML Schema is designed to be self-descriptive
- XML is a W3C Standard Recommendation

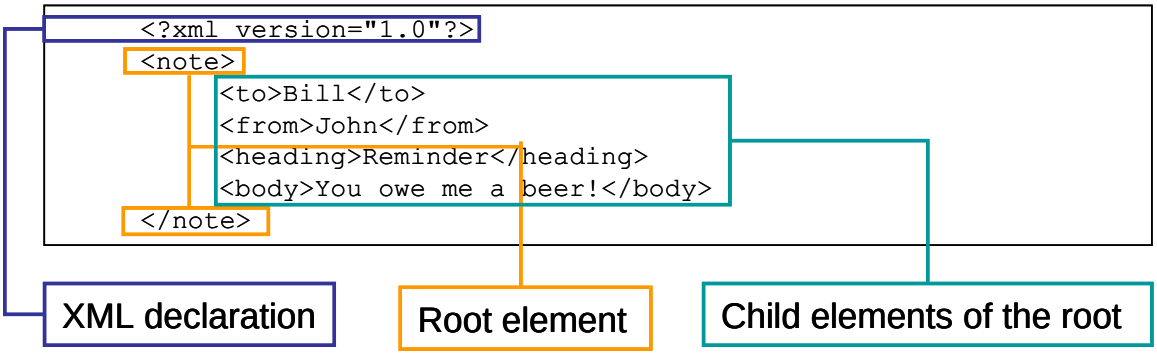


Figure 1: Basic XML document

More information available at http://www.xml.com

2.2 FUNCTIONALITY

The XML browser in Mitel IP phones allows developers to create custom services that they can use via the phone's keypad and display. These services include things like weather and traffic reports, contact information, company info, stock quotes, or custom call scripts.

With firmware release 5.0.0, the Mitel 6739i XML API supports 5 proprietary objects that allow the creation of powerful XML applications.

There are 2 types of XML objects:

- UI objects, XML objects which will use the display of the phone when they are received.
- Non UI objects, XML objects which have no direct impact on the current display.

The supported objects are:

- TextMenu object (UI)
- IconMenu (UI)
- TextScreen object (UI)
- FormattedTextScreen (UI)
- InputScreen object (UI)
- ImageScreen (UI)
- ImageMenu (UI)
- Execute object
- Configuration object

- Status object
- PhoneSoftkey

Some of these objects also support customizable softkeys that are declared as an independent object.

The following sections describe the process of creating XML objects for the Mitel SIP phones.

2.3 HOW DOES IT WORK?

Leveraging on the IP infrastructure, Mitel has decided to develop the browser capability on the phone using the HTTP transport protocol but as a direct support of HTML would not be suitable for the phone horsepower and limited display, the choice has been made to support only XML objects in the browser.

The Mitel SIP phones support two types of applications:

- Phone-initiated
- Server-initiated

2.3.1 PHONE INITIATED APPLICATION

The phone issues an HTTP (or HTTPS) GET command to the Web server, waits for the answer, decodes and displays this answer as any Web browser such as Microsoft Internet Explorer or Firefox would do as a Web client.

This can be done through a phone custom softkey and from the list of custom features (see chapter 7 for more details).

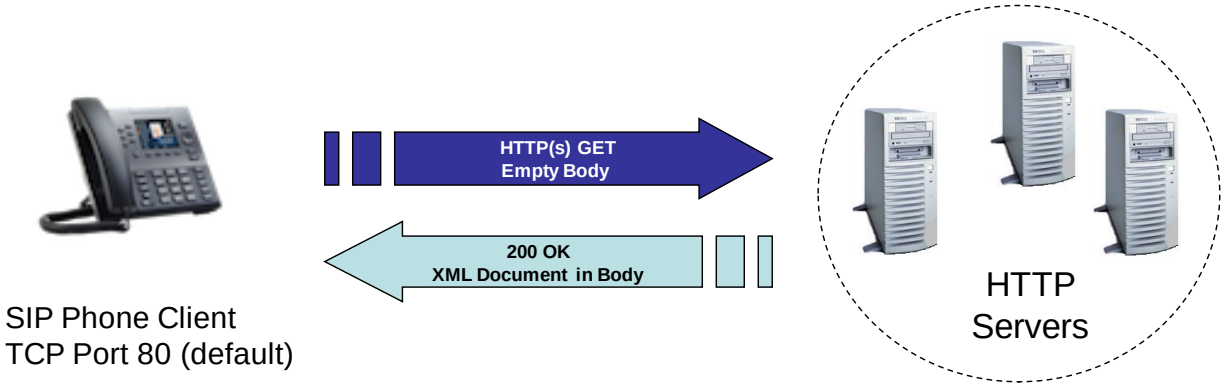


Figure 2: Mitel IP Phone acting as a client

When the phone performs an HTTP GET, it is a non blocking operation (phone will keep processing other events) as long as the GET is not requested by an action uri.

2.3.2 SERVER INITIATED APPLICATION

The other type of application would be more used for alerting as an application is pushing an XML object to the phone. The phone is now acting as a limited Web Server.

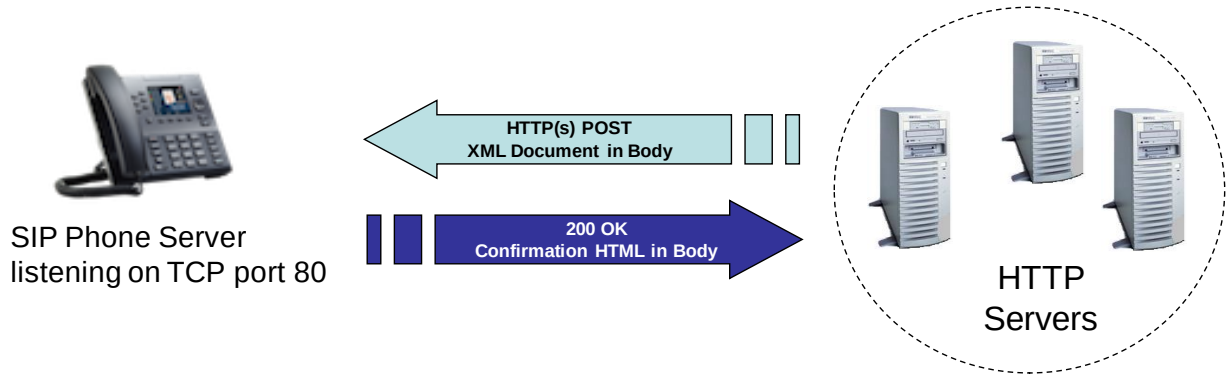


Figure 3: Mitel IP Phone acting as a server

2.4 SYSTEM ARCHITECTURE

The XML applications are hosted by one or multiple Web servers which will serve as a proxy to either other applications or to Internet Web servers.

2.4.1 CORPORATE APPLICATIONS

The following figure details the architecture to allow Mitel IP Phones to access an internal application. The application hosted by the Web server translates the phone requests to a protocol specific to the target application and formats the answer as an XML object to be displayed on the phone.

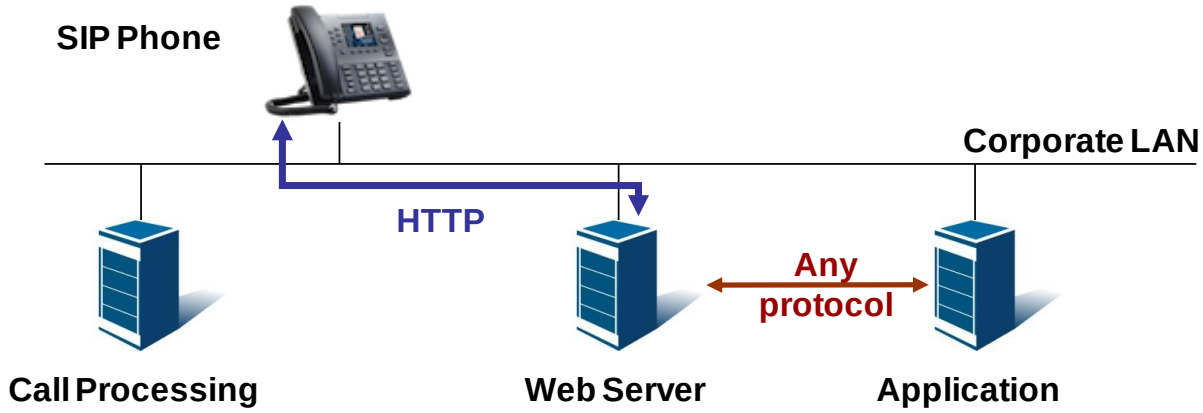


Figure 4: Access to an internal application

The following figure details the architecture of an XML application that would retrieve data from the internet such as a real-time stock-quote service.

Note: for certain Web applications that are not real time, the Internet content can be cached on the XML web server for a faster access.

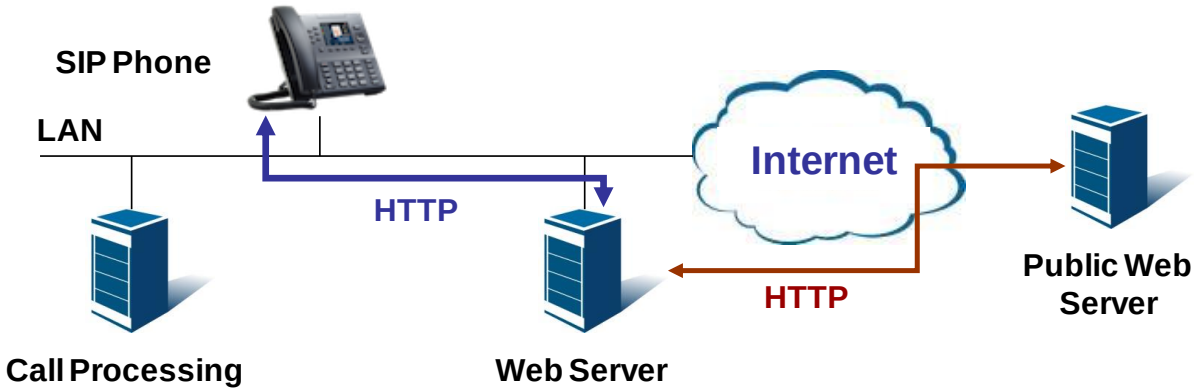


Figure 5: Access to an Internet application

2.4.2 TELEPHONY APPLICATIONS

The following figure details the architecture of an XML application that would provide more telephony features at the phone level.

The application could for instance

- show the list of the parked calls and perform a pick-up,
- activate the Call forward or the DND on the server side
- control a conference from the phone
- login/logout from a call center, access to the voice mail messages

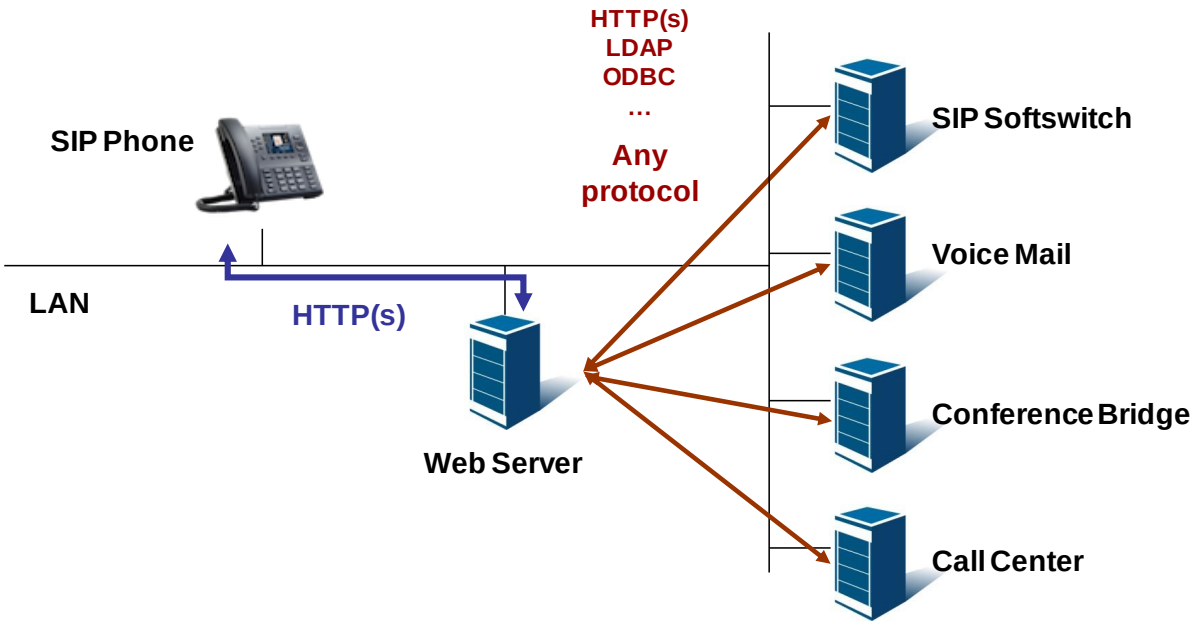


Figure 6: Access to a telephony application

2.5 DEVELOPMENT ENVIRONMENT

2.5.1 TYPICAL SOFTWARE ARCHITECTURE

The following diagram details the typical architecture of an XML application.

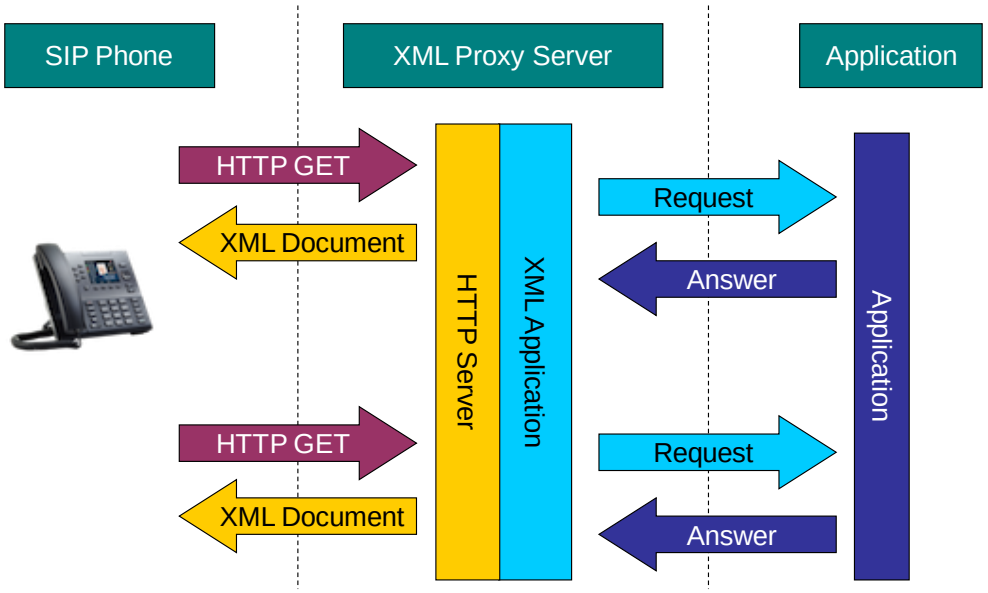


Figure 7: Typical software architecture of an XML application

The XML application is in fact translating requests from the phone to the protocol used by the external application and formats the answer into an XML document that the Mitel SIP phone can interpret.

2.5.2 WEB SERVER

There is no constraint in the choice of the Web Server software to be used for Mitel XML applications; the choice is more based on the tools needed for the development such as the script language, the corporate policy or even the cost of the platform.

The most common Web server applications supported are:

- Apache (http://www.apache.org) for Microsoft and Linux Operating systems
- Microsoft IIS for Microsoft Operating systems

2.5.3 SCRIPTS/LANGUAGES

As for the Web Server, there is no specific constraint on the tools to develop the applications. All the languages supported to develop a Web application are supported to develop XML applications. The most common are:

- Compiled languages: C, C++, C#...
- Scripting languages: VBscript, Perl, Python, PHP, asp…

2.5.4 XML VALIDATION TOOLS

A large number of tools are available to validate the XML document you will be sending to the phone, these tools use the XSD schema (provided at chapter 11) to check the syntax of the generated XML document.

Example of a Web based tool

- http://tools.decisionsoft.com/schemaValidate/

2.6 XML FORMAT

The text in the Mitel XML objects must be compliant with XML recommendations and special characters must be escape encoded:

Character	Name	Escape Sequence
&	Ampersand	&

Character	Name	Escape Sequence
“	Quote	"
‘	Apostrophe	'
<	Left angle bracket	<
>	Right angle bracket	>

Figure 8: XML conversion table

To respect XML recommendations, the following header can be set at the beginning of the XML document,

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

Or

```
<?xml version="1.0" encoding="UTF-8"?>
```

By default UTF-8 is used by the phone.

2.7 HTTP FORMAT

The HTTP message sent to the Mitel SIP Phone must respect HTTP/1.1 and must include the following parameters in the header:

- Content-Length
- Content-Type

The other parameters of the HTTP header are optional such as charset.

Example

```
HTTP/1.1 200 OK                                          } HTTP Header
Date: Tue, 15 May 2007 14:24:33 GMT
Server: Apache/2.0.52 (CentOS)
X-Powered-By: PHP/4.3.11
Content-Length: 564
Connection: close
Content-type: text/xml; charset=ISO-8859-1

<?xml version="1.0" encoding="ISO-8859-1"?>              } HTTP body
<AastraIPPhoneInputScreen type="string">
<Title>Title</Title>
<Prompt>Enter value</Prompt>
<URL>http://myserver.com/script.php</URL>
<Parameter>value</Parameter>
<Default></Default>
</AastraIPPhoneInputScreen>
```

2.8 XML DISPLAY CONTROL AND KEYS

This chapter describes the available part of the display for each Mitel SIP phones as well as the keys that are controlled by the XML objects.

2.8.1 MITEL 6863I

The display and keys available for XML applications on a Mitel 6863i are:

- 3 lines of 16 characters for the display
- the left and right arrow navigation keys
- the up and down arrow navigation keys

The 3rd line of the display is a command line and will be used to display the labels of the available actions. See chapter 3 for more details on how each XML object will use this line of command.

Depending on the XML object displayed on the phone,

- the left arrow navigation key can also be interpreted as a “cancel” key,
- the right arrow navigation key can also be interpreted as a “next” key,
- the “Delete” key is used as a “Backspace” for the AastraIPPhoneInputScreen object.

See chapter 3 for more detailed information on each object.

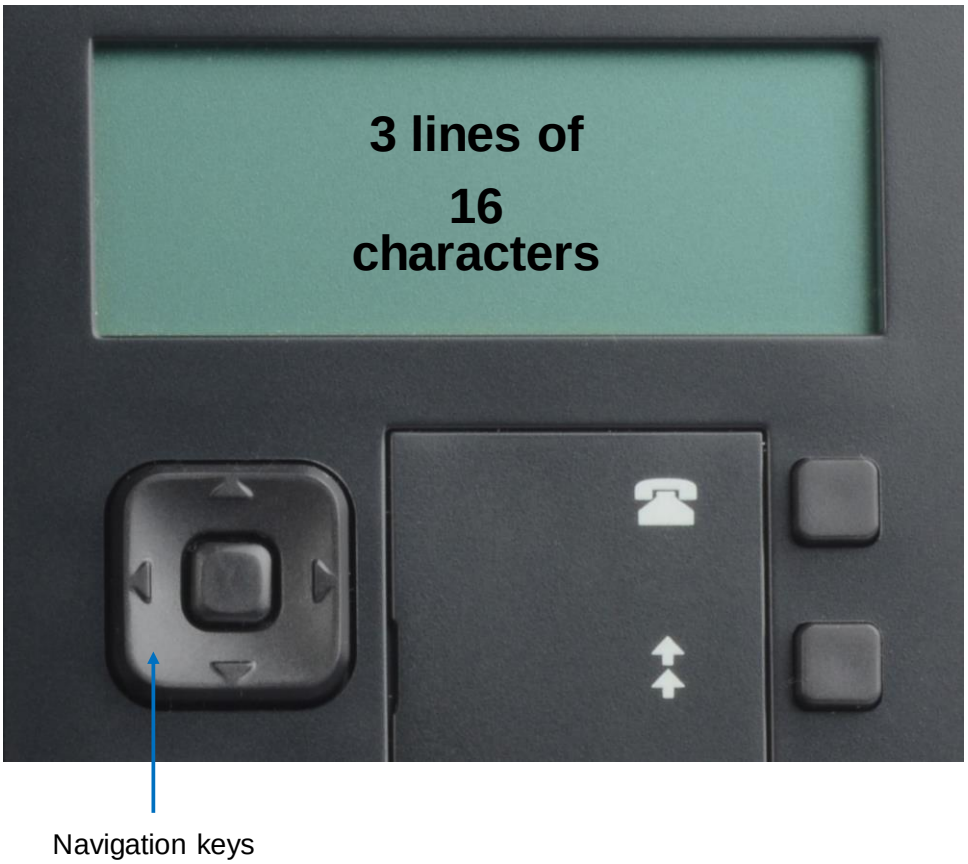


Figure 9: Mitel 6863i XML display and keys

2.8.2 MITEL 6865I

The display and keys available for XML applications on a Mitel 6865i are:

- lines of 16 characters for the display
- the left and right arrow navigation keys
- the up and down arrow navigation keys

The 3rd line of the display is a command line and will be used to display the labels of the available actions. See chapter 3 for more details on how each XML object will use this line of command.

Depending on the XML object displayed on the phone,

- the left arrow navigation key can also be interpreted as a “cancel” key,
- the right arrow navigation key can also be interpreted as a “next” key,
- the “Delete” key is used as a “Backspace” for the AastraIPPhoneInputScreen object.

See chapter 3 for more detailed information on each object.

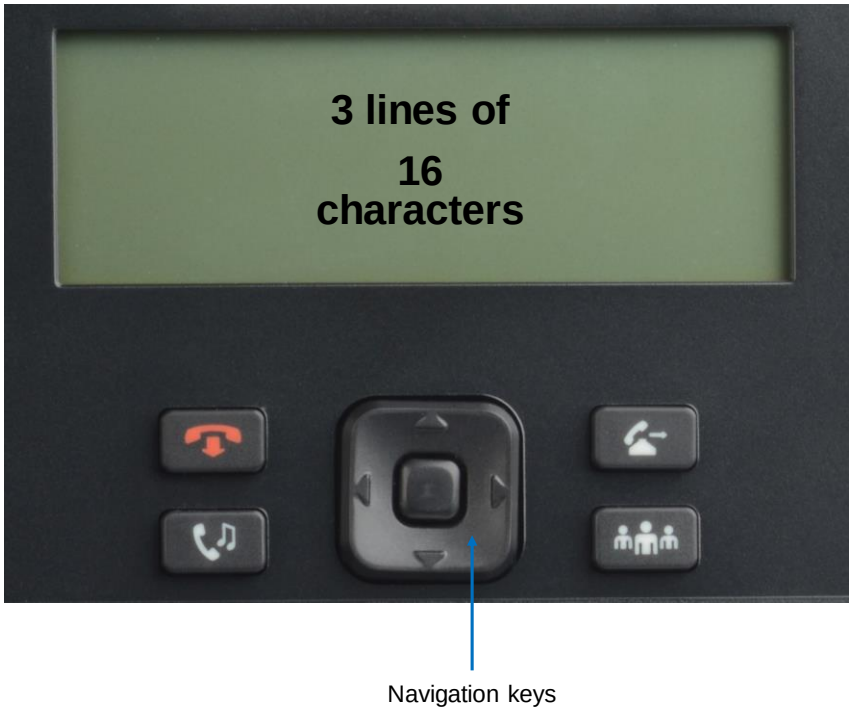


Figure 10: Mitel 6865i XML display and keys

2.8.3 MITEL 6867I / MITEL 6920

The display and keys available for XML applications on a Mitel 6867i or a Mitel 6920 are:

- 4 softkeys (6 logical softkeys for XML applications)
- A graphical zone for the XML objects
- Left/Right arrow navigation keys
- Up/Down arrow navigation keys
- Select key

See chapter 3 for more detailed information on each XML object.

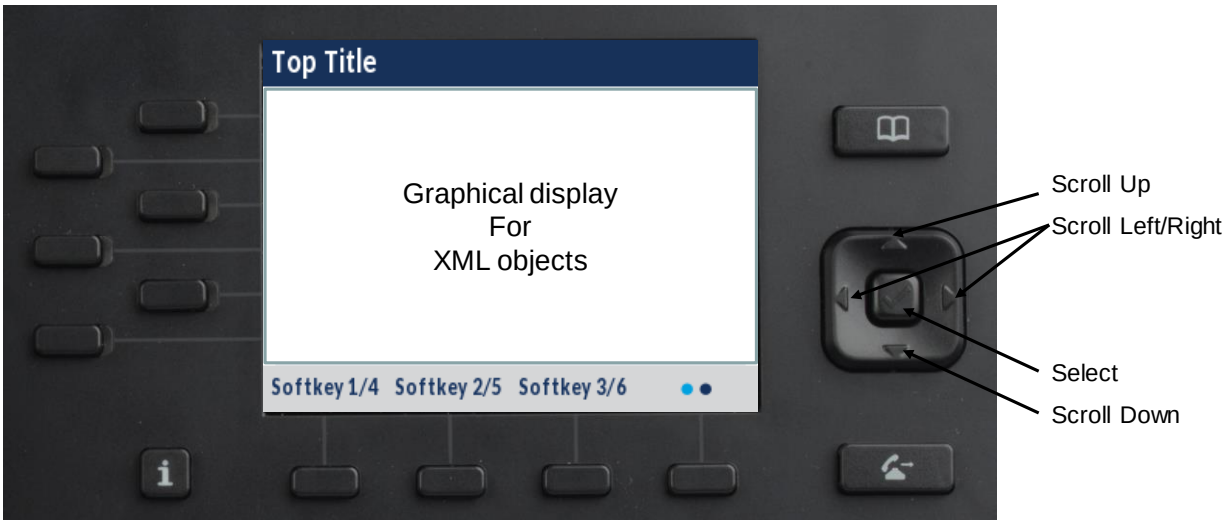


Figure 11: Mitel 6867i XML display and keys

2.8.4 MITEL 6869I / MITEL 6930

The display and keys available for XML applications on a Mitel 6869i or Mitel 6930 are:

- 5 softkeys (8 logical softkeys for XML applications)
- A graphical zone for the XML objects
- Left/Right arrow navigation keys
- Up/Down arrow navigation keys
- Select key

See chapter 3 for more detailed information on each XML object.

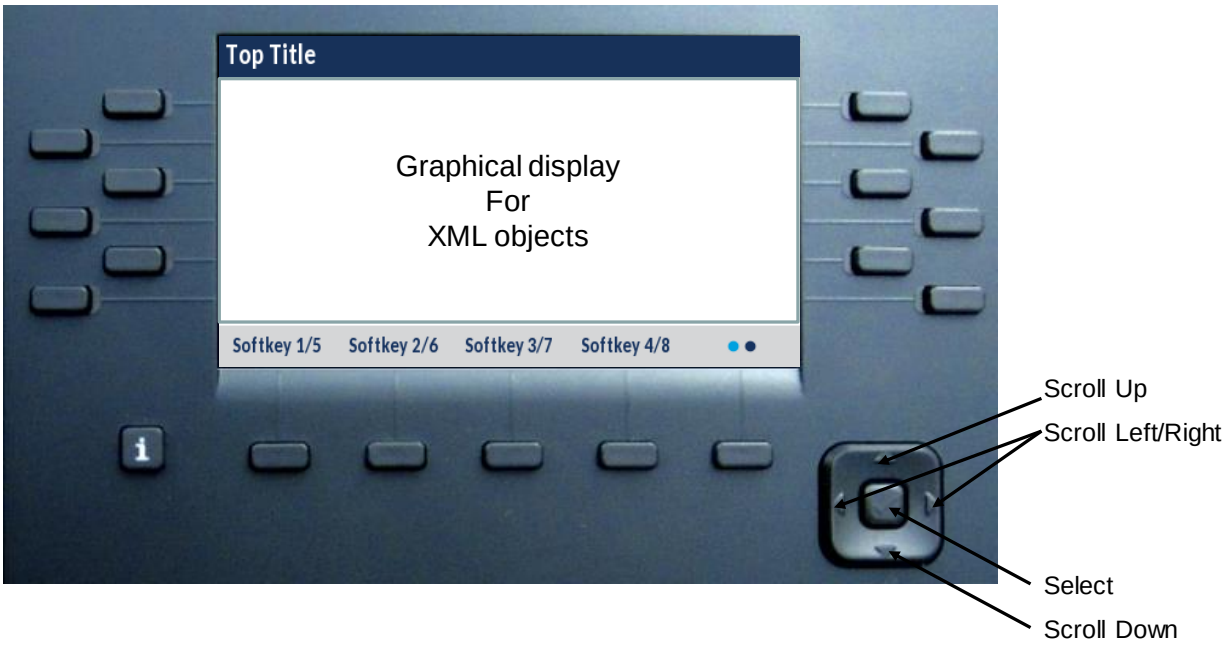


Figure 12: Mitel 6869i XML display and keys

2.8.5 MITEL 6873I / MITEL 6940

The display and keys available for XML applications on a Mitel 6873i or Mitel 6940are:

- 6 softkeys (10 logical softkeys for XML applications)
- A graphical zone for the XML objects
- No navigation key as the 6873i and 6940 are touch screen phones

See chapter 3 for more detailed information on each XML object.

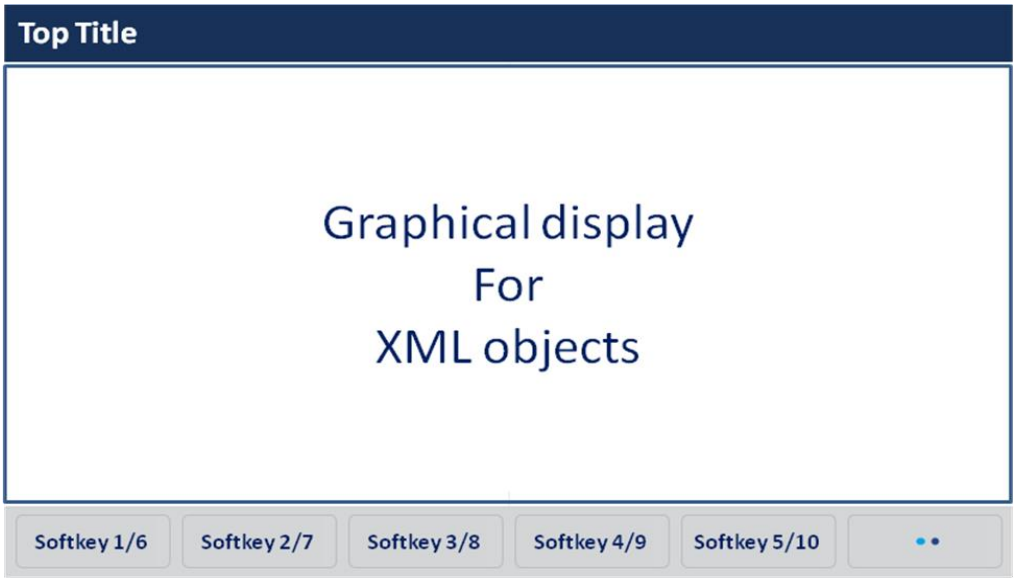


Figure 13: Mitel 6873i/6940 XML display

3 MITEL IP PHONE XML OBJECTS

This chapter details all the XML objects supported by the Mitel SIP phones

In this chapter:

Non softkey phones are Mitel phones without softkey:

- Mitel 6863i/6865i

Softkey phones are Mitel phones with softkeys:

- Mitel 6867i/6869i (graphical screen)
- Mitel 6920/6930 (graphical screen)

Touchscreen phones are Mitel phone(s) with a color graphical touch screen

- Mitel 6873i
- Mitel 6940

Notes:

.



- the size of an XML object can not exceed 10000 bytes (10 kb).
- per XML specifications, only one XML object is supported in the XML document sent to the phone.

3.1 TEXTMENU OBJECT (ALL MODELS)

The `TextMenu` object allows developers to create a list of menu items on the IP phones. Go-to line support, arrow indicator, and scroll key support are built into these objects, along with the "**Select"** and "**Done"** softkeys for the phones supporting softkeys. The TextMenu object allows users to navigate the application, by linking HTTP requests to menu items.

For more details on the TextMenu behavior using the `Selection` tag or the `Dial` tag, please refer to section 4.12 and 4.13.

3.1.1 IMPLEMENTATION (SOFTKEY AND NON SOFTKEY PHONES)

Object native interaction

- ***Select*** Executes the content of the URI field assigned to the selected MenuItem
- ***Exit*** Redisplays the previous XML object present in the phone browser.

Non softkey phone keys

The object is displayed on one line or one item at a time. The Up and Down arrow keys allow the user to browse the list up and down.

Line Selected	Label	Keys	
Title	`Use ^v to view`	Up and Down Arrow	Browse up or down
		Left Arrow	***Exit***
Item	`vNext`	Up and Down Arrow	Browse up or down
	`>Enter`	Right Arrow	***Select***
		Left Arrow	***Exit***

Softkey phone keys

The object is displayed on up to 6 lines, the title stays on the top of the display. The Up and Down arrow keys allow the user to browse the list up and down.

Line selected	Keys	
Item	Up and Down Arrow	Browse up or down
	Right Arrow	Select
	Left Arrow	Exit

Notes:

- the Left Arrow key interaction is disabled if the `LockIn` tag is set to "yes".
- the Left Arrow key interaction can be modified using the `cancelAction` tag.
- If the `LockIn` tag is set to "yes" and the `cancelAction` tag is configured, pressing the Left Arrow key triggers the configured `cancelAction`.

Object default Softkeys (softkey phones)

Position	Label	Interaction	URIs
1	Select	***Select***: Executes the content of the URI field assigned to the selected MenuItem;	SoftKey:Select
6	Done	***Exit:*** Redisplays the previous XML object present	SoftKey:Exit

Position	Label	Interaction	URIs
		in the phone browser.	

3.1.2 IMPLEMENTATION (6873I/6940)

The title is displayed at the top of the XML area and uses up to 2 lines. The menu items are displayed after the title area and up to 6 items (7 if no title) can be displayed at a time.

The item prompts are automatically wrapped on 2 lines if needed.

The scrolling is done by swiping the finger on the elements, pressing the left button of an item selects it.

The teal background on the button indicates that the item is selected.

The › icon located on the right side of each item is the “Select” button and triggers the default URI of the object.

Object default Softkeys

Ten customizable softkeys are available for this object.

Position	Label	Interaction	URIs
1	Select	***Select***: Executes the content of the URI field assigned to the selected MenuItem;	SoftKey:Select
6	Done	***Exit****:* Redisplays the previous XML object present in the phone browser.	SoftKey:Exit

The following figure details how the `AastraIPPhoneTextMenu` is implemented on the Mitel 6873i.

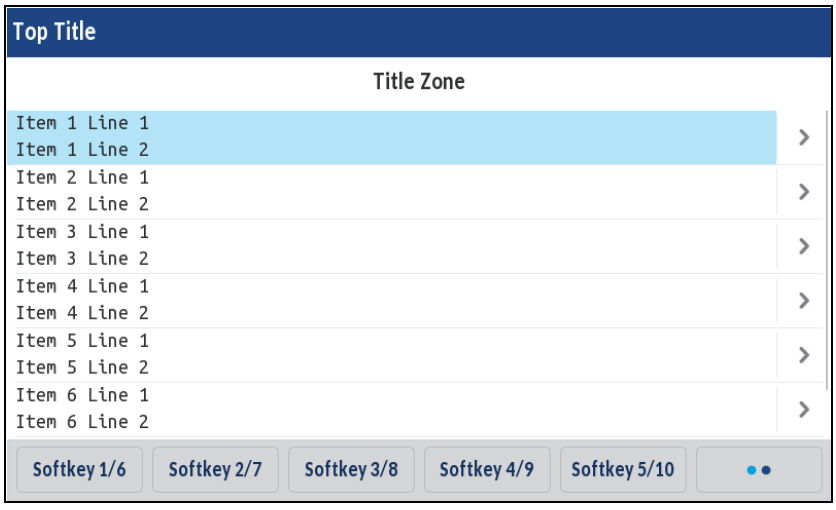


Figure 14: TextMenu Implementation on 6873i/6940

touchLaunch tag usage

The “touchLaunch” tag specific to the 6873i and 6940 changes the overall behavior of the AastraIPPhoneTextMenu object, when set to “yes”:

- The › button is no longer available
- User can select and launch the uri configured in the item by just pressing anywhere on the item
- Selection background is no longer displayed
- The default “Select” softkey is no longer displayed as well as custom softkeys using “Softkey:Select”, “Softkey:Dial” and “Softkey:Dial2”.

This allows the creation of quick touch menus.



Figure 15: TextMenu touchLaunch tag usage

fontMono tag usage

The “fontMono” tag specific to the 6867i, 6869i and 6873i changes the rendering of the TextMenu object, by allowing the usage of a proportional font instead of the default monotype font.



Figure 16: fontMono tag usage

icon tag usage

The “icon” tag allows to display an icon to the left of the MenuItem Label

The “iconr1” to “iconr4” tags specific to the 6867i, 6869i, 6873i, 6920, 6930 and 6940 allow to display up to 4 icons to the right side of the MenuItem. If one or more right icons are configured in one item of the list, the phone clips the label for all the MenuItems in the list to the the size of the MenuItem with the higher right icon.



Figure 17: icon positions in MenuItem

3.1.3 XML DESCRIPTION

“Red” tags indicate that the tag does not apply to all phones, when not supported the tags are just ignored.

```
<AastraIPPhoneTextMenu
    defaultIndex = “some integer”
    destroyOnExit = “yes/no”
    style = “numbered/none/radio”
    Beep = “yes/no”
    Timeout = “some integer”
    bgColor = white/black…”
    LockIn = “yes/no”
    CallProtection = “yes/no/notif”
    GoodbyeLockInURI = “some URI”
    allowAnswer = “yes/no”
    allowDrop = “yes/no”
    allowXfer = “yes/no”
    allowConf = “yes/no”
    cancelAction = “some URI”
    wrapList = “yes/no”
    scrollConstrain = “yes/no”
    scrollUp = “Some URI”
    scrollDown = “Some URI”
    numberLaunch = “yes/no”
    touchLaunch = “yes/no”
    fontMono = “yes/no”
    >
       <Title       wrap=”yes/no”
                    Color=”white/black…”
       >Menu Title</Title>
       <TopTitle    icon=”icon index”
                    Color=”white/black…”
       >Top Title</TopTitle>
       <MenuItem base = ”http://base/”
                 icon = “icon index”
                 iconr1 = “icon index”
                 iconr2 = “icon index”
                 iconr3 = “icon index”
                 iconr4 = “icon index”
              >
              <Prompt Color=”white/black…”
                      split=”integer”
              >First Choice</Prompt>
              <URI>http://somepage.xml</URI>
              <Dial line=”SIP line”>Number to dial</Dial>
              <Selection>Selection</Selection>
```

```
        </MenuItem>
        <!—Additional Menu Items may be added (up to 30)-->
        <IconList>
              <Icon index = "int">Icon:Iconname or HEX string</Icon>
              <!—As many as different icons used in the object -->
              <!—Up to 22 icons, index can be 1 to 21 -->
        </IconList>
        <!—Additional Softkey Items may be added (softkey phones)-->
</AastraIPPhoneTextMenu>
```



Notes:

- The number of items in a TextMenu object is limited to 30.
- You must declare at least one item in a TextMenu or the phone will generate a parsing error.
- Non softkey phones (6863i/6865i) support only "numbered" style.
- On graphic phones, picture ID icons ("Picture:XXXX" and "Picture:XXXX:YY") are supported see chapter 4.2.2 for more details

XML Document Objects

Document Object	Position	Type	Comments
AastraIPPhoneTextMenu	Root tag	Mandatory	Root object
defaultIndex	Root tag	Optional	Position of the cursor when the XML object is open. If not specified, the selection indicator is positioned on the first menu item.
style	Root tag	Optional	"numbered/none/radio" indicates the style of the TextMenu. Default is "numbered". *Ignored on 6863i and 6865i, style is always "numbered"* *Ignored on 6867i, 6869i, 6873i, 6920, 6930 and 6940, style is always "none"*
destroyOnExit	Root tag	Optional	"yes/no" indicates if the object is kept or not in the phone browser after exit. If not specified, the object is kept in the browser.
cancelAction	Root tag	Optional	Defines the URI to be called when the user cancels the XML object.
wrapList	Root tag	Optional	"yes" or "no" to indicate if the items will be text wrapped on 2 lines, default is "no". *Ignored on 6867i, 6869i, 6873i, 6920, 6930 and 6940, lines are automatically wrapped.*
Beep	Root tag	Optional	"yes" or "no" to indicate if a notification beep must be generated by the phone.

Document Object	Position	Type	Comments
Timeout	Root tag	Optional	Overrides the default 45 seconds timeout of the UI XML object. A Timeout set to “0” will disable the timeout feature. See section 4.10.2 for more details
bgColor	Root tag	Optional	Set the background color of the object. the possible values are detailed in chapter 4.2.3. If not specified, the default value is “white”. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*.
LockIn	Root tag	Optional	If set to “yes”, the phone ignores all events that would cause the screen to exit without using the keys defined by the object. Default value is “no”. See section 4.10.3 for more details.
CallProtection	Root tag	Optional	If set to “yes”, the phone will not destroy the XML object being displayed on an incoming call. If set to “notif”, the behavior is the same as “yes” but a notification is displayed on the screen providing the caller ID details. *“notif” is equivalent to “yes” on 6863i and 6865i as screen notification are not supported*.
GoodbyeLockInURI	Root tag	Optional	Valid only if LockIn=”yes”, this tag defines a URI to be called when the “Goodbye” key is pressed during a locked XML session. This URI overrides the native behavior of the “Goodbye” key which is to destroy the current XML object displayed.
allowAnswer	Root tag	Optional	This tag applies only to the non-softkey phones. If set to “yes”, the phone will display “Ignore” and “Answer” if the XML object is displayed when the phone is in the ringing state. Default value is “no”. See section 6.3 for more details. *Only for 6863i and 6865i.*
allowDrop	Root tag	Optional	This tag applies only to the non-softkey phones. If set to “yes”, the phone will display “Drop” if the XML object is displayed when the phone is in the connected state. Default value is “no”. See section 6.4 for more details. *Only for 6863i and 6865i.*

Document Object	Position	Type	Comments
allowXfer	Root tag	Optional	This tag applies only to the non-softkey phones. If set to “yes”, the phone will display “Xfer” if the XML object is displayed when the phone is in the connected state. Default value is “no”. See section 6.4 for more details. *Only for 6863i and 6865i.*
allowConf	Root tag	Optional	This tag applies only to the non-softkey phones. If set to “yes”, the phone will display “Conf” if the XML object is displayed when the phone is in the connected state. Default value is “no”. See section 6.4 for more details. *Only for 6863i and 6865i.*
scrollConstrain	Root tag	Optional	If set to “yes”, the phone will not “wrap” the list: scrolling down the last item does not move the cursor to the first item. Default value is “no”. *Not supported on 6873i and 6940*
numberLaunch	Root tag	Optional	If set to “yes”, the phone will allow the user to “launch” an item URI using the keypad (items 1-9 only). Default value is “no”.
touchLaunch	Root tag	Optional	If set to “yes”, selection is no longer needed user can press anywhere on the item to launch the uri. Default value is “no”. *6873i and 6940 only*
fontMono	Root tag	Optional	If set to “no”, a proportional font will be used to display the object instead of a monotype font. Default value is “yes”. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*
scrollUp	Root tag	Optional	This tag allows overriding the default behavior of the Up arrow key once the scrolling reaches an end.
scrollDown	Root tag	Optional	This tag allows overriding the default behavior of the Down arrow key once the scrolling reaches an end.
Title	Body	Optional	Text to be used as title for the object
Wrap	Title tag	Optional	If set to “yes” the title of the object will be wrapped on 2 lines. *Ignored on 6867i, 6869i, 6873i, 6920, 6930 and 6940, lines are automatically*

Document Object	Position	Type	Comments
			wrapped.
Color	Title tag	Optional	Label color, the possible values are detailed in chapter 4.2.3. If not specified, the default value is “white”. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*
TopTitle	Body	Optional	Text to be used as top title for the object *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*.
Icon	TopTitle tag	Optional	Index of the icon to be used for the top title *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*.
Color	TopTitle tag	Optional	Label color, the possible values are detailed in chapter 4.2.3. If not specified, the default value is “white”. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*.
MenuItem	Body	Mandatory	Choice Item (up to 30 instances, minimum is 1 instance)
Base	MenuItem tag	Optional	The value of this attribute is pre-pended to the value in the URI tags.
icon	MenuItem tag	Optional	Index of the icon to be used for this menu entry *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*.
iconr1	MenuItem tag	Optional	Index of the icon to be used for the first right icon *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*.
iconr2	MenuItem tag	Optional	Index of the icon to be used for the second right icon *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*.
iconr3	MenuItem tag	Optional	Index of the icon to be used for the third right icon *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*.

Document Object	Position	Type	Comments
iconr4	MenuItem tag	Optional	Index of the icon to be used for the fourth right icon *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*.
Prompt	MenuItem body	Mandatory	Label of the item
Color	Prompt tag	Optional	Label color, the possible values are detailed in chapter 4.2.3. If not specified, the default value is “white”. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*.
split	Prompt tag	Optional	This attribute tells where to split the prompt on 2 lines giving the position of the first character of the second line. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*.
URI	MenuItem body	Mandatory	URI to be used if the user press “Select” with the cursor on this item
Dial	MenuItem body	Optional	Defines what number will be dialed when an offhook action is performed on the phone or if the “Dial2” custom softkey is pressed
line	Dial tag	Optional	Defines which SIP line to use when the Dial command is executed. If omitted, the Dial command is performed using the first available SIP line.
Selection	MenuItem body	Optional	This tag must be used in conjunction with custom softkeys. See Section 4.11 for details
IconList	Body	Optional	List of icon definitions *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*.
Icon	IconList body	Optional	Icon value, it can be “Icon:Iconname”, or the URL to a png file . See section 4.2.2 for more details. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only.*
Index	Icon tag	Optional	Index of the icon must be consistent with the ‘Icon’ used in the MenuItems. Possible values are 1 to 21. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*.

Document Object	Position	Type	Comments
SoftKey	Body	Optional	See section 4.1 for details

Note: the `numberLaunch` tag works for all TextMenu styles.

Limitations on non softkey phones

- Custom Softkeys are not supported
- Icons are not supported
- Only "`numbered`" is supported for the `Style` tag.

3.1.4 EXAMPLES

XML Example 1

```
<AastraIPPhoneTextMenu>
       <Title>Phone Services</Title>
       <MenuItem base = "http://10.50.10.53/">
              <Prompt>Traffic Reports</Prompt>
              <URI> rss_to_xml.pl</URI>
       </MenuItem>
       <MenuItem>
              <Prompt>Employee List</Prompt>
              <URI>employees.xml</URI>
       </MenuItem>
        <MenuItem base ="">
              <Prompt>Weather</Prompt>
              <URI>http://10.50.10.52/weather.pl</URI>
       </MenuItem>
</AastraIPPhoneTextMenu>
```

Resulting Screens (6863i/6865i)

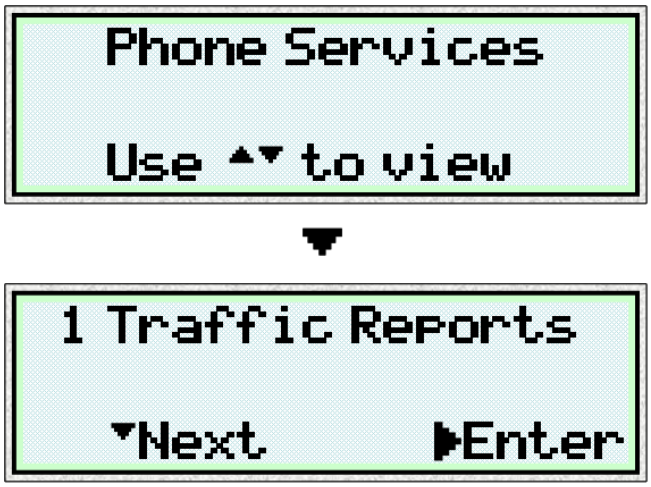


Figure 18: TextMenu Example 1 (6863i/6865i)

Resulting Screen (6869i/6930)

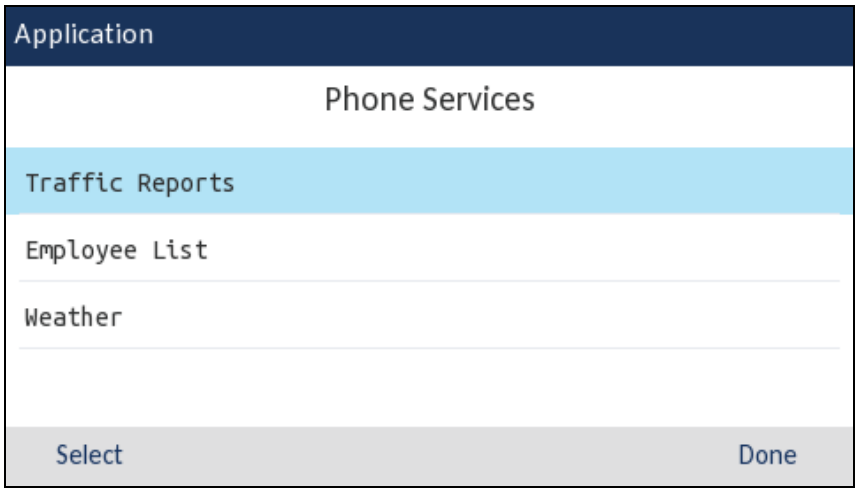


Figure 19: TextMenu Example 1 (6869i/6930)

Resulting Screen (6873i/6940)

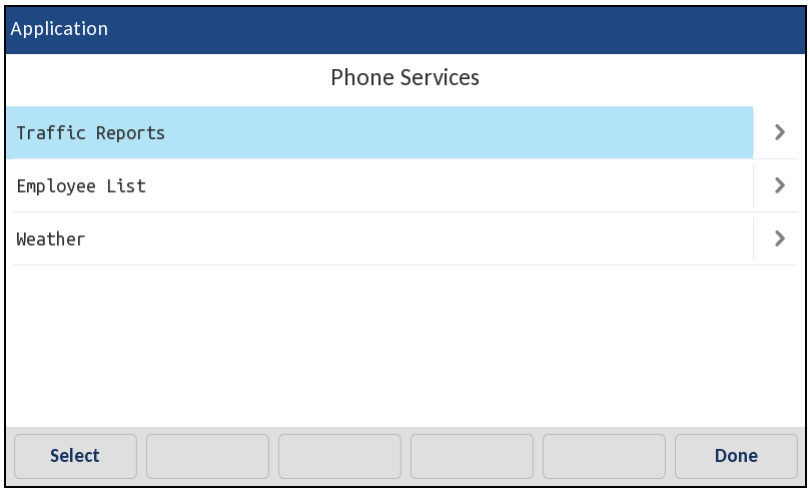


Figure 20: TextMenu Example 1 (6873i/6940)

XML Example 2 (softkey phones)

```
<AastraIPPhoneTextMenu>
       <Title>Phone Services</Title>
       <MenuItem base = "http://10.50.10.53/" icon="1">
              <Prompt>Traffic Reports</Prompt>
              <URI> rss_to_xml.pl</URI>
       </MenuItem>
       <MenuItem icon="2">
              <Prompt>Employee List</Prompt>
              <URI>employees.xml</URI>
       </MenuItem>
<MenuItem base ="" icon="3">
              <Prompt>Weather</Prompt>
              <URI>http://10.50.10.52/weather.pl</URI>
       </MenuItem>
       <IconList>
              <Icon index="1">Icon:PresenceMeeting</Icon>
              <Icon index="2">Icon:Book</Icon>
              <Icon index="3">Icon:World</Icon>
        </IconList>
</AastraIPPhoneTextMenu>
```

Resulting Screen

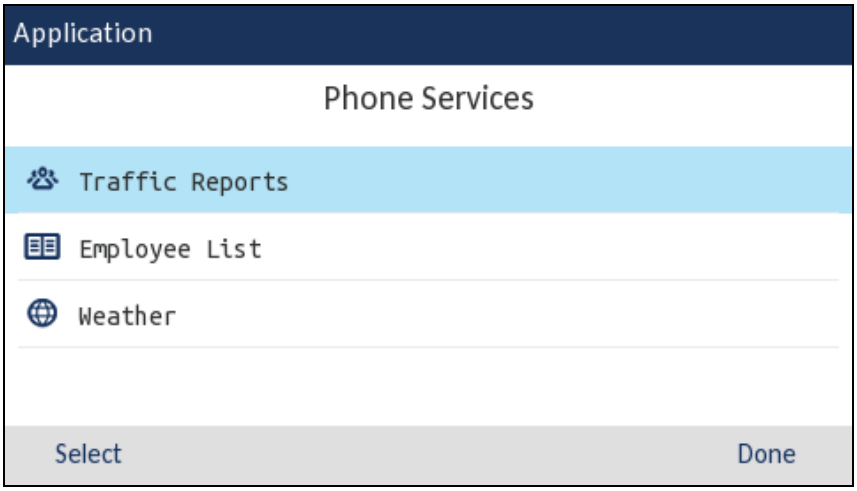


Figure 21: TextMenu Example 2 (6869i/6930)

XML Example 3 (touchscreen phone)

```
<AastraIPPhoneTextMenu fontMono="no">
       <TopTitle icon="1">John Doe</TopTitle>
       <MenuItem icon="3">
              <Prompt>Work Number</Prompt>
              <URI>http://myserver/myscript.php?a=zoom%26n=1234</URI>
              <Dial>1234</Dial>
              <Selection>1234</Selection>
       </MenuItem>
       <MenuItem icon="4">
              <Prompt>Cell Number</Prompt>
              <URI>http://myserver/myscript.php?a=zoom%26n=2345</URI>
              <Dial>2345</Dial>
              <Selection>2345</Selection>
       </MenuItem>
<MenuItem icon="2">
              <Prompt>Home Number</Prompt>
              <URI>http://myserver/myscript.php?a=zoom%26n=3456</URI>
              <Dial>3456</Dial>
              <Selection>3456</Selection>
       </MenuItem>
       <IconList>
              <Icon index="1">Picture:1234</Icon>
              <Icon index="2">Icon:Home</Icon>
              <Icon index="3">Icon:Office</Icon>
              <Icon index="4">Icon:CellPhone</Icon>
  </IconList>
<SoftKey index = "1">
       <Label>Dial</Label>
       <URI>SoftKey:Dial2</URI>
</SoftKey>
<SoftKey index = "2">
       <Label>Delete</Label>
       <URI>http://myserver/myscript.php?a=delete</URI>
</SoftKey>
<SoftKey index = "3">
       <Label>Edit</Label>
       <URI>http://myserver/myscript.php?a=edit</URI>
</SoftKey>
<SoftKey index = "6">
       <Label>Done</Label>
       <URI>SoftKey:Exit</URI>
</SoftKey>
</AastraIPPhoneTextMenu>
```

In this example, with item 1 selected:

- Pressing the  icon on the item, will call http://myserver/myscript.php?a=zoom&n=1234
- Pressing the “Delete” softkey, will call http://myserver/myscript.php?a=delete&selection=1234
- Pressing the “Edit” softkey, will call http://myserver/myscript.php?a=edit&selection=1234
- Going off will dial “1234”

Resulting Screen

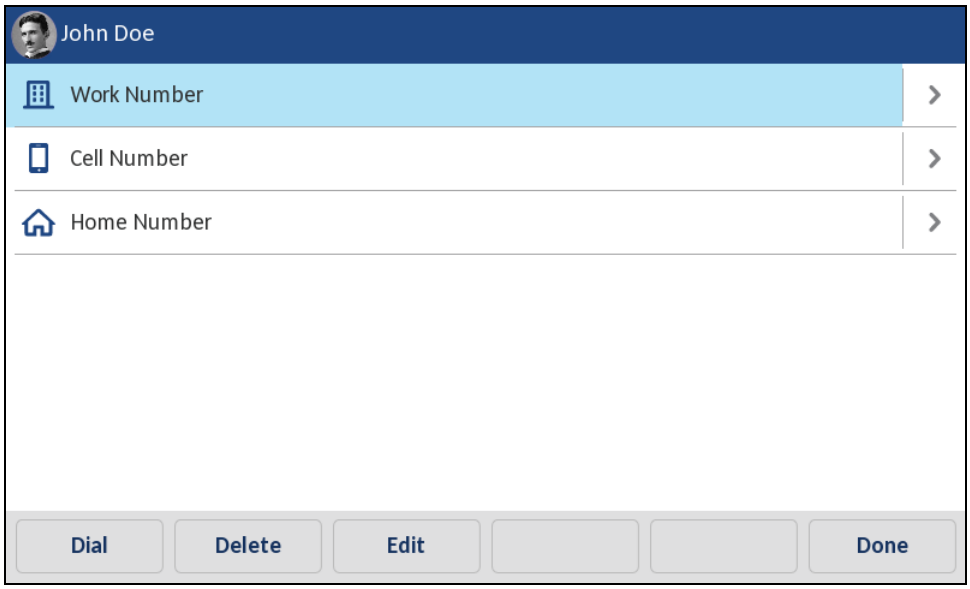


Figure 22: TextMenu Example 3 (6873i/6940)

3.2 ICONMENU OBJECT (6867I/6869I AND 6873I)

The `IconMenu` object allows developers to create a list of menu items on the IP phones just like TextMenu but in this case menus are represented by an icon or a sequence of text lines.

The IconMenu object allows users to navigate the application, by linking HTTP requests to menu items.

For more details on the IconMenu behavior using the `Selection` tag or the `Dial` tag, please refer to section 4.12 and 4.13.

This object allows to configure up to 24 menu items which are represented as cells on the screen, the user can navigate to select the cell and then launch the attached URI or action.

The number, format and size of the visible cells depends on the selected layout and screen mode. If the total number of menu items is less than the number of cells per row, the cells are centered on the screen.

Icons can be scaled to fit the placeholder size, which allows a picture to be displayed in a cell.

Layouts

Two layouts are available for this object

- Layout “1” screen is split in 2 rows with either 4 (6867i - grid of 8 portrait cells) or 6 (6869i/6873i - grid of 12 portrait cells) colums respectively.
- Layout “2” screen is split in 2 rows with 2 columns, forming a grid of 4 landscape cells.

Each cell can be composed of an icon and/or lines of text.

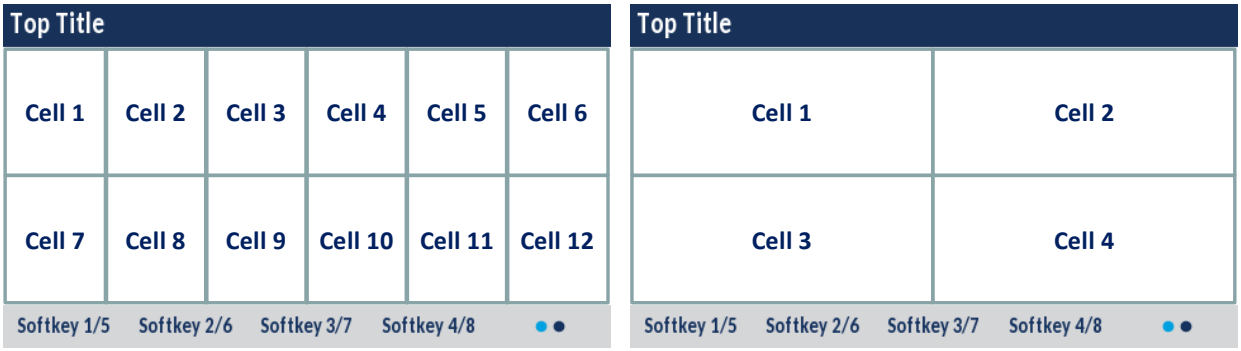


Figure 23: IconMenu cell layout

Portrait mode (layout=1)

If an icon is configured it is positioned at the top of the cell and any respective cell text is then positioned below it. However, if there is no icon configured, any respective cell text will be positioned at the top of the cell in its place.

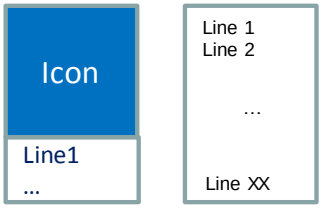


Figure 24: IconMenu cell rendering in portrait mode

Landscape mode (layout=2)

If an icon is configured it is positioned on the left hand side of the cell and any respective cell text is then positioned to the right of the icon. However, if there is no icon configured, any respective cell text will be positioned on the left hand side of the cell in its place.

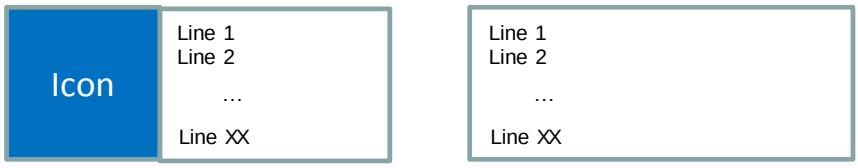


Figure 25: IconMenu cell rendering in landscape mode

Screen modes

Two screen modes are available for this object:

- Regular Mode: Full screen, where both the top bar and bottom softkeys are displayed.
- Extended Mode: Same as regular mode but without the top bar displayed.

The screen mode does not affect the number of cells in the grid just their size.

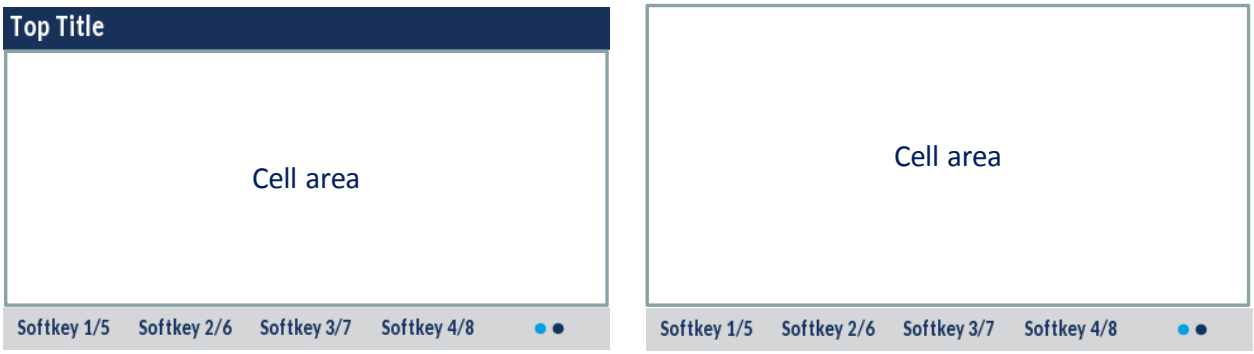


Figure 26: IconMenu screen modes "regular" and "extended"

Object native interaction

- ***Select*** Executes the content of the URI field assigned to the selected MenuItem
- ***Exit*** Redisplays the previous XML object present in the phone browser.

3.2.1 IMPLEMENTATION (6867I)

The 6867i supports the 2 grid layouts.

- Layout 1 screen has2 rows of 4 cells
- Layout 2 screen has 2 rows of 2 cells

Up to 24 cells can be created and a maximum of 8 cells per screen can be displayed at once. Any additional cells are placed underneath on the next virtual screen. The up and down scroll keys can be used to highlight/select and navigate up/down over the current screen icons or to navigate further to the next screen(s). Likewise, the left and right arrow keys can also be used to highlight/select and navigate left/right over the current screen icons.

Selected cell is highlighted.

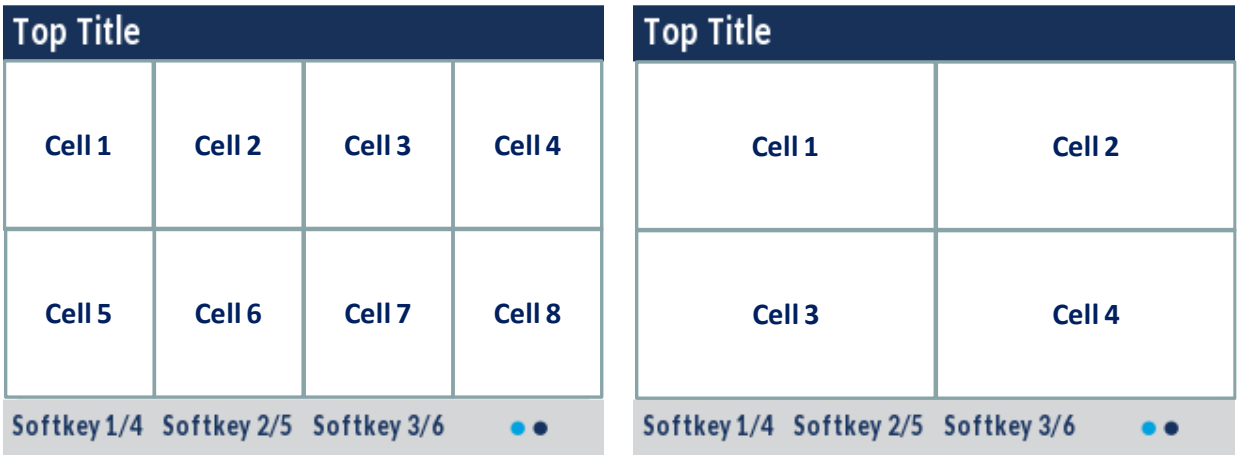


Figure 27: 6867i - Layout 1 and 2 mode "regular"

Navigation

- Up moves a row up and selects cell
- Down moves a row down and selects cell, if no cell configured below the cell selection moves to the last cell of the row
- Left moves a cell to the left and selects cell
- Right moves a cell to the right and selects cell
- OK is equivalent to select softkey

Component size

Layout	Mode	Icon size	Number of text lines
1	regular	60x60	1
1	extended	60x60	1
2	regular	66x66	3
2	extended	81x81	4

Object default Softkeys

Six customizable softkeys are available for this object.

Position	Label	Interaction	URIs
1	Select	***Select***: Executes the content of the URI field assigned to the selected MenuItem;	SoftKey:Select
4	Done	***Exit:*** Redisplays the previous XML object present in the phone browser.	SoftKey:Exit

3.2.2 IMPLEMENTATION (6869I)

The 6869i supports the 2 grid layouts.

- Layout 1 screen has 2 rows of 6 cells
- Layout 2 screen has 2 rows of 2 cells

Up to 24 cells can be created and a maximum of 12 cells per screen can be displayed at once. Any additional cells are placed underneath on the next virtual screen. The up and down scroll keys can be used to highlight/select and navigate up/down over the current screen icons or to navigate further to the next screen(s). Likewise, the left and right arrow keys can also be used to highlight/select and navigate left/right over the current screen icons.

Selected cell is highlighted.

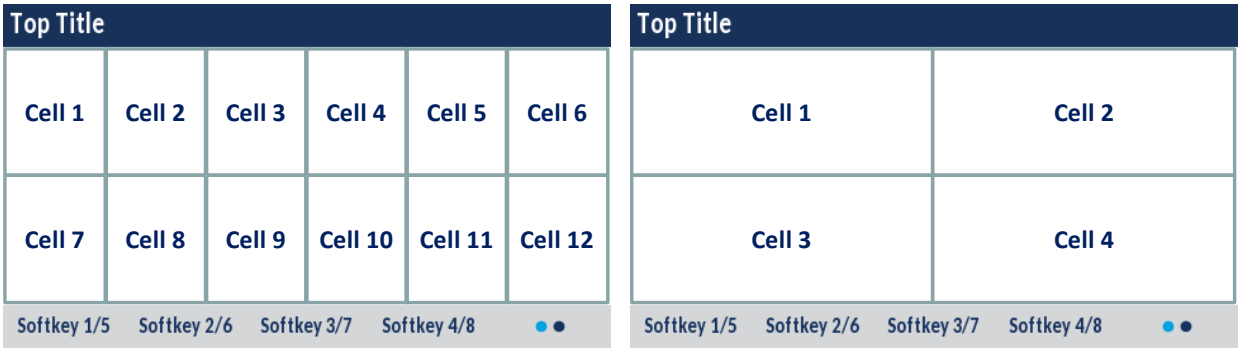


Figure 28: 6869i - Layout 1 and 2 in "regular" mode

Navigation

- Up moves a row up and selects cell
- Down moves a row down and selects cell, if no cell configured below the cell selection moves to the last cell of the row
- Left moves a cell to the left and selects cell
- Right moves a cell to the right and selects cell
- OK is equivalent to select softkey

Component size

Layout	Mode	Icon size	Number of text lines
1	regular	61x61	1
1	extended	61x61	2
2	regular	78x78	4
2	extended	95x95	5

Object default Softkeys

Eight customizable softkeys are available for this object.

Position	Label	Interaction	URIs
1	Select	***Select***: Executes the content of the URI field assigned to the selected MenuItem;	SoftKey:Select
5	Done	***Exit:*** Redisplays the previous XML object present in the phone browser.	SoftKey:Exit

3.2.3 IMPLEMENTATION (6873I)

The 6873i supports the 2 grid layouts.

- Layout 1 screen has 2 rows of 6 cells

- Layout 2 has 2 rows of 2 cells

Up to 24 cells can be created and a maximum of 12 cells per screen can be displayed at once. Any additional cells are placed on the next virtual screen. Swipe left or right to access the next screen.

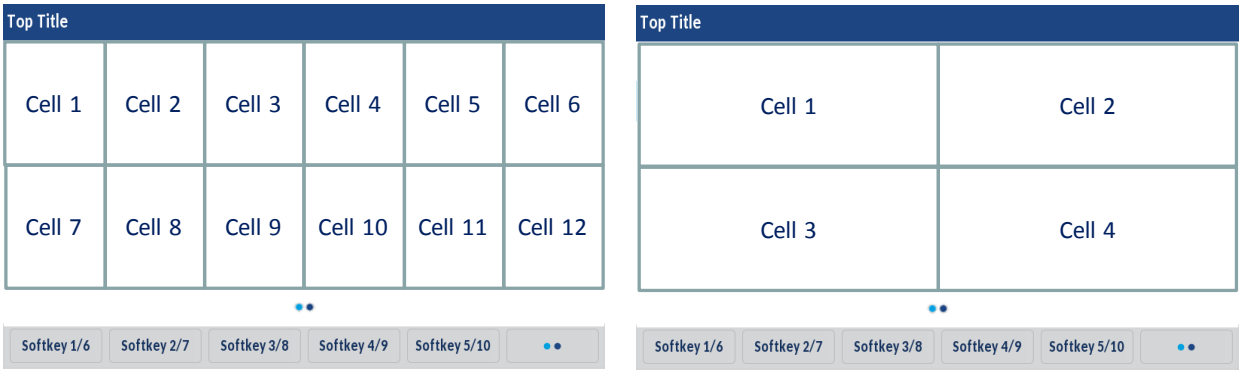


Figure 29: 6873i - Layout 1 and 2 in "regular" mode

Navigation

- Touch screen, clicking on a cell selects it and launches the associated URI
- Swipe changes page.

User cannot select and then launch the URI on the 73i, it is all in one operation, if user clicks on a cell, the URI attached to it is launched.

Custom softkeys using "Softkey:Select" or "Softkey:Dial" or "Softkey:Dial2" are not displayed on the 73i as as user cannot select a cell, it is comparable to the TextMenu with touchLaunch tag enabled.

Component size

Layout	Mode	Icon size	Number of text lines
1	regular	114x114	1
1	extended	114x114	2
2	regular	147x147	6
2	extended	171x171	7

Object default Softkeys

Ten customizable softkeys are available for this object.

Position	Label	Interaction	URIs
6	Done	***Exit:*** Redisplays the previous XML object present in the phone browser.	SoftKey:Exit

3.2.4 XML DESCRIPTION

"Red" tags indicate that the tag does not apply to all phones, when not supported the tags are just ignored.

```
<AastraIPPhoneIconMenu
    defaultIndex = "some integer"
    destroyOnExit = "yes/no"
```

```
    Beep = "yes/no"
    Timeout = "some integer"
    bgColor="white/black…"
    LockIn = "yes/no"
    CallProtection = "yes/no/notif"
    GoodbyeLockInURI = "some URI"
    fontMono = "yes/no"
    layout = "integer 1 or 2"
    mode = "regular/extended"
    >
       <TopTitle    icon="icon index"
                    Color="white/black…"
       >Top Title</TopTitle>
       <MenuItem fontMono = "yes/no">
              <iconName scaled="yes/no">IconName or location</iconName>
              <URI>http://somepage.xml</URI>
              <Dial line="SIP line">Number to dial</Dial>
              <Selection>Selection</Selection>
             <Line Align="left/center/right" Color="white/black…"
              >A line of static text</Line>
              <!—Additional Lines may be added -->
       </MenuItem>
       <!—Additional Menu Items may be added (up to 24)-->
       <IconList>
              <Icon index = "int">Icon:Iconname or icon location</Icon>
              <!—As many as different icons used in the object -->
              <!—Up to 22 icons, index can be 1 to 21 -->
       </IconList>
       <!—Additional Softkey Items may be added -->
<AastraIPPhoneIconMenu>
```

Notes:

- The number of items in a IconMenu object is limited to 24.
- You must declare at least one item in a IconMenu or the phone will generate a parsing error.
- "cancelAction" tag is not supported by this object as the left navigation key is used for screen navigation.
- It is not recommended to scale an icon up.

XML Document Objects

Document Object	Position	Type	Comments
AastraIPPhoneIconMenu	Root tag	Mandatory	Root object
defaultIndex	Root tag	Optional	Position of the selected item when the XML object is open. If not specified, the arrow is positioned on the first menu item. On the 6873i the object will open at the page containing the selected item.
destroyOnExit	Root tag	Optional	"yes/no" indicates if the object is kept or not in the phone browser after exit. If

Document Object	Position	Type	Comments
			not specified, the object is kept in the browser.
Beep	Root tag	Optional	“yes” or “no” to indicate if a notification beep must be generated by the phone.
Timeout	Root tag	Optional	Overrides the default 45 seconds timeout of the UI XML object. A Timeout set to “0” will disable the timeout feature. See section 4.10.2 for more details
bgColor	Root tag	Optional	Set the background color of the object. the possible values are detailed in chapter 4.2.3. If not specified, the default value is “white”.
LockIn	Root tag	Optional	If set to “yes”, the phone ignores all events that would cause the screen to exit without using the keys defined by the object. Default value is “no”. See section 4.10.3 for more details.
CallProtection	Root tag	Optional	If set to “yes”, the phone will not destroy the XML object being displayed on an incoming call. If set to “notif”, the behavior is the same as “yes” but a notification is displayed on the screen providing the caller ID details. *“notif” is equivalent to “yes” on 6863i and 6865i as screen notification are not supported*.
GoodbyeLockInURI	Root tag	Optional	Valid only if LockIn=”yes”, this tag defines a URI to be called when the “Goodbye” key is pressed during a locked XML session. This URI overrides the native behavior of the “Goodbye” key which is to destroy the current XML object displayed.
fontMono	Root tag	Optional	If set to “no”, a proportional font will be used to display the object instead of a monotype font. Default value is “no”.
layout	Root tag	Optional	Object layout, “1” or “2”. Default value is “1”.
mode	Root tag	Optional	Object mode “regular” or “extended”. Default value is “regular”.
TopTitle	Body	Optional	Text to be used as top title for the object.

Document Object	Position	Type	Comments
Icon	TopTitle tag	Optional	Index of the icon to be used for the top title.
Color	TopTitle tag	Optional	Label color, the possible values are detailed in chapter 4.2.3. If not specified, the default value is “white”.
MenuItem	Body	Mandatory	Choice Item (up to 24 instances, minimum is 1 instance)
fontMono	MenuItem tag	Optional	Indicates if a monotype font must be used in this menu item. If set, it overrides the value defined in the root tag.
iconName	MenuItem body	Optional	Name or location of the icon to be used for this menu entry
scaled	iconName tag	Optional	Indicates if the icon needs to be scaled to the size of the placeholder or not. Values can be “yes” or “no”. If not specified, the icon is not scaled.
URI	MenuItem body	Mandatory	URI to be used if the user press “Select” with the cursor on this item
Dial	MenuItem body	Optional	Defines what number will be dialed when an offhook action is performed on the phone or if the “Dial2” custom softkey is pressed. *Ignored on 6873i as custom softkeys based on item selection are not supported.*
line	Dial tag	Optional	Defines which SIP line to use when the Dial command is executed. If omitted, the Dial command is performed using the first available SIP line. *Ignored on 6873i as custom softkeys based on item selection are not supported.*
Line	MenuItem body	Optional	Defines the text to display
Align	Line tag	Optional	Text alignment, the possible values are “left”, “right” and “center”. If not specified, the default value is “center”.
Color	Line tag	Optional	Text color, the possible values are detailed in chapter 4.2.3. If not specified, the default value is

Document Object	Position	Type	Comments
			"white".
Selection	MenuItem body	Optional	This tag must be used in conjunction with custom softkeys. See Section 4.11 for details *Ignored on 6873i as custom softkeys based on selection not supported.*
IconList	Body	Optional	List of icon definitions
Icon	IconList body	Optional	Icon value, it can be "Icon:Iconname" or or the URL to a png file. See section 4.2.2 for more details.
Index	Icon tag	Optional	Index of the icon must be consistent with the 'Icon' used in the TopTitle Possible values are 1 to 21.
SoftKey	Body	Optional	See section 4.1 for details *Softkeys using "Softkey:Select" or "Softkey:Dial" or "Softkey:Dial2" are ignored on the 6873i*

3.2.5 EXAMPLES

XML Example 1

```
<AastraIPPhoneIconMenu
destroyOnExit="yes"
defaultIndex = "2"
layout = "1"
mode="regular"
>
<TopTitle>IconMenu</TopTitle>
<MenuItem>
<Line>Menu 1</Line>
<iconName>http://192.168.0.135/6869i-icon1-1.png</iconName>
<URI>http://myserver/myscript.php?menu=1</URI>
</MenuItem>
<MenuItem>
<Line>Menu 2</Line>
<iconName>http://192.168.0.135/6869i-icon2-1.png</iconName>
<URI>http://myserver/myscript.php?menu=2</URI>
</MenuItem>
<MenuItem>
<Line>Menu 3</Line>
<iconName>http://192.168.0.135/6869i-icon3-1.png</iconName>
<URI>http://myserver/myscript.php?menu=3</URI>
</MenuItem>
</AastraIPPhoneIconMenu>
```

Resulting Screen (6869i)

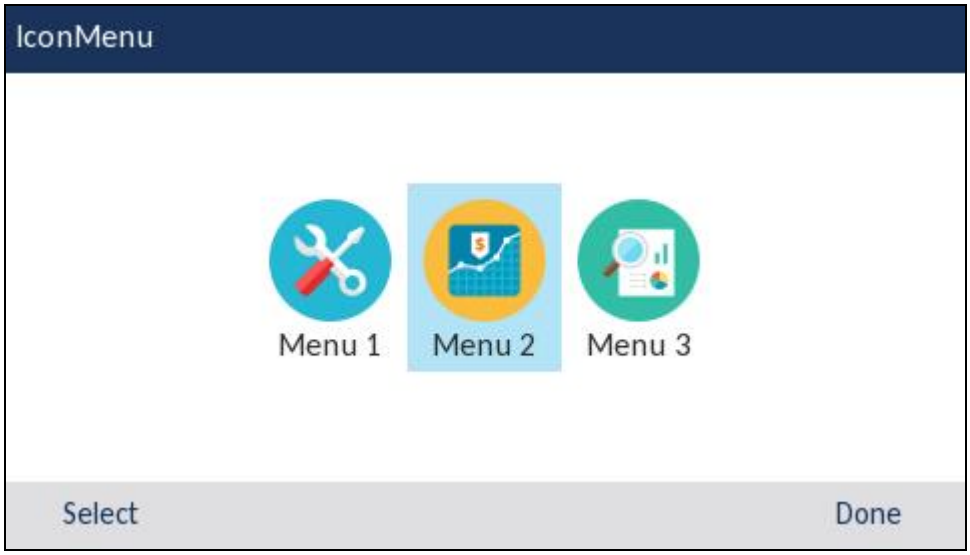


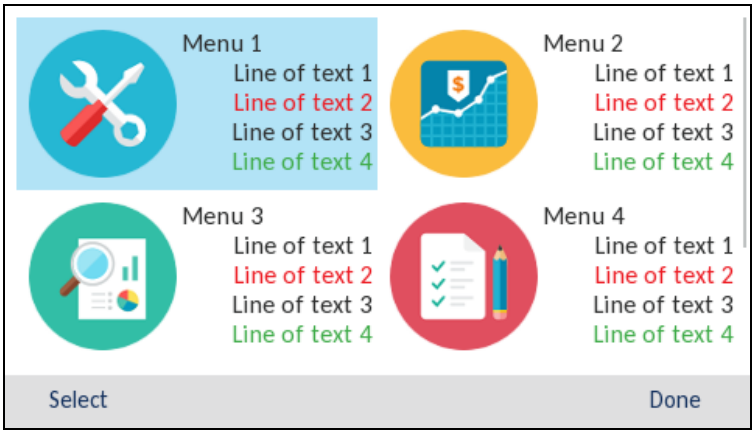
Figure 30: IconMenu Example 1 (6869i)

XML Example 2

```
<AastraIPPhoneIconMenu
destroyOnExit="yes"
defaultIndex = "1"
layout = "2"
mode="extended"
>
<TopTitle>IconMenu</TopTitle>
<MenuItem>
<Line Align="left">Menu 1</Line>
<Line Align="right">Line of text 1</Line>
<Line Align="right" Color="red">Line of text 2</Line>
<Line Align="right">Line of text 3</Line>
<Line Align="right" Color="green">Line of text 4</Line>
<Line Align="right">Line of text 5</Line>
<Line Align="right">Line of text 6</Line>
<Line Align="right">Line of text 7</Line>
<iconName>http://192.168.0.135/6869i-icon1-2.png</iconName>
<URI>http://myserver/myscript.php?menu=1</URI>
</MenuItem>
<MenuItem>
<Line Align="left">Menu 2</Line>
<Line Align="right">Line of text 1</Line>
<Line Align="right" Color="red">Line of text 2</Line>
<Line Align="right">Line of text 3</Line>
<Line Align="right" Color="green">Line of text 4</Line>
<Line Align="right">Line of text 5</Line>
<Line Align="right">Line of text 6</Line>
<Line Align="right">Line of text 7</Line>
<iconName>http://192.168.0.135/6869i-icon2-2.png</iconName>
<URI>http://myserver/myscript.php?menu=2</URI>
</MenuItem>
<MenuItem>
<Line Align="left">Menu 3</Line>
<Line Align="right">Line of text 1</Line>
<Line Align="right" Color="red">Line of text 2</Line>
<Line Align="right">Line of text 3</Line>
<Line Align="right" Color="green">Line of text 4</Line>
<Line Align="right">Line of text 5</Line>
<Line Align="right">Line of text 6</Line>
<Line Align="right">Line of text 7</Line>
<iconName>http://192.168.0.135/6869i-icon3-2.png</iconName>
<URI>http://myserver/myscript.php?menu=3</URI>
</MenuItem>
<MenuItem>
<Line Align="left">Menu 4</Line>
<Line Align="right">Line of text 1</Line>
<Line Align="right" Color="red">Line of text 2</Line>
<Line Align="right">Line of text 3</Line>
<Line Align="right" Color="green">Line of text 4</Line>
<Line Align="right">Line of text 5</Line>
<Line Align="right">Line of text 6</Line>
<Line Align="right">Line of text 7</Line>
<iconName>http://192.168.0.135/6869i-icon4-2.png</iconName>
<URI>http://myserver/myscript.php?menu=4</URI>
</MenuItem>
<MenuItem>
<Line Align="left">Menu 5</Line>
<Line Align="right">Line of text 1</Line>
<Line Align="right" Color="red">Line of text 2</Line>
<Line Align="right">Line of text 3</Line>
<Line Align="right" Color="green">Line of text 4</Line>
```

```
<Line Align="right">Line of text 5</Line>
<Line Align="right">Line of text 6</Line>
<Line Align="right">Line of text 7</Line>
<iconName>http://192.168.0.135/6869i-icon5-2.png</iconName>
<URI>http://myserver/myscript.php?menu=5</URI>
</MenuItem>
<MenuItem>
<Line Align="left">Menu 6</Line>
<Line Align="right">Line of text 1</Line>
<Line Align="right" Color="red">Line of text 2</Line>
<Line Align="right">Line of text 3</Line>
<Line Align="right" Color="green">Line of text 4</Line>
<Line Align="right">Line of text 5</Line>
<Line Align="right">Line of text 6</Line>
<Line Align="right">Line of text 7</Line>
<iconName>http://192.168.0.135/6869i-icon6-2.png</iconName>
<URI>http://myserver/myscript.php?menu=6</URI>
</MenuItem>
</AastraIPPhoneIconMenu>
```

Resulting Screens (6869i)



Scroll down twice to display the rest of the items

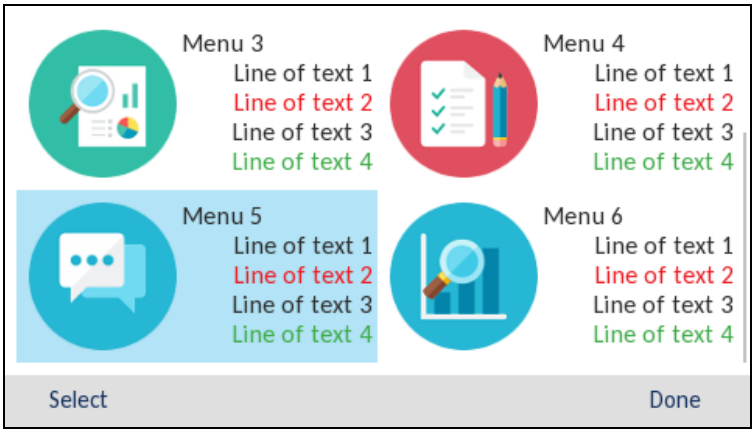
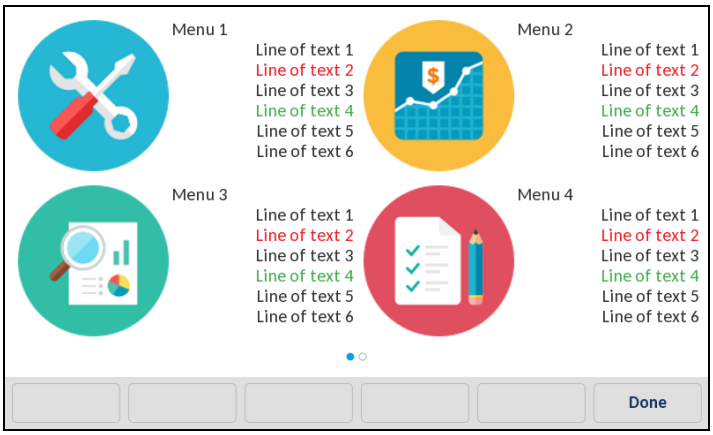


Figure 31: IconMenu Example 2 (6869i)

Resulting Screens (6873i)



Swipe to the left to reach the second page

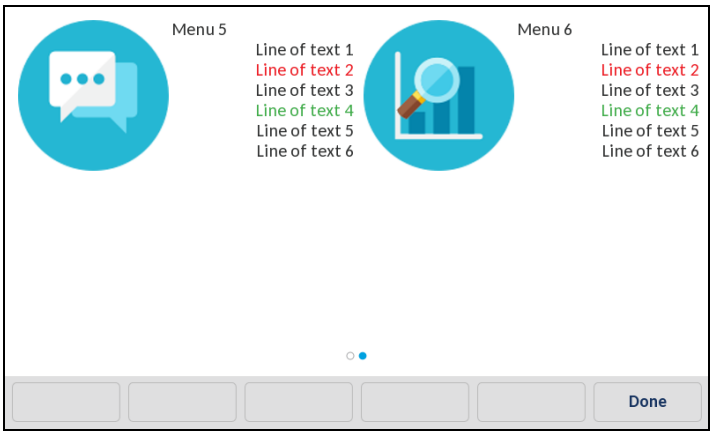


Figure 32: IconMenu Example 2 (6873i)

3.3 IMAGEMENU OBJECT (6867I/6869I/6873I/6920/6930/6940)

The `ImageMenu` object allows using a bitmap image to serve as a menu. This is desirable when a user wants to display menu choices in some non-ASCII character set or with pictures only. Each menu selection is linked to a keypad key (0-9, *, #).allows developers to create a numerical list of choices.

3.3.1 IMPLEMENTATION (6867I/6920)

For the 6867i, the image is either a 24/32 bit depth “png” or a “jpeg” file located on a server and which can be downloaded by the phone using TFTP, FTP, HTTP or HTTPS. See chapter 4.2.1.1 for more details.

Two image sizes are supported:

- Up to 320x180 pixels (mode=regular or extended).
- Up to 320x240 pixels (mode=fullscreen). In that case, the softkeys are not displayed.

If the image is bigger than the size supported by the requested mode, the image is clipped based on the requested alignment.

If the image is smaller than the size supported by the requested mode, the image is displayed based on the requested alignment

The ***Left*** key is mapped to the cancelAction tag, if not specified the XML object is destroyed.

Note:

- the Left Arrow key default interaction is disabled if the `LockIn` tag is set to “yes”.
- If the `LockIn` tag is set to “yes” and the `cancelAction` tag is configured, pressing the Left Arrow key triggers the configured `cancelAction`.
- The up, down, left and right arrow key default interactions can be overridden using the `scrollUp`, `scrollDown`, `scrollLeft` and `scrollRight` tags.

Object default Softkeys

Six (four physical) customizable softkeys are available for this object.

Position	Label	Interaction	URIs
4	Done	***Exit:*** Redisplays the previous XML object present in the phone browser.	SoftKey:Exit

The following figure details how the `AastraIPPhoneImageMenu` is implemented on the Mitel 6867i.





Figure 33: ImageMenu 6867i/6920 implementation (mode regular/extended and full screen)

Also a URI can be programmed to be called when the user presses the ***Select*** key (imageAction root tag). If this tag is empty, phone will use the URI configured by the doneAction tag (SoftKey:Exit by default).

So by default pressing the ***Select*** key destroys the current XML object unless an imageAction or a doneAction is configured.

Native Interaction

- "SoftKey:Exit"

3.3.2 IMPLEMENTATION (6869I/6930)

For the 6869i, the image is either a 24/32 bit depth "png" or a "jpeg" file located on a server and which can be downloaded by the phone using TFTP, FTP, HTTP or HTTPS. See chapter 4.2.1.1 for more details.

Two image sizes are supported:

- Up to 480x204 pixels (mode=regular or extended).
- Up to 480x272 pixels (mode=fullscreen). In that case, the softkeys are not displayed.

If the image is bigger than the size supported by the requested mode, the image is clipped based on the requested alignment.

If the image is smaller than the size supported by the requested mode, the image is displayed based on the requested alignment

The ***Left*** key is mapped to the cancelAction tag, if not specified the XML object is destroyed.

Note:

- the Left Arrow key default interaction is disabled if the `LockIn` tag is set to "yes".
- If the `LockIn` tag is set to "yes" and the `cancelAction` tag is configured, pressing the Left Arrow key triggers the configured `cancelAction`.
- The up, down, left and right arrow key default interactions can be overridden using the `scrollUp`, `scrollDown`, `scrollLeft` and `scrollRight` tags.

Object default Softkeys

Eight (five physical) customizable softkeys are available for this object.

Position	Label	Interaction	URIs
5	Done	***Exit:*** Redisplays the previous XML object present in the phone browser.	SoftKey:Exit

The following figure details how the `AastraIPPhoneImageMenu` is implemented on the Mitel 6869i.

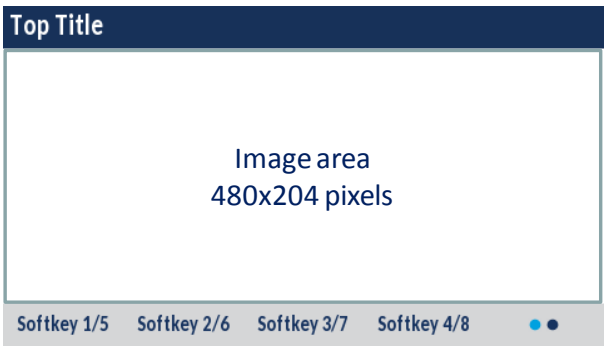




Figure 34: ImageMenu 6869i/6930 implementation (mode regular/extended and full screen)

Also a URI can be programmed to be called when the user presses the ***Select*** key (imageAction root tag). If this tag is empty, phone will use the URI configured by the doneAction tag (SoftKey:Exit by default).

So by default pressing the ***Select*** key destroys the current XML object unless an imageAction or a doneAction is configured.

Native Interaction

- "SoftKey:Exit"

3.3.3 IMPLEMENTATION (6873I/6940)

For the 6873i, the image is either a 24/32 bit depth "png" or a "jpeg" file located on a server and which can be downloaded by the phone using TFTP, FTP, HTTP or HTTPS. See chapter 4.2.1.1 for more details.

Two image sizes are supported:

- Up to 800x372 pixels (mode=regular or extended).
- Up to 800x480 pixels (mode=fullscreen). In that case, the softkeys are not displayed.

If the image is bigger than the size supported by the requested mode, the image is clipped following the requested alignment.

If the image is smaller than the size supported by the requested mode, the image is displayed following the requested alignment

Note: as the 6873i and 6940 do not have navigation keys, the scroll actions are mapped to screen swipes

- `scrollUp`, by swiping from top to bottom
- `scrollDown`, by swiping from bottom to top
- `scrollLeft`, by swiping from left to right
- `scrollRight`, by swiping from right to left

Object default Softkeys

Ten (six physical) customizable softkeys are available for this object.

Position	Label	Interaction	URIs
6	Done	***Exit:*** Redisplays the previous XML object present in the phone browser.	SoftKey:Exit

The following figure details how the `AastraIPPhoneImageMenu` is implemented on the Mitel 6873i.

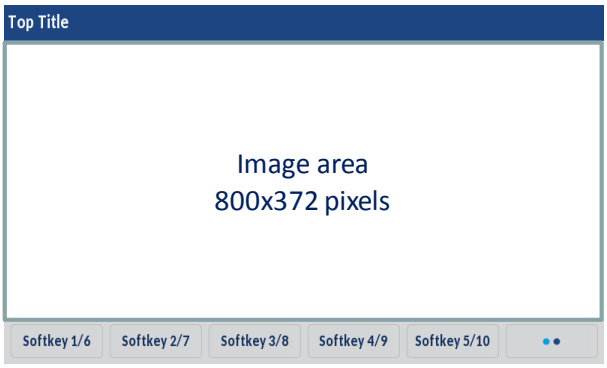




Figure 35: ImageMenu 6873i/6940 implementation (mode regular/extended and full screen)

Also a URI can be programmed to be called when the user presses on the displayed image (imageAction root tag). If this tag is empty, phone will use the URI configured by the doneAction tag (SoftKey:Exit by default).

So, by default, pressing on the image destroys the current XML object unless an imageAction or a doneAction is configured.

Native Interaction

- "SoftKey:Exit"

3.3.4 XML DESCRIPTION

```
<AastraIPPhoneImageMenu
    destroyOnExit = "yes/no"
    cancelAction = "some URI"
    doneAction = "some URI"
    imageAction = "some URI"
    Beep = "yes/no"
    Timeout = "some integer"
    bgColor="white/black…"
    LockIn = "yes/no"
    CallProtection = "yes/no/notif"
    GoodbyeLockInURI = "some URI"
    allowDTMF = "yes/no"
    scrollUp = "some URI"
    scrollDown = "some URI"
    scrollLeft = "some URI"
    scrollRight = "some URI"
    mode = "regular/extended/fullscreen"
    >
       <TopTitle     icon="icon index"
                     Color="white/black…"
       >Top Title</TopTitle>
       <Image
             verticalAlign = "top,middle,bottom"
             horizontalAlign = "left,middle,right"
             height = "height in pixels"
             width = "width in pixels"
       >Image as hexadecimal characters or URL</Image>
       <!—Base attribute is optional-->
       <URIList base = "http://someserver/">
             <URI key = "0">link1.php</URI>
             <URI key = "#">link3.php</URI>
             <!—Additional URI entries may be added (0-9,* and #)-->
       </URIList>
<!—Additional Softkey Items may be added -->
</AastraIPPhoneImageMenu>
```

XML Document Objects

Document Object	Position	Type	Comments
AastraIPPhoneImageMenu	Root tag	Mandatory	Root object
destroyOnExit	Root tag	Optional	"yes/no" indicates if the object is kept or not in the phone browser after exit. If not specified, the object is kept in the browser.
cancelAction	Root tag	Optional	Defines the URI to be called when the user cancels the XML object.

Document Object	Position	Type	Comments
doneAction	Root tag	Optional	Defines the URI to be called when the user selects the “Done” softkey or ‘SoftKey:Exit’.
imageAction	Root tag	Optional	Defines the URI to be called when the user presses on the image. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only.*
Beep	Root tag	Optional	“yes” or “no” to indicate if a notification beep must be generated by the phone.
Timeout	Root tag	Optional	Overrides the default 45 seconds timeout of the UI XML object. A Timeout set to “0” will disable the timeout feature. See section 4.10.1 for more details
bgColor	Root tag	Optional	Set the background color of the object. the possible values are detailed in chapter 4.2.3. If not specified, the default value is “white”.
LockIn	Root tag	Optional	If set to “yes”, the phone ignores all events that would cause the screen to exit without using the keys defined by the object. Default value is “no”. See section 4.10.3 for more details.
CallProtection	Root tag	Optional	If set to “yes”, the phone will not destroy the XML object being displayed on an incoming call. If set to “notif”, the behavior is the same as “yes” but a notification is displayed on the screen providing the caller ID details. *“notif” is equivalent to “yes” on 6863i and 6865i as screen notification are not supported*.
GoodbyeLockInURI	Root tag	Optional	Valid only if LockIn=”yes”, this tag defines a URI to be called when the “Goodbye” key is pressed during a locked XML session. This URI overrides the native behavior of the “Goodbye” key which is to destroy the current XML object displayed.
allowDTMF	Root tag	Optional	This tag allows letting keypad strokes as DTMF when the phone is in the connected status.
scrollUp	Root tag	Optional	This tag allows overriding the default behavior of the Up arrow key.
scrollDown	Root tag	Optional	This tag allows overriding the default behavior of the Down arrow key.
scrollLeft	Root tag	Optional	This tag allows overriding the default

Document Object	Position	Type	Comments
			behavior of the Left arrow key.
scrollRight	Root tag	Optional	This tag allows overriding the default behavior of the Right arrow key. *Ignored on 6739i*
mode	Root tag	Optional	Configures the display mode, “regular”, “extended” or “fullscreen” *6867i, 6869i, 6873i and 6739i only*
TopTitle	Body	Optional	Text to be used as top title for the object *6867i, 6869i, 6873i and 6739i only*.
Icon	TopTitle tag	Optional	Index of the icon to be used for the top title *6867i, 6869i, 6873i and 6739i only*.
Color	TopTitle tag	Optional	Label color, the possible values are detailed in chapter 4.2.3. If not specified, the default value is “white”. *6867i, 6869i, 6873i and 6739i only*
Image	Body	Mandatory	Image to be displayed as a series of hexadecimal characters or a URL. See section 4.2.1 for more details.
verticalAlign	Image tag	Optional	Vertical position of the image (“top”, “middle” or “bottom”). If the tag is not specified, the object will use “middle” as a default value.
horizontalAlign	Image tag	Optional	Horizontal position of the image (“left”, “middle” or “right”). If the tag is not specified, the object will use “middle” as a default value.
height	Image tag	Mandatory	Height in pixels. Must match the image height. *Optional and Ignored on 6867i, 6869i and 6873i*
width	Image tag	Mandatory	Width in pixels. Must match the image width. *Optional and Ignored on 6867i, 6869i and 6873i*
URIList	Body	Mandatory	Master tag of the URI list linked to a keypad key (0-9, * and #)
Base	URIList tag	Optional	The value of this attribute is pre-pended to the value in the URI tags.
URI	URIList body	Mandatory	URI to be used if the user presses the key defined in the “key” tag.
key	URI tag	Mandatory	This tag defines the key that will trigger the selection (0-9,* and #).
SoftKey	Body	Optional	See section 4.1 for details

3.3.5 EXAMPLES

XML Example (6869i/6930)

```
<AastraIPPhoneImageMenu destroyOnExit="yes">
<Image>tftp://server.com/menu.png</Image>
<URIList>
<URI key="1">http://myserver.com?choice=1</URI>
<URI key="2">http://myserver.com?Choice=2</URI>
</URIList>
<SoftKey index="1">
<Label>Label</Label>
<URI>http://myserver.com/script.php?action=1</URI>
</SoftKey>
<SoftKey index="5">
<Label>Exit</Label>
<URI>SoftKey:Exit</URI>
</SoftKey>
</AastraIPPhoneImageMenu>
```

Resulting Screen



Figure 36: ImageMenu Example (6869i/6930i)

3.4 TEXTSCREEN OBJECT (ALL MODELS)

The `TextScreen` object can be used to display some text. The screen word-wraps appropriately and user can scroll to display a message longer then the physical display.

3.4.1 IMPLEMENTATION (6863I/6865I)

The object is displayed on 2 lines. The Up and Down arrow keys allow the user to browse the rest of the text.

Line selected	Label	Keys	
Title/Text	`vNext`	Up and Down Arrow	Browse up or down
	`>Done`	Right Arrow	Done
		Left Arrow	Exit

.

Note:

- the Left Arrow key default interaction is disabled if the `LockIn` tag is set to “yes”.
- the Left Arrow key default interaction can be modified using the `cancelAction` tag on a non-softkey phone.
- If the `LockIn` tag is set to “yes” and the `cancelAction` tag is configured, pressing the Left Arrow key triggers the configured `cancelAction`.
- the Right arrow key default interaction can be modified using the `doneAction` tag on a non-softkey phone.
- The up, down, left and right arrow key default interactions can be overridden using the `scrollUp`, `scrollDown`, `scrollLeft` and `scrollRight` tags.

Object native interaction

- ***Done*** Redisplays the previous XML object present in the phone browser.

3.4.2 IMPLEMENTATION (6867I/6869I)

The title is displayed at the top of the XML area and uses up to 2 lines. The text is displayed after the title area and up to 7 lines can be displayed (9 lines if no title).

Note: The Text and Title area use the same font (size and type) but the Title is displayed in bold.

The ***Left*** key is mapped to the cancelAction tag, if not specified the XML object is destroyed.

The ***Up/Down*** keys allow up and down scrolling in order to display the complete text.

The ***Right*** and ***Select*** keys are mapped to the doneAction tag, if not specified the previous XML object in the stack is displayed.

Six (eight) customizable softkeys are available for this object. If more than 4 (5) softkeys are configured the softkeys are displayed on 2 pages.

Note: the ***Left*** key interaction is disabled if the `LockIn` tag is set to “yes”.

Object default Softkeys (6867i/6920)

Six customizable softkeys are available for this object.

Position	Label	Interaction	URIs
4	Done	***Exit:*** Redisplays the previous XML object present in the phone browser.	SoftKey:Exit

Object default Softkeys (6869i/6930)

Eight customizable softkeys are available for this object.

Position	Label	Interaction	URIs
5	Done	***Exit:*** Redisplays the previous XML object present in the phone browser.	SoftKey:Exit

The following figures detail how the `AastraIPPhoneTextScreen` is implemented on the Mitel 6867i. The size of the text zone depends on the presence of the optional title.

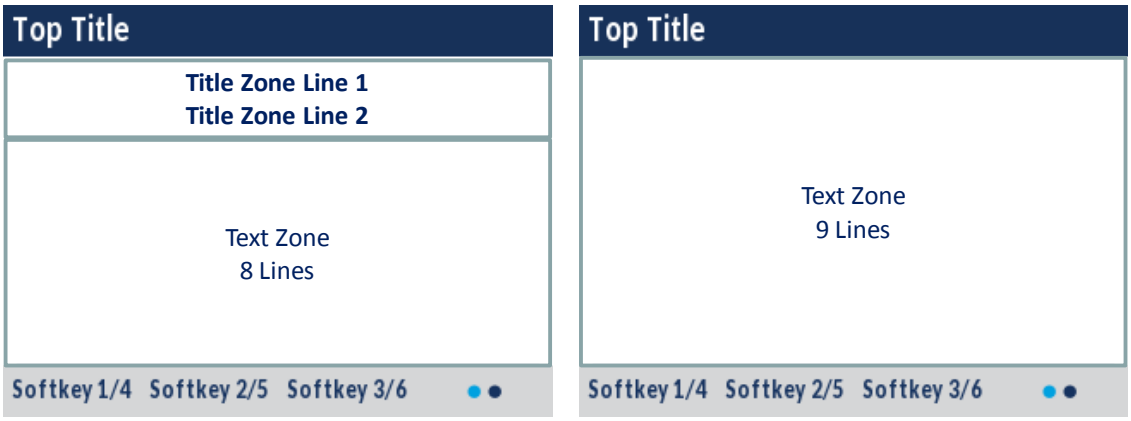


Figure 37: TextScreen implementation on 6867i/6920

Native Interaction

- "SoftKey:Exit"

3.4.3 IMPLEMENTATION (6873I/6940)

The title is displayed at the top of the XML area and uses up to 2 lines. The text is displayed after the title area and up to 13 lines can be displayed (15 lines if no title).

Note: The Text and Title area use the same font (size and type) but the Title is displayed in bold.

As the 6873i and the 6940 have no navigation keys, scrolling the text is done by swiping the screen up and down.

Ten customizable softkeys are available for this object. If more than 6 softkeys are configured the softkeys are displayed on 2 pages.

Object default Softkeys

Ten customizable softkeys are available for this object.

Position	Label	Interaction	URIs
6	Done	***Exit:*** Redisplays the previous XML object present in the phone browser.	SoftKey:Exit

The following figures detail how the `AastraIPPhoneTextScreen` is implemented on the Mitel 6873i and Mitel 6940. The size of the text zone depends on the presence of the optional title.



Figure 38: TextScreen implementation on 6873i/6940

Native Interaction

- "SoftKey:Exit"

3.4.4 XML DESCRIPTION

"Red" tags indicate that the tag does not apply to all phones, when not supported the tags are just ignored.

```
<AastraIPPhoneTextScreen
    destroyOnExit = "yes/no"
    cancelAction = "some URI"
    doneAction = "some URI"
    Beep = "yes/no"
    Timeout = "some integer"
    bgColor="white/black…"
    allowAnswer = "yes/no"
    allowDrop = "yes/no"
    allowXfer = "yes/no"
    allowConf = "yes/no"
    LockIn = "yes/no"
    CallProtection = "yes/no/notif"
    GoodbyeLockInURI = "some URI"
    allowDTMF = "yes/no"
    scrollUp = "some URI"
    scrollDown = "some URI"
    scrollLeft = "some URI"
    scrollRight = "some URI"
    >
       <Title       wrap="yes/no"
                    Color="white/black…"
       >Screen Title</Title>
       <TopTitle    icon="icon index"
                    Color="white/black…"
       >Top Title</TopTitle>
       <Text       Color="white/black…"
       >The screen text goes here</Text>
      <Dial        line="SIP line"
      >Number to dial</Dial>
```

```
<!—Additional Softkey Items may be added-->
</AastraIPPhoneTextScreen>
```

XML Document Objects

Document Object	Position	Type	Comments
AastraIPPhoneTextScreen	Root tag	Mandatory	Root object
destroyOnExit	Root tag	Optional	“yes/no” indicates if the object is kept or not in the phone browser after exit. If not specified, the object is kept in the browser.
cancelAction	Root tag	Optional	Defines the URI to be called when the user cancels the XML object.
doneAction	Root tag	Optional	Defines the URI to be called when the user selects the “Done” softkey or ‘SoftKey:Exit’.
Beep	Root tag	Optional	“yes” or “no” to indicate if a notification beep must be generated by the phone.
Timeout	Root tag	Optional	Overrides the default 45 seconds timeout of the UI XML object. A Timeout set to “0” will disable the timeout feature. See section 4.10.2 for more details
bgColor	Root tag	Optional	Set the background color of the object. the possible values are detailed in chapter 4.2.3. If not specified, the default value is “white”. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*
LockIn	Root tag	Optional	If set to “yes”, the phone ignores all events that would cause the screen to exit without using the keys defined by the object. Default value is “no”. See section 4.10.3 for more details.
CallProtection	Root tag	Optional	If set to “yes”, the phone will not destroy the XML object being displayed on an incoming call. If set to “notif”, the behavior is the same as “yes” but a notification is displayed on the screen providing the caller ID details. *“notif” is equivalent to “yes” on 6863i and 6865i as screen notification are not supported*.
GoodbyeLockInURI	Root tag	Optional	Valid only if LockIn=”yes”, this tag defines a URI to be called when the “Goodbye” key is pressed during a locked XML session. This URI overrides the native behavior of the “Goodbye” key which is to destroy the current XML object displayed.
allowAnswer	Root tag	Optional	This tag applies only to the non-softkey

Document Object	Position	Type	Comments
			phones. If set to “yes”, the phone will display “Ignore” and “Answer” if the XML object is displayed when the phone is in the ringing state. Default value is “no”. See section 6.3 for more details. *Only for 6863i and 6865i*
allowDrop	Root tag	Optional	This tag applies only to the non-softkey phones. If set to “yes”, the phone will display “Drop” if the XML object is displayed when the phone is in the connected state. Default value is “no”. See section 6.4 for more details. *Only for 6863i and 6865i*
allowXfer	Root tag	Optional	This tag applies only to the non-softkey phones. If set to “yes”, the phone will display “Xfer” if the XML object is displayed when the phone is in the connected state. Default value is “no”. See section 6.4 for more details. *Only for 6863i and 6865i*
allowConf	Root tag	Optional	This tag applies only to the non-softkey phones. If set to “yes”, the phone will display “Conf” if the XML object is displayed when the phone is in the connected state. Default value is “no”. See section 6.4 for more details. *Only for 6863i and 6865i*
allowDTMF	Root tag	Optional	This tag allows letting keypad strokes as DTMF when the phone is in the connected status.
scrollUp	Root tag	Optional	This tag allows overriding the default behavior of the Up arrow key once the scrolling reaches an end. *Not available on 6873i and 6940.*
scrollDown	Root tag	Optional	This tag allows overriding the default behavior of the Down arrow key once the scrolling reaches an end. *Not available on 6873i and 6940.*
scrollLeft	Root tag	Optional	This tag allows overriding the default behavior of the Left arrow key once the scrolling reaches an end. *Not available on 6873i and 6940.*
scrollRight	Root tag	Optional	This tag allows overriding the default behavior of the Right arrow key once the scrolling reaches an end. *Not available on 6873i and 6940.*
Title	Body	Optional	Label to be used as title for the object
Wrap	Title tag	Optional	If set to “yes” the title of the object will be wrapped on 2 lines. *Only for 6863i and 6865i, lines are*

Document Object	Position	Type	Comments
			automatically wrapped on other phone models.
Color	Title tag	Optional	Label color, the possible values are detailed in chapter 4.2.3. If not specified, the default value is “white”. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*
TopTitle	Body	Optional	Text to be used as top title for the object *6867i, 6869i, 6873i, 6920, 6930 and 6940i only.*
Icon	TopTitle tag	Optional	Index of the icon to be used for the top title *6867i, 6869i, 6873i, 6920, 6930 and 6940 only.*
Color	TopTitle tag	Optional	Label color, the possible values are detailed in chapter 4.2.3. If not specified, the default value is “white”. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*
Text	Body	Mandatory	Text to be displayed.
Color	Text tag	Optional	Text color, the possible values are detailed in chapter 4.2.3. If not specified, the default value is “darkgray”. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*
Dial	Body	Optional	Defines what number will be dialed when an offhook action is performed on the phone or if the “Dial2” custom softkey is pressed
line	Dial tag	Optional	Defines which SIP line to use when the Dial command is executed. If omitted, the Dial command is performed using the first available SIP line.

Note: the URI configured via the `scrollUp` and `scrollDown` tags is triggered only when the scrolling reaches an end. If no scrolling is needed, the URI are immediately triggered otherwise, they are triggered after the scroll reaches its beginning or its end.

3.4.5 EXAMPLES

XML Example 1

```
<AastraIPPhoneTextScreen>
      <Title>Screen Object</Title>
      <Text>The screen object can be implemented similar to the firmware info
screen. Note that white space is preserved in XML so the display should word-
wrap appropriately. Only three lines can display at a time.</Text>
</AastraIPPhoneTextScreen>
```

Resulting Screens (6863i/6865i)

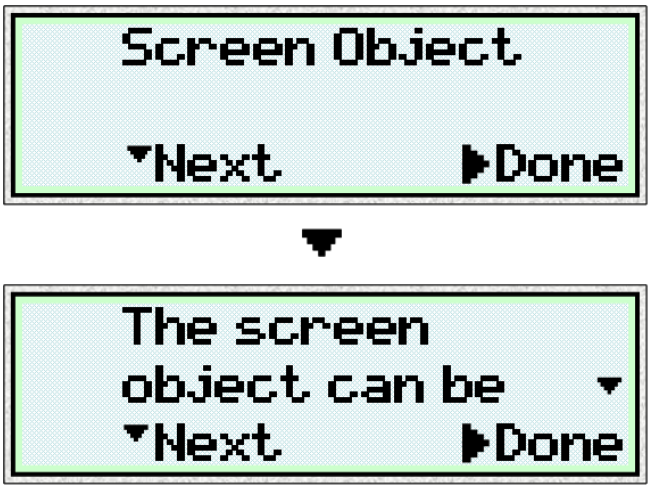


Figure 39: TextScreen Example (6863i/6865i)

Resulting Screen (6869i/6930)

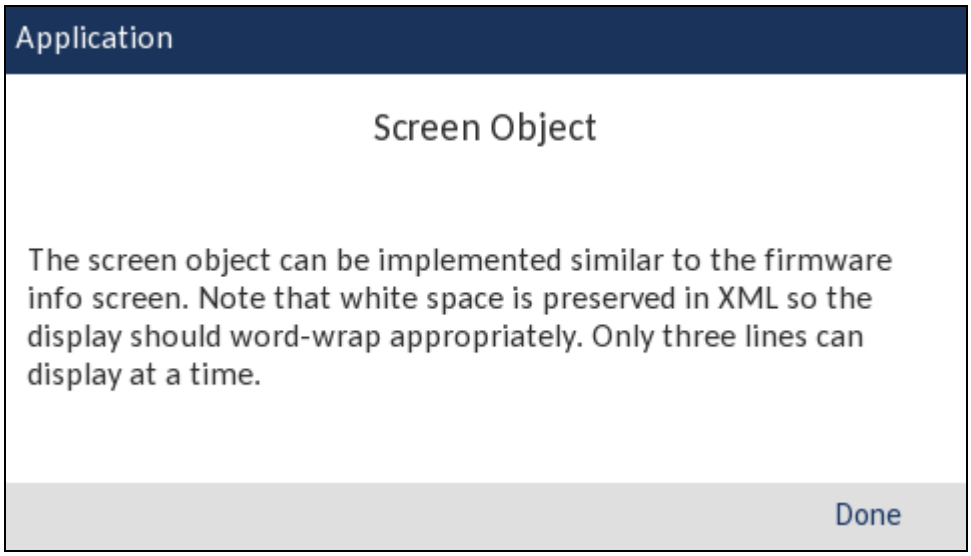


Figure 40: TextScreen Example (6869i/6930)

XML Example 2

```
<AastraIPPhoneTextScreen>
      <Title>This is a very long title which should be on two lines</Title>
      <Text>Lorem Ipsum is simply dummy text of the printing and typesetting
industry. Lorem Ipsum has been the industry's standard dummy text ever since
the 1500s, when an unknown printer took a galley of type and scrambled it to
make a type specimen book. It has survived not only five centuries, but also
the leap into electronic typesetting, remaining essentially unchanged. It was
popularised in the 1960s with the release of Letraset sheets containing Lorem
Ipsum passages, and more recently with desktop publishing software like Aldus
PageMaker including versions of Lorem Ipsum.</Text>
</AastraIPPhoneTextScreen>
```

Resulting Screen (6869i/6930)

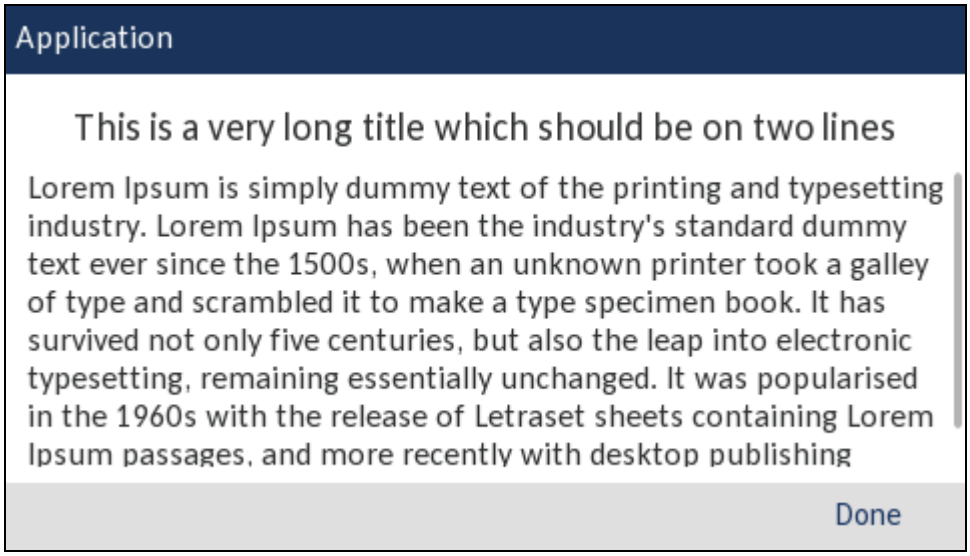


Figure 41: TextScreen Example (6869i/6930)

3.5 FORMATTEDTEXTSCREEN OBJECT (ALL MODELS)

The `FormattedTextScreen` object allows the XML designer to display formatted (alignment, size, color and scrolling) text.

3.5.1 IMPLEMENTATION (6863I/6865I)

This text is divided into 3 distinct blocks, any of which can be empty.

The first block is displayed at the top of the display and contains static text. This block takes up as many lines as the XML object specifies and can range from 0 up to the size of the physical screen.

The next block, displayed below the first block, displays scrolling text and takes up as many lines as the designer specifies up to the size of the screen.

The final block of contains static text and will take up whatever lines are left on the screen.

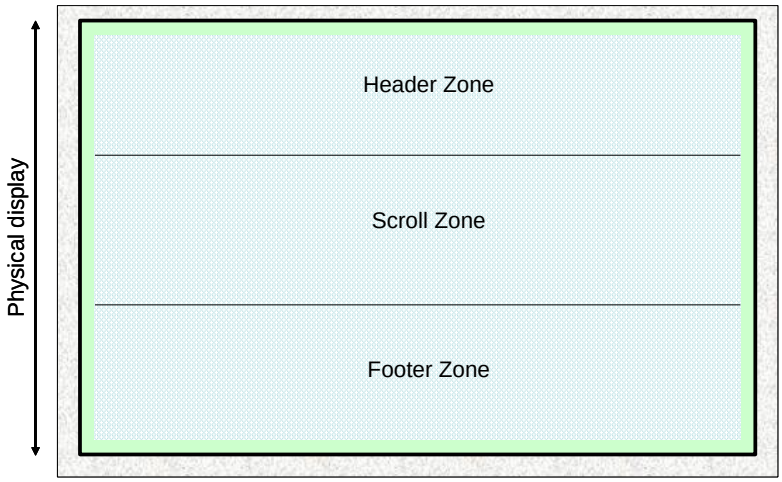


Figure 42: FormattedTextScreen layout

Object native interaction

- ***Done*** Redisplays the previous XML object present in the phone browser.

Non softkey phone keys

The object is displayed on 2 lines. The Up and Down arrow keys allow the user to browse the rest of the text.

Line selected	Label	Keys	
Text		Up and Down Arrow	Browse up or down (if scrollable)
	`>Done`	Right Arrow	Done
		Left Arrow	Exit

Note:

- the Left Arrow key default interaction is disabled if the `LockIn` tag is set to “yes”.
- the Left Arrow key default interaction can be modified using the `cancelAction` tag.

- the Right arrow key default interaction can be modified using the `doneAction` tag.
- The up, down, left and right arrow key default interactions can be overridden using the `scrollUp`, `scrollDown`, `scrollLeft` and `scrollRight` tags.

3.5.2 IMPLEMENTATION (6867I/6869I/6920/6930)

The text zone is divided into 3 distinct blocks, any of which can be empty.

The first block (header zone) is displayed at the top of the display and contains static text. This block takes up as many lines as the XML object specifies and can range from 0 up to the size of the physical screen.

The next block (scrolling zone), displayed below the first block, displays scrolling text and takes up as many lines as available, the size of the scrolling zone is automatically determined by using what's left of the display between the header zone and the footer zone.

The final block (footer zone) contains static text and will take up whatever lines are left on the screen starting from the bottom.



Figure 43: FormattedTextScreen layout

The *Left* key is mapped to the cancelAction tag, if not specified the XML object is destroyed.

The *Up/Down* keys allow up and down scrolling in order to display the complete text if needed.

Note: the *Left* key interaction is disabled if the `LockIn` tag is set to "yes".

Object default Softkeys (6867i)

Six customizable softkeys are available for this object.

Position	Label	Interaction	URIs
4	Done	***Exit:*** Redisplays the previous XML object present in the phone browser.	SoftKey:Exit

Object default Softkeys (6869i)

Eight customizable softkeys are available for this object.

Position	Label	Interaction	URIs
5	Done	***Exit:*** Redisplays the previous XML object present in the phone browser.	SoftKey:Exit

The following figure details how the `AastraIPPhoneFormattedTextScreen` is implemented on the Mitel 6867i and Mitel 6920.

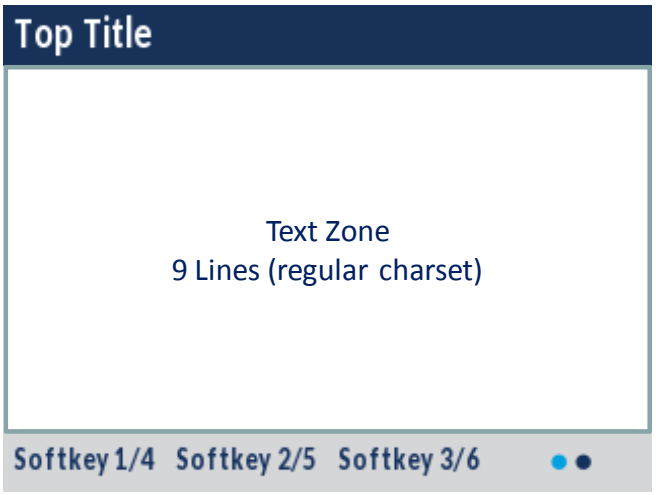


Figure 44: FormattedTextScreen implementation on 6867i/6920

Native Interaction

- "SoftKey:Exit"

3.5.3 IMPLEMENTATION (6873I/6940)

The text zone is divided into 3 distinct blocks, any of which can be empty.

The first block (header zone) is displayed at the top of the display and contains static text. This block takes up as many lines as the XML object specifies and can range from 0 up to the size of the physical screen.

The next block (scrolling zone), displayed below the first block, displays scrolling text and takes up as many lines as available, the size of the scrolling zone is automatically determined by using what's left of the display between the header zone and the footer zone.

The final block (footer zone) contains static text and will take up whatever lines are left on the screen starting from the bottom.



Figure 45: FormattedTextScreen layout on 6873i/6940

As the 6873i does not have navigation keys, the scrolling is performed by swiping the scrollable zone up and down.

Object default Softkeys

Ten customizable softkeys are available for this object.

Position	Label	Interaction	URIs
6	Done	***Exit:*** Redisplays the previous XML object present in the phone browser.	SoftKey:Exit

The following figure details how the `AastraIPPhoneFormattedTextScreen` is implemented on the Mitel 6873i and Mitel 6940.

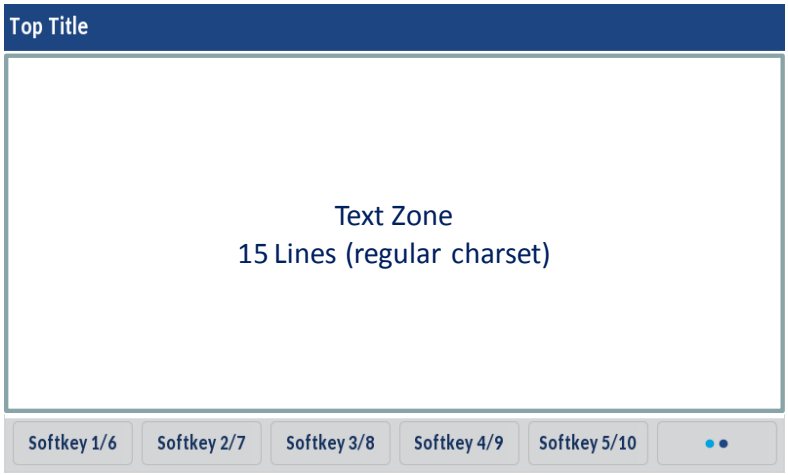


Figure 46: FormattedTextScreen implementation on 6873i/6940

Native Interaction

- “SoftKey:Exit”

3.5.4 XML DESCRIPTION

"Red" tags indicate that the tag does not apply to all phones, when not supported the tags are just ignored.

```
<AastraIPPhoneFormattedTextScreen
    destroyOnExit = "yes/no"
    cancelAction = "some URI"
    doneAction = "some URI"
    Beep = "yes/no"
    Timeout = "some integer"
    bgColor="white/black…"
    allowAnswer = "yes/no"
    allowDrop = "yes/no"
    allowXfer = "yes/no"
    allowConf = "yes/no"
    LockIn = "yes/no"
    CallProtection = "yes/no/notif"
    GoodbyeLockInURI = "some URI"
    allowDTMF = "yes/no"
    scrollUp = "some URI"
    scrollDown = "some URI"
    scrollLeft = "some URI"
    scrollRight = "some URI"
    fontMono = "yes/no"
>
     <TopTitle      icon="icon index"
                    Color="white/black…"
       >Top Title</TopTitle>
     <Line    wrap="yes/no"
              blink="no/slow/fast"
              Size="small/normal/double/large"
              Align="left/center/right"
              Color="white/black…"
       >A line of static text</Line>
       <!—Additional Lines may be added -->
     <Scroll Height="integer">
              <Line  wrap="yes/no"
                     blink="no/slow/fast"
                     Size="small/normal/double/large"
                     Align="left/center/right"
                     Color="white/black…"
              >Scrolling text</Line>
              <!—Additional Lines may be added -->
    </Scroll>
    <Line     wrap="yes/no"
              blink="no/slow/fast"
              Size="small/normal/double/large"
              Align="left/center/right"
              Color="white/black…"
       >Some static footer text</Line>
<!—Additional Lines may be added -->
    <Dial   line="SIP line">Number to dial</Dial>
    <IconList>
              <Icon index = "int">Icon:Iconname or HEX string</Icon>
              <!—As many as different icons used in the object -->
              <!—Up to 22 icons, index can be 1 to 21 -->
    </IconList>
<!—Additional Softkey Items may be added -->
</AastraIPPhoneFormattedTextScreen>
```

Notes:

- Any lines that would display past the bottom of the screen will be ignored.
- Text will not display with the top or bottom cut off.
- Text that extends past the edge of the screen will be cropped to the last fully displayed word.

XML Document Objects

Document Object	Position	Type	Comments
AastraIPPhoneFormattedTextScreen	Root tag	Mandatory	Root object
destroyOnExit	Root tag	Optional	"yes/no" indicates if the object is kept or not in the phone browser after exit. If not specified, the object is kept in the browser.
cancelAction	Root tag	Optional	Defines the URI to be called when the user cancels the XML object.
doneAction	Root tag	Optional	Defines the URI to be called when the user selects the "Done" softkey or 'SoftKey:Exit'.
Beep	Root tag	Optional	"yes" or "no" to indicate if a notification beep must be generated by the phone.
Timeout	Root tag	Optional	Overrides the default 45 seconds timeout of the UI XML object. A Timeout set to "0" will disable the timeout feature. See section 4.10.2 for more details
bgColor	Root tag	Optional	Set the background color of the object. the possible values are detailed in chapter 4.2.3. If not specified, the default value is "white". *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*
LockIn	Root tag	Optional	If set to "yes", the phone ignores all events that would cause the screen to exit without using the keys defined by the object. Default value is "no". See section 4.10.3 for more details.
CallProtection	Root tag	Optional	If set to "yes", the phone will not destroy the XML object being displayed on an incoming call. If set to "notif", the behavior is the same as "yes" but a notification is displayed on the screen

Document Object	Position	Type	Comments
			providing the caller ID details. *"notif" is equivalent to "yes" on 6863i and 6865i as screen notification are not supported*.
GoodbyeLockInURI	Root tag	Optional	Valid only if LockIn="yes", this tag defines a URI to be called when the "Goodbye" key is pressed during a locked XML session. This URI overrides the native behavior of the "Goodbye" key which is to destroy the current XML object displayed.
allowAnswer	Root tag	Optional	This tag applies only to the non-softkey phones. If set to "yes", the phone will display "Ignore" and "Answer" if the XML object is displayed when the phone is in the ringing state. Default value is "no". See section 6.3 for more details. *Only for 6863i and 6865i*
allowDrop	Root tag	Optional	This tag applies only to the non-softkey phones. If set to "yes", the phone will display "Drop" if the XML object is displayed when the phone is in the connected state. Default value is "no". See section 6.4 for more details. *Only for 6863i and 6865i*
allowXfer	Root tag	Optional	This tag applies only to the non-softkey phones. If set to "yes", the phone will display "Xfer" if the XML object is displayed when the phone is in the connected state. Default value is "no". See section 6.4 for more details. *Only for 6863i and 6865i*
allowConf	Root tag	Optional	This tag applies only to the non-softkey phones. If set to "yes", the phone will display "Conf" if the XML object is displayed when the phone is in the connected state. Default value is "no". See section 6.4 for more details. *Only for 6863i and 6865i*
allowDTMF	Root tag	Optional	This tag allows letting keypad strokes as DTMF when the phone is in the connected status.

Document Object	Position	Type	Comments
scrollUp	Root tag	Optional	This tag allows overriding the default behavior of the Up arrow key once the scrolling reaches an end. *Not available on 6873i and 6940*
scrollDown	Root tag	Optional	This tag allows overriding the default behavior of the Down arrow key once the scrolling reaches an end. *Not available on 6873i and 6940*
scrollLeft	Root tag	Optional	This tag allows overriding the default behavior of the Left arrow key once the scrolling reaches an end. *Not available on 6873i and 6940*
scrollRight	Root tag	Optional	This tag allows overriding the default behavior of the Right arrow key once the scrolling reaches an end. *Not available on 6873i and 6940*
fontMono	Root tag	Optional	If set to “no”, a proportional font will be used to display the object instead of a monotype font. Default value is “yes”. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*
TopTitle	Body	Optional	Text to be used as top title for the object *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*
Icon	TopTitle tag	Optional	Index of the icon to be used for the top title *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*
Color	TopTitle tag	Optional	Label color, the possible values are detailed in chapter 4.2.3. If not specified, the default value is “white”. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*
Line	Body	Optional	Text to be displayed on the line. If the text is larger than the display, line is cropped to the last word.
wrap	Line tag	Optional	Indicates if the text for the line is wrapped on multiple lines or limited to a single line

Document Object	Position	Type	Comments
			(default). *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*
blink	Line tag	Optional	Indicates if the blink mode of the line “slow”, “fast” or “no” (no by default). *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*
Size	Line tag	Optional	Size of the font for the line: “small” for the small font “normal” for the regular font, “double” for the larger font. “large” for the largest font. “normal” is the default value if not specified.
Align	Line tag	Optional	Alignment of the line, “left”, “right” or “center”. If not specified, the default value is “left”.
Color	Line tag	Optional	Color of the line, the possible values are detailed in chapter 4.2.3. If not specified, the default value is “white”. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*
Scroll	Body	Optional	Defines the scrolling section of the display.
Height	Scroll tag	Optional	Specifies the height of the scroll zone. If not specified, the phone uses the remaining lines of the physical screen and does not allow footer lines.
Line	Scroll Body	Optional	Text to be displayed on the line in the scrolled zone. If the text is larger than the display, line is cropped to the last word.
wrap	Line tag	Optional	Indicates if the text for the line is wrapped on multiple lines or limited to a single line (default). *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*
blink	Line tag	Optional	Indicates if the blink mode of the line “slow”, “fast” or “no” (no by default). *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*

Document Object	Position	Type	Comments
Size	Line tag	Optional	Size of the font for the line: “small” for the small font “normal” for the regular font, “double” for the larger font. “large” for the largest font. “normal” is the default value if not specified.
Align	Line tag	Optional	Alignment of the scrolled line, “left”, “right” or “center”. If not specified the default value is “left”.
Color	Line tag	Optional	Color of the line, the possible values are detailed in chapter 4.2.3. If not specified, the default value is “white”. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*
Dial	Body	Optional	Defines what number will be dialed when an offhook action is performed on the phone or if the “Dial2” custom softkey is pressed
line	Dial tag	Optional	Defines which SIP line to use when the Dial command is executed. If omitted, the Dial command is performed using the first available SIP line.
IconList	Body	Optional	List of icon definitions *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*.
Icon	IconList body	Optional	Icon value, it can be “Icon:Iconname”, or the URL to a png file . See section 4.2.2 for more details. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only.*
Index	Icon tag	Optional	Index of the icon must be consistent with the ‘Icon’ used in the MenuItems. Possible values are 1 to 21. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*.
SoftKey	Body	Optional	See section 4.1 for details Not available on non softkey phones

Limitations on the non softkey phones

- Custom Softkeys are not supported
- Only 2 lines are available to display the object.
- A double size line on line 1 does not allow any scrolling
- The scroll height can not be more than 2.

3.5.5 EXAMPLES

XML Example 1 (6863i/6865i)

```
<AastraIPPhoneFormattedTextScreen destroyOnExit = "yes">
     <Scroll Height="2">
             <Line>Line 1</Line>
             <Line>Line 2</Line>
             <Line Size="double">Line 3</Line>
             <Line>Line 4</Line>
             <Line>Line 5</Line>
    </Scroll>
    <Line Align="center">Footer</Line>
</AastraIPPhoneFormattedTextScreen>
```

Resulting Screen

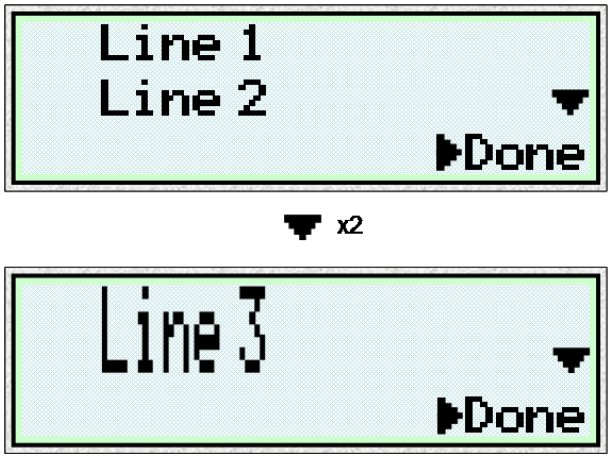


Figure 47: FormattedTextScreen Example 1 (6863i/6865i)

XML Example 2

In this example, there is no need for scrolling.

```
<AastraIPPhoneFormattedTextScreen destroyOnExit="yes">
      <Line Size="large" Align="center" Color="red">Header Line</Line>
      <Scroll>
            <Line>Scrolled Line 1</Line>
            <Line>Scrolled Line 2</Line>
            <Line>Scrolled Line 3</Line>
            <Line>Scrolled Line 4</Line>
            <Line>Scrolled Line 5</Line>
      </Scroll>
      <Line Size="double" Align="center" Color="blue">Footer Line</Line>
</AastraIPPhoneFormattedTextScreen>
```

Resulting Screen

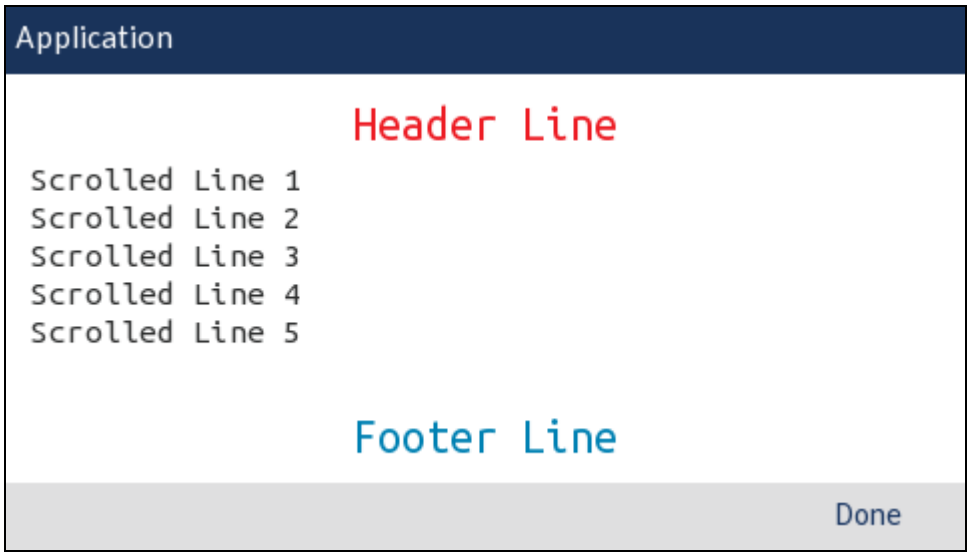


Figure 48: FormattedTextScreen Example 2 (6869i/6930)

XML Example 3

In this example, scrolling is needed.

```
<AastraIPPhoneFormattedTextScreen destroyOnExit="yes" fontMono="no">
      <TopTitle Color="yellow">FormattedTextScreen</TopTitle>
      <Line Size="double" Align="center" Color="red">Header Line</Line>
      <Scroll>
            <Line wrap="yes">Scrolled Line 1 - This is an example where the
line can be wrapped like a paragraph.</Line>
            <Line>Scrolled Line 2 - This is an example where the line is not
wrapped like a paragraph.</Line>
            <Line>Scrolled Line 3</Line>
            <Line>Scrolled Line 4</Line>
            <Line>Scrolled Line 5</Line>
            <Line>Scrolled Line 6</Line>
            <Line>Scrolled Line 7</Line>
            <Line>Scrolled Line 8</Line>
            <Line>Scrolled Line 9</Line>
            <Line>Scrolled Line 10</Line>
      </Scroll>
      <Line Size="double" Align="center" Color="blue">Footer Line</Line>
</AastraIPPhoneFormattedTextScreen>
```

Resulting Screen

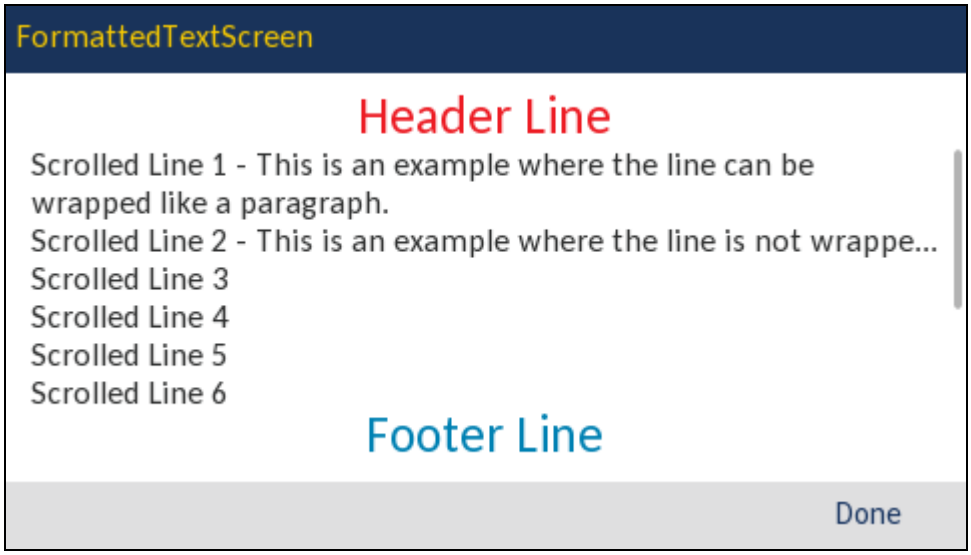


Figure 49: FormattedTextScreen Example 3 (6869i/6930)

3.6 IMAGESCREEN OBJECT (6867I/6869I/6873I/6920/6930/6940)

The `ImageScreen` object can be used to display single image (bitmap for the 55i/57i/57iCT/6735i/6737i, jpeg or png file for 6867i, 6869i, 6873i and 6739i).

The user can specify where the image should be placed by setting horizontal and vertical alignment of the upper left hand corner, along with the height and width of the image.

3.6.1 IMPLEMENTATION (6867I/6869I/6920/6930)

For the 6867i/6920 and 6869i/6930, the image is either a 24/32 bit depth “png” or a “jpeg” file located on a server and which can be downloaded by the phone using TFTP, FTP, HTTP or HTTPS. See chapter 4.2.1.1 for more details.

Two image sizes are supported:

6867i/6920

- Up to 320x180 pixels (mode=regular or extended).
- Up to 320x240 pixels (mode=fullscreen). In that case, the softkeys are not displayed.

6869i/6930

- Up to 480x204 pixels (mode=regular or extended).
- Up to 480x272 pixels (mode=fullscreen). In that case, the softkeys are not displayed

If the image is bigger than the size supported by the requested mode, the image is clipped based on the requested alignment.

If the image is smaller than the size supported by the requested mode, the image is displayed based on the requested alignment

The ***Left*** key is mapped to the cancelAction tag, if not specified the XML object is destroyed.

Note:

- the Left Arrow key default interaction is disabled if the `LockIn` tag is set to “yes”.
- If the `LockIn` tag is set to “yes” and the `cancelAction` tag is configured, pressing the Left Arrow key triggers the configured `cancelAction`.
- The up, down, left and right arrow key default interactions can be overridden using the `scrollUp`, `scrollDown`, `scrollLeft` and `scrollRight` tags.

Object default Softkeys (6867i/6920)

Six (four physical) customizable softkeys are available for this object.

Position	Label	Interaction	URIs
4	Done	***Exit:*** Redisplays the previous XML object present in the phone browser.	SoftKey:Exit

Object default Softkeys (6869i/6930)

Eight (five physical) customizable softkeys are available for this object.

Position	Label	Interaction	URIs
5	Done	***Exit:*** Redisplays the previous XML object present in the phone browser.	SoftKey:Exit

The following figure details how the `AastraIPPhoneImageScreen` is implemented on the Mitel 6867i and 6920.

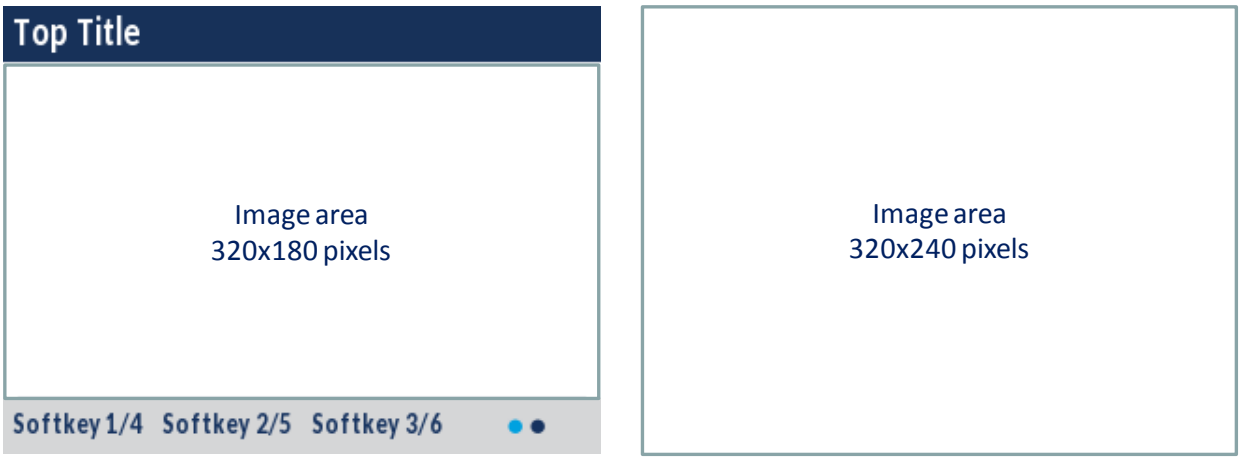


Figure 50: ImageScreen 6867i/6920 implementation (mode regular/extended and full screen)

Also a URI can be programmed to be called when the user presses the ***Select*** key (imageAction root tag). If this tag is empty, phone will use the URI configured by the doneAction tag (SoftKey:Exit by default).

So by default pressing the ***Select*** key destroys the current XML object unless an imageAction or a doneAction is configured.

Native Interaction

- "SoftKey:Exit"

3.6.2 IMPLEMENTATION (6873I/6940)

For the 6873i/6940, the image is either a 24/32 bit depth "png" or a "jpeg" file located on a server and which can be downloaded by the phone using TFTP, FTP, HTTP or HTTPS. See chapter 4.2.1.1 for more details.

Two image sizes are supported:

- Up to 800x372 pixels (mode=regular or extended).
- Up to 800x480 pixels (mode=fullscreen). In that case, the softkeys are not displayed.

If the image is bigger than the size supported by the requested mode, the image is clipped following the requested alignment.

If the image is smaller than the size supported by the requested mode, the image is displayed following the requested alignment

Note: as the 6873i and 6940 do not have navigation keys, the scroll actions are mapped to screen swipes

- `scrollUp`, by swiping from top to bottom
- `scrollDown`, by swiping from bottom to top
- `scrollLeft`, by swiping from left to right

- `scrollRight`, by swiping from right to left

Object default Softkeys

Ten (six physical) customizable softkeys are available for this object.

Position	Label	Interaction	URIs
6	Done	***Exit:*** Redisplays the previous XML object present in the phone browser.	SoftKey:Exit

The following figure details how the `AastraIPPhoneImageScreen` is implemented on the Mitel 6873i and Mitel 6940.

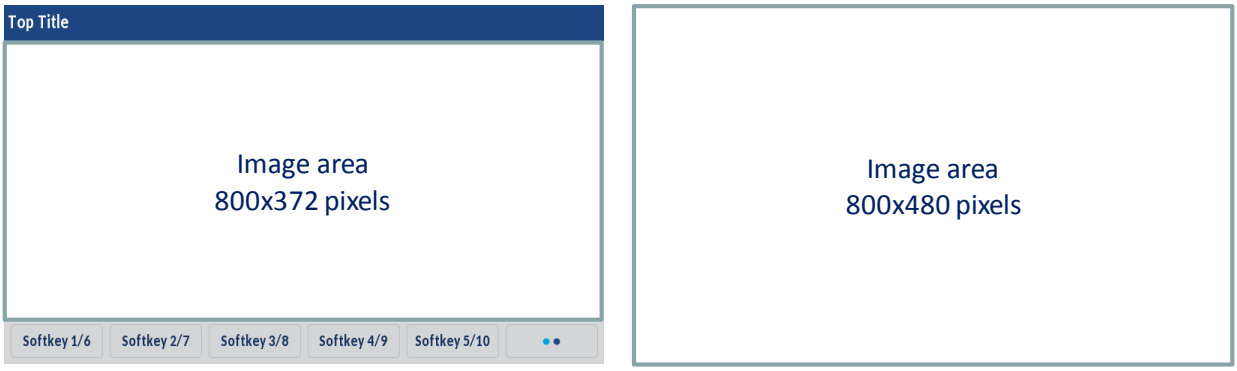


Figure 51: ImageScreen 6873i/6940 implementation (mode regular/extended and full screen)

Also a URI can be programmed to be called when the user presses on the displayed image (imageAction root tag). If this tag is empty, phone will use the URI configured by the doneAction tag (SoftKey:Exit by default).

So, by default, pressing on the image destroys the current XML object unless an imageAction or a doneAction is configured.

Native Interaction

- "SoftKey:Exit"

3.6.3 XML DESCRIPTION

"Red" tags indicate that the tag does not apply to all phones, when not supported the tags are just ignored.

```
<AastraIPPhoneImageScreen
    destroyOnExit = "yes/no"
    cancelAction = "some URI"
    doneAction = "some URI"
    imageAction = "some URI"
    Beep = "yes/no"
    Timeout = "some integer"
    bgColor = "white,black…"
    LockIn = "yes/no"
    GoodbyeLockInURI = "some URI"
    CallProtection = "yes/no/notif"
    allowDTMF = "yes/no"
    scrollUp = "some URI"
    scrollDown = "some URI"
    scrollLeft = "some URI"
    scrollRight = "some URI"
```

```
    mode = "regular/extended/fullscreen"
    >
       <TopTitle     icon="icon index"
                     Color="white/black…"
       >Top Title</TopTitle>
       <Image
             verticalAlign = "top,middle,bottom"
             horizontalAlign = "left,middle,right"
             height = "height in pixels"
             width = "width in pixels"
       >Image as hexadecimal characters or URL</Image>
<!—Additional Softkey Items may be added -->
</AastraIPPhoneImageScreen>
```

XML Document Objects

Document Object	Position	Type	Comments
AastraIPPhoneImageScreen	Root tag	Mandatory	Root object
destroyOnExit	Root tag	Optional	"yes/no" indicates if the object is kept or not in the phone browser after exit. If not specified, the object is kept in the browser.
cancelAction	Root tag	Optional	Defines the URI to be called when the user cancels the XML object.
doneAction	Root tag	Optional	Defines the URI to be called when the user selects the "Done" softkey or 'SoftKey:Exit'.
imageAction	Root tag	Optional	Defines the URI to be called when the user presses on the image (6873i, 6940) or presses Select on 6867i, 6869i, 6920 and 6930.
Beep	Root tag	Optional	"yes" or "no" to indicate if a notification beep must be generated by the phone.
Timeout	Root tag	Optional	Overrides the default 45 seconds timeout of the UI XML object. A Timeout set to "0" will disable the timeout feature. See section 4.10.2 for more details
bgColor	Root tag	Optional	Set the background color of the object. the possible values are detailed in chapter 4.2.3. If not specified, the

Document Object	Position	Type	Comments
			default value is “white”.
LockIn	Root tag	Optional	If set to “yes”, the phone ignores all events that would cause the screen to exit without using the keys defined by the object. Default value is “no”. See section 4.10.3 for more details.
CallProtection	Root tag	Optional	If set to “yes”, the phone will not destroy the XML object being displayed on an incoming call. If set to “notif”, the behavior is the same as “yes” but a notification is displayed on the screen providing the caller ID details. *“notif” is equivalent to “yes” on 6863i and 6865i as screen notification are not supported*.
GoodbyeLockInURI	Root tag	Optional	Valid only if LockIn=”yes”, this tag defines a URI to be called when the “Goodbye” key is pressed during a locked XML session. This URI overrides the native behavior of the “Goodbye” key which is to destroy the current XML object displayed.
allowDTMF	Root tag	Optional	This tag allows letting keypad strokes as DTMF when the phone is in the connected status.
scrollUp	Root tag	Optional	This tag allows overriding the default behavior of the Up arrow key.
scrollDown	Root tag	Optional	This tag allows overriding the default behavior of the Down arrow key.
scrollLeft	Root tag	Optional	This tag allows overriding the default behavior of the Left arrow key.
scrollRight	Root tag	Optional	This tag allows

Document Object	Position	Type	Comments
			overriding the default behavior of the Right arrow key.
mode	Root tag	Optional	Configures the display mode, “regular”, “extended” or “fullscreen”. “extended” mode is equivalent to “regular” and kept for compatibility reasons.
TopTitle	Body	Optional	Text to be used as top title for the object
Icon	TopTitle tag	Optional	Index of the icon to be used for the top title
Color	TopTitle tag	Optional	Label color, the possible values are detailed in chapter 4.2.3. If not specified, the default value is “white”.
Image	Body	Mandatory	Image to be displayed as a predefined icon name or URL to get the png file. See section 4.2.1 for more details.
verticalAlign	Image tag	Optional	Vertical position of the image (“top”, “middle” or “bottom”). If the tag is not specified, the object will use “middle” as a default value.
horizontalAlign	Image tag	Optional	Horizontal position of the image (“left”, “middle” or “right”). If the tag is not specified, the object will use “middle” as a default value.
Height	Image tag	Mandatory	Height in pixels. Must match the image height. (not on 6867i, 6869i) *Optional and Ignored on 6867i, 6869i and 6873i*
Width	Image tag	Mandatory	Width in pixels. Must match the image width. (not on 6867i, 6869i) *Optional and Ignored on 6867i, 6869i and 6873i*
SoftKey	Body	Optional	See section 4.1 for details

3.6.4 EXAMPLES

XML Example (6739i)

```
<AastraIPPhoneImageScreen destroyOnExit="yes">
<Image height="64" width="380">http://myserver.com/images/mitel.png</Image>
<SoftKey index="1" icon="1">
<Label>Mail</Label>
<URI>http://myserver.com/script.php?action=1</URI>
</SoftKey>
<SoftKey index="10">
<Label>Exit</Label>
<URI>SoftKey:Exit</URI>
</SoftKey>
<IconList>
<Icon index="1">Icon:Envelope</Icon>
</IconList>
</AastraIPPhoneImageScreen>
```

Resulting Screen



Figure 52: ImageScreen Example (6739i)

3.7 INPUTSCREEN OBJECT – SINGLE INPUT FIELD (ALL MODELS)

The InputScreen object allows developers to create a screen capable of gathering user input. The Mitel IP phones support seven input types:

- IP Addresses,
- Numbers (integers plus * and #),
- Strings,
- Dates (US and international format)
- Time (US and international format)

Each parameter has a URL tag that is used to send information back to the HTTP server. The label in the parameter tag is appended to the address in the URL tag and sent via HTTP GET.

3.7.1 IMPLEMENTATION (6863I/6865I)

Object native interactions

- ***Done/Submit*** Completes the user input by submitting the programmed URI and value
- ***Cancel*** Redisplays the previous XML object present in the phone browser.

Non softkey phone keys

The object is displayed on two lines, one line for the prompt message and one line for the input field, the title is not displayed.

Line selected	Label	Keys	
Input field	^Cancel	Up Arrow	Exit
	vDone	Down Arrow	Done
		Right Arrow	Next Character
		Left Arrow	Previous Character
		prgkey2 (Delete)	Backspace

Note:

- the Up Arrow key interaction is disabled if the `LockIn` tag is set to “yes”.
- the Up Arrow key interaction can be modified using the `cancelAction` tag.

3.7.2 IMPLEMENTATION (6867I/6869I/6920/6930)

On the 6867i/6869i, the input is very similar to the softkey phones.

The ***Left/Right*** keys are used to navigate between characters in the input field.

The ***Up/Down*** keys allow up and down scrolling in order to display the previous/next input field.

The ***Select*** key is mapped to the doneAction tag (Submit by default).

Object native interaction

- ***Submit*** (SoftKey:Submit) Completes the user input by submitting the programmed URI and value.

“SoftKey:Exit”

Common default Softkeys (6867i/6920)

Position	Label	Description	URIs
3	Done	Completes the user input by submitting the programmed URI and value.	SoftKey:Submit
4	Cancel	Redisplays the previous XML object present in the phone browser. Not available if `LockIn` set to “yes”.	SoftKey:Exit

Common default Softkeys (6869i/6930)

Position	Label	Description	URIs
4	Done	Completes the user input by submitting the	SoftKey:Submit

Position	Label	Description	URIs
		programmed URI and value.	
5	Cancel	Redisplays the previous XML object present in the phone browser. Not available if `LockIn` set to “yes”.	SoftKey:Exit

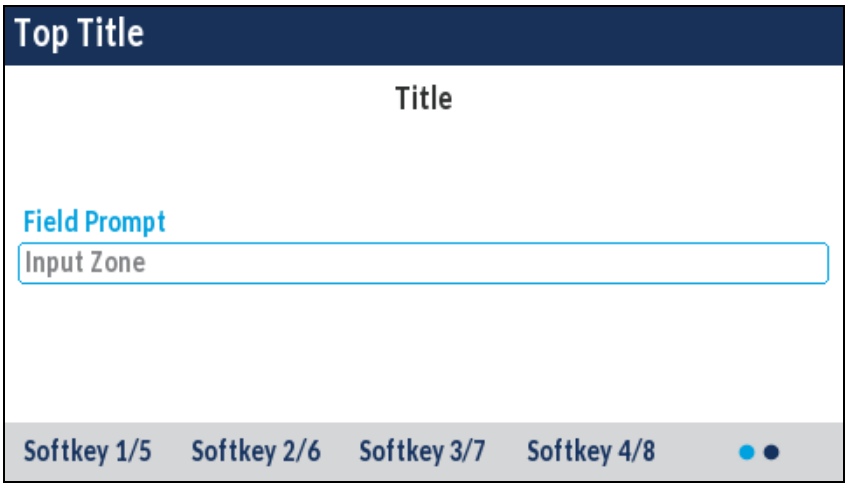


Figure 53: InputScreen implementation on 6869i/6930

On the 6867i/6869i, like “Softkey::Submit” the URI custom softkeys are also appended with the parameter values as well as the selection, this allows to exit the InputScreen object without the validity check.

3.7.3 IMPLEMENTATION (6873I/6940)

On the 6873i and 6940, the input for a field is done via a virtual keyboard which is displayed when the user touches the input zone.

The ↵ key on the virtual keyboard triggers the submission of the entered value.

The ⌨ key on the virtual keyboard removes the keyboard from the display.

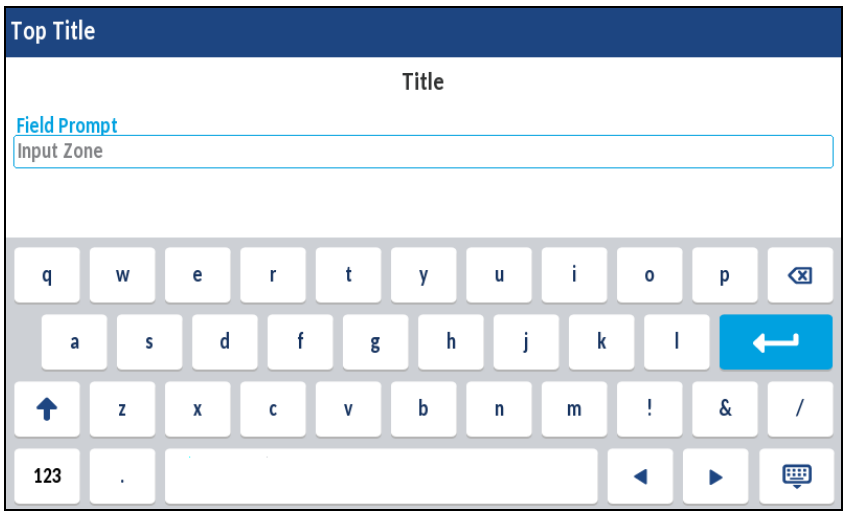


Figure 54: InputScreen implementation on 6873i/6940

Object native interaction

- ***Submit*** (SoftKey:Submit) Completes the user input by submitting the programmed URI and value.
- "SoftKey:Exit"

Object default Softkeys

Ten customizable softkeys are available for this object.

Position	Label	Interaction	URIs
5	Done	Completes the user input by submitting the programmed URI and value.	SoftKey:Submit
6	Cancel	***Exit:*** Redisplays the previous XML object present in the phone browser.	SoftKey:Exit

3.7.4 XML DESCRIPTION

"Red" tags indicate that the tag does not apply to all phones, when not supported the tags are just ignored.

```
<AastraIPPhoneInputScreen
    type = "IP/string/stringN/number/timeUS/timeInt/dateUS/dateInt"
    password = "yes/no"
    editable = "yes/no"
    destroyOnExit = "yes/no"
    cancelAction = "some URI"
    Beep = "yes/no"
    Timeout = "some integer"
    bgColor="white/black…"
    allowAnswer = "yes/no"
    allowDrop = "yes/no"
    allowXfer = "yes/no"
    allowConf = "yes/no"
    defaultFocus = "yes/no"
    LockIn = "yes/no"
    CallProtection = "yes/no/notif"
    GoodbyeLockInURI = "some URI"
    inputLanguage = "English / French / German / Italian / Spanish /
Portuguese/ Russian / Nordic"
>
       <Title       wrap="yes/no"
                    Color="white/black…"
       >Title string</Title>
       <TopTitle    icon="icon index"
                    Color="white/black…"
       >Top Title</TopTitle>
       <Prompt      Color="white/black…"
       >Guidance for the input</Prompt>
       <URL>Target receiving the input</URL>
       <Parameter   Color="white/black…"
       >name of the parameter added to URL</Parameter>
       <Default>Default Value</Default>
<!—Additional Softkey Items may be added -->
<!—Additional Icon Items may be added -->
</AastraIPPhoneInputScreen>
```

XML Document Objects

Document Object	Position	Type	Comments
AastraIPPhoneInputScreen	Root tag	Mandatory	Root object

Document Object	Position	Type	Comments
type	Root tag	Optional	Specifies the type of input, possible values are “IP”, “string”, “stringN”,“number”, “timeUS”, “timeInt”, “dateUS”, “dateInt”. Default value is “string”
password	Root tag	Optional	Specifies if the input is masked by “*” characters. Default value is “no”
editable	Root tag	Optional	Specifies if the user is allowed to modify the input. Default value is “yes”
destroyOnExit	Root tag	Optional	“yes/no” indicates if the object is kept or not in the phone browser after exit. If not specified, the object is kept in the browser.
cancelAction	Root tag	Optional	Defines the URI to be called when the user cancels the XML object.
Beep	Root tag	Optional	“yes” or “no” to indicate if a notification beep must be generated by the phone.
Timeout	Root tag	Optional	Overrides the default 45 seconds timeout of the UI XML object. A Timeout set to “0” will disable the timeout feature. See section 4.10.2 for more details
bgColor	Root tag	Optional	Set the background color of the object. the possible values are detailed in chapter 4.2.3. If not specified, the default value is “white”. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*
LockIn	Root tag	Optional	If set to “yes”, the phone ignores all events that would cause the screen to exit without using the keys defined by the object. Default value is “no”. See section 4.10.3 for more details.
CallProtection	Root tag	Optional	If set to “yes”, the phone will not destroy the XML object being displayed on an incoming call. If set to “notif”, the behavior is the same as “yes” but a notification is displayed on the screen providing the caller ID details. *“notif” is equivalent to “yes” on 6863i and 6865i as screen*

Document Object	Position	Type	Comments
			notification are not supported.
GoodbyeLockInURI	Root tag	Optional	Valid only if LockIn=”yes”, this tag defines a URI to be called when the “Goodbye” key is pressed during a locked XML session. This URI overrides the native behavior of the “Goodbye” key which is to destroy the current XML object displayed.
defaultFocus	Root tag	Optional	If set to “yes”, the input field automatically goes to edit mode (keyboard displayed). Default value is “no”. *6873i and 6940 only*
inputLanguage	Root tag	Optional	Defines the language character set used for the input. English by default. See section 15 for the localized key mapping.
allowAnswer	Root tag	Optional	This tag applies only to the non-softkey phones. If set to “yes”, the phone will display “Ignore” and “Answer” if the XML object is displayed when the phone is in the ringing state. Default value is “no”. See section 6.3 for more details. *Only for 6863i and 6865i.*
allowDrop	Root tag	Optional	This tag applies only to the non-softkey phones. If set to “yes”, the phone will display “Drop” if the XML object is displayed when the phone is in the connected state. Default value is “no”. See section 6.4 for more details. *Only for 6863i and 6865i.*
allowXfer	Root tag	Optional	This tag applies only to the non-softkey phones. If set to “yes”, the phone will display “Xfer” if the XML object is displayed when the phone is in the connected state. Default value is “no”. See section 6.4 for more details. *Only for 6863i and 6865i.*
allowConf	Root tag	Optional	This tag applies only to the non-softkey phones. If set to “yes”, the phone will display “Conf” if the XML object is displayed when the phone is in the connected state. Default value is “no”. See section 6.4 for more details.

Document Object	Position	Type	Comments
			Only for 6863i and 6865i.
Title	Body	Optional	Text to be used as title for the object
Wrap	Title tag	Optional	If set to “yes” the title of the object will be wrapped on 2 lines. *Only for 6863i and 6865i.*
Color	Title tag	Optional	Label color, the possible values are detailed in chapter 4.2.3. If not specified, the default value is “white”. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*
TopTitle	Body	Optional	Text to be used as top title for the object *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*
Icon	TopTitle tag	Optional	Index of the icon to be used for the top title *6867i, 6869i, 6873i, 6920, 6930 and 6940 only only.*
Color	TopTitle tag	Optional	Label color, the possible values are detailed in chapter 4.2.3. If not specified, the default value is “white”. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*
Prompt	Body	Mandatory	Text to be displayed as guidance for the user input.
Color	Prompt tag	Optional	Label color, the possible values are detailed in chapter 4.2.3. If not specified, the default value is “yellow”. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*
URL	Body	Mandatory	URI called when user completes his input.
Parameter	Body	Mandatory	Name of the parameter to be added to the URL after input is complete.
Color	Parameter tag	Optional	Label color, the possible values are detailed in chapter 4.2.3. If not specified, the default value is “white”. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*
Default	Body	Mandatory	Default value to be displayed in the input field.

Document Object	Position	Type	Comments
SoftKey	Body	Optional	See section 4.1 for more details *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*

3.7.5 INPUT TYPE: IP

When the type is set to IP, the user input is restricted to integers only. The phone will validate the user input; if an invalid IP address is entered, nothing will be sent to the server and the user will receive an error message.

Object default Softkeys (6867i/6869i/6873i/6920/6930/6940)

Position	Label	Description	URIs
1	Backspace	Deletes the character before the cursor in the input field.	SoftKey:BackSpace
2	Dot "."	Inserts a "." in the user input at the cursor position	SoftKey:Dot
One before last	Done	Completes the user input by submitting the programmed URI and value.	SoftKey:Submit
Last	Cancel	Redisplays the previous XML object present in the phone browser. Does not appear if `LockIn` set to "yes".	SoftKey:Exit

XML Example

```
<AastraIPPhoneInputScreen type = "IP">
       <Title>Proxy Server</Title>
       <Prompt>Server IP:</Prompt>
       <URL>http://10.50.10.53/script.pl</URL>
       <Parameter>proxy</Parameter>
       <Default></Default>
</AastraIPPhoneInputScreen>
```

Note: In this example, when the user press "Done" or "Submit" or "Enter" on the phone after entering "192.168.0.100", the phone will call the following URL

"http://10.50.10.53/script.pl?proxy=192.168.0.100".

Resulting Screen (6863i/6865i)

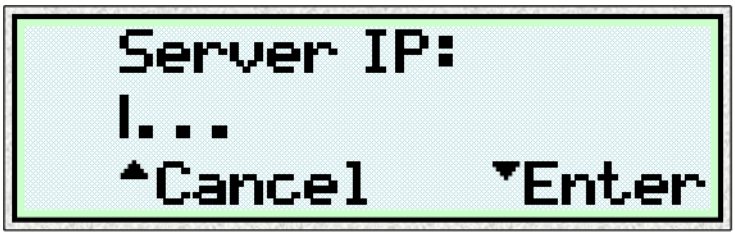


Figure 55: InputScreen "IP" Example (6863i/6865i)

Resulting Screen (6869i/6930)

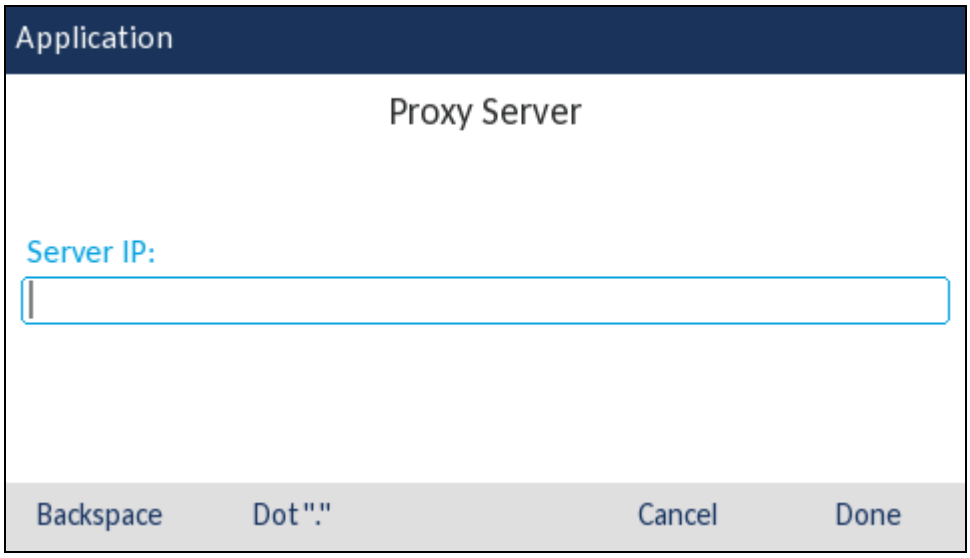


Figure 56: InputScreen "IP" Example (6869i/6930)

Resulting Screen (6873i/6940)

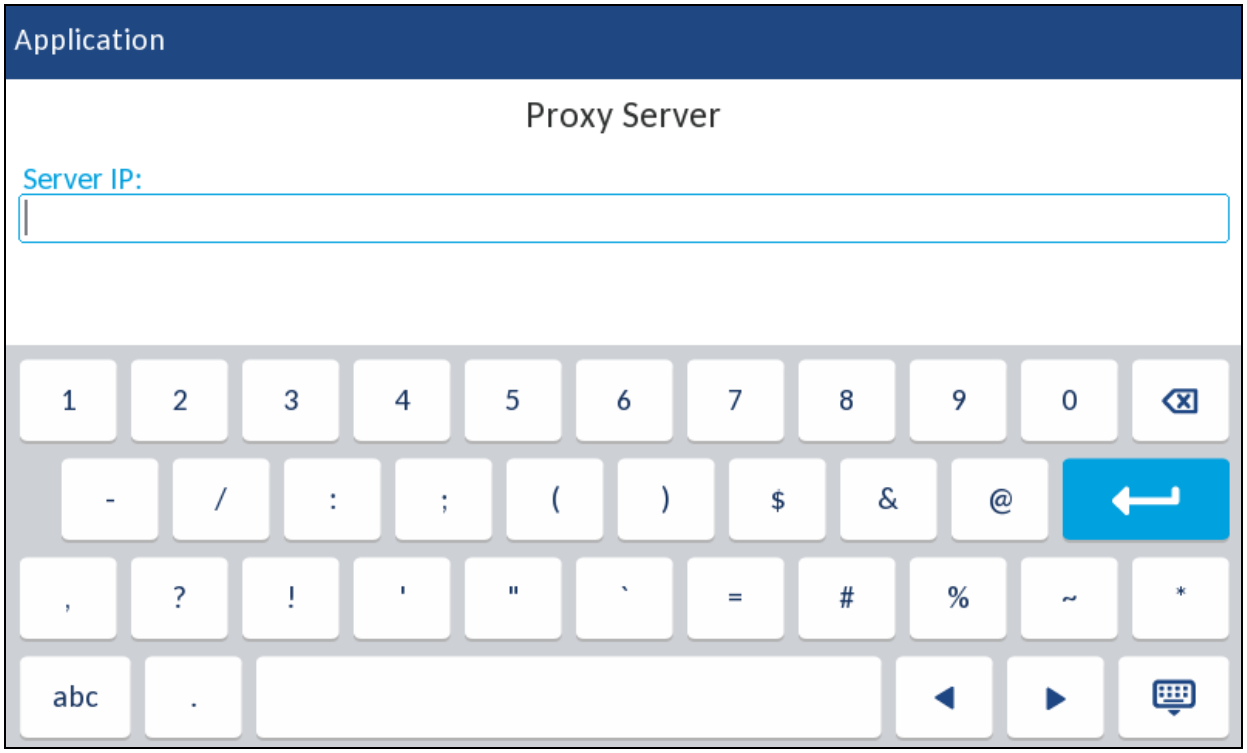


Figure 57: InputScreen "IP" Example (6873i/6940)

3.7.6 INPUT TYPE: NUMBER

Like an IP screen, a number input screen restricts the user to numbers only. Field validation is performed on the user input.

Object default Softkeys (6867i/6869i/6873i/6920/6930/6940)

Position	Label	Description	URIs
1	Backspace	Deletes the character before the cursor in the input field.	SoftKey:BackSpace
One before last	Done	Completes the user input by submitting the programmed URI and value.	SoftKey:Submit
Last	Cancel	Redisplays the previous XML object present in the phone browser. Does not appear if `LockIn` set to “yes”.	SoftKey:Exit

XML Example

```
<AastraIPPhoneInputScreen type=”number”>
       <Title>Proxy Port</Title>
       <Prompt>Port:</Prompt>
       <URL>http://10.50.10.53/script.pl</URL>
       <Parameter>port</Parameter>
       <Default>5060</Default>
</AastraIPPhoneInputScreen>
```

Resulting Screen (6863i/6865i)



Figure 58: InputScreen “Number” Example (6863i/6865i)

Resulting Screen (6869i/6930)

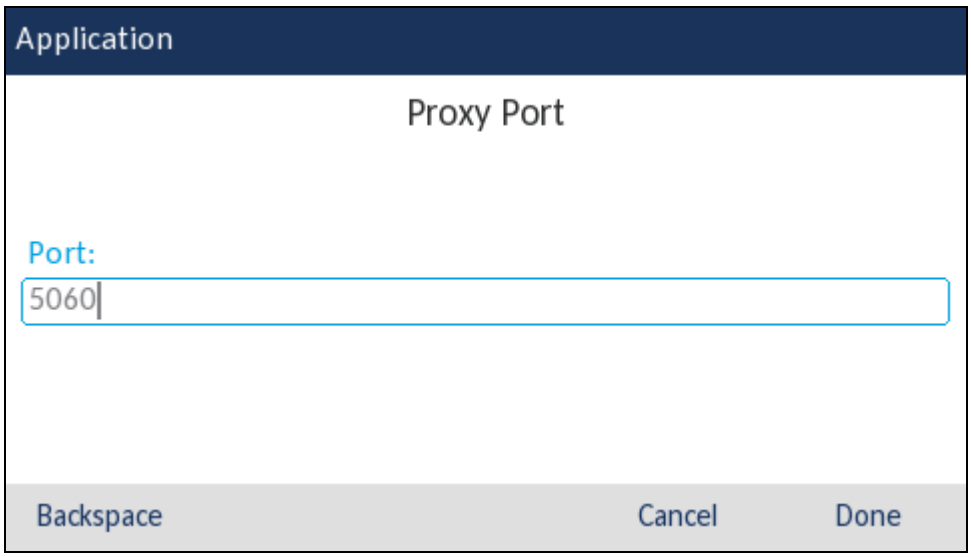


Figure 59: InputScreen “Number” Example (6869i/6930)

Resulting Screen (6873i/6940)

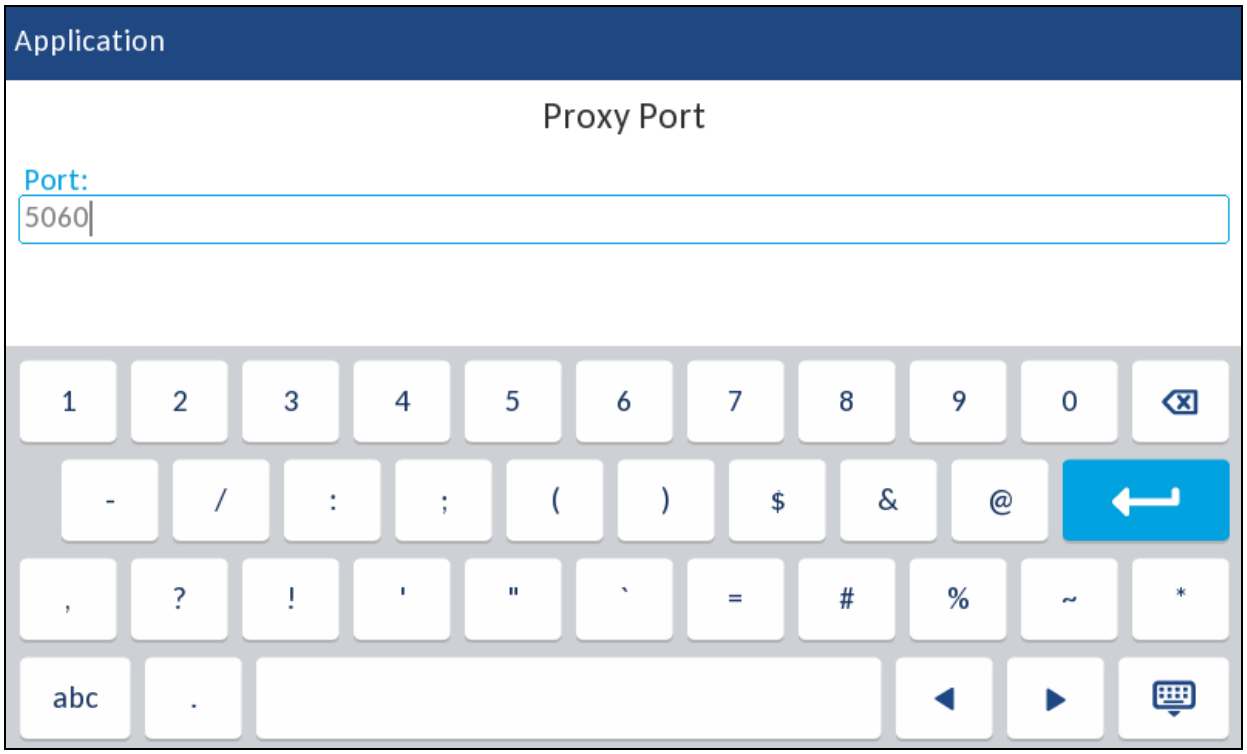


Figure 60: InputScreen "Number" Example (6873i/6940)

Note: In this example, when the user presses "Done" or "Submit" or "Enter" on the phone after entering "5060", the phone will call the following URL "`http://10.50.10.53/script.pl?port=5060`".

3.7.7 INPUT TYPE: STRING OR STRINGN

When the input type is set to string, the user can enter uppercase letters, lowercase letters, symbols, and numbers.

6863i/6865i input

The user can scroll through some available input symbols

- `. : ; , = _ , - ' & ( ) " $ !` by pressing the 1 key repeatedly.
- `# / \ @` by pressing the # key repeatedly.
- `Space` by pressing the * key repeatedly

Keys 2-9 scroll through the keypad letters in upper case and lower case (e.g. pressing repeatedly 2 will scroll through ABC2abc or abc2ABC if type is string and 2ABCabc or 2abcABC if type is stringN)

6867i/6869i/6873i/6920/6930/6940 input

The input mode can be switched via the Mode Key (Upper Case, Lower Case and Digits).

The user can scroll through some available input symbols

- `. : ; , = _ , - ' & ( ) " $ !` by pressing the 1 key repeatedly.
- `# / \ @` by pressing the # key repeatedly.
- `Space` by pressing the * key repeatedly

In Upper Case or Lower Case mode, keys 2-9 scroll through the keypad letters (e.g. pressing repeatedly 2 in Upper case mode will scroll through ABC2)

Object default Softkeys (6867i/6869i/6873i/6920/6930/6940)

Position	Label	Description	URIs
1	Backspace	Deletes the character before the cursor in the input field.	SoftKey:BackSpace
2	Dot “.”	Inserts a “.” in the user input at the cursor position	SoftKey:Dot
3	ABC>	Toggle between input modes, “ABC”, “123”, “abc”. Type is “string”	SoftKey:ChangeMode
3	123>	Toggle between input modes, “123”, “ABC”, “abc”. Type is “stringN”	SoftKey:ChangeMode
4	NextSpace	Inserts a space in the user input at the cursor position	SoftKey:NextSpace
One before last	Done	Completes the user input by submitting the programmed URI and value.	SoftKey:Submit
Last	Cancel	Redisplays the previous XML object present in the phone browser. Does not appear if `LockIn` set to “yes”.	SoftKey:Exit

XML Example

```
<AastraIPPhoneInputScreen
  type = ”string”
  password = “yes”>
       <Title>SIP Settings</Title>
       <Prompt>Enter something</Prompt>
       <URL>http://10.50.10.53/script.pl</URL>
       <Parameter>passwd</Parameter>
       <Default></Default>
</AastraIPPhoneInputScreen>
```

Resulting Screen (6863i/6865i)



Figure 61: InputScreen “String” Example (6863i/6865i)

Resulting Screen (6869i/6930)

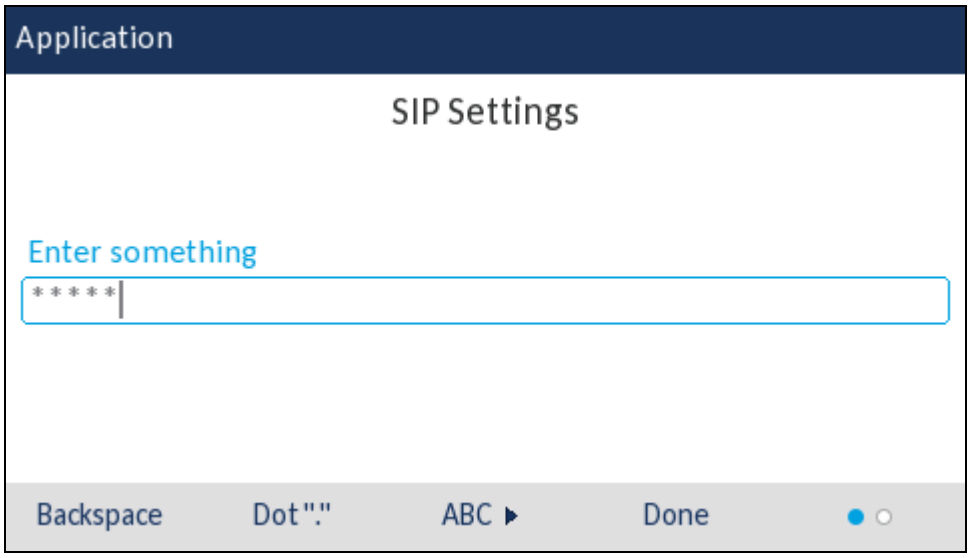


Figure 62: InputScreen "String" Example (6869i/6930)

Resulting Screen (6873i/6940)

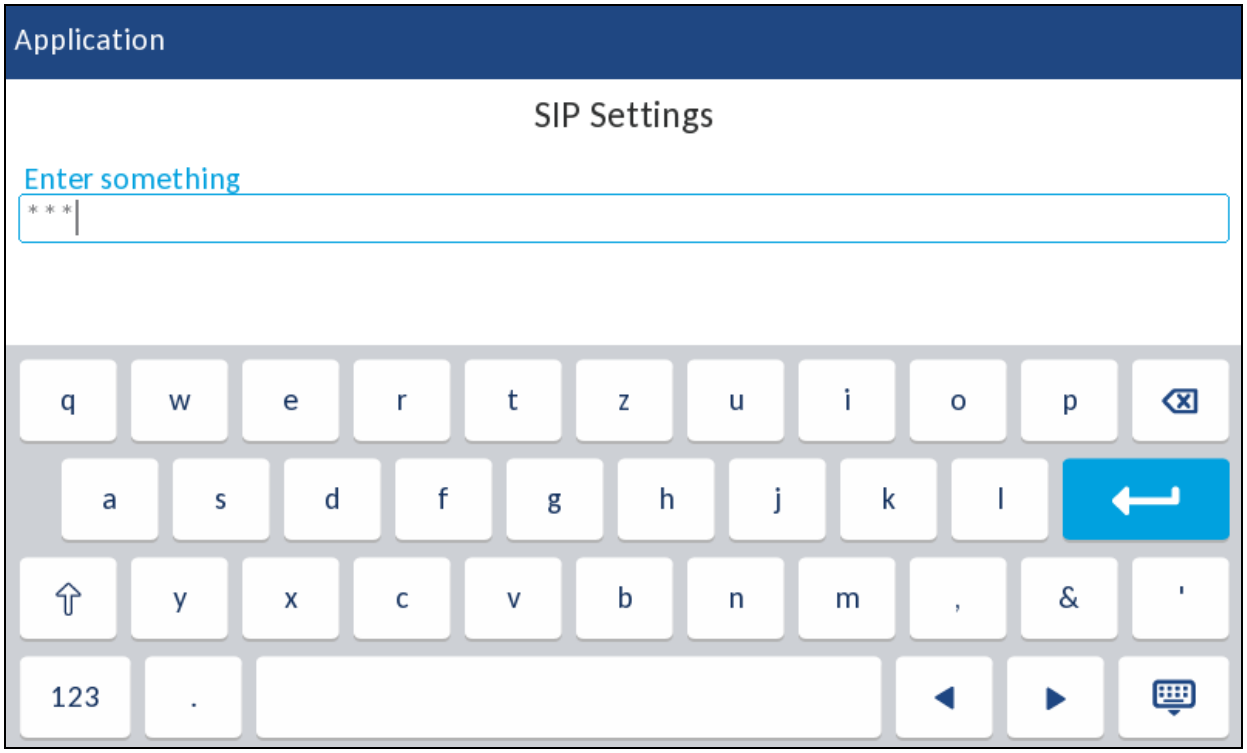


Figure 63: InputScreen "String" Example (6873i/6940)

Note: In this example, when the user press "Done" or "Submit" or "Enter" on the phone after entering "12345", the phone will call the following URL "http://10.50.10.53/script.pl?passwd=12345".

3.7.8 INPUT TYPE: TIMEUS

When the input type is set to "`timeUS`", the user can enter a time using the US format with 12 hours cycle (HH:MM:SS AM/PM with HH from 00 to 12).

The user navigates between the various fields using the left and right navigation arrow keys, the toggle between AM and PM is done using the right navigation arrow key with the cursor positioned right before the AM/PM field.

Notes:

- the "`password`" attribute has no effect on this input type.
- the format of the "`Default`" attribute must be HH:MM:SSXX
- where XX is AM or PM, HH between 00 and 12 and MM/SS between 00 and 59.
- If the "Default" tag is empty, the phone displays "12:00:00 AM".
- On the softkey phones, an extra softkey "AM/PM" is also available.

Object default Softkeys (6867i/6869i/6873i/6920/6930/6940)

Position	Label	Description	URIs
One before last	Done	Completes the user input by submitting the programmed URI and value.	SoftKey:Submit
Last	Cancel	Redisplays the previous XML object present in the phone browser. Does not appear if `LockIn` set to "yes".	SoftKey:Exit

XML Example

```
<AastraIPPhoneInputScreen type = "timeUS">
      <Title>Time US</Title>
      <Prompt>Enter time:</Prompt>
      <URL>http://10.50.10.53/script.pl</URL>
      <Parameter>time</Parameter>
      <Default></Default>
</AastraIPPhoneInputScreen>
```

Resulting Screen (6863i/6865i)

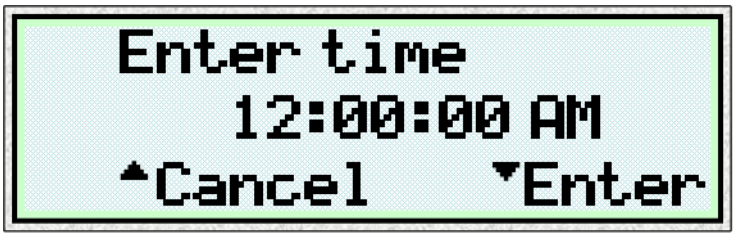


Figure 64: InputScreen "TimeUS" Example (*6863i/6865i*)

Resulting Screen (6869i/6930)

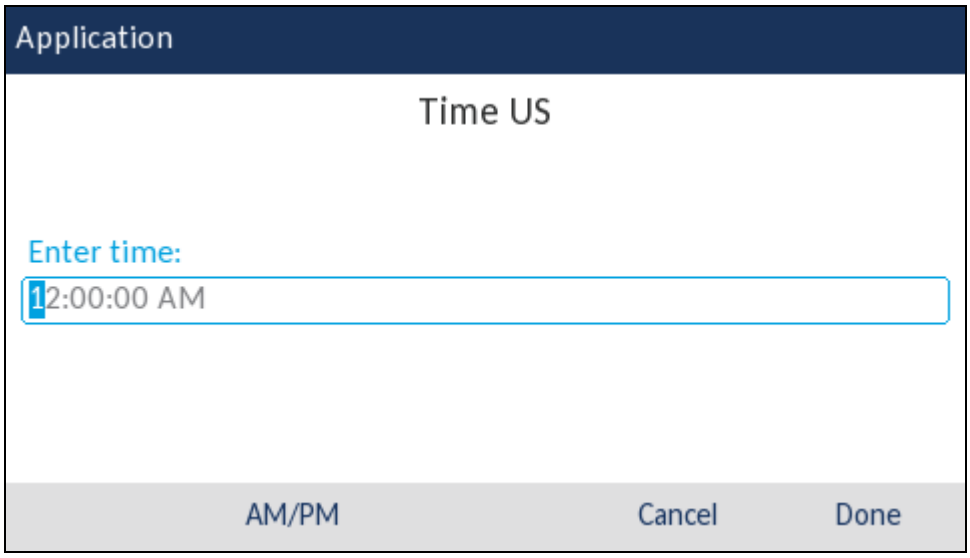


Figure 65: InputScreen "TimeUS" Example (*6869i/6930*)

Resulting Screen (6873i/6940)

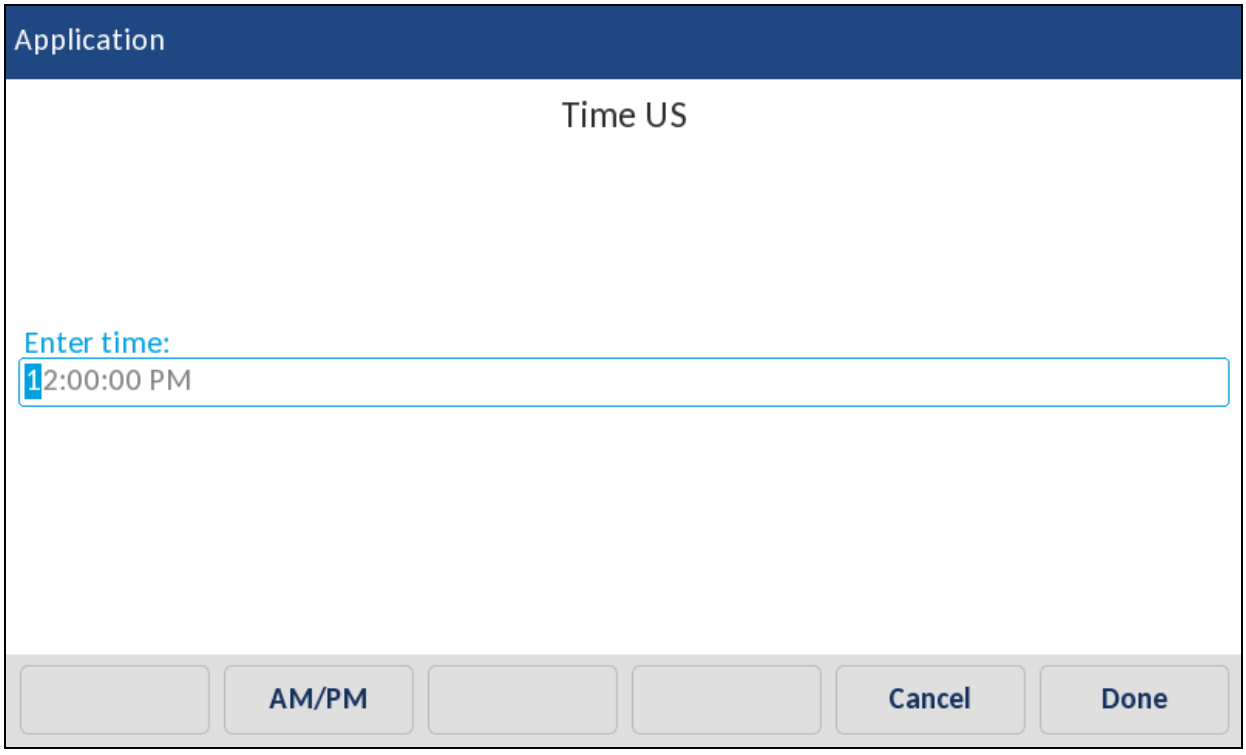


Figure 66: InputScreen "TimeUS" Example (6873i/6940)

Note: In this example, when the user press "Done" or "Submit" or "Enter" on the phone after entering "10:20:00 AM", the phone will call the following URL "http://10.50.10.53/script.pl?time=10:20:00AM" (actually the phone URL encodes the URL so the ":" character is replaced by %3A but transparent to the receiving code)

3.7.9 INPUT TYPE: TIMEINT

When the input type is set to "`timeInt`", the user can enter a time using the international format with a 24 hours cycle (HH:MM:SS with HH from 00 to 23).

The user navigates between the various fields using the left and right navigation arrow keys.

Notes:

- the "`password`" attribute has no effect on this input type.
- the format of the "`Default`" attribute must be HH:MM:SS with HH from 00 to 23. If the "`Default`" tag is empty, the phone displays "00:00:00".

Object default Softkeys (6867i/6869i/6873i/6920/6930/6940)

Position	Label	Description	URIs
One before last	Done	Completes the user input by submitting the programmed URI and value.	SoftKey:Submit
Last	Cancel	Redisplays the previous XML object present in the phone browser. Does not appear if `LockIn` set to "yes".	SoftKey:Exit

XML Example

```
<AastraIPPhoneInputScreen type = "timeInt">
      <Title>Time International</Title>
      <Prompt>Enter time:</Prompt>
      <URL>http://10.50.10.53/script.pl</URL>
      <Parameter>time</Parameter>
      <Default></Default>
</AastraIPPhoneInputScreen>
```

Resulting Screen (6863i/6865i)



Figure 67: InputScreen "TimeInt" Example (*6863i/6865i*)

Resulting Screen (6869i/6930)

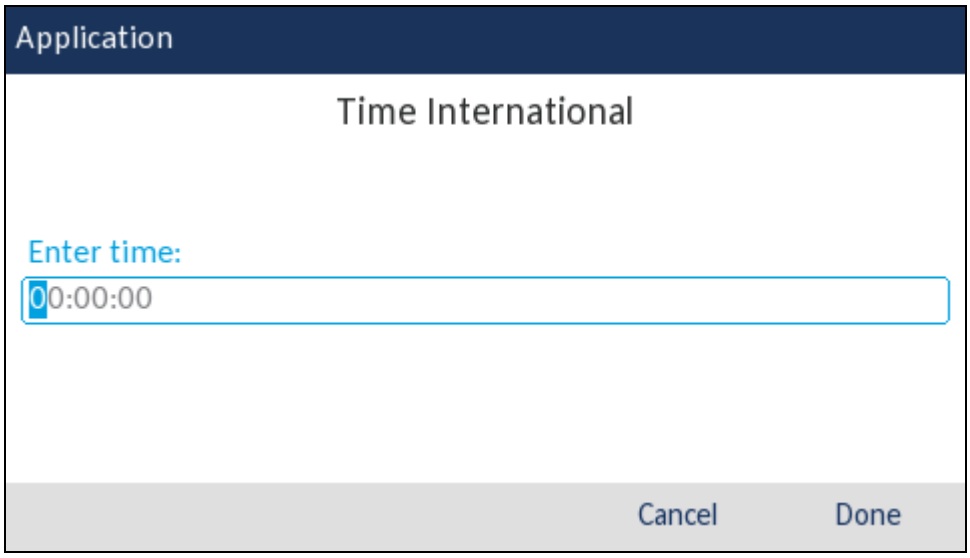


Figure 68: InputScreen "TimeInt" Example (6869i/6930)

Resulting Screen (6873i/6940)

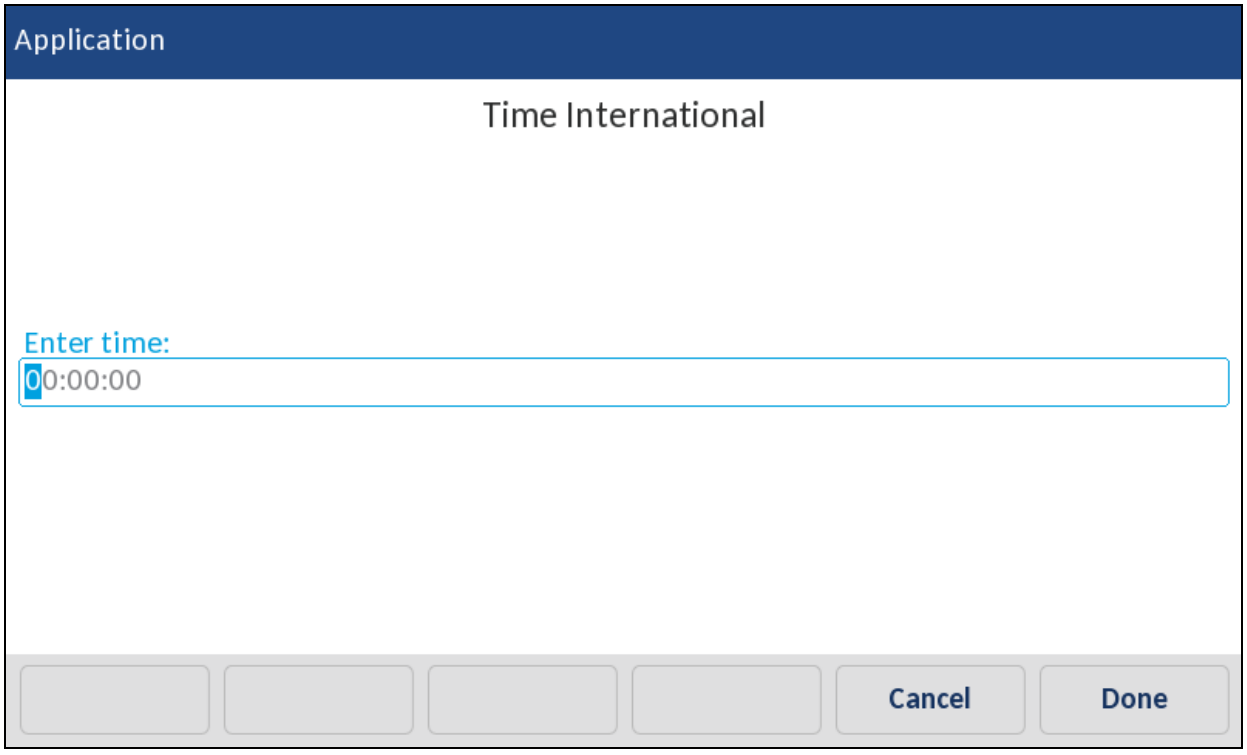


Figure 69: InputScreen "TimeInt" Example (6873i/6940)

Note: In this example, when the user press "Done" or "Submit" or "Enter" on the phone after entering "14:20:00", the phone will call the following URL http://10.50.10.53/script.pl?time=14:20:00 (actually the phone URL encodes the URL so the ":" character is replaced by %3A but transparent to the receiving code)

3.7.10 INPUT TYPE: DATEUS

When the input type is set to "`dateUS`", the user can enter a date using the US format (MM/DD/YYYY).

The user navigates between the various editable fields using the left and right navigation arrow keys.

Notes:

- the "`password`" attribute has no effect on this input type.
- the format of the "`Default`" attribute must be MM/DD/YYYY. If "`Default`" tag is empty, today's date is displayed.

Object default Softkeys (6867i/6869i/6873i/6920/6930/6940)

Position	Label	Description	URIs
One before last	Done	Completes the user input by submitting the programmed URI and value.	SoftKey:Submit
Last	Cancel	Redisplays the previous XML object present in the phone browser. Does not appear if `LockIn` set to "yes".	SoftKey:Exit

XML Example

```
<AastraIPPhoneInputScreen type = "dateUS">
      <Title>Date US</Title>
      <Prompt>Enter date</Prompt>
      <URL>http://10.50.10.53/script.pl</URL>
      <Parameter>date</Parameter>
      <Default></Default>
</AastraIPPhoneInputScreen>
```

Resulting Screen (6863i/6865i)



Figure 70: InputScreen "DateUS" Example (*6863i/6865i*)

Resulting Screen (6869i/6930)

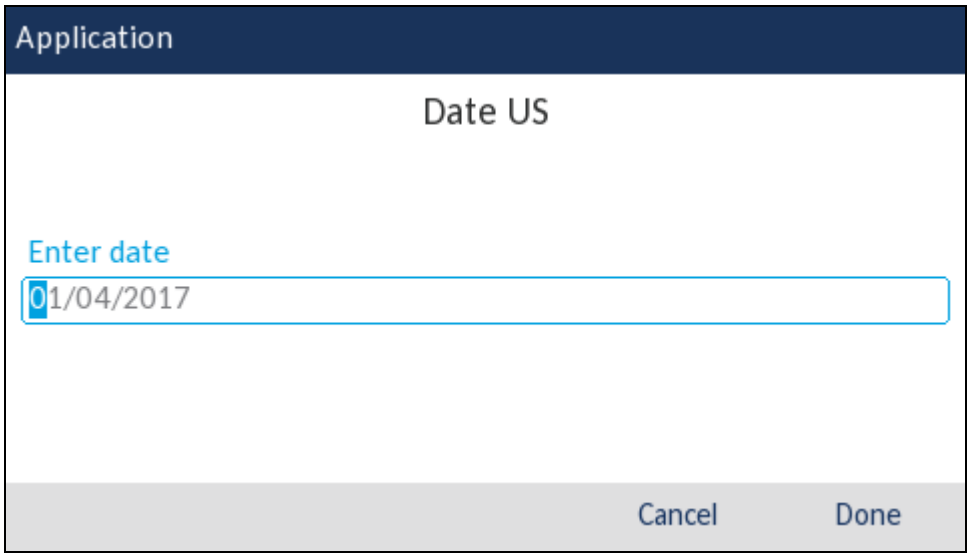


Figure 71: InputScreen "DateUS" Example (*6869i/6930*)

Resulting Screen (6873i/6940)

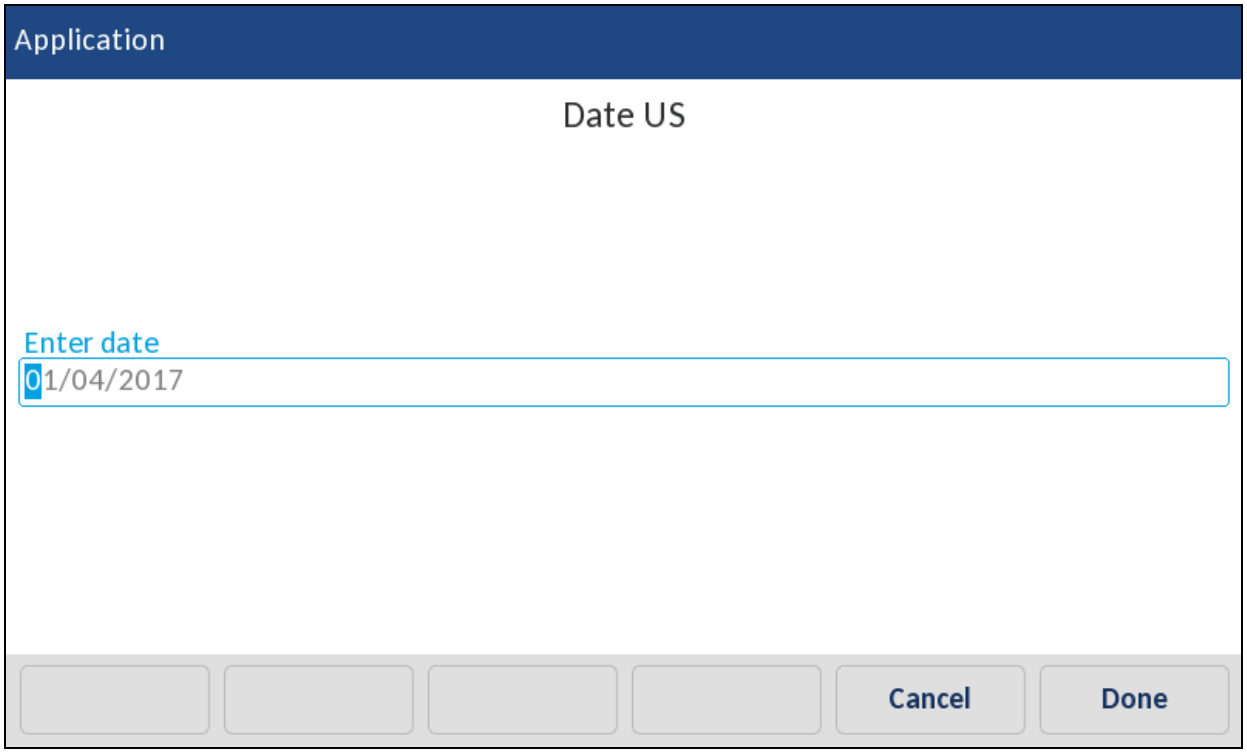


Figure 72: InputScreen "DateUS" Example (6873i/6940)

Notes:

- In this example, when the user press "Done" or "Submit" or "Enter" on the phone after entering "06/14/2009", the phone will call the following URL "`http://10.50.10.53/script.pl?date=06/14/2009`".
- the `InputScreen` object does not perform any control on the validity of the date entered by

the user.

3.7.11 INPUT TYPE: DATEINT

When the input type is set to "`dateInt`", the user can enter a date using the international format (DD/MM/YYYY).

The user navigates between the various editable fields using the left and right navigation arrow keys.

Notes:

- the "`password`" attribute has no effect on this input type.
- the format of the "`Default`" attribute must be DD/MM/YYYY. If "`Default`" tag is empty, today's date is displayed.

Object default Softkeys (6867i/6869i/6873i/6920/6930/6940)

Position	Label	Description	URIs
One before last	Done	Completes the user input by submitting the programmed URI and value.	SoftKey:Submit
Last	Cancel	Redisplays the previous XML object present in the phone browser. Does not appear if `LockIn` set to "yes".	SoftKey:Exit

XML Example

```
<AastraIPPhoneInputScreen type = "dateInt">
      <Title>Date International</Title>
      <Prompt>Enter date</Prompt>
      <URL>http://10.50.10.53/script.pl</URL>
      <Parameter>date</Parameter>
      <Default></Default>
</AastraIPPhoneInputScreen>
```

Resulting Screen (non softkey phone)

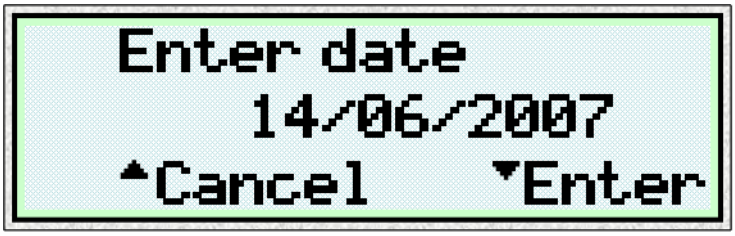


Figure 73: InputScreen "DateInt" Example (*non softkey phone*)

Resulting Screen (6869i/6930)

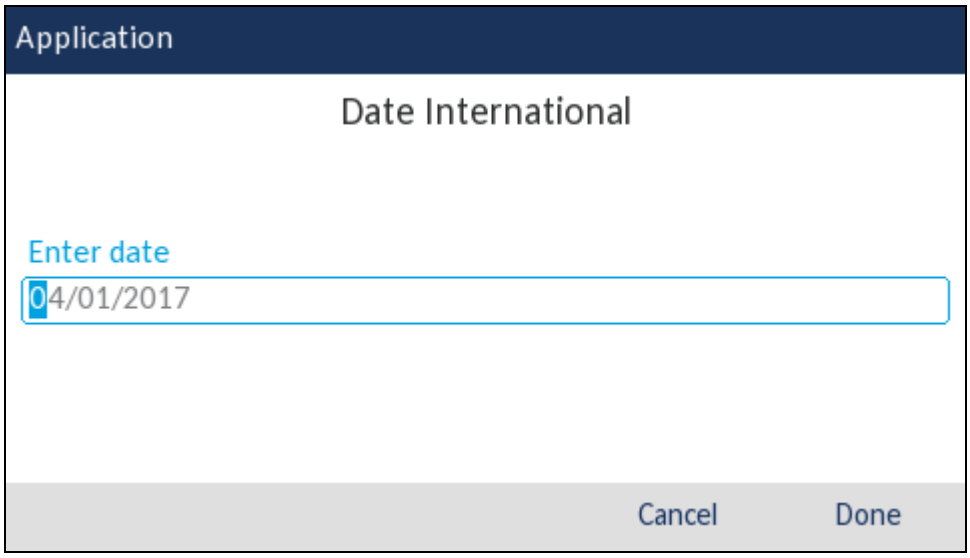


Figure 74: InputScreen "DateInt" Example (6869i/6930)

Resulting Screen (6973i/6940)

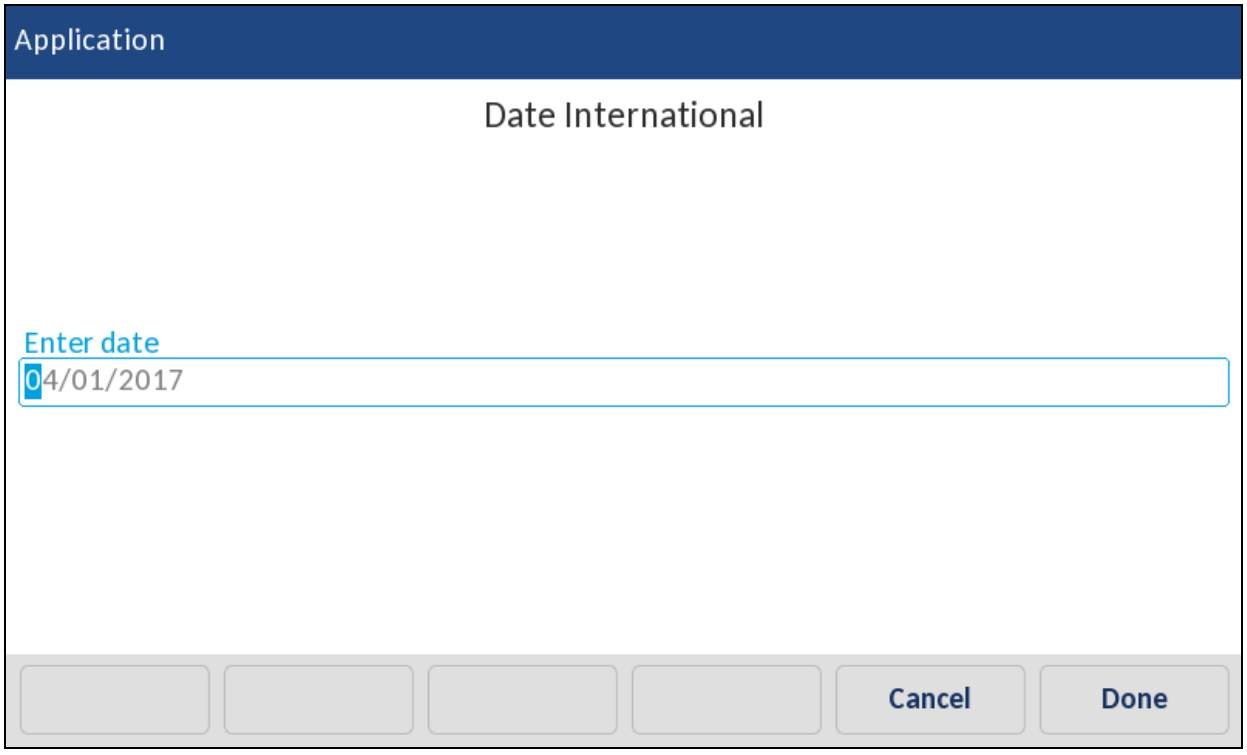


Figure 75: InputScreen "DateInt" Example (6873i/6940)

Notes:

- In this example, when the user press "Done" or "Submit" or "Enter" on the phone after entering "14/06/2009", the phone will call the following URL "`http://10.50.10.53/script.pl?date=14/06/2009`".

- **Note:** the `InputScreen` object does not perform any control on the validity of the date entered by the user.

3.8 INPUTSCREEN OBJECT – MULTIPLE INPUT FIELDS (6867I / 6869I / 6873I / 6920 / 6930 / 6940)

The InputScreen object can support up to 10 input fields, each of them potentially having different individual attributes.

The supported input types for each field are the same than the ones supported for the single input object plus an 'empty' type.

For each input field, the following attributes can be defined overriding the ones declared in the main object:

- Type
- Editable
- Password
- Prompt
- Default
- Parameter

Of course, only one URL can be defined to be called when the user has completed his inputs; the label in the parameter tags is appended to the address in the URL tag and sent via HTTP GET. If the Selection tag is used the value of the tag is also appended to the URL tag.

Two display modes are available for the Multiple Input AastraIPPhoneInputScreen:

- Normal: similar aspect to the single input field with a prompt and the input field on 2 separate lines, three (6867i/6869i/6920/6930) or seven (6873i/6940) input fields will be displayed per screen and the user will be able to scroll through them.
- Condensed: the prompt and the input field are on the same line, the prompts being right aligned on the longest prompt. Up to 7 fields are displayed on the same screen depending on the phone model.

3.8.1 IMPLEMENTATION (6867I/6869I/6920/6930)

The ***Left/Right*** keys are used to navigate between characters in the input field.

The ***Up/Down*** keys allow up and down scrolling in order to display the previous/next input field.

When multiple fields are displayed, the ***Select*** key is mapped to go to the next field (like the arrow down key) until the last field is reached where it is then mapped to the doneAction tag (Submit by default).

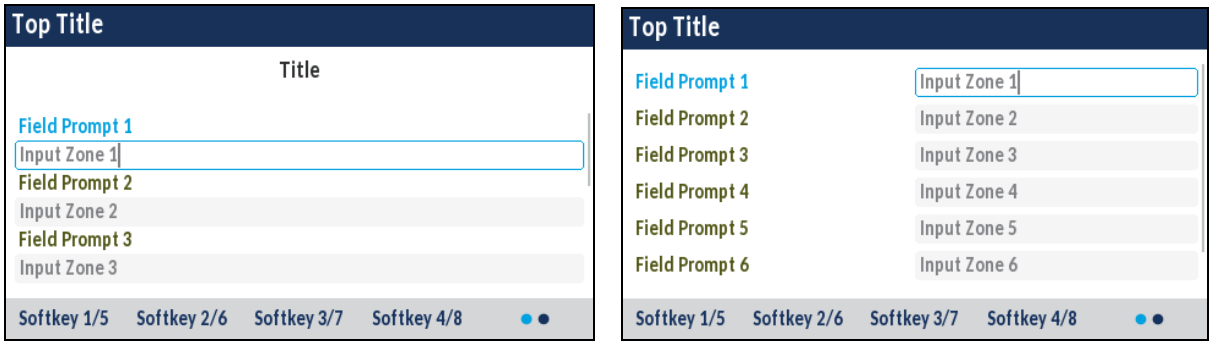


Figure 76: InputScreen implementation on 6869i/6930 "normal" mode

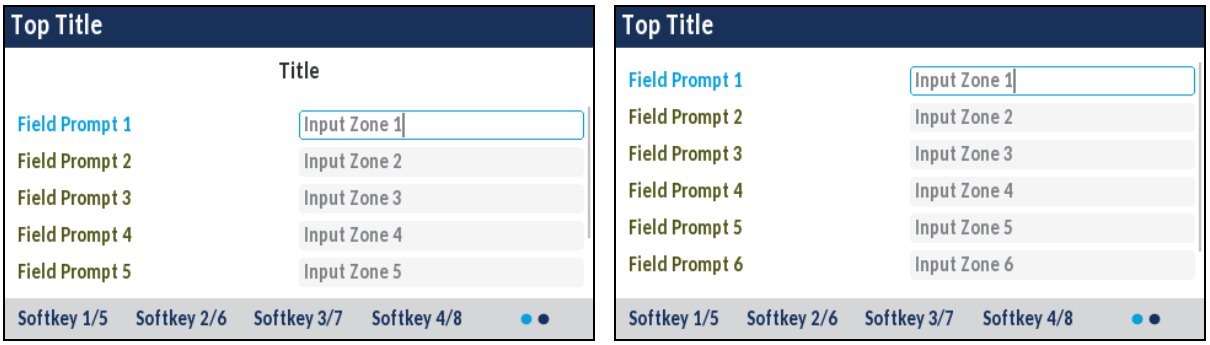


Figure 77: InputScreen implementation on 6869i/6930 "condensed" mode

The number of fields displayed on a single screen depends on the presence of the optional title.

Object native interaction

- ***Done/Submit*** Completes the user input by submitting the programmed URI and value.
- ***Cancel*** Redisplays the previous XML object present in the phone browser

Object default Softkeys (6867i/6920)

Six customizable softkeys are available for this object. If more than 4 softkeys are configured or needed the softkeys are displayed on 2 pages.

Position	Label	Interaction	URIs
3	Done	Completes the user input by submitting the programmed URI and value.	SoftKey:Submit
4	Cancel	Redisplays the previous XML object present in the phone browser. Not available if `LockIn` set to "yes".	SoftKey:Exit

Object default Softkeys (6869i/6930)

Eight customizable softkeys are available for this object. If more than 5 softkeys are configured or needed the softkeys are displayed on 2 pages.

Position	Label	Interaction	URIs
4	Done	Completes the user input by submitting the programmed URI and value.	SoftKey:Submit
5	Cancel	Redisplays the previous XML object present in the phone browser. Not available if `LockIn` set to "yes".	SoftKey:Exit

On the 6867i/6869i, like "Softkey::Submit" the URI custom softkeys are also appended with the parameter values as well as the selection, this allows to exit the InputScreen object without the validity check. This allows for instance to create an input screen with a field which could be a selection amongst a list of values and come back to the screen without losing the user inputs.

3.8.2 IMPLEMENTATION (6873I/6940)

The ***Left/Right*** keys on the virtual keyboard are used to navigate between characters in the input field.

Scrolling between the fields is done swiping the finger on the screen up and down. Scrolling is also available when the virtual keyboard is displayed.

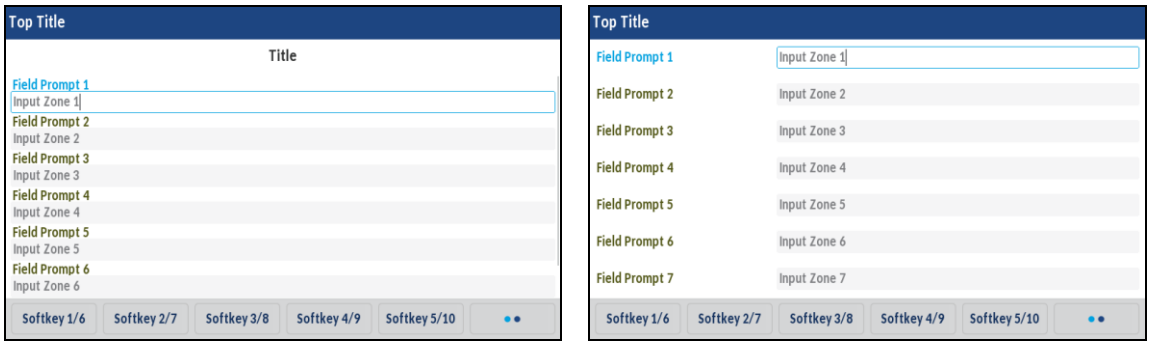


Figure 78: InputScreen implementation on 6873i/6940 "normal" mode

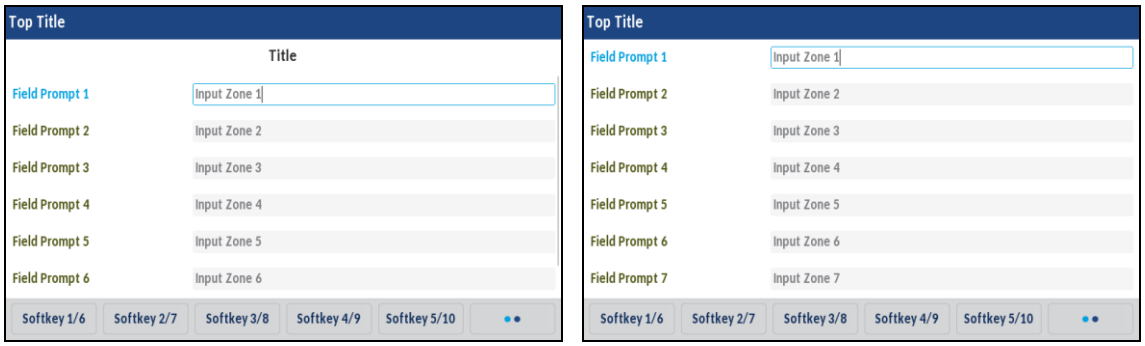


Figure 79: InputScreen implementation on 6873i/6940 "condensed" mode

The number of fields displayed on a single screen depends on the presence of the optional title.

Object native interaction

- ***Done/Submit*** Completes the user input by submitting the programmed URI and value.
- ***Cancel*** Redisplays the previous XML object present in the phone browser

Object default Softkeys

Eight customizable softkeys are available for this object. If more than 5 softkeys are configured or needed the softkeys are displayed on 2 pages.

Position	Label	Interaction	URIs
5	Done	Completes the user input by submitting the programmed URI and value.	SoftKey:Submit
6	Cancel	Redisplays the previous XML object present in the phone browser. Not available if `LockIn` set to "yes".	SoftKey:Exit

On the 6873i/6940, like "Softkey::Submit" the URI custom softkeys are also appended with the parameter values as well as the selection, this allows to exit the InputScreen object without the validity check. This allows for instance to create an input screen with a field which could be a selection amongst a list of values and come back to the screen without losing the user inputs.

3.8.3 XML DESCRIPTION

"Red" tags indicate that the tag does not apply to all phones, when not supported the tags are just ignored.

```
<AastraIPPhoneInputScreen
    type = "IP/string/stringN/number/timeUS/timeInt/dateUS/dateInt"
```

```
    password = "yes/no"
    editable = "yes/no"
    destroyOnExit = "yes/no"
    cancelAction = "some URI"
    Timeout = "some integer"
    bgColor="white/black…"
    LockIn = "yes/no"
    CallProtection = "yes/no/notif"
    GoodbyeLockInURI = "some URI"
    allowAnswer = "yes/no"
    allowDrop = "yes/no"
    allowXfer = "yes/no"
    allowConf = "yes/no"
    defaultIndex = "some integer (1 to 10)"
    defaultFocus = "yes/no"
    displayMode = "normal/condensed"
>
      <Title       wrap="yes/no"
                   Color="white/black…"
      >Title string</Title>
      <TopTitle    icon="icon index"
                   Color="white/black…"
      >Top Title</TopTitle>
      <Prompt      Color="white/black…"
      >Guidance for the input</Prompt>
      <URL>Target receiving the input</URL>
      <Parameter   Color="white/black…"
      >name of the parameter added to URL</Parameter>
      <Default>Default Value</Default>
     <InputField
          type = "IP/string/stringN/number
                  timeUS/timeInt/dateUS/dateInt"
          password = "yes/no"
          editable = "yes/no"
     >
           <Prompt Color="white/black…"
             >Guidance for the input</Prompt>
             <Parameter   Color="white/black…"
      >parameter name added to URL</Parameter>
           <Default>Default Value</Default>
           <Selection>Selection</Selection>
     <!—Additional Softkey Items may be added -->
     </InputField>
<!—Additional Input fields Items may be added -->
<!—Additional Softkey Items may be added -->
<!—Additional Icon Items may be added -->
</AastraIPPhoneInputScreen>
```

XML Document Objects

Document Object	Position	Type	Comments
AastraIPPhoneInputScreen	Root tag	Mandatory	Root object
Type	Root tag	Optional	Specifies the type of input, possible values are "IP", "string","stringN", "number", "timeUS", "timeInt", "dateUS", "dateInt". Default value is "string".
Password	Root tag	Optional	Specifies if the input is masked by "*" characters. Default value

Document Object	Position	Type	Comments
			is “no”
Editable	Root tag	Optional	Specifies if the user is allowed to modify the input. Default value is “yes”
destroyOnExit	Root tag	Optional	“yes/no” indicates if the object is kept or not in the phone browser after exit. If not specified, the object is kept in the browser.
cancelAction	Root tag	Optional	Defines the URI to be called when the user cancels the XML object.
Timeout	Root tag	Optional	Overrides the default 45 seconds timeout of the UI XML object. A Timeout set to “0” will disable the timeout feature. See section 4.10.2 for more details
bgColor	Root tag	Optional	Set the background color of the object. the possible values are detailed in chapter 4.2.3. If not specified, the default value is “white”. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*
LockIn	Root tag	Optional	If set to “yes”, the phone ignores all events that would cause the screen to exit without using the keys defined by the object. Default value is “no”. See section 4.10.3 for more details.
CallProtection	Root tag	Optional	If set to “yes”, the phone will not destroy the XML object being displayed on an incoming call. If set to “notif”, the behavior is the same as “yes” but a notification is displayed on the screen providing the caller ID details. *“notif” is equivalent to “yes” on 6863i and 6865i as screen notification are not supported*.
GoodbyeLockInURI	Root tag	Optional	Valid only if LockIn=”yes”, this tag defines a URI to be called when the “Goodbye” key is pressed during a locked XML session. This URI overrides the native behavior of the “Goodbye” key which is to destroy the current XML object displayed.

Document Object	Position	Type	Comments
allowAnswer	Root tag	Optional	This tag applies only to the non-softkey phones. If set to “yes”, the phone will display “Ignore” and “Answer” if the XML object is displayed when the phone is in the ringing state. Default value is “no”. See section 6.4 for more details. Only for 6863i, 6865i
allowDrop	Root tag	Optional	This tag applies only to the non-softkey phones. If set to “yes”, the phone will display “Drop” if the XML object is displayed when the phone is in the connected state. Default value is “no”. See section 6.4 for more details. *Only for 6863i, 6865i*
allowXfer	Root tag	Optional	This tag applies only to the non-softkey phones. If set to “yes”, the phone will display “Xfer” if the XML object is displayed when the phone is in the connected state. Default value is “no”. See section 6.4 for more details. *Only for 6863i, 6865i*
allowConf	Root tag	Optional	This tag applies only to the non-softkey phones. If set to “yes”, the phone will display “Conf” if the XML object is displayed when the phone is in the connected state. Default value is “no”. See section 6.4 for more details. *Only for 6863i, 6865i*
defaultFocus	Root tag	Optional	If set to “yes”, the first input field defined by defaultIndex automatically goes to edit mode (keyboard displayed). Default value is “no”. *6739i only*
displayMode	Root tag	Optional	If set to “normal” the input fields will be displayed with the prompt and the input field on 2 lines. If set to “condensed” both the prompt and the input field are on the same line. Default value is “normal”
defaultIndex	Root tag	Optional	Defines the field where the user will start his input amongst the multiple field inputs. Default 1.
Title	Body	Optional	Text to be used as title for the object

Document Object	Position	Type	Comments
Wrap	Title tag	Optional	If set to “yes” the title of the object will be wrapped on 2 lines. *Ignored on 6867i, 6869i, 6873i and 6739i*
Color	TopTitle tag	Optional	Label color, the possible values are detailed in chapter 4.2.3. If not specified, the default value is “white”. *6867i, 6869i and 6739i only*
TopTitle	Body	Optional	Text to be used as top title for the object *6867i, 6869i, 6873i and 6739i only*.
Icon	TopTitle tag	Optional	Index of the icon to be used for the top title *6867i, 6869i, 6873i and 6739i only*.
Color	TopTitle tag	Optional	Label color, the possible values are detailed in chapter 4.2.3. If not specified, the default value is “white”. *6867i, 6869i, 6873i and 6739i only*
Prompt	Body	Optional	Text to be displayed as guidance for the user input. Is used as the default value for each input field.
Color	Prompt tag	Optional	Label color, the possible values are detailed in chapter 4.2.3. If not specified, the default value is “yellow”. *6867i, 6869i, 6873i and 6739i only*
URL	Body	Mandatory	URI called when user completes his input.
Parameter	Body	Optional	Name of the parameter to be added to the URL after input is complete. Is used as the default value for each input field.
Color	Parameter tag	Optional	Label color, the possible values are detailed in chapter 4.2.3 If not specified, the default value is “white”. *6867i, 6869i, 6873i and 6739i only*
Default	Body	Optional	Default value to be displayed in the input field. Is used as the default value for each input

Document Object	Position	Type	Comments
			field.
InputField	Body	Optional	
Type	InputField tag	Optional	Specifies the type of input for the field, possible values are “IP”, “string”, “number”, “timeUS”, “timeInt”, “dateUS”, “dateInt” or “empty”. Overrides the value set in the root tag for the field. An “empty” value will create one blank line in condensed mode and 2 blank lines in normal mode.
Password	InputField tag	Optional	Specifies if the input is masked by “*” characters. Default value is “no”. Overrides the value set in the root tag for the field.
Editable	InputField tag	Optional	Specifies if the user is allowed to modify the input. Default value is “yes”. Overrides the value set in the root tag for the field
Prompt	InputField Body	Optional	Text to be displayed as guidance for the user input. Overrides the value set in the object for the field.
Color	Prompt tag	Optional	Label color, the possible values are detailed in chapter 4.2.3. If not specified, the default value is “yellow” for 6867i, 6869i and “white” for 6739i. *6867i, 6869i, 6873i and 6739i only*
Parameter	InputField Body	Optional	Name of the parameter to be added to the URL after input is complete. Overrides the value set in the object for the field.
Color	Parameter tag	Optional	Label color, the possible values are detailed in chapter 4.2.3. If not specified the default value is “white” for 6867i, 6869i and “black” for 6739i *6867i, 6869i, 6873i and 6739i only*
Default	InputField Body	Optional	Default value to be displayed in the input field. Overrides the value set in the object for the field.
Selection	InputField Body	Optional	The content of this tag will be added when the “Submit” key is pressed while editing this field.
SoftKey	Body	Optional	See section 4.1 for details.

Document Object	Position	Type	Comments
			Only on 6867i, 6869i, 6873i, 6739i, 6755i, 6757i, 6757iCT, 6735i and 6737i.

Note: when the `InputField type` is set to `'empty'`, a non editable blank line replaced the input field on the display when the XML object is in condensed mode, 2 blank lines for the normal mode. Also an empty field is a proper field; the `defaultIndex` value must consider the empty field as a plain field.

3.8.4 EXAMPLES

XML Example 1

```
<AastraIPPhoneInputScreen
     type="string"
     destroyOnExit="yes"
     displayMode="condensed"
>
     <Title>Restricted application</Title>
     <URL>http://myserver.com/script.php</URL>
     <Default/>
     <InputField type="empty">
     </InputField>
     <InputField type="string">
          <Prompt>Username:</Prompt>
          <Parameter>user</Parameter>
          <Selection>1</Selection>
     </InputField>
     <InputField type="number" password="yes">
          <Prompt>Password:</Prompt>
          <Parameter>passwd</Parameter>
          <Selection>2</Selection>
     </InputField>
</AastraIPPhoneInputScreen>
```

Note: In this example, when the user press “Submit” on the phone after entering “admin” for the username and “22222” for the password, the phone will call the following URLs:

- http://myserver.com/script.php?user=admin&passwd=22222&selection=1, if the “Submit” or “Enter” key is pressed while editing the “User name” field,
- http://myserver.com/script.php?user=admin&passwd=22222&selection=2, if the “Submit” key or “Enter” is pressed while editing the “Password” field.

Resulting Screen (6869i/6930)

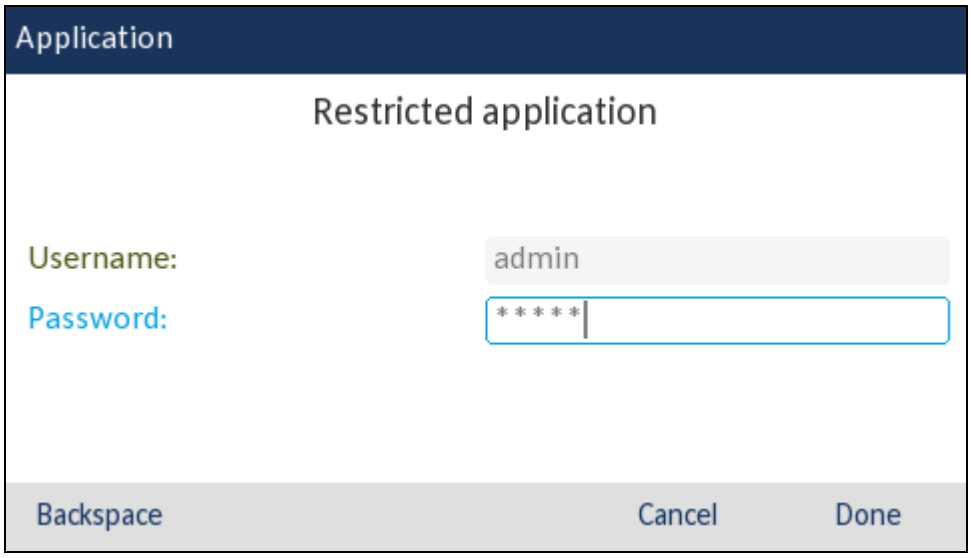


Figure 80: InputScreen multiple inputs "condensed" (6869i/6930)

Resulting Screen (6873i/6940)

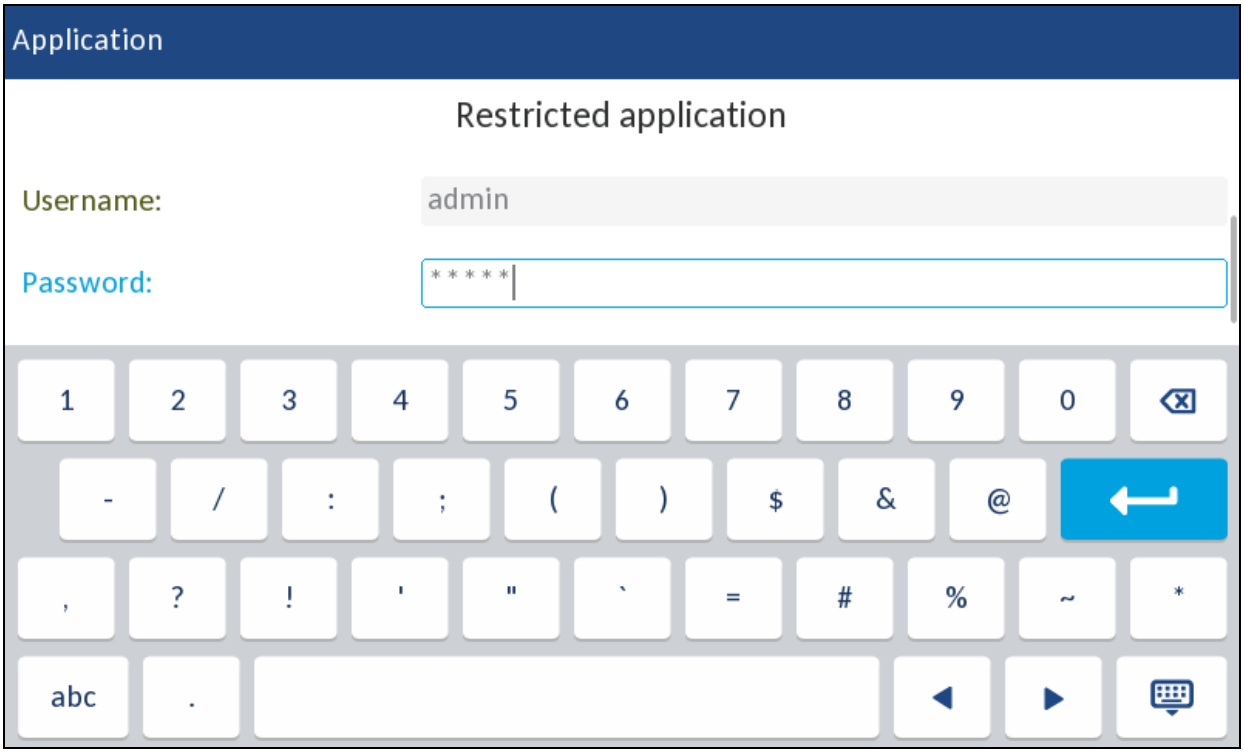


Figure 81: InputScreen multiple inputs "condensed" (6873i/6940)

XML Example 2

```
<AastraIPPhoneInputScreen
     type="string"
     destroyOnExit="yes"
>
     <Title>Date and Time</Title>
     <URL>http://myserver.com/script.php</URL>
     <Default/>
     <InputField type="dateUS">
          <Prompt>Enter Date</Prompt>
          <Parameter>date</Parameter>
     </InputField>
     <InputField type="timeUS">
          <Prompt>Enter Time</Prompt>
          <Parameter>time</Parameter>
     </InputField>
</AastraIPPhoneInputScreen>
```

Note: In this example, when the user press “Done” on the phone after entering “2/21/2010” for the date and “12:00:00 AM” for the time, the phone will call the following URL “http://myserver.com/script.php?date=02/21/2010&time=12:00:00AM”.

Resulting Screen (6869i/6930)

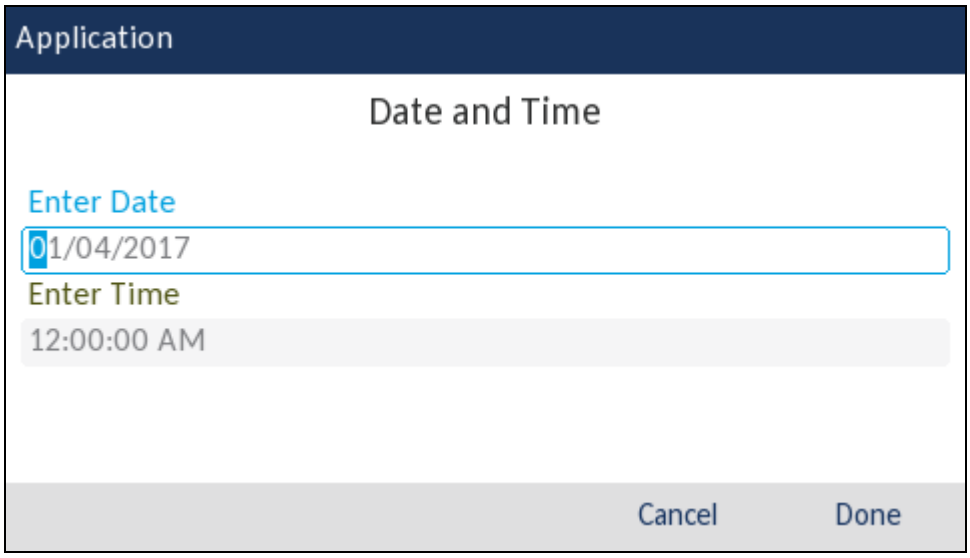


Figure 82: InputScreen multiple inputs “normal” (6869i/6930)

Resulting Screen (6873i/6940)

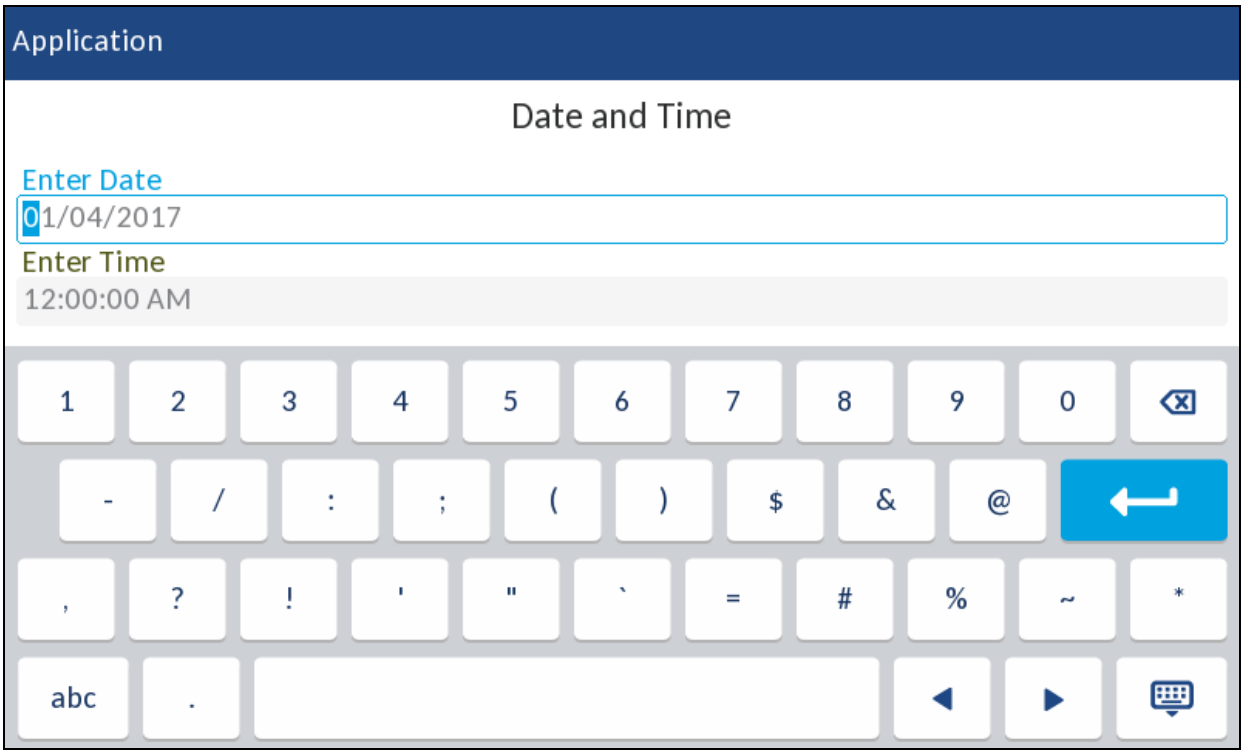


Figure 83: InputScreen multiple inputs “normal” (6873i/6940)

3.9 PHONESTATUS OBJECT

The `AastraIPPhoneStatus` object provides the ability to display a status message or icon on the phone's idle screen when XML information is pushed from the servers.

If the phone receives multiple messages, the first message received displays first and the remaining messages scroll consecutively one at a time.

The AastraIPPhoneStatus object supports 3 modes for the information to be displayed on the idle screen:

- Normal mode: messages remain displayed until they are removed (by the server) or after a phone reboot.
- Alert mode: **one** message is displayed on top of the existing messages only for a limited time (3 seconds by default) or permanently until removed or replaced by another one.
- Icon mode (6867i/6869i/6873i/6920/6930 and 6940 only): **up to 4 icons** are displayed on the top line of the display and remain there until they are removed/replaced (by the server) or after a phone reboot.
- Toaster mode (6867i/6869i/6873i/6920/6930 and 6940 only): the message is displayed in the "toaster" notification which scrolls under the status bar.

Note: You can set the amount of time, in seconds, that a message displays to the phone before scrolling to the next message (See *Scroll delay* below).

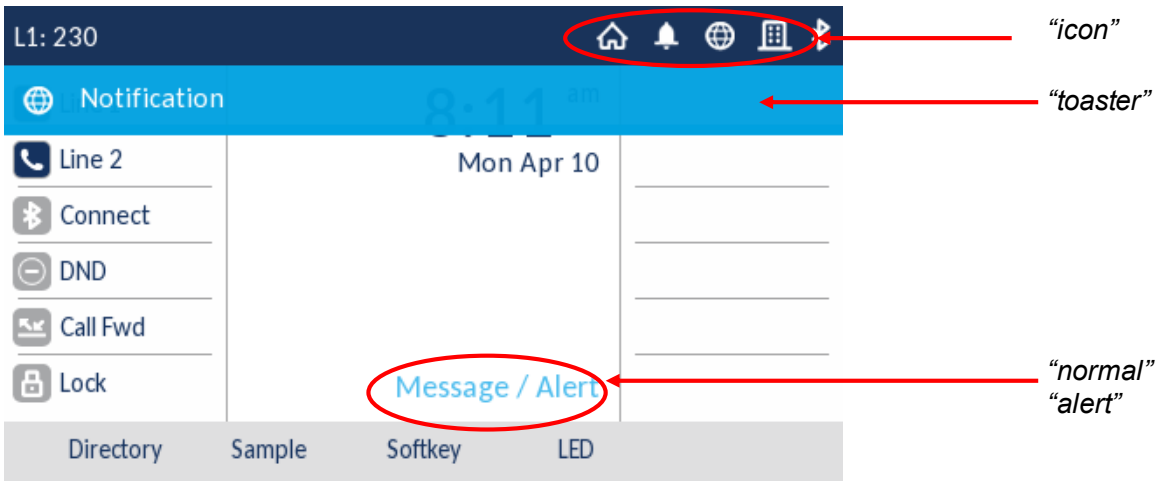
3.9.1 IMPLEMENTATION (6863I/6865I)

The phones display normal and alert messages on the second line. Long messages that are wider then the phone screen get truncated.

3.9.2 IMPLEMENTATION (6867I/6869I/6873I/6920/6930/6940)

The phones display the messages on the second line in the phone idle screen (where "No Service" would be displayed if there was no service. If there is no service on the phone, the "No Service" message overrides the XML object message).

The following graphic shows how the different types of message are displayed.



On the 6873i and 6940 the user can also press the "toaster" notification to execute the URI attached to the message if any.

Note: On the 6867i/6869i/6873i/6920/6930 and 6940 alert messages are displayed in red by default

3.9.3 XML DESCRIPTION

"Red" tags indicate that the tag does not apply to all phones, when not supported the tags are just ignored.

```
<AastraIPPhoneStatus
   Beep = "yes/no"
   triggerDestroyOnExit = "yes/no"
>
       <Session>Session ID</Session>
       <Message
        Index = "index"
       Color="white/black…"
        Type = "normal/alert/icon/toaster"
        Timeout = "timeout"
        URI = "some URI"
       icon = "icon index"
        >Message</Message>
<!—Additional Message Items may be added -->
<!—Additional Icon Items may be added -->
</AastraIPPhoneStatus>
```

Notes:

- The *Session ID* must be unique to the application sending the XML object to the phone. It is up to the application to generate that session ID, which does not have to be limited to just numbers. It could be a combination of letters and numbers. There could only be one Session tag per PhoneStatusMsg object. If the `Session` tag is not provided, the phone assumes a default value (0) for it; this can be used if you don't have multiple applications displaying messages on the idle screen.
- The `type="alert"` tag indicates the alert mode. In this mode a timeout of "0" indicates that the alert message is displayed until a new alert is posted or until the phone is rebooted or a new empty alert is posted. In this mode as only one alert is displayed, the session and index values are ignored.
- The `type="icon"` indicates the icon mode. The four icons are referenced by the index value (0 to 3), the session value is ignored. Dynamic icons are supported.
- The `type="toaster"` indicates the toaster mode. The message is displayed in an animated bar under the phone status bar, the session value is ignored. Dynamic icons are not supported.

The following table details which tags are relevant for each message type.

	Session	Index	Color	Timeout	icon	URI
normal	✓	✓	✓			
alert	✓		✓	✓		
icon		✓			✓	
toaster					✓	✓ (6873i/6940)

Scroll Delay

The Scroll delay can be configured via the configuration files and the Mitel Web UI using the following parameters:

`xml status scroll delay` (via configuration files)

Status Scroll Delay (in seconds) (via the Mitel Web UI see chapter 7.5)

XML Document Objects

Document Object	Position	Type	Comments
AastraIPPhoneStatus	Root tag	Mandatory	Root object
Beep	Root tag	Optional	"yes" or "no" to indicate if a notification beep must be generated by the phone.
triggerDestroyOnExit	Root tag	Optional	If the XML object is sent as an answer to a UI object, setting this parameter to "yes" will trigger the exit of the calling object as if it was a UI object. See section 4.10.4 for more details.
Session	Body	Optional	Session ID used to identify the application displaying message. It allows message change and message reset. Ignored if type is "alert" or "icon"
Message	Body	Mandatory	Message to be displayed or empty to reset the message. In "icon" mode this is the value of the counter to be displayed on the icon if value is superior to zero.
index	Message tag	Mandatory	Index of the message for the session, the index starts at 0. Ignored if type

Document Object	Position	Type	Comments
			is “alert” or “toaster”. If type is “icon” the index must be from 0 to 3 (4 icons).
Color	Prompt tag	Optional	Label color, the possible values are detailed in chapter 4.2.3. If not specified, the default value is “darkgray” for standard messages and “red” for alert messages. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*
type	Message tag	Optional	Type of message, “alert” to indicate that the message is displayed for a limited time. “icon” to indicate an update of the status bar icons. “toaster” to indicate that the message is displayed in the animated notification under the status bar. If not specified, the message stays on the screen until it is reset by an empty message or after a phone reboot. *“icon” and “toaster” values are for 6867i, 6869i, 6873i, 6920, 6930 and 6940 only*
Timeout	Message tag	Optional	Timeout of the “alert” message, overrides the 3s default value. If set to “0” the message is displayed indefinitely.
URI	Message tag	Optional	URI to be called when the user presses the notification message button *6873i and 6940 only*
icon	Message tag	Optional	Index of the icon to be used for this message/alert. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*
IconList	Body	Optional	List of icons used in the object *6867i, 6869i, 6873i, 6920,*

Document Object	Position	Type	Comments
			6930 and 6940 only

3.9.4 EXAMPLES

XML Example 1

```
<AastraIPPhoneStatus Beep="yes">
      <Session>abc12345</Session>
      <Message index="0">Message 1 displayed</Message>
      <Message index="1" type="alert" Timeout="5">Alert displayed</Message>
</AastraIPPhoneStatus>
```

Resulting Screen (6863i/6865i)

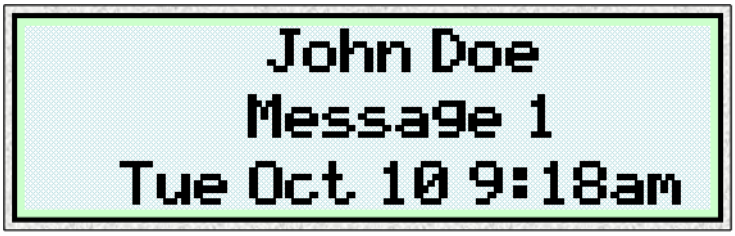


Figure 84: PhoneStatus Example (6863i/6865i)

Resulting Screen (6869i/6930)

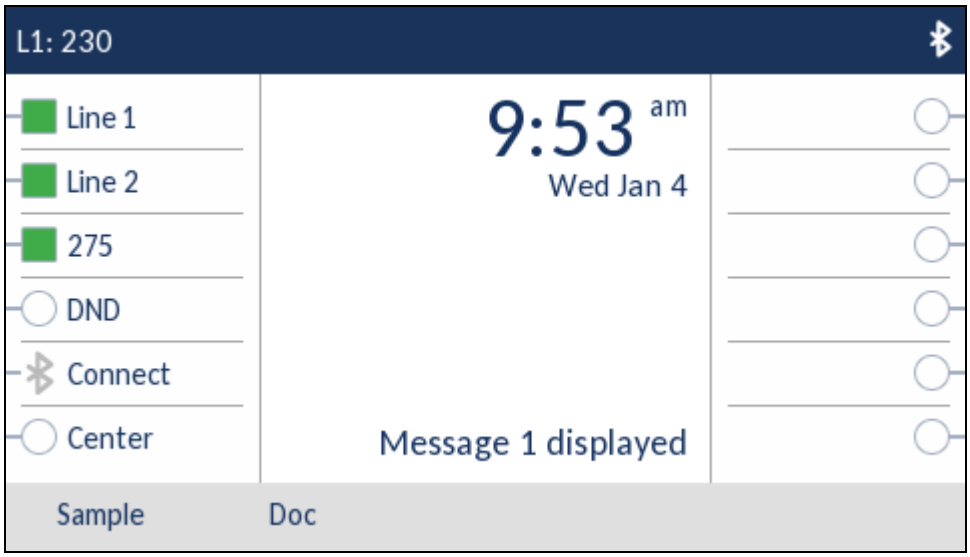


Figure 85: PhoneStatus Example (6869i/6930)

Note: The `PhoneStatus` object can also be used to remove status messages from the display using the same `SessionID` and the same `index`. This can be accomplished by setting an empty tag for the `Message` tag. For example, here is the XML object to remove the previous messages (as the second message was in alert mode it does not remain on the display so no need to remove it).

```
<AastraIPPhoneStatus>
       <Session>abc12345</Session>
       <Message index="0"/>
       <Message index="1"/>
<AastraIPPhoneStatus/>
```

XML Example 2

This XML example displays in the status bar

- the “DND” icon (index 0)
- the “Call Forward” icon (index 1)
- the “Message” icon (index 2) with a counter of 2.

It is critical that the mapping between feature and index remains constant if the XML application is dealing with multiple features.

```
<AastraIPPhoneStatus>
  <Session/>
  <Message index="0" type="icon" icon="1"/>
  <Message index="1" type="icon" icon="2"/>
  <Message index="2" type="icon" icon="3">2</Message>
  <IconList>
    <Icon index="1">Icon:DND</Icon>
    <Icon index="2">Icon:CallFwd</Icon>
    <Icon index="3">Icon:Envelope</Icon>
  </IconList>
</AastraIPPhoneStatus>
```



Figure 86: PhoneStatus icons example

Note (6867i, 6869i, 6873i, 6920, 6930 and 6940): The `PhoneStatus` object can also be used to remove icons from the display using the same `index` but setting the icon to “Icon:None”.

To just remove the DND icon, the following content must be sent

```
<AastraIPPhoneStatus>
       <Message index="0" type="icon" icon="1"></Message>
       <IconList>
              <Icon index=“1”>Icon:None</Icon>
       </IconList>
</AastraIPPhoneStatus>
```

Or

```
<AastraIPPhoneStatus>
       <Message index="0" type="icon"></Message>
</AastraIPPhoneStatus>
```



Figure 87: PhoneStatus icons example

And to just change the number of counter for “Message” (index 2) to the value of 5, the following content must be sent

```
<AastraIPPhoneStatus>
  <Session/>
  <Message index="2" type="icon" icon="1">5</Message>
  <IconList>
    <Icon index="1">Icon:Envelope</Icon>
  </IconList>
</AastraIPPhoneStatus>
```



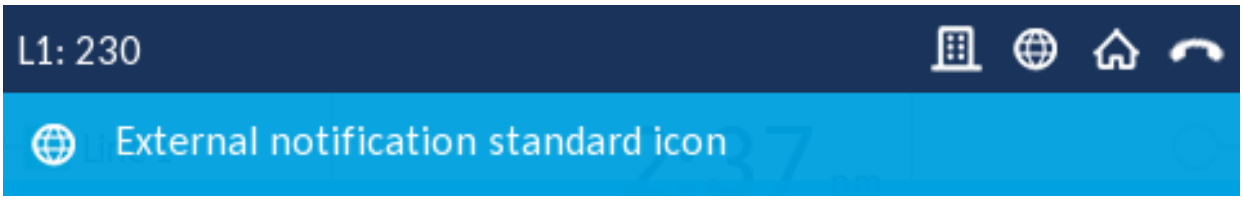
Figure 88: PhoneStatus icons example

XML Example 3

This XML example displays a “toaster” notification using the standard icon and another one with a custom icon.

```
<AastraIPPhoneStatus>
  <Session/>
  <Message type="toaster">External notification standard icon</Message>
  <Message type="toaster" icon="1">External notification custom icon</Message>
  <IconList>
    <Icon index="1">Icon:Home</Icon>
  </IconList>
</AastraIPPhoneStatus>
```

Resulting screen



Followed by

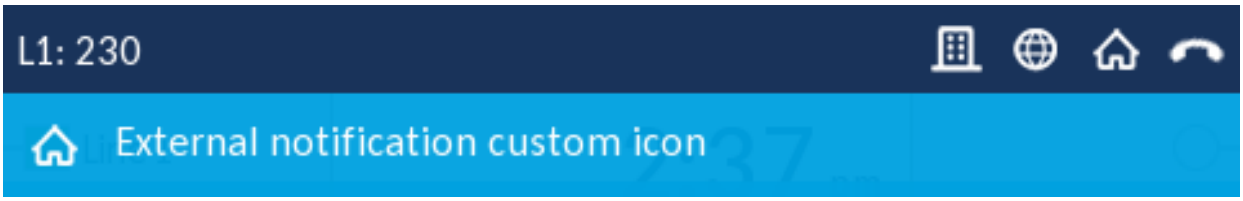


Figure 89: PhoneStatus toaster example

3.10 PHONEEXECUTE OBJECT (ALL MODELS)

The `PhoneExecute` object allows an external application to ask the phone to execute a sequence of local actions using a URI.

3.10.1 IMPLEMENTATION

The actions can be:

- Any supported uri
- http(s)://myserver.com/myscript.pl, you can now use variables in the URI, variables are parsed by the phone.
- Dial:XXXXX
- DialLine:X:YYYYY (X is the SIP line number, YYYYY is the number or URI to dial) see chapter 4.7 for more details on this feature.
- Led: XXXXXX=on/off/slowflash/fastflash:red/yellow/green/active/inactive/1..10 see chapter 4.34.2.3 for more details on the LED control.
- Led: all=on/off/slowflash/fastflash:red/yellow/green/active/inactive/1..10 to control all the phone XML keys see chapter 4.3 for more details on the LED control.
- Key: XXXXX see chapter 4.5 for more details on the simulated keypress feature.
- RTPRx:i:p:v:[mix]:[disableIcon] or RTPRx:Stop to control the reception of a Unicast RTP stream, see chapter 0 for more details.
- RTPTx:i:p or RTPTx:Stop to control the transmission of a Unicast RTP stream, see chapter 0 for more details.
- RTPMRx:i:p:v:[mix]:[disableIcon] to control the reception of a Multicast RTP stream, see chapter 0 for more details.
- RTPMTx:i:p:[mix]:[disableIcon] to control the transmission of a Multicast RTP stream, see chapter 0 for more details.
- Stream a wav file, see chapter 4.6 for more details.
- Download and play a wav file, see chapter 4.7 for more details.
- Phone Reboot (URI="Command: Reset"), the phone will restart (if the phone is idle) and process the complete boot sequence (configuration, language packs, firmware…)
- Phone Fast Reboot (URI="Command: FastReboot"), the phone will restart (if the phone is idle) but will reduce the boot sequence as it will not check for new firmware and will only download language packs if there is a change in supported languages
- Phone Lock (URI="Command: Lock"), this command will lock the phone and will allow only emergency calls.
- Phone Unlock (URI="Command: Unlock"), this command will unlock the phone.
- Clear local configuration and reboot (URI="Command: ClearLocal")
- Clear the callers list and missed calls indicator (URI="Command: ClearCallersList")
- Clear the local directory (URI="Command:ClearDirectory")
- Clear the redial list (URI="Command:ClearRedialList")
- Retrieve crash and configuration file (URI="Command:UploadSystemInfo") , see chapter 4.9 for more details.
- Launch the directory application (URI="Command: LaunchDirectory")
- Launch the callers list application (URI="Command: LaunchCallersList")
- Launch the redial list application (URI="Command: LaunchRedialList")
- Launch the services application (URI="Command: LaunchServices"), 6863i and 6865i only.
- Clear the Mobile directory and Mobile device (URI="Command: ClearMobile")
- Do nothing (URI="")

More actions will be implemented in future firmware versions.

In order to prevent a pushed dial uri from putting an active call on hold the PhoneExecute object supports an optional tag ″interruptCall″. By default, the attribute will allow the current call to be interrupted. To prevent this, set the attribute to “no”.

3.10.2 XML DESCRIPTION

```
<AastraIPPhoneExecute
   Beep                                =                                    "yes/no"
   triggerDestroyOnExit="yes/no"
>
      <ExecuteItem URI="URI" interruptCall="yes/no" title="any string"/>
<!—Additional ExecuteItems may be added -->
</AastraIPPhoneExecute>
```

XML Document Objects

Document Object	Position	Type	Comments
AastraIPPhoneExecute	Root tag	Mandatory	Root object
Beep	Root tag	Optional	“yes” or “no” to indicate if a notification beep must be generated by the phone.
triggerDestroyOnExit	Root tag	Optional	If the XML object is sent as an answer to a UI object, setting this parameter to “yes” will trigger the exit of the calling object as if it was a UI object. See section 4.10.4 for more details.
ExecuteItem	Body	Optional	Tag for the action to be executed
URI	ExecuteItem tag	Optional	URI describing the action to be executed.
interruptCall	ExecuteItem tag	Optional	When the URI is a dial command, if this tags is set to ‘no’ an existing call will not be put on hold.
title	ExecuteItem tag	Optional	WavPlay command the content of this tag is used on the Wav screen instead of the truncated name of the file being played. *6867i, 6869i, 6873i, 6920, 6930 and 6940 only*

3.10.3 EXAMPLES

```
<AastraIPPhoneExecute>
      <ExecuteItem URI=http://myserver.com/myscript.php/>
      <ExecuteItem URI="Dial:12345" interruptCall="no"/>
       <ExecuteItem URI="Command: ClearCallersList"/>
      <ExecuteItem URI="Command: ClearDirectory"/>
      <ExecuteItem URI="Command: ClearRedialList"/>
      <ExecuteItem URI="Command: Reset"/>
</AastraIPPhoneExecute>
```

This example will make the phone execute 6 actions:

- Do an HTTP GET to myserver.com/myscript.php
- Dial 12345 without putting any current call on hold
- Clear the Callers List
- Clear the local directory
- Clear the Redial List
- Reset the phone

Notes:

- the “do nothing” can be used when an application needs to display nothing as an answer to a HTTP GET.
- you must be careful when you use the “Dial:” URI as the state of phone is unknown at the time of the XML GET.
- the FastReboot command will speed up the boot process of the phone which may be useful in the self-configuration application.

3.11 PHONECONFIGURATION OBJECT (ALL MODELS)

The `PhoneConfiguration` object allows an external application to modify the phone configuration dynamically. The configuration parameters are the ones that are used in the configuration files (aastra.cfg and <MAC>.cfg) and detailed in the administrator guide.

3.11.1 IMPLEMENTATION

The phone parameters have 2 levels of precedence: 'server' where the parameters are coming from the configuration files and 'local where the parameters are locally set via the TUI or WebUI. The 'local' parameters override the 'server' parameters.

The AastraIPPhoneConfiguration object supports a parameter called 'setType' which allows defining at which level the parameter change will apply.

- `remote` - parameter will be saved with the same precedence as the server settings. This set will not persist through a reboot.
- `local` - parameter will be saved with the same precedence as TUI or WebUI settings. This set will persist through reboots of the phone. The only way to unset this type is via the TUI, WebUI, or another local type set.
- `oneBoot` - parameter will be saved with a precedence above server settings and below local settings. This parameter will persist through a single reboot and be lost after that. Using this value for setType anywhere in the object will cause the phone to reboot when the `AastraIPPhoneConfiguration` object is done parsing
- `override` - parameter will be saved with the same precedence as server settings. This is different from remote because it will also override a conflicting local value. This override will persist through a reboot but the new value will be lost.

Note: Default value is 'remote' as it was the object behavior prior to firmware release 2.4.0.

If setType is set in the AastraIPPhoneConfiguration root tag, it affects all of the ConfigurationItem sub-elements but when set in the ConfigurationItem tag it will affect that set only (override the global setting for the object).

The number of parameters to be sent in a single `Phoneconfiguration` object is only limited by the overall size of the XML object (10000 bytes). Practically 30 parameters can be sent in a single object.

Note: Not all the configuration parameters are dynamic; specifically, the network parameters are static and need a reboot.

The list of the dynamic configuration parameters is detailed in section 14.

3.11.2 XML DESCRIPTION

```
<AastraIPPhoneConfiguration
   Beep = "yes/no"
   triggerDestroyOnExit = "yes/no"
   setType = "remote/local/oneBoot/override"
>
       <ConfigurationItem setType = "remote/local/oneBoot/override">
              <Parameter
              firstItem="first instance"
              lastItem="last instance"
              >parameter#</Parameter>
Or
              <Parameter>parameter</Parameter>
              <Value>value</Value>
       </ConfigurationItem>
<!—Additional ConfigurationItems may be added -->
</AastraIPPhoneConfiguration>
```

XML Document Objects

Document Object	Position	Type	Comments
AastraIPPhoneConfiguration	Root tag	Mandatory	Root object
Beep	Root tag	Optional	"yes" or "no" to indicate if a notification beep must be generated by the phone.
triggerDestroyOnExit	Root tag	Optional	In case the XML object is sent as an answer to a UI object, setting this parameter to "yes" will trigger the exit of the calling object as if it was a UI object. See section 4.10.4 for more details.
setType	Root tag	Optional	Tag to define the type of configuration change. It applies to all sub-items.
ConfigurationItem	Body	Optional	Tag for the configuration change
setType	ConfigurationItem tag		Tag to define the type of configuration change. It applies only to this item and overrides global policy.
Parameter	ConfigurationItem tag	Mandatory	Configuration parameter to be changed
firstItem	Parameter tag	Optional	These parameters define the range of the parameter values. The parameter must include the character '#' in the name.
lastItem	Parameter tag	Optional	
Value	ConfigurationItem tag	Mandatory	New value of the configuration parameter.

3.11.3 EXAMPLES

XML Example 1: ‘remote’

```
<AastraIPPhoneConfiguration>
   <ConfigurationItem>
      <Parameter>softkey1 label</Parameter>
      <Value>Test</Value>
   </ConfigurationItem>
</AastraIPPhoneConfigutation>
```

or

```
<AastraIPPhoneConfiguration setType="remote">
   <ConfigurationItem>
      <Parameter>softkey1 label</Parameter>
      <Value>Test</Value>
   </ConfigurationItem>
</AastraIPPhoneConfigutation>
```

If `softkey1 label` parameter is unset, set via the configuration files, or set via another `setType="remote"` it will be assigned the value "Test". This value will be lost when the phone reboots. If "softkey1 label" has been set locally, this object will not change it.

XML Example 2: ‘local’

```
<AastraIPPhoneConfiguration setType="local">
   <ConfigurationItem>
      <Parameter>softkey1 label</Parameter>
      <Value>Test</Value>
   </ConfigurationItem>
</AastraIPPhoneConfiguration>
```

The `softkey1 label` parameter will be given the value "Test" whichever way this parameter was set beforeand this value will persist through reboots.

XML Example 3: ‘override’

```
<AastraIPPhoneConfiguration setType="override">
   <ConfigurationItem>
      <Parameter>softkey1 label</Parameter>
      <Value>Test</Value>
   </ConfigurationItem>
</AastraIPPhoneConfiguration>
```

The `softkey1 label` parameter will be set to the value "Test". When the phone reboots, `softkey1 label` will revert to the server configured value or its default of now server configured value is present.

XML Example 4: ‘oneBoot’

```
<AastraIPPhoneConfiguration setType="oneBoot">
   <ConfigurationItem>
      <Parameter>tftp server</Parameter>
      <Value>10.50.103.12</Value>
   </ConfigurationItem>
</AastraIPPhoneConfiguration>
```

If the `tftp server` parameter is unset, set via the configuration files, or another `setType="remote"` it will be assigned the value "10.50.103.12". When this object is done parsing the phone will reboot. Upon booting `tftp server` will still have the value "10.50.103.12". This value will be lost after configuration files have been downloaded from the server.

XML Example 5: Mixed types

```
<AastraIPPhoneConfiguration setType="local">
   <ConfigurationItem setType="oneBoot">
      <Parameter>softkey1 label</Parameter>
      <Value>Test</Value>
   </ConfigurationItem>
   <ConfigurationItem>
      <Parameter>softkey2 label</Parameter>
      <Value>Test</Value>
   </ConfigurationItem>
   <ConfigurationItem setType="override">
      <Parameter>softkey3 label</Parameter>
      <Value>Test</Value>
   </ConfigurationItem>
</AastraIPPhoneConfigutation>
```

Parameter `softkey1 label` will be set using the oneBoot rules (the phone will not reboot yet).
Parameter `softkey2 label` will be set using the local rules.
Parameter `softkey3 label` will be set using the override rules.

When all three items are done being set, the phone reboots as at least one configuration parameter was using the `oneBoot` policy.

XML Example 6: Parameter range

```
<AastraIPPhoneConfiguration>
   <ConfigurationItem>
      <Parameter firstItem=”1” lastItem=”20”>
       softkey# type</Parameter>
      <Value></Value>
   </ConfigurationItem>
</AastraIPPhoneConfiguration>
```

This exemple sets the parameters “softkey1 type” to “softkey20 type” to a blank value. This feature is very useful to limit the number of configuration parameters to be sent.

3.12 PHONESOFTKEY OBJECT (6867I/6869I/6873I/6920/6930/6940)

The `PhoneSoftkey` object allows an external application to modify the configuration of an “xmladvanced” top softkey dynamically. This includes changing icons, label, background color…

3.12.1 IMPLEMENTATION

“xmladvanced” key layout

The following diagram shows the visual elements of the layout for the “xmladvanced” softkey

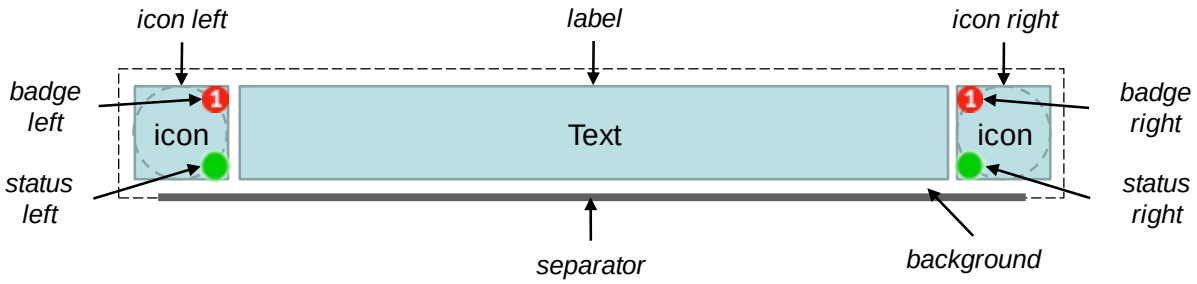


Figure 90: “xmladvanced” key attributes

Dynamic attributes

The attributes are indexed by the top softkey index the same way it is done in the phone configuration file. So the attributes are defined like regular top softkey attributes “topsoftkeyXX attribute” for instance “topsoftkey1 separator”.

Attribute	Description	Default value
label	text or sequences of text to be displayed on the key	Comes from the static configuration
value	URI to be called when the key is pressed	Comes from the static configuration
separator	a Boolean (“0” or ”1”) and indicates if the softkey separator is displayed or not Other values are rejected	“1”
background color	represents the color of the softkey background, possible values are detailed in chapter 4.2.3. an empty value indicates the default background (semi-transparent grey)	empty
label color	represents the color of the label displayed in the softkey, possible values are detailed in chapter 4.2.3 an empty value indicates the default label color (dark blue)	empty
label alignment	allows to set the label alignment, possible values are • left	“left”

Attribute	Description	Default value
	• right • center Other values are rejected	
icon left	allows to set the icon displayed on the left of the softkey, possible values are • “Icon:NoIcon” to reserve the space for an icon, unlike an empty value the label is not extended • “Icon:XXXXXX” to display a canned icon • “Icon:XXXXXX:r” to display a canned icon in reverse mode (dark background) • “Icon location” to display a dynamic icon located on a server • “Picture:XXXX:YY” to display the avatar matching number XXXX from the image server (XXXX.png) or optional initials (YY) picture not available. • An empty value indicates no icon and the label is now extended to the left	Icon:NoIcon
icon right	allows to set the icon displayed on the right of the softkey, possible values are • “Icon:NoIcon” to reserve the space for an icon, unlike an empty value the label is not extended • “Icon:XXXXXX” to display a canned icon • “Icon:XXXXXX:r” to display a canned icon in reverse mode (dark background) • “Icon location” to display a dynamic icon located on a server • “Picture:XXXX:YY” to display the avatar matching number XXXX from the image server (XXXX.png) or optional initials (YY) picture not available. • An empty value indicates no icon and the label is now extended to the right	empty
badge left	Allows to display or hide a notification badge on the top right corner of the left icon (if icon displayed). Values are integers,	0

Attribute	Description	Default value
	• a value strictly superior to 0 displays the value (if value superior to 9 a “!” is displayed) • a value equals to 0 or empty hides the notification badge	
badge right	Allows to display or hide a notification badge at the top left corner of the right icon (if icon displayed). Values are integers, • a value strictly superior to 0 displays the value (if value superior to 9 a “!” is displayed) • a value equals to 0 or empty hides the notification badge	0
status left	Allows to display or hide a status icon at the bottom right corner of the left icon (if icon displayed). Values are labels matching colors: • black, • green, • grey, • orange, • unknown, • red, • yellow, • None	none
status right	Allows to display or hide a status icon at the bottom left corner of the right icon (if icon displayed). Values are labels matching colors: • black, • green, • grey, • orange, • unknown, • red, • yellow, • None	none
button	“button” repositions the touch button on the softkey, possible values are • “left” can press on the left icon • “key” standard mode user can press anywhere on the softkey	“key”

Attribute	Description	Default value
	• "right" user can press on the right icon Other values are rejected. *6873i and 6940 only*	

Notes:

- Dynamic attribute configurations are lost when phone reboots or when the key has a configuration change via WebUI or via an AastraIPPhoneConfiguration command.
- For the "button" attribute, the left and right buttons are only displayed if there is a matching left/right icon defined or a placeholder for it.

Default layout

When a key with type "xmladvanced" is created, this is the default layout

- Icon place holder on the left
- No icon placeholder on the right
- Label left justified and default color
- No background color
- Separator displayed
- No left or right notification badge
- No left or right status
- For 6873i and 6940 the active touch zone is the whole key



"xmladvanced" key uri

The "xmladvanced" key like the "xml" key has a label field and an assigned uri as a value which is called when the button configured for the respective topsoftkey is pressed.

Furthermore, it also supports "Local::ToggleLabel", a special uri value which triggers a local action on the label field display. When configured, pressing the key will sequentially toggle the label display between all configured labels separated by the "|" sign.

In the following example, pressing the label key will sequentially toggle the label display from "Message 1", "Message 2", "Message 3" and then back to "Message 1"

```
topsoftkey1 type: xmladvanced
topsoftkey1 label: Message 1|Message 2|Message 3
topsoftkey1 value: Local::ToggleLabel
```

.

Notes:

- The label attribute is a dynamic parameter that lists the label messages that can also be

updated via the AastraIPPhoneSoftkey object.

- It is not recommended to use AastraIPPhoneConfiguration to change the label and value of an “xmladvanced” typed key as doing this with reinitialize the key to its default layout, all dynamic attributes are lost.

3.12.2 XML DESCRIPTION

```
<AastraIPPhoneSoftkey
   Beep = "yes/no"
   triggerDestroyOnExit = "yes/no"
>
      <SoftkeyItem>
            <Parameter>parameter#</Parameter>
            <Value>value</Value>
      </SoftkeyItem>
<!—Additional SoftkeyItems may be added -->
</AastraIPPhoneConfiguration>
```

The number of parameters to be sent in a single `PhoneSoftkey` object is only limited by the overall size of the XML object (10000 bytes). Practically 30 parameters can be sent in a single object.

XML Document Objects

Document Object	Position	Type	Comments
AastraIPPhoneSoftkey	Root tag	Mandatory	Root object
Beep	Root tag	Optional	“yes” or “no” to indicate if a notification beep must be generated by the phone.
triggerDestroyOnExit	Root tag	Optional	In case the XML object is sent as an answer to a UI object, setting this parameter to “yes” will trigger the exit of the calling object as if it was a UI object. See section 4.10.4 for more details.
SoftkeyItem	Body	Optional	Tag for the configuration change
Parameter	SoftkeyItem tag	Mandatory	Key attribute to be changed
Value	SoftkeyItem tag	Mandatory	New value of the key attribute.

3.12.3 EXAMPLES

Example 1

Configuration

```
topsoftkey1 type: xmladvanced
topsoftkey1 label:
topsoftkey1 value:
```

XML script

```
<AastraIPPhoneSoftkey>
	<SoftkeyItem>
		<Parameter>topsoftkey1 label color</Parameter>
		<Value>white</Value>
	</SoftkeyItem>
	<SoftkeyItem>
		<Parameter>topsoftkey1 label alignment</Parameter>
		<Value>center</Value>
	</SoftkeyItem>
	<SoftkeyItem>
		<Parameter>topsoftkey1 label</Parameter>
		<Value>Center</Value>
	</SoftkeyItem>
	<SoftkeyItem>
		<Parameter>topsoftkey1 background color</Parameter>
		<Value>blue</Value>
	</SoftkeyItem>
	<SoftkeyItem>
		<Parameter>topsoftkey1 icon left</Parameter>
		<Value>Icon:CircleYellow</Value>
	</SoftkeyItem>
	<SoftkeyItem>
		<Parameter>topsoftkey1 icon right</Parameter>
		<Value>Icon:Park:r</Value>
	</SoftkeyItem>
	<SoftkeyItem>
</AastraIPPhoneSoftkey>
```

Resulting screen (6873i/6940)



Example 2

Configuration

```
topsoftkey1 type: xmladvanced
topsoftkey1 label:
topsoftkey1 value:
```

XML script

```
<AastraIPPhoneSoftkey>
	<SoftkeyItem>
		<Parameter>topsoftkey4 icon left</Parameter>
		<Value>Picture:1234</Value>
	</SoftkeyItem>
	<SoftkeyItem>
		<Parameter>topsoftkey4 icon right</Parameter>
		<Value>Picture:5678</Value>
	</SoftkeyItem>
	<SoftkeyItem>
		<Parameter>topsoftkey4 label</Parameter>
		<Value>Nikola||Available</Value>
	</SoftkeyItem>
	<SoftkeyItem>
		<Parameter>topsoftkey4 label alignment</Parameter>
		<Value>left</Value>
	</SoftkeyItem>
```

```
        <SoftkeyItem>
                <Parameter>topsoftkey4 label color</Parameter>
                <Value>black</Value>
        </SoftkeyItem>
        <SoftkeyItem>
                <Parameter>topsoftkey4 separator</Parameter>
                <Value>1</Value>
        </SoftkeyItem>
        <SoftkeyItem>
                <Parameter>topsoftkey4 status right</Parameter>
                <Value>red</Value>
        </SoftkeyItem>
        <SoftkeyItem>
                <Parameter>topsoftkey4 status left</Parameter>
                <Value>green</Value>
        </SoftkeyItem>
</AastraIPPhoneSoftkey>
```

Resulting screen (6873i/6940)



In this script the left icon is a picture associated to 1234 available on image server, and the right icon is a picture associated to 5678 not available on the server, therefore replaced by a colored avatar. Both icons have a status indication.

Example 3

Configuration

```
topsoftkey1 type: xmladvanced
topsoftkey1 label:
topsoftkey1 value:
topsoftkey2 type: xmladvanced
topsoftkey2 label:
topsoftkey2 value: Local::ToggleLabel
```

XML script

```
<AastraIPPhoneSoftkey>
        <SoftkeyItem>
                <Parameter>topsoftkey1 label color</Parameter>
                <Value>black</Value>
        </SoftkeyItem>
        <SoftkeyItem>
                <Parameter>topsoftkey1 label alignment</Parameter>
                <Value>left</Value>
        </SoftkeyItem>
        <SoftkeyItem>
                <Parameter>topsoftkey1 label</Parameter>
                <Value>Label</Value>
        </SoftkeyItem>
        <SoftkeyItem>
                <Parameter>topsoftkey1 background color</Parameter>
                <Value>cyan</Value>
        </SoftkeyItem>
        <SoftkeyItem>
                <Parameter>topsoftkey1 icon left</Parameter>
                <Value>Icon:PhoneConnected</Value>
        </SoftkeyItem>
        <SoftkeyItem>
```

```
            <Parameter>topsoftkey1 separator</Parameter>
            <Value>0</Value>
      </SoftkeyItem>
      <SoftkeyItem>
            <Parameter>topsoftkey1 icon right</Parameter>
            <Value></Value>
      </SoftkeyItem>
      <SoftkeyItem>
            <Parameter>topsoftkey2 label color</Parameter>
            <Value>black</Value>
      </SoftkeyItem>
      <SoftkeyItem>
            <Parameter>topsoftkey2 label alignment</Parameter>
            <Value>left</Value>
      </SoftkeyItem>
      <SoftkeyItem>
            <Parameter>topsoftkey2 label</Parameter>
            <Value>Label 1|Label 2|Label 3</Value>
      </SoftkeyItem>
      <SoftkeyItem>
            <Parameter>topsoftkey2 background color</Parameter>
            <Value>cyan</Value>
      </SoftkeyItem>
      <SoftkeyItem>
            <Parameter>topsoftkey2 icon left</Parameter>
            <Value></Value>
      </SoftkeyItem>
      <SoftkeyItem>
            <Parameter>topsoftkey2 icon right</Parameter>
            <Value>Icon:ArrowRight</Value>
      </SoftkeyItem>
      <SoftkeyItem>
            <Parameter>topsoftkey2 button</Parameter>
            <Value>right</Value>
      </SoftkeyItem>
</AastraIPPhoneSoftkey>
```

Resulting screen (6873i/6940)



In this example the two keys appear to be merged (no separator) and pressing on the second key will toggle between the labels.

4 XML EXTENSIONS

4.1 CUSTOMIZABLE SOFTKEYS ((6867I/6869I/6873I/6920/6930/6940))

The Softkey object can be used to override the default softkeys in each of the XML objects. It allows developers to link arbitrary URIs to keys in the XML screens and invoke softkey behavior native to each XML screen type.

XML Description

6 softkeys are available (8 for the 6869i/6930, 10 for the 6873i/6940)

```
<SoftKey index = "1-10" >
<Label>Text</Label>
<URI>
http(s)://someserver/somepage
OR SoftKey:XXXXXX
OR Dial:somenumber
</URI>
</SoftKey>
<!—As many as used in the softkey definition -->
```

Notes:

- Custom softkeys are only available for the UI XML objects.
- If you use custom softkeys, the default softkeys of the XML object are no longer displayed. This means they have to be recreated as custom softkeys.

XML Document Objects

Document Object	Position	Type	Comments
SoftKey	Body	Mandatory	Softkey Root object (up to 6 or 10 for 6739i)
Index	SoftKey Root tag	Mandatory	Indicates the softkey number
Label	SoftKey Body	Mandatory	Label of the softkey
URI	SoftKey Body	Mandatory	URI called if the softkey is pressed

Available object commands

The following softkey functionality is available to the developer for the purpose of reordering or preserving the default functionality of the XML screens.

- "`SoftKey:Select`" is available for AastraIPPhoneTextMenu only and calls the `URI` tag of the selected `MenuItem`.
- "`SoftKey:Exit`" is available for all UI XML objects and redisplays the previous XML object present in the phone browser.
- "`SoftKey:Dial`" is available to screens that allow input. The dial string for the "Dial" function is taken from the menu items URI on the Menu Screen, and from the editor field input on the Input Screen.
- "`SoftKey:Dial2`" is available only for the AastraIPPhoneTextMenu, AastraIPPhoneTextScreen and AastraIPPhoneFormattedTextScreen objects and will dial the number set by the "`Dial`" tag.
- "`SoftKey:Submit`" is available only for the AastraIPPhoneInputScreen object, it completes the user input by submitting the programmed URI (`URL` tag) and value (`Parameter` tag).
- "`SoftKey:BackSpace`" is available only for the AastraIPPhoneInputScreen object, it deletes the character placed before the cursor.

- "SoftKey:NextSpace" is available only for the AastraIPPhoneInputScreen object, it inserts a "space" character at the cursor position.
- "SoftKey:Dot" is available only for the AastraIPPhoneInputScreen object, it inserts a "." character at the cursor position.
- "SoftKey:ChangeMode" is available only for the AastraIPPhoneInputScreen object, it allows a toggle between lower case, upper case and digit inputs.
- "SoftKey:Answer" is available for all UI XML objects and allows the user to answer an incoming call. This softkey will only be displayed if it is created while the phone is in an incoming state and diseappears when the call is answered or ignored.
- "SoftKey:Ignore" is available for all UI XML objects and allows the user to ignore an incoming call. This softkey will only be displayed if it is created while the phone is in an incoming state and diseappears when the call is answered or ignored.
- "Softkey: Drop" is available for all UI XML objects and allows the user to drop the current call. This softkey will only be displayed if it is created while the phone is in connected state and diseappears when the call is dropped.



Note: Dropping the call using the drop softkey will maintain the current XML display.

"Softkey:Conf" is available for all UI XML objects and allows the user to start a 3-way conference. This softkey will only be displayed if it is created while the phone is in connected state and diseappears when the call is dropped or the conference started.



Note: Using this softkey will destroy the current XML display.

"Softkey:Xfer" is available for all UI XML objects and allows the user to transfer the current call. This softkey will only be displayed if it is created while the phone is in connected state and diseappears when the call is transferred or dropped.



Note: Using this softkey will destroy the current XML display.

"SoftKey:SymbolList="XYZ"" is available only for the AastraIPPhoneInputScreen object, it allows an input of a custom list of characters at the cursor position.

The format is:

```
<SoftKey index="1">
      <Label>Symbols</Label>
      <URI>SoftKey:SymbolList="@."</URI>
</SoftKey>
```

The content of the Symbol List must be encapsulated with quotes.



Note: SymbolList supports only standard ASCII characters.

Note that there can be multiple SymbolList softkeys with different list of symbols. There are some special characters that need to be encoded due to XML limitations (see chapter 2.6 for the encoded value).

Available object commands

"SoftKey:Exit" is available for all UI XML objects and redisplays the previous XML object present in the phone browser.

"`SoftKey:Submit`" is available for AastraIPPhoneInputScreen.

Softkey availability per object

SoftKey	TextScreen	TextMenu	IconMenu	InputScreen
Select		X	X	
Exit	X	X	X	X
Dial		X	X	X
Dial2	X	X	X	
Submit				X
BackSpace				X
NextSpace				X
Dot				X
ChangeMode				X
Answer[1]	X	X	X	X
Ignore[1]	X	X	X	X
Drop[2]	X	X	X	X
Xfer[2]	X	X	X	X
Conf[2]	X	X	X	X
SymbolList				X

SoftKey	Formatted TextScreen	ImageScreen	ImageMenu
Select			
Exit	X	X	X
Dial			
Dial2	X		
Submit			

[1] This softkey will only be displayed if it is created while the phone is in an incoming state.

[2] This softkey will only be displayed if it is created while the phone is in a connected state.

SoftKey	Formatted TextScreen	ImageScreen	ImageMenu
BackSpace			
NextSpace			
Dot			
ChangeMode			
Answer[1]	X	X	X
Ignore[1]	X	X	X
Drop[2]	X	X	X
Xfer[2]	X	X	X
Conf[2]	X	X	X
SoftKey List			

You can define up to six, eight or ten softkeys depending on the phone model before the closing tag of any object.

4.2 GRAPHICS (6867I/6869I/6873I/6920/6930/6940)

4.2.1 IMAGES

4.2.1.1 Implementation on the 6867i/6920

The `image` tag is used by the ImageScreen and ImageMenu XML objects.

Format

The 6867i and 6920 support 24/32 bits depth png or jpeg files, the `image` tag must include the URL to retrieve the png or jpeg file. The supported protocols are:

- TFTP
- FTP
- HTTP
- HTTPS

The area available to display the picture is 320x240 pixels, the phone uses the provided width and height tags as the display zone, if the image sent is bigger than the display zone, the picture is clipped using the top left corner as the origin of the clipping area.

Same thing if the width or height is larger than the supported resolution, the phone will use the minimum value and will clip if needed.

If the phone cannot retrieve the picture using the provided URL, the “unknown” image is displayed.

4.2.1.2 Implementation on the 6869i/6930

The `image` tag is used by the ImageScreen and ImageMenu XML objects.

Format

The 6869i and 6930 support 24/32 bits depth png or jpeg files, the `image` tag must include the URL to retrieve the png or jpeg file. The supported protocols are:

- TFTP
- FTP
- HTTP
- HTTPS

The area available to display the picture is 480x272 pixels, the phone uses the provided width and height tags as the display zone, if the image sent is bigger than the display zone, the picture is clipped using the top left corner as the origin of the clipping area.

Same thing if the width or height is larger than the supported resolution, the phone will use the minimum value and will clip if needed.

If the phone cannot retrieve the picture using the provided URL, the “unknown” image is displayed.

4.2.1.3 Implementation on the 6873i/6940

The `image` tag is used by the ImageScreen and ImageMenu XML objects.

Format

The 6869i and 6940 support 24/32 bits depth png or jpeg files, the `image` tag must include the URL to retrieve the png or jpeg file. The supported protocols are:

- TFTP
- FTP
- HTTP
- HTTPS

The area available to display the picture is 800x480 pixels, the phone uses the provided width and height tags as the display zone, if the image sent is bigger than the display zone, the picture is clipped using the top left corner as the origin of the clipping area.

Same thing if the width or height is larger than the supported resolution, the phone will use the minimum value and will clip if needed.

If the phone cannot retrieve the picture using the provided URL, the “unknown” image is displayed.

4.2.2 ICONS

The 6867i/6869i/6873i/6920/6930/6940 also support icons in some XML objects:

- AastraIPPhoneTextMenu in the menuItem tags
- AastraIPPhoneStatus in the icon tags when type is set to “icon”
- AastraIPPhoneSoftkey for the left and right icons
- All UI objects in TopTitle tag

But not in the Custom softkeys for the UI XML objects.

Three types of icons are supported:

- Predefined icons residing in the phone identified by a unique name
- Dynamic icons identified by a URI, the icon files are downloaded by the phone every time they are needed

- Icons as pictures using Picture:XXXX or Picture:XXXX:YY where XXXX is a picture identification like an extension or a phone number, image will be automatically retrieved from the image server (if “image server uri” configured). YY being the optional initials to be displayed if picture not available, a sngle letter for initials is accepted. YY must be composed with letters [A-Z] or initials are not displayed.

Predefined icons

The predefined icons shown below are directly available from the phone. To use the predefined icons you must set the Icon tag to Icon:IconName where IconName is the name of the predefined icon.

Each XML icon has 2 variants:

- One for the regular UI XML application designed for a white background
- One for the status bar designed for a dark background

The phone automatically selects the appropriate icon except for AatraIPPhoneSoftkey object where the XML app can decide which type of icon to use ie. regular “Icon::XXXXX” or reverse “Icon::XXXXX:r”..

Name	Icon (XML app)	Icon (status bar)	Name	Icon (XML app)	Icon (status bar)
Icon:Add			Icon:OutgoingMissed		
Icon:Alarm			Icon:Park		
Icon:AlarmFilled			Icon:PhoneConnected		
Icon:ArrowDown			Icon:PhoneDial		
Icon:ArrowLeft			Icon:PhoneDiverted		
Icon:ArrowRight			Icon:PhoneDivertedA		
Icon:ArrowUp			Icon:PhoneDivertedB		
Icon:ArrowUpAndDown			Icon:PhoneLockActive		
Icon:BLF_Busy			Icon:PhoneLockInactive		
Icon:BLF_Hold			Icon:PhoneOffHook		
Icon:BLF_Ringing			Icon:PhoneOnHold		
Icon:BLF_Unknown			Icon:PhoneOnHook		

Name	Icon (XML app)	Icon (status bar)	Name	Icon (XML app)	Icon (status bar)
Icon:Book			Icon:PhoneRecall		
Icon:Calendar			Icon:PhoneRinging		
Icon:CallFailed			Icon:PhoneRingingA		
Icon:CallFwd			Icon:PhoneRingingB		
Icon:CallFwdActive			Icon:PresenceAbsent		
Icon:CallFwdInactive			Icon:PresenceAvailable		
Icon:CallTransfer			Icon:PresenceBusy		
Icon:CellPhone			Icon:PresenceMeeting		
Icon:CheckBoxCheck			Icon:PresenceNotAvailable		
Icon:CheckBoxUnCheck			Icon:PresenceSignedOut		
Icon:CircleBlue			Icon:PresenceUnknown		
Icon:CircleGreen			Icon:Prohibit		
Icon:CircleRed			Icon:RecordingOn		
Icon:CircleYellow			Icon:RecordingOff		
Icon:Delete			Icon:Reset		
Icon:DND			Icon:RingTone		
Icon:DndActive			Icon:RoomInspected		
Icon:DndInactive			Icon:RoomNotClean		

Name	Icon (XML app)	Icon (status bar)	Name	Icon (XML app)	Icon (status bar)
Icon:Edit			Icon:RoomNotInspected		
Icon:Envelope			Icon:RoomOccupied		
Icon:EnvelopeOpen			Icon:RoomVacant		
Icon:GroupRoomOccupied			Icon:Save		
Icon:GroupRoomVacant			Icon:Search		
Icon:Home			Icon:Settings		
Icon:Incoming			Icon:Speaker		
Icon:IncomingMissed			Icon:StarBlue		
Icon:Information			Icon:StarYellow		
Icon:Key			Icon:StateActive		
Icon:Lock			Icon:StateActive2		
Icon:MakeBusyActive			Icon:StateInactive		
Icon:MakeBusyInactive			Icon:TailArrowDown		
Icon:MissedCall			Icon:TailArrowUp		
Icon:Mute			Icon:UnLock		
Icon:NightServiceActive			Icon:WakeUpExpired		
Icon:NightServiceInactive			Icon:WakeUpPending		

Name	Icon (XML app)	Icon (status bar)	Name	Icon (XML app)	Icon (status bar)
Icon:Office			Icon:Warning		
Icon:Outgoing			Icon:World		

Icons declaration

Icons are identified by an index in the XML objects; this index refers to the `IconList` tag which must be present each an icon is used. For AastraIPPhoneSoftkey the icon name is directly used.

```
<IconList>
      <Icon index = "iconindex">Icon:Iconname or file location</Icon>
      <Icon index = "iconindex">Icon:Iconname or file location</Icon>
<!—As many as used in the object definition -->
<!-Up to 22 icons, index can be 1 to 21 -->
</IconList>
```

Dynamic icons

Dynamic icons are identified by an URI which is the location of the graphical file on a server.

Icon files must be:

- png files
- resolution 28x28 pixels (6867i/6869i) and 40x40 pixels (6873i), if a larger icon is used, the icon is clipped by the phone to the expected size
- 24 bits or 32 bits depth
- Less than 150 kB

Supported protocols are:

- TFTP
- FTP
- HTTP
- HTTPS

Example

```
<IconList>
  <Icon index="1">http://myserver/icons/icon.png</Icon>
</IconList>
```

Picture icons

Picture icons are identified by an URI defined as “Picture:XXXX” or “Picture:XXXX:YY”. Which translates into the phone downloading

```
[image server uri]/XXXX.png
```

If YY is set, phone displays YY as initials when picture is not available.

Picture icons are available for

- All UI objects for TopTitle icon

- AastraIPPhoneTextMenu all icons
- AastraIPPhoneSoftkey for “icon left” and “icon right”

Example

```
image server uri: http://myserver.com/images

<IconList>
  <Icon index=“1”>Picture :1234</Icon>
</IconList>

Downloads
http://myserver.com/images/1234.png
```

4.2.3 COLORS

The phones support 28 colors which can be used in the XML applications for labels and backgrounds.

The following table provides the XML color map.

Color name	HTML Value	RGB value	Display
white	#FFFFFF	255,255,255	
black	#000000	0,0,0	
darkred	#C51D23	197,29,35	
red	#ED1C24	237,28,36	
lightred	#F26065	242,96,101	
darkgreen	#399041	57,144,65	
green	#3FAC49	63,172,73	
lightgreen	#78C57F	120,197,127	
darkblue	#15325F	21,50,95	
blue	#0081B3	0,129,179	
lightblue	#4CBDEF	76,189,239	
darkyellow	#CDA806	205,168,6	
yellow	#F8CA00	248,202,0	
lightyellow	#FADA4C	250,218,76	
darkgray	#404141	64,65,65	
gray	#BFBFC0	191,191,192	
lightgray	#EDEEEF	237,238,239	
darkbrown	#2B1D0E	43,29,14	

Color name	HTML Value	RGB value	Display
brown	#52361B	40,26,13	
lightbrown	#9F6934	159,105,52	
darkmagenta	#8B008B	139,0,139	
magenta	#FF00FF	255,0,255	
lightmagenta	#FF46FF	255,70,255	
darkcyan	#008B8B	0,139,139	
cyan	#00FFFF	0,255,255	
lightcyan	#E0FFFF	224,255,255	

Figure 91: XML colors (6867i/6869i/6873i/6920/6930/6940)

4.3 LED CONTROL

You can control the status of the LED associated to a phone key **as long as the key is configured as an XML typed "xml" or "xmladvanced" key**.

For the graphical phones (6867i, 6869i and 6873i) the same command gives access to the pseudo-led which is an icon on the key. Even if the 6739i or the 6873i do not have LED attached to the softkeys on its main display, an equivalent feature has been developed to mimic the LED behavior still available on the expansion modules.

"Led: XXXXX=on/off/fastflash/slowflash[:green/yellow/red/active/inactive/integer]"

Where XXXXX represents the key you want to modify the LED or "all" for all the keys.

The first part of the command controls the state of the physical LED attached to a softkey and, if the second part of the command is omitted, a matching color for the pseudo-LED.

The second part of the command supported only on 6867i/6920, 6869i/6930 and 6873i/6940 controls the state of the pseudo-LED by allowing a color (red, green, yellow, active or inactive) or an integer to represent a counter indicator.

Note: the LED state is lost after a phone reboot.

The supported LED states are:

- on, the LED is steady on
- off, the LED is steady off
- slowflash, the LED blinks at a slow pace
- fastflash, the LED blinks at a fast pace

Keys supported on an expansion module (6865i/6867i/6869i/6920/6930/6940)

- M680i expmodX key1 to expmodX key16
- M685i expmodX key1 to expmodX key74
- Where X is the expansion module number (1 to 3)

Keys supported on a 6865i

- prgkey1 to prgkey8

Keys supported on a 6867i/6920

- topsoftkey1 to topsoftkey20
- softkey1 to softkey18
- hardkey1 (L1)
- hardkey2 (L2)
- hardkey3 (Redial)
- hardkey4 (Callers)

Keys supported on a 6869i/6930

- topsoftkey1 to topsoftkey44
- softkey1 to softkey24
- hardkey1 (L1)
- hardkey2 (L2)
- hardkey3 (Redial)
- hardkey4 (Callers)

Keys supported on a 6873i/6940

- topsoftkey1 to topsoftkey48
- softkey1 to softkey30
- hardkey1 (L1)
- hardkey2 (L2)
- hardkey3 (Redial)
- hardkey4 (Callers)

4.3.1 IMPLEMENTATION (6867I/6869I/6920/6930)

Though the 6867i/6869i/6920/6930 has physical LEDs for the topsoftkeys, the LED status is also indicated on the screen via the pseudo-LED. The pseudo-LED also supports a counter notification, a value between 1 and 9 is displayed as is any value over 9 is displayed as “!”.

Notes:

- The pseudo-LED feature is also supported on the M685i
- The counter notification is not supported on the bottom softkeys and on the M680i.
- The counter notification is displayed on top of the LED state

---

The following table shows the mapping between the LED state providing in the command with the physical LED and the pseudo-LED when the pseudo-LED state is not provided.

LED state	LED	Pseudo-LED state	Pseudo-LED (top)	Pseudo-LED (bottom)
“off”	On	“default”		
“on”	Off	“red”		
“slowflash”	Slow flashing	“yellow”		
“fastflash”	Fast Flashing	“yellow”		

The following table shows the pseudo-LED based on its state, when specified the pseudo-LED state overrides the state mapping coming from the LED state except for the counter notification which is displayed on top on the LED state.

Pseudo-LED state	Pseudo-LED (top)	Pseudo-LED (bottom)
“default”		
“red”		
“green”		
“yellow”		
“active”		
“inactive”		
“on:1” to “on:9”		N/A

Pseudo-LED state	Pseudo-LED (top)	Pseudo-LED (bottom)
	… 	
“on:10+”		N/A
“slowflash:1” to “slowflash:9”	 … 	N/A
“slowflash:10+”		N/A
“fastflash:1” to “fastflash:9”	 … 	N/A
“fastflash:10+”		N/A

4.3.2 IMPLEMENTATION (6873I/6940)

As the 6873i and the 6940 have no physical LED attached to the softkeys, the LED status is indicated on the screen via the pseudo-LED which blinks at the requested rate instead of showing a different color. The pseudo-LED also supports a counter notification, a value between 1 and 9 is displayed as is any value over 9 is displayed as “!”.

Notes:

- The pseudo-LED feature is also supported on the M685i
- The counter notification is not supported on the bottom softkeys and on the M680i.
- The counter notification is displayed on top of the LED state

The following table shows the mapping between the LED state providing in the command with the physical LED and the pseudo-LED when the pseudo-LED state is not provided.

LED state	Pseudo-LED state	Icon/Border	Pseudo-LED (top)	Pseudo-LED (bottom)
“off”	“default”	On		

LED state	Pseudo-LED state	Icon/Border	Pseudo-LED (top)	Pseudo-LED (bottom)
“on”	“red”	Steady		
“slowflash”	“red”	Slow blink		
“fastflash”	“red”	Fast blink		

The following table shows the pseudo-LED based on its state, when specified the pseudo-LED state overrides the state mapping coming from the LED state.

Pseudo-LED state	Pseudo-LED (top)	Pseudo-LED (bottom)
“default”		
“red”		
“green”		
“yellow”		
“active”		
“inactive”		
“on:1” to “on:9”	 …	N/A

Pseudo-LED state	Pseudo-LED (top)	Pseudo-LED (bottom)
		
“on:10+”		N/A
“slowflash:1” to “slowflash:9”	 … 	N/A
“slowflash:10+”		N/A
“fastflash:1” to “fastflash:9”	 … 	N/A
“fastflash:10+”		N/A

4.3.3 XML EXAMPLES

```
<AastraIPPhoneExecute>
   <ExecuteItem URI="Led: softkey1=on"/>
   <ExecuteItem URI="Led: softkey1=off"/>
   <ExecuteItem URI="Led: prgkey2=fastflash"/>
   <ExecuteItem URI="Led: expmod2 key20=slowflash"/>
   <ExecuteItem URI="Led: all=off"/>
</AastraIPPhoneExecute>
```

On the 6867i/6869i/6873i/6920/6930/6940 pseudo-LED can also be controlled.

```
<AastraIPPhoneExecute>
   <ExecuteItem URI="Led: softkey1=on:green"/>
   <ExecuteItem URI="Led: topsoftkey1=off:1"/>
   <ExecuteItem URI="Led: all=on:99"/>
</AastraIPPhoneExecute>
```

The following example shows the difference in pseudo-LED behavior between a 6867i/6869i/6920/6930 and a 6873i/6940

```
<AastraIPPhoneExecute>
   <ExecuteItem URI="Led: softkey1=slowflash"/>
   <ExecuteItem URI="Led: topsoftkey1=fastflash"/>
</AastraIPPhoneExecute>
```

Results on screen

Phone	Key	LED	Icon	Key
6867i 6869i 6920 6930	topsoftkey1	Fast flashing	Steady	
	softkey1	N/A	Steady	
6873i 6940	topsoftkey1	N/A	Fast blink	
	softkey1	N/A	Slow blink	

4.4 RTP STREAMING

The Mitel SIP Phones have XML commands to use with the `AastraIPPhoneExecute` object, these commands allow the phone to send/receive an RTP stream to/from any given multicast/unicast addresses (without involving any SIP signaling).

The XML commands to use with the `AastraIPPhoneExecute` object in an XML application are:

- **RTPRx** Receive (Rx) a Unicast RTP stream or stop receiving a Unicast/Multicast RTP streams
- **RTPTx** Transmit (Tx) a Unicast RTP stream or stop transmitting a Unicast/Multicast RTP streams
- **RTPMRx** Receive (Rx) a Multicast RTP stream
- **RTPMTx** Transmit (Tx) a Multicast RTP stream

Note: Though the RTP commands could be considered as URIs they can not be used directly as a URI, they must be launch using a `AastraIPPhoneExecute` command.

The phones support **only** the following RTP stream format:

- G.711 µ-law Codec
- 20 ms packet size

Note: If one of the previous commands is sent to phone which is already in a communication, current call is placed on hold and the RTP streaming commands are performed using a new audio path.

Each RTP streaming command supports 2 modes:

- With audio path mixing
- Without audio path mixing

When the RTP streaming command is sent to the phone with an existing audio path (the phone is in the “connected” state), depending on the mixing parameter, the phone will behave differently.

With the audio mixing set to on, the command will apply on the existing audio path, for instance, sending a RTPTx will “add” the RTP streaming to the audio path. The typical use is for “agent whisper” in a contact center environment, “page whisper”, call recording…

With the audio mixing set to off, the command will create a new audio path putting the initial call on hold, for instance sending a RTPTx will create a new audio path. The typical use is for paging.

Notes: Mixing is not supported:

when both the cordless handset (57iCT/9480iCT) and the base are connected. 57i/9480iCT.

when the phone is already in a 3-way conference.

4.4.1 RTPRX

The RTPRx URI instructs the phone to receive a Unicast RTP stream or stop receiving Unicast or Multicast RTP streams. The RTPRx formats to use with the `AastraIPPhoneExecute` XML object in the URI are:

```
RTPRx:i:p:[v]:[mix]:[disableIcon]
RTPRx:Stop
RTPRx:Resume
```

Where:

- i specifies the IP Address from which the stream is coming.
- p specifies the UDP port on which to receive the RTP stream. You must ensure that this is a number greater than 3100 to make sure not to use a port already bound to the phone.
- v (optional) indicates the optional volume setting that controls the volume of the stream playout. The supplied value is an offset to the current volume setting. After the initial volume level gets set and the stream starts, you can manually change the volume level as required using this "v" option. If you do not specify the optional volume parameter, the phone uses the current volume setting on the phone as the default.
- mix (optional) specifies that the incoming RTP stream will be added to the existing audio stream if it exists.
- disableIcon (optional) specifies that the "mix" icon is not displayed (if "mix" is configured in the command).

"Stop" specifies to stop any active custom RTP stream from being received.

"Resume" requests the phone to resume the last RTPRx stream after user dropped it.

Scenarios for "mix" RTP

Phone State	Action
Phone is in "idle" state	Phone initiates a new RTP session on the paging line and the paging line is displayed.
Phone is in "connected" state	Phone starts playing the incoming RTP stream on top of the existing call. Paging line is not displayed.
Phone is in a 3-way conference.	Request for receving RTP is declined.
57iCT/9480iCT, base and handsets are in a "connected" state.	Request for receving RTP is declined.
57iCT/9480iCT, handset is in "connected" state.	Phone starts playing the incoming RTP stream on top of the existing call. Paging line is not displayed.
The active voice call is dropped (RTP stream was being played on top of this voice call).	RTP stream is dropped as well.
A new call comes in while the active voice call (RTP stream was being played on top of this voice call) is put on hold.	RTP mixed stream is played on top of the currently active call.

Example1:

This example orders the phone to receive a unicast RTP stream from 10.30.100.20 at UDP port 21000 with the voice settings at 3 levels more than the current offset.

```
<AastraIPPhoneExecute>
<ExecuteItem URI = ”RTPRx:10.30.100.20:21000:3”/>
</AastraIPPhoneExecute>
```

Example 2:

This example orders the phone to receive a unicast RTP stream from 10.30.100.20 at UDP port 21000 using the current voice settings.

```
<AastraIPPhoneExecute>
<ExecuteItem URI = ”RTPRx:10.30.100.20:21000”/>
</AastraIPPhoneExecute>
```

Example 3:

This example orders the phone to receive a unicast RTP stream from 10.30.100.20 at UDP port 21000 using the current voice settings and audio mixing.

```
<AastraIPPhoneExecute>
<ExecuteItem URI = ”RTPRx:10.30.100.20:21000:mix”/>
</AastraIPPhoneExecute>
```

Example 4:

This example orders the phone to receive a unicast RTP stream from 10.30.100.20 at UDP port 21000 with the voice settings at 3 levels more than the current offset, audio mixing and disabling the mixing icon.

```
<AastraIPPhoneExecute>
<ExecuteItem URI = ”RTPRx:10.30.100.20:21000:3:mix:disableIcon”/>
</AastraIPPhoneExecute>
```

Example 5:

This example orders the phone to stop an incoming active multicast/unicast RTP stream.

```
<AastraIPPhoneExecute>
<ExecuteItem URI=”RTPRx:Stop”/>
</AastraIPPhoneExecute>
```

Note: Once the RTPRx command in the URI is sent to the phone, it stops the phone from listening to any previous RTPRx or RTPMRx commands. The phone starts listening based on the most recent RTPRx command received. This behavior also applies to the RTPRx:Stop and RTPMRx:Stop commands as well but does not enable any further listening.

4.4.2 RTPTX

The RTPTx URI instructs the phone to transmit a Unicast RTP stream or to stop transmitting Unicast or Multicast RTP streams. The RTPTx formats to use with the `AastraIPPhoneExecute` object in the URI are:

```
RTPTx:i:p:[mix]:[disableIcon]
RTPTx:Stop
```

Where

- i specifies the IP Address to which an RTP stream is transmit ed.
- p specifies the UDP port on which to transmit the RTP stream. You must ensure that this is a number greater than 3100 to make sure not to use a port already bound to the phone.

- mix (optional) specifies that the RTP stream sent will be the existing audio stream if it exists.
- disableIcon (optional) specifies that the “mix” icon is not displayed (if “mix” is configured in the command).

Stop specifies to stop any active custom RTP stream from being received.

Scenarios for “mix” RTP

Phone State	Action
Phone is in “idle” state	Phone initiates a new RTP session on the paging line and the paging line is displayed.
Phone is in “connected” state	Phone starts sending the mixed RTP stream. Paging line is not displayed.
Phone is in a 3-way conference.	Request for sending RTP is declined.
57iCT/9480iCT, base and handsets are in a “connected” state.	Request for sending RTP is declined.
57iCT/9480iCT, handset is in “connected” state.	Paging call is initiated using the voice stream with the mixed audio from the conference stream, paging line is not displayed.
The active voice call is dropped (RTP stream was being played on top of this voice call).	RTP stream is dropped as well.
A new call comes in while the active voice call (RTP stream was being played on top of this voice call) is put on hold.	RTP mixed stream is sent for the current active call.

Example 1:

This example orders the phone to send a unicast RTP stream to 10.30.100.20 on UDP port 21000.

```
<AastraIPPhoneExecute>
<ExecuteItem URI = ”RTPTx:10.30.100.20:21000”/>
</AastraIPPhoneExecute>
```

Example 2:

This example orders the phone to send a unicast RTP stream to 10.30.100.20 on UDP port 21000 with audio mixing.

```
<AastraIPPhoneExecute>
<ExecuteItem URI = ”RTPTx:10.30.100.20:21000:mix”/>
</AastraIPPhoneExecute>
```

Example 3:

This example orders the phone to send a unicast RTP stream to 10.30.100.20 on UDP port 21000 with audio mixing and “mix” icon disabled.

```
<AastraIPPhoneExecute>
<ExecuteItem URI = ”RTPTx:10.30.100.20:21000:mix:disableIcon”/>
</AastraIPPhoneExecute>
```

Example 4:

This example orders the phone to stop an outgoing active multicast/unicast RTP stream.

```
<AastraIPPhoneExecute>
<ExecuteItem URI="RTPTx:Stop"/>
</AastraIPPhoneExecute>
```

4.4.3 RTPMRX

The RTPMRx URI instructs the phone to receive a Multicast RTP stream. The RTPMRx format to use with the `AastraIPPhoneExecute` object in the URI is:

```
RTPMRx:i:p:[v]:[mix]:[disableIcon]
RTPMRx:Stop
RTPMRx:Resume
```

Where

- i specifies the multicast IP Address from which to receive an RTP stream.
- p specifies the UDP port on which to receive the RTP stream. You must ensure that this is a number greater than 3100 to make sure not to use a port already bound to the phone.
- v (optional) indicates the optional volume setting that controls the volume of the stream playout. The supplied value is an offset to the current volume setting. After the initial volume level gets set and the stream starts, you can manually change the volume level as required using this "v" option. If you do not specify the optional volume parameter, the phone uses the current volume setting on the phone as the default.
- mix (optional) specifies that the incoming RTP stream will be added to the existing audio stream if it exists.
- disableIcon (optional) specifies that the "mix" icon is not displayed (if "mix" is configured in the command).

"Stop" specifies to stop any active custom RTP stream from being received.

"Resume" requests the phone to resume the last RTPRx or RTPMRx stream after user dropped it.

Scenarios for "mix" RTP

Phone State	Action
Phone is in "idle" state	Phone initiates a new RTP session on the paging line and the paging line is displayed.
Phone is in "connected" state	Phone starts playing the incoming RTP stream on top of the existing call. Paging line is not displayed.
Phone is in a 3-way conference.	Request for receving RTP is declined.
57iCT/9480iCT, base and handsets are in a "connected" state.	Request for receving RTP is declined.
57iCT/9480iCT, handset is in "connected" state.	Phone starts playing the incoming RTP stream on top of the existing call. Paging line is not displayed.
The active voice call is dropped (RTP stream was being played on top of this voice call).	RTP stream is dropped as well.
A new call comes in while the active voice call (RTP stream was being played on top of this voice call) is put on hold.	RTP mixed stream is played on top of the currently active call.

Example 1:

This example orders the phonr to receive a multicast RTP stream from 239.0.1.20 on UDP port 21000 with the voice settings at -3 levels less than the current offset.

```
<AastraIPPhoneExecute>
<ExecuteItem URI = "RTPMRx:239.0.1.20:21000:-3"/>
</AastraIPPhoneExecute>
```

Example 2:

This example orders the phone to receive a multicast RTP stream from 239.0.1.20 at UDP port 21000 with the current voice settings.

```
<AastraIPPhoneExecute>
<ExecuteItem URI = "RTPMRx:239.0.1.20:21000"/>
</AastraIPPhoneExecute>
```

Example 3:

This example orders the phone to receive a multicast RTP stream from 239.0.1.20 at UDP port 21000 with the current voice settings and audio mixing.

```
<AastraIPPhoneExecute>
<ExecuteItem URI = "RTPMRx:239.0.1.20:21000:mix"/>
</AastraIPPhoneExecute>
```

Example 4:

This example orders the phone to receive a multicast RTP stream from 239.0.1.20 at UDP port 21000 with the current voice settings and audio mixing with the "mix" icon disabled.

```
<AastraIPPhoneExecute>
<ExecuteItem URI = "RTPMRx:239.0.1.20:21000:mix:disableIcon"/>
</AastraIPPhoneExecute>
```

Note: Once the RTPMRx command in the URI is sent to the phone, it stops the phone from listening to any previous RTPRx or RTPMRx commands. The phone starts listening based on the most recent RTPMRx command received. This behavior also applies to the RTPRx:Stop and RTPMRx:Stop commands as well, but tbe "Stop" command does not enable any further listening.

4.4.4 RTPMTX

The RTPMTx URI instructs the phone to transmit a Multicast RTP stream. The RTPMTx format to use with the AastraIPPhoneExecute object in the URI is:

```
RTPMTx:i:p:[mix]:[disableIcon]
```

Where

- i specifies the Multicast IP Address to which an RTP stream is transmitted.
- p specifies the UDP port on which to transmit the RTP stream. You must ensure that this is a number greater than 3100 to make sure not to use a port already bound to the phone.
- mix (optional) specifies that the RTP stream sent will be the existing audio stream if it exists.
- disableIcon (optional) specifies that the "mix" icon is not displayed (if "mix" is configured in the command).

Scenarios for "mix" RTP

Phone State	Action
Phone is in "idle" state	Phone initiates a new RTP session on the paging line and the paging line is displayed.
Phone is in "connected" state	Phone starts sending the mixed RTP stream. Paging line is not displayed.

Phone State	Action
Phone is in a 3-way conference.	Request for sending RTP is declined.
57iCT/9480iCT, base and handsets are in a “connected” state.	Request for sending RTP is declined.
57iCT/9480iCT, handset is in “connected” state.	Paging call is initiated using the voice stream with the mixed audio from the conference stream, paging line is not displayed.
The active voice call is dropped (RTP stream was being played on top of this voice call).	RTP stream is dropped as well.
A new call comes in while the active voice call (RTP stream was being played on top of this voice call) is put on hold.	RTP mixed stream is sent for the current active call.

Example 1:

This example orders the phone to send a multicast RTP stream to 239.0.1.20 from UDP port 21000.

```
<AastraIPPhoneExecute>
<ExecuteItem URI = ”RTPMTx:239.0.1.20:21000”/>
</AastraIPPhoneExecute>
```

Example 2:

This example orders the phone to send a multicast RTP stream to 239.0.1.20 from UDP port 21000 with audio mixing.

```
<AastraIPPhoneExecute>
<ExecuteItem URI = ”RTPMTx:239.0.1.20:21000:mix”/>
</AastraIPPhoneExecute>
```

Example 3:

This example orders the phone to send a multicast RTP stream to 239.0.1.20 from UDP port 21000 with audio mixing and “mix” icon disabled.

```
<AastraIPPhoneExecute>
<ExecuteItem URI = ”RTPMTx:239.0.1.20:21000:mix:disableIcon”/>
</AastraIPPhoneExecute>
```

4.4.5 INTERACTION WITH ACTION URIS

action uri incoming

The action uri incoming is not triggered when the phone starts streaming RTP. This is a known issue which will be fixed in a future firmware release.

action uri connected

The action uri connected is not triggered when the phone starts streaming RTP. This is a known issue which will be fixed in a future firmware release.

action uri onhook

The action uri onhook is triggered at the end of the RTP streaming but the URL variables are not bound. This is a known issue which will be fixed in a future firmware release.

action uri disconnected

The action uri disconnected is not triggered at the end of theRTP streaming. This is a known issue which will be fixed in a future firmware release.

4.5 KEYPRESS EMULATION

The Mitel SIP Phones support a “Key” URI to allow the XML application to send an event that a key has been pressed on the phone. The system initiates the event as if the button was physically pressed.

The phone maps key events to the phone physical keys and not to their mapped logical keys.

The exceptions to this rule are

- Xfer,
- Icom,
- Conf

as they can be dynamically mapped on the phone.

Notes:

- If the key is a valid key, the phone executes the key regardless of the current phone status.
- Phone does not send a response detailing the success or failure of a simulated key.

Lines

• Key:Line1 to Key:Line4	Line 1 to 4

Note: Since no Mitel phone has Line 5 to 9 physical keys, Line5 to Line9 are not supported.

Keypad

• Key:KeyPad0 to Key:KeyPad9	Numeric Keypad buttons 0-9
• Key:KeyPadStar	*Star button
• Key:KeyPadPound	# Hash button

Programmable keys (depending on model)

• Key:SoftKey1 to Key:SoftKey55	soft keys 1-55
• Key:PrgKey1 to Key:PrgKey11	programmable keys 1-11
• Key:TopSoftKey1 to Key:TopSoftKey10	top softkeys 1-10
• Key:ExpMod1SoftKey1 to Key:ExpMod1SoftKey60	exp. module 1 keys 1 to 60
• Key:ExpMod2SoftKey1 to Key:ExpMod2SoftKey60	exp. module 2 keys 1 to 60
• Key:ExpMod3SoftKey1 to Key:ExpMod3SoftKey60	exp. module 3 keys 1 to 60

Expansion modules pages (M685i and M695)

• Key:ExpMod1Page1 to ExpMod1Page3	exp module 1 page 1 to 3
• Key:ExpMod2Page1 to ExpMod2Page3	exp module 2 page 1 to 3
• Key:ExpMod3Page1 to ExpMod3Page3	exp module 3 page 1 to 3

Note: The phone ignores expansion module keys that are not present on the expansion module.

Volume and Voice path control

- Key:VolDwn — Decrease button
- Key:VolUp — Increase button
- Key:Headset — Headset
- Key:Speaker — Speaker
- Key:Mute — Mute

Note: for both keys (headset and speaker) the behaviour is as if the "speaker/headset" button is pressed, and it does not switch to headset for “Headset” key event or to speaker for “Speaker” key event.

Telephony

- Key:Xfer — Transfer
- Key:Conf — Conference
- Key:Hold — Hold
- Key:Redial — Redial
- Key:Callers — Callers List
- Key:Services — Services
- Key:Intercom — Intercom
- Key:Directory — Directory
- Key:Options — Options
- Key:Save — Save
- Key:Delete — Delete
- Key:Goodbye — GoodBye
- Key:Voicemail — Voicemail

Note: Since the phones do not have physical keys for Pickup and Park, those features will be only available if they are mapped to a programmable or soft key

Navigation (not available on 6873i and 6940)

- Key:NavUp — Navigation up key
- Key:NavDwn — Navigation down key
- Key:NavEnter — Navigation center key (Enter)
- Key:NavLeft — Navigation left key
- Key:NavRight — Navigation right key

Example:

This example asks the phone to display the phone directory.

```
<AastraIPPhoneExecute>
<ExecuteItem URI = ”Key:Directory”/>
</AastraIPPhoneExecute>
```

4.6 WAV FILE STREAMING

The phone has the capability to stream a wav file from a TFTP or a HTTP server. In order to limit bandwidth fluctuations, the phone buffers 4 seconds of audio data (or the complete file) before playing the audio.

The Wav command follows the configured phone behavior for the audio path (speaker, headset and handset) and supports volume adjustments.

Note: The wav file is played only if the phone is idle.

The user can abort the streaming with:

- Goodbye key
- Soft 'Drop' key
- On hook (when handset active)
- Selecting a line

The beginning and the end of the streaming trigger the following action uris:

- action uri incoming/connected at the beginning of the streaming
- action uri onhook/disconnected at the end of the streaming (end of the file or aborted by the user)

As the phone displays a “Wav streaming” screen, the beginning of the streaming destroys the current XML display so diverting the action uri incoming and onhook is the only way to regain control for the XML application.

The typical use for this feature is a voicemail application or podcasts.

Note: Paging will not interrupt the streaming unless barge-in is set.

4.6.1 XML COMMANDS

The phone supports 2 `AastraIPPhoneExecute` commands. 'Wav.Play' and 'Wav.Stop'.

XML Command: Wav.Play

This command initiates the streaming of a WAV file to the phone.

Syntax:

```
Wav.Play:[v]:[tftp://|http://[username[:password]@]<host>[:port][/<path>]/file
```

Where v is the volume level of the audio path from 0 (lowest level) to 9 (highest level), this volume level overrides the current configured volume level of the phone.

Notes:

- The default mode is HTTP and the default port for HTTP is 80
- HTTPS is not supported.

Examples:

```
<ExecuteItem URI="Wav.Play:tftp://10.30.101.26/example.wav"/>
<ExecuteItem URI="Wav.Play:5:http://10.30.101.26/example.wav"/>
<ExecuteItem URI="Wav.Play:http://10.30.101.26:8080/example.wav"/>
<ExecuteItem URI="Wav.Play:2:http://user:password@10.30.101.26/sample.wav"/>
<ExecuteItem URI="Wav.Play:7:10.30.101.26/example.wav"/>
```

The last example is equivalent to

```
<ExecuteItem URI="Wav.Play:7:http://10.30.101.26/example.wav"/>
```

As HTTP is the default protocol.

Note:

On the 6867i, 6869i and 6873i an extra tag “title” can be used to become the title to be displayed on the Wav.Play screen.

XML Command: Wav.Stop

This command will abort WAV streaming.

Syntax:

```
Wav.Stop:
```

Example:

```
<ExecuteItem URI="Wav.Stop:"/>
```

4.6.2 FILE FORMAT

- The phones support **only** the following file format:
- Wav file
- G.711 µ-law and a-law Codec
- 20 ms packet size
- Mono 8KHz

If the format is not supported by the phone, the request is declined.

4.6.3 INTERACTION WITH ACTION URIS

action uri incoming/connected

When the phone starts streaming the wav file, the action uri incoming as well as action uri connected are triggered and the XML variables get the following values.

Variable Name	Value
$$SIPUSERNAME$$	“Streaming”
$$DISPLAYNAME$$	“Streaming”
$$SIPAUTHNAME$$	“Streaming”
$$PROXYURL$$	“Streaming”
$$ACTIVEPROXY$$	“Streaming”
$$INCOMINGNAME$$	Name of the wav file

Variable Name	Value
$$REMOTENUMBER$$	“”
$$LOCALIP$$	Local IP address of the phone
$$CALLDURATION$$	0
$$CALLDIRECTION$$	Empty
$$LINEINDEX$$	0
$$REMOTEURI$$	Empty
$$DIVERSIONURI$$	Empty
$$DIVERSIONREASON$$	Empty

action uri onhook/disconnected

At the end of the streaming (end of file or aborted by user), the action uri onhook is triggered as well as the action uri disconnected and the XML variables get the following values.

Variable Name	Value
$$SIPUSERNAME$$	“Streaming”
$$DISPLAYNAME$$	“Streaming”
$$SIPAUTHNAME$$	“Streaming”
$$PROXYURL$$	“Streaming”
$$ACTIVEPROXY$$	“Streaming”
$$INCOMINGNAME$$	Name of the wav file
$$REMOTENUMBER$$	“”
$$LOCALIP$$	Local IP address of the phone
$$CALLDURATION$$	Duration of streaming
$$CALLDIRECTION$$	Empty
$$LINEINDEX$$	0
$$REMOTEURI$$	Empty
$$DIVERSIONURI$$	Empty
$$DIVERSIONREASON$$	Empty

4.7 WAV FILE LOOP PLAYBACK (MELODY)

The phone also has the capability to download a wav file from a TFTP / FTP / HTTP server and play it continuously until it is stopped by a stop command or a user intervention.

Unlike the “Wav” streaming commands the file is first downloaded from the server and them played in loop locally on the phone.

For performance reasons, up to 8 wav files can be cached on the phone to allow a faster playback for following commands. The cache is limited to 400k and lost at reboot.

This command follows the configured phone behavior for the audio path (speaker, headset and handset) and supports volume adjustments.



Notes:

- The wav file is locally played only if the phone is idle.
- The wav file size is limited to 200K

The user can abort the playback with:

- Goodbye key
- Soft 'Drop' key
- On hook (when handset active)
- Selecting a line

Unlike the streaming capability, the beginning and the end of a playback does not trigger any following action uri and the phone display is not changed.



Note: Paging will not interrupt the streaming unless barge-in is set.

4.7.1 XML COMMANDS

The phone supports 2 `AastraIPPhoneExecute` commands. 'Melody.Play' and 'Melody.Stop'.

XML Command: Melody.Play

This command initiates the download then playback of a WAV file to the phone.

Syntax:

```
melody.Play:[tftp://|ftp://|http://[username[:password]@]<host>[:port][/<path>]
/file:[v]
```

Where v is the volume level of the audio path from 0 (lowest level) to 9 (highest level), this volume level overrides the current configured volume level of the phone.

Examples:

```
<ExecuteItem URI="Melody.Play:tftp://10.30.101.26/example.wav"/>
<ExecuteItem URI="Melody.Play:ftp://user:password@10.30.101.26/example.wav:5"/>
<ExecuteItem URI="Melody.Play:http://10.30.101.26:8080/example.wav"/>
<ExecuteItem URI="Melody.Play:http://user:password@10.30.101.26/sample.wav:2"/>
```

XML Command: Melody.Stop

This command will abort the wav file playback.

Syntax:

```
Melody.Stop:
```

Example:

```
<ExecuteItem URI="Melody.Stop:"/>
```

4.7.2 FILE FORMAT

The phones support **only** the following file format:

- Wav file
- G.711 µ-law and a-law Codec
- 20 ms packet size
- Mono 8KHz
- 200 kb

If the format is not supported by the phone, the request is declined.

4.8 DIAL AND DIALLINE URIS

The phone supports 2 URI commands to launch an outgoing call from XML.

- Dial:XXXXX
- DialLine:Y:XXXXX

Where XXXXX is the number/URI to dial and Y is the SIP line number.

Note: The AastraIPPhoneDirectory object does not support the “Dial” and “DialLine” commands.

The difference between Dial and DialLine is that with the DialLine command you can specify which SIP line to use, Dial will use the first available SIP line on the phone.

Dial and DialLine are generic URI and can be used anywhere a URI is requested by an XML object, this can be a custom softkey, an item in a TextMenu.

Although the line range is defined as 1 to 9, you must have a SIP account configured on the line to use the dial tag.

For example, if you configure lines 1 to 4 on a phone using a line value of 5 results in undefined behavior. The range of lines you can use for values is limited to the number of lines that the phone supports.

The following table identifies the number of lines supported on each phone type.

Phone model	Max Line
6863i	2
6865i/6867i/6869i/6873i/6920/6930/6940	24

4.9 CRASH AND CONFIGURATION FILES RETRIEVAL

The phone has the capability to upload the last crash file as well as the configuration files to a pre-configured location. This operation can be triggered manually via the Web UI or via the phone (if configured) but also via an XML command.

Example

In the configuration file (aastra.cfg or MAC.cfg)

```
upload system info server: tftp://myserver/path
```

Command

```
<AastraIPPhoneExecute>
<ExecuteItem URI="Command: UploadSystemInfo"/>
</AastraIPPhoneExecute>
```

Sending this command to the phone make the phone uploads the last crash file as well as the configuration files (server.cfg and local.cfg) to the configured server (using TFTP in this example). Uploaded files are indexed by the MAC address, date and time.

TFTP, FTP, HTTP and HTTPS are supported for the upload protocol.

4.10 SPECIAL ATTRIBUTES

4.10.1 BEEP

You can enable or disable a `Beep` option in all the Mitel XML objects via the Beep attribute in the root tag. When the phone receives an XML object, the `Beep` notifies the user that an object is being displayed.

This attribute is optional. If the `Beep` attribute is set to "yes" (i.e. `Beep="yes")` then it is an indication to the phone to sound a beep when it receives the object. If the `Beep` attribute is set to "no" (i.e. `Beep="no"`) or not present, then the default behavior is no beep is heard when the object arrives to the phone.

The `Beep` option can also be enabled or disabled via the configuration files and the Mitel Web UI using the following parameters:

`xml beep notification` (via configuration files)

XML Beep Support (via the Mitel Web UI see chapter 7.4)

The value set in the configuration files and Mitel Web UI override the attribute you specify in the XML object.

For example, if an `AastraIPPhoneStatus` object has the attribute of Beep="yes", and you uncheck (disable) the "XML Beep Support" in the Mitel Web UI, the phone does not beep when it receives an `AastraIPPhoneStatus` object.

4.10.2 TIMEOUT

The `Timeout` attribute is an optional root tag attribute for all of the current UI XML objects. When the phone receives an XML object with this attribute set it will override the default 45 seconds timeout specified for custom applications.

Setting `Timeout` to "0" will disable the timeout feature.

This timeout is reset by button presses.

4.10.3 LOCKIN

The `LockIn` attribute is an optional root tag attribute for all of the current UI XML objects. This attribute allows the XML application designer to specify that a screen can not be cancelled.

When a phone receives an UI XML object with the attribute set to "yes" it ignores all events that would cause the screen to exit without using the keys defined by the object.

Note: An incoming call still cancels the "locked" screen as telephony events have priority over XML applications.

Setting `LockIn` will disable the default timeout feature (45 seconds) unless the `Timeout` attribute is also set in the root tag.

4.10.4 CALLPROTECTION

The `CallProtection` attribute is an optional root tag for all the current UI XML objects. This attribute allows the XML application designer to controil the effect of an incoming call on the XML screen. It complements the `LockIn` attribute but is completely independent of it.

The possible values are:

- "no", an incoming call will cancel the current screen (even if `LockedIn` is enabled), this is the default behavior.
- "yes", an incoming call will not cancel the current screen
- "notif", an incoming call will not cancel the current screen and a toaster notification is displayed providing the caller ID (name/number). Not supported on 6863i and 6865i.

When set to "yes" or "notif", the capability for the user to answer the call (off-hook, Line key, hands-free, …) will depend on the `LockedIn` attribute.

4.10.5 TRIGGERDESTROYONEXIT

The `triggerDestroyOnExit` attribute is an optional root tag attribute for all of the current non-UI XML objects (PhoneStatus, PhoneExecute and PhoneConfiguration). Its default value is "no".

By default, if a UI XML object gets a non-UI XML object as an answer to an exit URI ("Select" on a TextMenu or "Done" on a InputScreen…) the UI XML object is not destroyed and stays on the phone display even if its `destroyOnExit` tag is set to "yes".

By setting `triggerDestroyOnExit` to "yes", the previous UI XML object is destroyed if its `destroyOnExit` tag is set to "yes".

4.11 TEXTMENU OR ICONMENU USER SELECTION (6867I, 6869I, 6873I, 6920, 6930, 6940)

It is also possible to send information related to a user's choice in a Text Menu.

"Select" and "Dial" system keys use the URI passed in the `MenuItem` attribute, a custom key might need to know the user selection in the TextMenu, and this is the purpose of the **`Selection`** attribute.

When a user accesses an arbitrary URI via a custom softkey, the Text Menu will send along a bit of information regarding the user's currently selected option using an optional XML attribute **`Selection`**.

XML Example

```
<AastraIPPhoneTextMenu destroyOnExit = "yes">
<Title>Parameter Tester</Title>
<MenuItem>
       <Prompt>First Selection</Prompt>
       <URI>http://someserver/somepage.xml</URI>
       <Selection>dataToAppend</Selection>
</MenuItem>
<SoftKey index = "1">
       <Label>Custom Key</Label>
       <URI>http://someotherserver/someotherpage.xml</URI>
</SoftKey>
</AastraIPPhoneTextMenu>
```

When the user selects item 1 and presses softkey 1, the URI requested is **http://someotherserver/someotherpage.xml?selection=dataToAppend**

Notes:

- If a "?" already exists in the URI, then a "&" is used to separate the parameters. Note the parameter name "selection" is automatic. If the `Selection` attribute is omitted, then nothing extra is appended to the URI.

- the `Selection` attribute can be more than one parameter, for instance, the following `Selection` attribute is valid `<Selection>200&action=set</Selection>` (don't forget to escape encode the attribute for the "&").

4.12 "SELECT" AND "DIAL" IN THE SAME TEXTMENU OR ICONMENU OBJECT

The only way to have a "Select" and "Dial" keys in the same text menu, the only option is to use "Dial" as a system command and mimic "Select" with a custom key.

```
<AastraIPPhoneTextMenu destroyOnExit = "yes">
<Title>Dial and Select</Title>
<MenuItem>
<Prompt>First Selection</Prompt>
<URI>200</URI>
<Selection>200</Selection>
</MenuItem>
<MenuItem>
<Prompt>Second Selection</Prompt>
<URI>201</URI>
<Selection>201</Selection>
</MenuItem>
<SoftKey index = "1">
<Label>Dial</Label>
<URI>SoftKey:Dial</URI>
</SoftKey>
<SoftKey index = "2">
<Label>Select</Label>
<URI>http://myserver.com/script.php</URI>
</SoftKey>
</AastraIPPhoneTextMenu>
```

When the user selects item 1 and presses

Softkey 1, the phone dials 200

Softkey 2, the requested URI requested is http://myserver.com/script.php?selection=200

4.13 "SELECT", "DIAL" AND "DIAL2" BEHAVIOR IN A TEXTMENU OR ICONMENU OBJECT

The following table describes what actions will be performed for each possible combination of items and actions in a `<MenuItem>`.

MenuItem Content	Select Softkey	Offhook	Dial2 Softkey	Dial Softkey
<URI>http://XYZ.</URI>	GET(http://XYZ)	-	-	Dial(http://XYZ)
<Dial>123</Dial>	-	Dial(123)	Dial(123)	-
<URI>http://XYZ</URI> <Dial>123</Dial>	GET(http://XYZ)	Dial(123)	Dial(123)	Dial(http://XYZ)
<URI>Dial:456</URI> <Dial>123</Dial>	Dial(456)	Dial(123)	Dial(123)	Dial(Dial:456)

Note: the main purpose of the "Dial" tag in the TextMenu is to allow the Mitel SIP phones which do not support custom softkeys to dial a number going off-hook.

4.14 REFRESH OF AN XML PAGE

All XML UI objects have the ability to be refreshed. This is accomplished by adding a Refresh setting to the HTTP header. This setting comprises two parameters:

- a time in seconds
- a URL.

The Refresh setting is set by the XML application and it is up to the application to decide which objects they want to refresh. So, it is an optional setting.

If the setting appears, then both parameters must be set for the Refresh to work.

The format of the HTTP header is:

```
Refresh: timeout; url=page to load
```

Example

```
Refresh: 3; url=http://10.50.10.140/update.xml
```

All of the parameters are mandatory.

With php you will have to use the `header` command in your script.

```
header("Refresh: 1; url= http://10.50.10.140/update.xml");
```

4.15 HTTP HEADERS FORMAT FOR A PHONE HTTP GET

When the Mitel SIP performs a GET to an HTTP server, it sets multiple variables in the HTTP header:

- 'User-Agent' providing information on the characteristics of the phone (type, MAC address, firmware),
- 'X-Aastra-ExpMod1' providing information on the expansion module 1 if present,
- 'X-Aastra-ExpMod2' providing information on the expansion module 2 if present,
- 'X-Aastra-ExpMod3' providing information on the expansion module 3 if present,
- 'Accept-Language' providing information on the language used on the phone.

Example:

```
User-Agent: Aastra6867i MAC:00-08-5D-19-94-B7 V:2.3.1.26-SIP
Accept-Language: en
X-Aastra-ExpMod1: 560M
X-Aastra-ExpMod2: 536M
```

4.15.1 USER-AGENT

The User-Agent header includes:

- The type of phone
- The MAC address of the phone
- The firmware version of the phone

Usually, the HTTP server is able to retrieve the User Agent parameters. The Mitel SIP phones send the following string for the User-Agent.

Phone_Model[space]MAC:-XX-XX-XX-XX-XX-XX[space]V:Version

This allows the application to adapt to the type of phone it is dealing with but also allows to design a single application to provide XML (for the phones) and HTML (for a web browser).

The following source code is a php example that can be used to decode the User-Agent header sent by a Mitel SIP phone.

```
function Decode_HTTP_header()
{
$user_agent=$_SERVER["HTTP_USER_AGENT"];
if(stristr($user_agent,"Aastra"))
	{
	$value=preg_split("/ MAC:/",$user_agent);
	$end=preg_split("/ /",$value[1]);
	$value[1]=preg_replace("/\-/","",$end[0]);
	$value[2]=preg_replace("/V:/","",$end[1]);
	}
else
	{
	$value[0]="Unknown";
	$value[1]="NA";
	$value[2]="NA";
	}
$header['model']=$value[0];
$header['mac']=$value[1];
$header['firmware']=$value[2];

return($header);
}
```

This function returns an array with

Array['model] is the phone model

- "Aastra6863i"
- "Aastra6865i"
- "Aastra6867i"
- "Aastra6869i"
- "Aastra6873i"
- "Mitel6920"
- "Mitel6930"
- "Mitel6940"

Array['mac'] is the MAC address

Array['firmware'] is the firmware version

Example of execution

- Array['model'] => Aastra6869i
- Array['mac'] => 00085D031C81
- Array['firmware'] => 4.0.0.66

4.15.2 X-AAASTRA-EXPMODI

If the phone has expansion modules (all models but 6863i) connected, the phone sends this information in proprietary http headers:

- X-Aastra-ExpMod1
- X-Aastra-ExpMod2
- X-Aastra-ExpMod3

The header is sent only if an expansion module is present at the given position, the possible values are:

- M680i for the 16 keys paper labeled expansion module (6865i/6867i/6869i/6873i/6920/6930/6940)
- M685i for the 3x28 keys self labeled expansion module (6865i/6867i/6869i/6873i/6920/6930/6940)

The following source code is a php example that can be used to decode the X-Aastra_ExpMod header sent by a Mitel SIP phone.

```
function Decode_HTTP_X_Aastra()
{
$header['module1']=$_SERVER["HTTP_X_AASTRA_EXPMOD1"];
$header['module2']=$_SERVER["HTTP_X_AASTRA_EXPMOD2"];
$header['module3']=$_SERVER["HTTP_X_AASTRA_EXPMOD3"];

return($header);
}
```

Example of execution

A Mitel 6869i with one M680 in position 1 and a M685 in position 2.

- Array['module1'] => M680
- Array['module2'] => M685
- Array['module3'] => Empty string

4.15.3 ACCEPT-LANGUAGE

The phone also sends the standard **Accept-Language** in the headers request which describes the language of the phone using 2 characters.

To access it using php the syntax is `$_SERVER["HTTP_ACCEPT_LANGUAGE"])`

Example

echo $_SERVER["HTTP_ACCEPT_LANGUAGE"]

returns "en"

4.16 SOME DEVELOPMENT GUIDELINES

There are some simple rules that you should follow when you develop XML applications for the Mitel SIP phones.

Don't forget the "Exit" key when you use custom softkeys

Place custom softkeys as they are for the standard objects, also it is better to use the same labels. The Exit key should preferably be placed in position 6.

Setting **`destroyOnExit`** attribute to "yes" is always preferable when you write complex applications as it is the only way to properly control the pages that are stacked in the phone.

Avoid using the Directory Object; TextMenu object with custom keys is much more flexible.

If you want to access data from the internet, it is preferable to use a RSS feed or a SOAP interface than Web scraping as Web sites frequently change their layout interface.

For XML applications using data from the Internet, it they are not real time you might consider to "cache" the data on the server and update them on a regular basis (a weekly update is necessary for Movie schedules but a 10 minutes update might be necessary for a weather application for severe conditions).

As the XML objects are limited to 10000 bytes, you might sometime hit this limit for a complex `TextMenu` with a lot of custom keys and custom icons. Don't forget to use the `base` attribute in the `MenuItem` object as it is a good way to reduce the size of the object.

5 URL FORMAT AND VARIABLES

5.1 URL FORMAT

To access to an XML application the phone performs a HTTP (or HTTPS) GET on a URL. The supported syntax is:

http://host[:port]/dir/file

or

https://host[:port]/dir/file

where:

- host is the hostname or IP address of the Web server supporting the XML application
- port is the port number the phones is using for the HTTP request.

Note: If the port is not specified, the phone uses port 80.

The phone also supports the following URLs:

- `Dial: XXXXX` to have the phone dial XXXXX
- `DialLine:X:YYYYY` to have the phone dial YYYYY (phone number or URI) using SIP line X see chapter 4.7 for more details on this feature..
- `Led:XXXXX=on/off/slowflash/fastflash` to change the status the LED associated to the XXXXX key if the key is configured as an XML key. See chapter 4.2 for more details on the LED control.
- `Key:XXXXX` to simulate a keypress on the phone. See chapter 4.5 for more details on keypress emulation.

5.2 URL VARIABLES

The phones support system variables in the HTTP URL to pass dynamic data to the XML applications.

The variables are in the form $$VARIABLENAME$$ and the following variables are supported:

Variable Name	Value
SIPUSERNAME	Active line user name
DISPLAYNAME	Active line display name
SIPAUTHNAME	Active line SIP authentication name
PROXYURL	Active line SIP proxy
ACTIVEPROXY	Active line active SIP proxy
INCOMINGNAME	Caller-ID name of the current incoming call
REMOTENUMBER	Remote party phone number (incoming or outgoing)
LOCALIP	Phone local IP address
CALLDURATION	Duration of the current call in seconds
CALLDIRECTION	Direction of the current call (“Incoming” or “Outgoing”)
REGISTRATIONCODE	3 digits registration code for the registration coming from the SIP Proxy server.

Variable Name	Value
REGISTRATIONSTATE	Registration state available from the Registration event, values are: “REGISTERED", "UNREGISTERED", "EXPIRED", "REFUSED", "TIMEOUT".
LINESTATE	Information on Disconnect event, values are “IDLE” “DIALING” “CALLING” “OUTGOING” “INCOMING” “CONNECTED” “CLEARING”
LINEINDEX	Line Index of focused line (SIP registration line number of the focused line)
REMOTEURI	The Remote party SIP URI (incoming or outgoing) (sip:user@server:port)
DIVERSIONURI	The full URI of the Diversion information (sip:user@server)
DIVERSIONREASON	The Diversion Reason information

At the time of the HTTP call, the variables are replaced with the value of the appropriate variable.

The variables are available in all the HTTP URL for programmable/soft keys as well as for the action URI. They can also be used in the XML objects themselves (TextMenu, InputScreen…) but not in a PhoneExecute command.

Example

For example, if the administrator specifies an XML softkey with the value:

http://10.50.10.140/script.pl?name=$$SIPUSERNAME$$

This softkey executes a GET on:

http://10.50.10.140/script.pl?name=42512

Assuming that the SIP username of the specific line is 42512.

The variables can be used, for instance, in a context where the phone has multiple SIP registrations and the XML application need to know which SIP registration the phone is currently using.



Note: the value of a variable depends on the status of the phone, if the variable has no meaning in the current status; the phone sends an empty string. For example, if the URL is “http://myserver.com/script.pl?number=$$REMOTENUMBER$$&key=5”, the phone will call “http://myserver.com/script.pl?number=”&key=5” if the phone is idle.

5.2.1 VARIABLES RELATED TO THE ACTIVE LINE

The following variables are related to the active line

- $$SIPUSERNAME$$ Active line user name

- $$DISPLAYNAME$$ Active line display name
- $$SIPAUTHNAME$$ Active line SIP authentication name
- $$PROXYURL$$ Active line SIP proxy
- $$ACTIVEPROXY$$ Active line active SIP proxy
- $$LINEINDEX$$ Active line index

They can be used anywhere where the active line is meaningful for your XML application but they should not be used with outside event action uris such as action uri poll or action uri xml sip notify as the active line can be anything when the event is processed.

5.2.2 VARIABLES RELATED TO THE CURRENT CALL

The following variables are related to the current call.

- $$INCOMINGNAME$$ Caller-ID name of the current incoming call
- $$REMOTENUMBER$$ Remote party phone number (incoming or outgoing)
- $$CALLDURATION$$ Duration of the current call in seconds
- $$CALLDIRECTION$$ Direction of the current call (“Incoming” or “Outgoing”)
- $$LINESTATE$$ Information on Disconnect event
- $$REMOTEURI$$ Remote party SIP URI (incoming or outgoing)
- $$DIVERSIONURI$$ Diversion URI
- $$DIVERSIONREASON$$ Diversion reason code

List of values for the LINESTATE variable

LINESTATE VALUE	Meaning	Disconnected uri
IDLE	The phone is idle	N/A
DIALING	The phone is offhook and ready to dial	N/A
CALLING	A SIP INVITE has been sent but no response has been received.	An error occured during the call.
OUTGOING	Remote party is ringing.	The call was canceled.
INCOMING	Local phone is ringing.	The call was missed or canceled.
CONNECTED	The parties are talking.	Call was successful
CLEARING	Call has been released but not acknowledged.	N/A

These variables are initialized at the beginning and at the end of the call which means for instance that the call duration variable is reset to ‘0’ after the phone goes on-hook, the last moment to retrieve the value is when you use the `action uri onhook` or `action uri disconnected`.

5.2.3 USAGE WITH ACTION URIS

The following table details when the variables are meaningful with an action URI.

	startup	registered	registration	onhook	offhook	incoming	outgoing	disc.	poll	notify
SIPUSERNAME	X	X	X	X	X	X	X	X		
DISPLAYNAME	X	X	X	X	X	X	X	X		
SIPAUTHNAME	X	X	X	X	X	X	X	X		
PROXYURL	X	X	X	X	X	X	X	X		
INCOMINGNAME				X[1]		X			X[2]	X[2]
REMOTENUMBER				X		X	X		X[3]	X[3]
LINESTATE				X				X	X	X
LOCALIP	X	X	X	X	X	X	X		X	X
CALLDURATION				X	X				X	X
CALLDIRECTION				X	X	X	X	X	X	X
REGISTRATION STATE		X	X						X	
REGISTRATION CODE		X	X						X	
LINEINDEX	X	X	X	X	X	X	X	X	X	X
REMOTEURI						X	X			
DIVERSIONURI						X	X			
DIVERSION REASON						X	X			

5.2.4 PHONE STATE

The following table details when the variables are meaningful for a programmable/soft key depending on the phone state.

	idle	connected	incoming	outgoing
SIPUSERNAME	X	X	X	X
DISPLAYNAME	X	X	X	X
SIPAUTHNAME	X	X	X	X
PROXYURL	X	X	X	X

[1] only if the previous call was an incoming call

[2] only if the phone is in the “connected” state answering an incoming call

[3] only if the phone is in the “connected” state

	idle	connected	incoming	outgoing
ACTIVEPROXY	X	X	X	X
INCOMINGNAME		X[1]	X	
REMOTENUMBER		X	X	X
LINESTATE	X			
LOCALIP	X	X	X	X
CALLDURATION		X		
CALLDIRECTION		X		
REGISTRATIONSTATE				
REGISTRATIONCODE				
LINEINDEX	X	X	X	X
REMOTEURI		X	X	X
DIVERSIONURI		X	X	X
DIVERSIONREASON		X	X	X

5.3 HTTPS

If you want to secure the communication between the Mitel SIP Phone and your HTTP server hosting the XML applications, you can use HTTPS instead of HTTP.

Note: HTTPS provides a reasonable level of security to your XML applications for man-in-the-middle attacks but you must still make sure that the HTTP server itself is secure, only the data transport is protected by HTTPS.

To allow HTTPS connections, the web server must have a root certificate; this certificate must be signed by a certificate authority of one form or another, which certifies that the certificate holder is indeed the entity it claims to be. The certificate can also be self-signed.

Mitel SIP phones come with the signing certificates of

- Verisign
- GeoTrust
- Thwate

So, only certificates signed by these Certification Authorities can be verified by the phone. Certificates that are signed by other providers will not verify on the phone but to overcome this problem, the phone can be loaded with user certificates.

The following parameters allow HTTPS validation configuration.

[1] only if the user has answered the current call.

```
https validate certificates
https validate expires
https validate hostname
https user certificates
```

Note: if you have `http validate expires` enabled, you must make sure that the phone clock is yet prior to using HTTPS.

When a certificate is rejected the phone displays "Bad Certificate" on the fifth line of the display for large screen phones and on the third line for small screen phones (6863i/6865i).

Note: For more information regarding HTTPS and associated certificates please refer to the x.509 standard at http://www.ietf.org/html.charters/pkix-charter.html .

5.3.1 USER CERTIFICATES

The user has the option to upload its own certificates onto the phone. These certificates must be uploaded in a single file in the PEM format.

The user certificates are persistent between firmware upgrades but are deleted during a factory default.

User provided certificates are downloaded as part of the boot time configuration downloads and be based on a filename specified in `http user certificates`. The certificate file must be located in the configuration server directory.

Note: In order to install a root certificate using HTTPS, the user must first disable verification, since the certificate will not be in the validation chain yet.

Example

Here is an example of a self-signed certificate; it is stored in a file called “mycertificate.pem” located in the configuration server directory.

```
-----BEGIN CERTIFICATE-----
MIIDyjCCAzOgAwIBAgIBADANBgkqhkiG9w0BAQQFADCBpTELMAkGA1UEBhMCLS0x
EjAQBgNVBAgTCVNvbWVTdGF0ZTERMA8GA1UEBxMIU29tZUNpdHkxGTAXBgNVBAoT
EFNvbWVPcmdhbml6YXRpb24xHzAdBgNVBAsTFlNvbWVPcmdhbml6YXRpb25hbFVu
aXQxEzARBgNVBAMUCm15X3RyaXhib3gxHjAcBgkqhkiG9w0BCQEWD3Jvb3RAbXlf
dHJpeGJveDAeFw0wODA2MTAxOTM4MzZaFw0wOTA2MTAxOTM4MzZaMIGlMQswCQYD
VQQGEwItLTESMBAGA1UECBMJU29tZVN0YXRlMREwDwYDVQQHEwhTb211Q2l0eTEZ
MBcGA1UEChMQU29tZU9yZ2FuaXphdGlvbjEfMB0GA1UECxMWU29tZU9yZ2FuaXph
dGlvbmFsVW5pdDETMBEGA1UEAxQKbXlfdHJpeGJveDEeMBwGCSqGSIb3DQEJARYP
cm9vdEBteV90cml4Ym94MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDUZa1m
3ienPcP1zmgp2jT7oGnVoSfKmOs7n7kLpWTUyWpmEfrleetVoRoaGL5j+zTwaAjO
lp9apC17nwe0qRC2BuPDyzkVE4f30Vn2pm1u5yYDoi+yRE1NFyVLbQwLFTffUGwQ
kfC7YabPdbjQ87/XU6/AhuquET0n3nNE9NwQHwIDAQABo4IBBjCCAQIwHQYDVR0O
BBYEFE6XNlxAlZxfYbmucEN2VUyOTIfNMIHSBgNVHSMEgcowgceAFE6XNlxAlZxf
YbmucEN2VUyOTIfNoYGrpIGoMIGlMQswCQYDVQQGEwItLTESMBAGA1UECBMJU29t
ZVN0YXRlMREwDwYDVQQHEwhTb211Q2l0eTEZMBcGA1UEChMQU29tZU9yZ2FuaXph
dGlvbjEfMB0GA1UECxMWU29tZU9yZ2FuaXphdGlvbmFsVW5pdDETMBEGA1UEAxQK
bXlfdHJpeGJveDEeMBwGCSqGSIb3DQEJARYPcm9vdEBteV90cml4Ym94ggEAMAwG
A1UdEwQFMAMBAf8wDQYJKoZIhvcNAQEEBQADgYEANdtuygpPobd7ICWnlor+UEJ5
3Tei51HDN2oKp6SaF3uQCUJUMMJwB9/i/rVa/c801Oqf06/MMCU6/VWS9uZspBvm
IJWaFpU8+VcbGH63KA61FSqKHSN+8ZRl0E58FgmHzEXwJr4P4Hb59FrpqkUwjDRF
86UP/ZyBYhH7EWtIJhA=
-----END CERTIFICATE-----
```

The following lines must be added to the phone configuration files (‘aastra.cfg’ or ‘MAC.cfg’) in order to use HTTPS for XML requests.

```
https user certificates: mycertificate.pem
```

5.3.2 CONFIGURATION

Using configuration files

Parameter – https validate certificates	Configuration Files aastra.cfg, <mac>.cfg
Description	When this parameter is set to 1 the https client will perform validation on SSL certificates before they are accepted.
Format	Boolean (0=disabled, 1=enabled)
Default Value	1 (enabled)
Range	0 or 1
Example	https validate certificates : 1

Parameter – https validate expires	Configuration Files aastra.cfg, <mac>.cfg
Description	When this parameter is set to 1 the https client will verify that a certificate has not expired prior to accepting it.
Format	Boolean (0=disabled, 1=enabled)
Default Value	1 (enabled)
Range	0 or 1
Example	https validate expires : 1

Parameter – https validate hostname	Configuration Files aastra.cfg, <mac>.cfg
Description	When this parameter is set to 1 the https client will verify the commonName of a certificate matches the server it is connecting to prior to accepting the certificate
Format	Boolean (0=disabled, 1=enabled)
Default Value	1 (enabled)
Range	0 or 1
Example	https validate hostname : 1

Parameter – https user certificates	Configuration Files aastra.cfg, <mac>.cfg
Description	This parameter is a file name on the configuration server that contains user provided certificates in PEM format. These certificates will be used to validate peer certificates.
Format	String
Default Value	Empty
Range	N/A
Example	https user certificates : mycertificate.pem

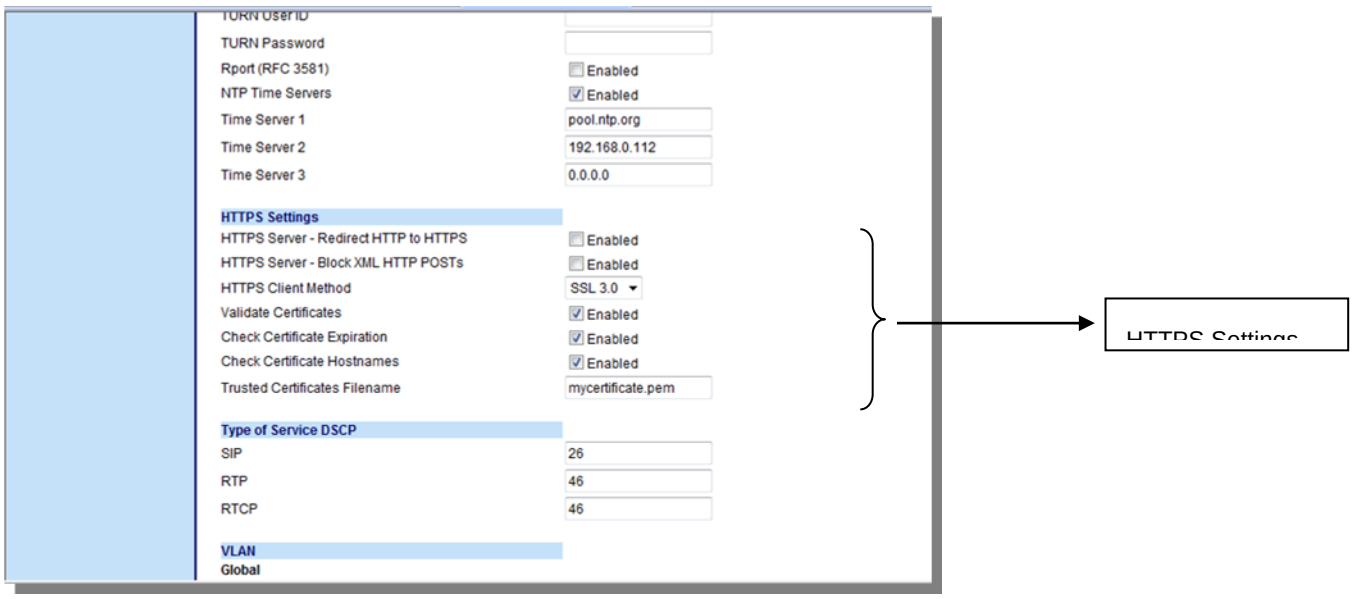
Web UI configuration

The HTTPS paramerets can also be configured using the WebUI.

Select Network.

Enter the parameters.

Click Save Settings.



Note: Depending on the changes performed, you might have to reboot the phone to apply them.

5.4 XML OBJECTS PUSHED TO THE PHONE

The phone can request an XML object via HTTP GET, or an object can be pushed to the phone via a POST. The phone parses this object immediately upon receipt and displays the information to the screen.

The HTTP POST packet must contain an “xml=” line in the message body. The string to parse is located after the equals sign in the message. HTML forms that post objects to the phone must use a field named “xml” to send their data. Any applications that construct HTTP packets on the fly must also specify this line.

Note: the content of the HTTP POST is not url-encoded.

Sample php source code

Below is a sample php source code that sends an XML object to a Mitel phone. In this example, the phone is located at 192.168.0.150 and the server pushing the XML object at 192.168.0.112.

```
<?php
#
function push2phone($server,$phone,$data)
{
$xml = "xml=".$data;

$post = "POST / HTTP/1.1\r\n";
$post .= "Host: $phone\r\n";
$post .= "Referer: $server\r\n";
$post .= "Connection: Keep-Alive\r\n";
$post .= "Content-Type: text/xml\r\n";
$post .= "Content-Length: ".strlen($xml)."\r\n\r\n";

$fp = @fsockopen ( $phone, 80, $errno, $errstr, 5);
if($fp)
       {
       fputs($fp, $post.$xml);
       flush();
       fclose($fp);
       }
}

############################
$xml = "<AastraIPPhoneTextScreen>\n";
$xml .= "<Title>Push test</Title>\n";
$xml .= "<Text>This is a test for pushing a screen to a phone. It is a way to
demonstrate that we can push XML objects to an Mitel Phone.</Text>\n";
$xml .= "</AastraIPPhoneTextScreen>\n";
push2phone("192.168.0.112","192.168.0.150",$xml);
?>
```

Sample perl source code

Below is a sample perl source code that sends an XML object to an Mitel phone. In this example, the phone is located at 192.168.0.150.

```
#!c:/perl/bin/Perl.exe
# Create a user agent object
     use LWP::UserAgent;
     use LWP::ConnCache;
     use HTTP::Request::Common;
my $ua = LWP::UserAgent->new(agent => 'Mozilla/4.0 (compatible; MSIE 5.5;
Windows 98)');

$ua->conn_cache(LWP::ConnCache->new());
$ua->request(
   POST 'http://192.168.0.150'
        ,Content_Type  => 'application/x-www-form-urlencoded'
        ,Content                => 'xml=<AastraIPPhoneTextScreen><Title>Push
test</Title><Text>This is a test for pushing a screen to a phone. It is a way
to demonstrate that we can push XML objects to an Mitel
Phone.</Text></AastraIPPhoneTextScreen>');
```

Notes:

- To accept a pushed page the “XML Push Server List” phone parameter must be configured with the list of the HTTP servers allowed pushing a page.
- When a page is pushed to the phone the MWI lamp blinks to indicate a new message to the phone.

When a XML object is pushed to the phone, the phone acts like a Web server listening to TCP port 80, which means that if the phone is behind a NAT and the server on the other side of the NAT, which is a typical configuration for a remote user, the port 80 must be forwarded to the IP phone at the router level.

6 ACTION URIS

Configuration parameters have been introduced to configure the phone to make an HTTP GET based on events, these events are:

- End of the boot sequence `action uri startup`
- Successful registration `action uri registered`
- On-hook action uri onhook
- Off-hook action uri offhook
- Incoming call action uri incoming
- Outgoing call action uri outgoing
- Time based action uri poll
- SIP Notify action uri xml sip notify
- Connect action uri connected
- Disconnect action uri disconnected
- Registration event action uri registration event

6.1 CONFIGURATION

A URI can be configured for each type of events; the configuration can be made using:

- The configuration files (aastra.cfg and/or MAC.cfg)
- The Web UI.

See chapter 7.3 for more details on the Web UI Configuration.

Configuration description

These URIs will be configurable via the configuration files using the following parameters.

- **action uri startup**: <URI to GET on startup>
- **action uri registered**: <URI to GET on successful registration>
- **action uri incoming**: <URI to GET on an incoming call>
- **action uri outgoing**: <URI to GET on outgoing call>
- **action uri offhook**: <URI to GET on an off-hook event>
- **action uri onhook**: <URI to get on an on-hook event>
- **action uri connected**: <URI to get on a connect event>
- **action uri disconnected**: <URI to get on a disconnect event>
- **action uri poll**: <URI to get when configured timer expires>
- **action uri poll interval**: <polling interval in seconds for action uri poll>
- **action uri registration event**: <URI to GET on any registration event>
- **action uri xml sip notify**: <URI to get when a aastra-xml SIP notify without content is received by the phone>
- **action uri blf** <URI to get when a BLF or BLF/Xfer key is pressed>
- **sip xml notify event**: <Enables or disables the phone to accept an Aastra-xml SIP Notify event>

As described in chapter 5.2 the action URI support variables in their configuration.

Example

action uri startup: http://myserver.com/startup.php

action uri incoming: http://myserver.com/incoming.php?number=$$REMOTENUMBER$$

sip xml notify event: 1

action uri xml sip notify: http://myserver.com/notify.php

6.2 ACTION URI DETAILED BEHAVIOR

Action uris are available on all the lines and line appearance of the phone so passing the system variables $$SIPUSERNAME$$ or $$SIPAUTHNAME$$ and $$PROXYURL$$ or $$ACTIVEPROXY$$ will define which line has triggered the action uri.

Notes:

The action uri GET requests are non blocking.

If the GET request triggered by an action uri fails, the phone does not display any error message.

6.2.1 ACTION URI OFFHOOK

The offhook uri is triggered only when the user performs one of the following actions:

- Release the handset
- Press a line key
- Press the Speaker/Headset key

This means that the offhook uri is not triggered when the user presses a speeddial key even if the phone goes off-hook and dials a number. Also if the user directly dials a number and press the “dial” key the action uri is not triggered (but the outgoing uri is)

6.2.2 ACTION URI ONHOOK

The *action uri onhook* is triggered at the end of any active call:

- The user presses the “Goodbye” key
- The user hangs up the handset after a connected call
- The user drops an established call pressing the “drop” softkey
- After a completed transfer (blind or monitored)
- The other party hangs up.

Note: The *onhook* event is a subset of the *disconnect* event, you should avoid to configure both action uri as most of the time they will both be called at the same time.

6.2.3 ACTION URI INCOMING

The *action uri incoming* is triggered each time there is an incoming call presented to the phone and processed to make the phone ring. If the incoming call includes an auto-answer info message (intercom mode), the action uri incoming is triggered as well.

Note: the action uri incoming is not triggered if the phone is call forwarded or is in DND mode or if call waiting is disabled (and the phone in a non idle mode).

6.2.4 ACTION URI OUTGOING

The *action uri outgoing* is triggered each time an outgoing call is performed by the phone (SIP INVITE message sent).

Note: the action uri outgoing is also triggered when the user makes a second call to perform a transfer or a conference.

6.2.5 ACTION URI CONNECTED

The *action uri connected* is triggered each time the phone transitions to a connected state. This includes regular phone calls but also Intercom, Paging, RTP streaming or Play Wav.

Notes:

The action uri connected is also triggered when the second leg of a 3-way conference is established.

During SLA calls, the phone uses the Action URI Connected parameter when the line is seized before the caller dials out

SCA and BLA calls on hold trigger the action uri connected, since the retrieval is a second call performed by the phone and the phone cannot link the retrieved call with the earlier held call.

6.2.6 ACTION URI DISCONNECTED

The *action uri disconnected* is triggered when the phone transitions from any active (non idle) call state to idle. This includes regular calls but also Intercom, Paging, RTP streaming or Play Wav.

The difference between action uri disconnected and action uri onhook is that the disconnect event is generated even when the call fails or if a call is not answered.

The action uri disconnect is triggered every time the phone comes back to the idle state where the action uri onhook is triggered only when the phone was on an active call.

The $$LINESTATE$$ variable provides more information on the reason of the disconnect event.

LINESTATE VALUE	Meaning	Disconnected uri
IDLE	The phone is idle	N/A
DIALING	The phone is offhook and ready to dial	N/A
CALLING	A SIP INVITE has been sent but no response has been received.	An error occured during the call.
OUTGOING	Remote party is ringing.	The call was canceled.
INCOMING	Local phone is ringing.	The call was missed or canceled.
CONNECTED	The parties are talking.	Call was successful
CLEARING	Call has been released but not acknowledged.	N/A

Note: the action uri disconnected is not triggered if the phone is call forwarded or is in DND mode or if call waiting is disabled (and the phone in a non idle mode).

6.2.7 ACTION URI STARTUP

The *action uri startup* is triggered at the end of the phone boot sequence.

6.2.8 ACTION URI REGISTERED

The *action uri registered* is triggered the first time the phone registers successfully a line to its SIP proxy/registrar.

If the phone has multiple SIP registrations, the action uri registered is triggered for each line.

6.2.9 ACTION URI REGISTRATION EVENT

The *action uri registration* event is triggered every time there is a registration state change, the registration states are:

- Registered
- Unregistered
- Registration timed out
- Registration refused
- Registration expired

The action uri registration event is not triggered when the same event is repeated.

Two URI variables have also been added to complement this feature

- $$REGISTRATIONSTATE$$
- $$REGISTRATIONCODE$$

See chapter 5.2 for more details on these variables.

6.2.10 ACTION URI POLL

The *action uri poll* is called each time the configured timer (*action uri poll interval* parameter) expires.

The first timer starts at the end of the phone boot sequence.

6.2.11 ACTION URI BLF

The action uri blf is called each time a BLF or BLF/Xfer key is pressed, this happens only when

- “blf key mode” parameter is set to the value 2
- “action uri blf” parameter is not empty

This action uri has dedicated variables on top of the common variables:

- $$BLFNO$$ indicates the monitored extension
- $$BLFSTATE$$ indicates the monitored line state which can be
    - IDLE
    - BUSY
    - RINGING
    - ONHOLD
    - DND
    - UNKNOWN
- $$BLFTRANSFER$$ indicates if a BLF/XFER or a BLF key was pressed
    - YES (BLF/XFER)
    - NO (BLF)

Example

action uri blf: http://myserver/blf.php?blf=$$BLFNO$$&state=$$BLFSTATE$$

6.2.12 ACTION URI XML SIP NOTIFY

The *action uri xml sip notify* is triggered when the phone receives an authorized SIP Notify aastra-xml event with an empty content. The XML call is triggered only if the *sip xml notify event* parameter is set to 1 (enabled). See chapter 6.2.12 for more details on the configuration.

If XML content is provided in the SIP NOTIFY, it is processed directly by the phone as it is done for a XML PUSH

If the content is empty in the SIP NOTIFY, the phone automatically triggers a new pre-configured action uri (**action uri xml sip notify**).

Example of a SIP NOTIFY with XML content

```
NOTIFY sip:200@10.30.100.103:5060 SIP/2.0
Via: SIP/2.0/UDP 10.30.100.103:5060;branch=z9hG4bK7bbc1fac;rport
From: <sip:201@10.30.100.103:5060>;tag=81be2861f3
To: Jacky200 <sip:200@10.30.100.103:5060>
Contact: <sip:201@10.30.100.103>
Call-ID: 59638f5d95c9d301
CSeq: 4 NOTIFY
Max-Forwards: 70
Event: aastra-xml
Content-Type: application/xml
Content-Length: 115
<AastraIPPhoneExecute><ExecuteItem
URI="http://10.30.100.39/XMLtests/SampleTextScreen.xml"/></AastraIPPhoneExecute>
```

When the phone receives the SIP NOTIFY, the XML content is processed as any XML object; in this example, the phone will call http://10.30.100.39/XMLtests/SampleTextScreen.xml after reception of the SIP NOTIFY.

Note: The phone supports all the current XML objects with all the existing limitations. For example if an AastraIPPhoneExecute is used, the embedded uri(s) cannot be HTTPS based.

Example of a SIP NOTIFY without XML content

```
NOTIFY sip:200@10.30.100.103:5060 SIP/2.0
Via: SIP/2.0/UDP 10.30.100.103:5060;branch=z9hG4bK7bbc1fac;rport
From: <sip:201@10.30.100.103:5060>;tag=81be2861f3
To: Jacky200 <sip:200@10.30.100.103:5060>
Contact: <sip:201@10.30.100.103>
Call-ID: 59638f5d95c9d301
CSeq: 4 NOTIFY
Max-Forwards: 70
Event: aastra-xml
Content-Type: application/xml
Content-Length: 0
```

When the phone receives the SIP NOTIFY, it will trigger the action uri xml sip notify if it has been previously configured using the configuration files or the phone Web UI. If the action uri xml sip notify is not enabled, the phone does not do anything.

Note: To ensure that the SIP NOTIFY is coming from a **trusted source**, it is recommended that you enable the Whitelist feature (Whitelist Proxy parameter) on the IP phone. If enabled, and the phone receives a SIP NOTIFY from a server that is NOT on the whitelist, the phone will reject the message.

Asterisk implementation

Asterisk custom SIP Notify messages are configured in 'sip_notify.conf' file in the asterisk root directory, usually '/etc/asterisk/'.

Asterisk does not support custom SIP Notify with XML content. The only way to use the SIP Notify event with Asterisk is to use the Aastra-xml event triggerint the action uri xml sip notify.

To do so, the following lines must be added in /etc/asterisk/sip_notify.conf'

```
; Aastra XML event
[aastra-xml]
Event=>aastra-xml
Content-Length=>0
```

Then you must restart Asterisk to have the new SIP Notify parsed by Asterisk.

To use it, just send a SIP Notify aastra-xml event to a Mitel SIP phone registered to the platform using either the asterisk CLI or the AGI.

sip notify aastra-xml 200 201….

Where 200, 201… are the SIP peers for the Mitel SIP phones.

This will trigger the action uri xml sip notify if it has been enabled and properly configured.

6.3 "ANSWER" AND "IGNORE" KEYS FOR ACTION URI INCOMING

When a UI XML object is displayed as an answer to an action uri incoming, the regular contextual keys ("Answer" and "Ignore") are not displayed as the display is managed by the XML object, the user can only take the call by pressing the blinking line key or go off-hook.

To keep a consistent behavior of the phone after an action uri incoming, the "allowAnswer" tag allows the native interactive keys "Answer" and "Ignore" to be displayed and allow the user to answer the call.

Once the call is answered or ignored, the 2 keys disappear automatically.

6.4 "DROP", "XFER" AND "CONF" KEYS FOR XML APPS IN CONNECTED STATE

When a UI XML object is displayed as with the phone in a connected state, the regular contextual keys ("Drop", "Xfer" and "Conf") are overridden by the XML object, the user cannot access anymore to these telephony features.

To keep a consistent behavior of the phone when the phone is connected, 3 tags have been introduced:

- allowDrop
- allowXfer
- allowConf

When they are enabled the native interactive keys are displayed on top of the XML object.

Once the call is ended, the 3 keys disappear automatically.

6.5 APPLICATIONS

Action URIs are very powerful as they allow an external application to take control of the display when an event occurs.

Here are some examples of potential applications.

Self-configuration

Using the startup URI, it is possible to develop self-provisioning on the phone. If a new phone boots and gets its configuration server IP address (through DHCP option 66 for example), it can download the `aastra.cfg` file with a startup URI set to an XML application, as it is a new phone the `<MAC>.cfg` config files does not exist. At the end of the boot, the phone will go to this XML application which can identify the phone and then generate the `<MAC>.cfg` file “on the fly” and ask the phone to reboot using the `PhoneExecute` object.

Screen pop

Using the action uri incoming, it is possible to display extra information on the phone for an incoming call. For instance, the XML application that is called when there is an incoming call can do a database lookup (Microsoft Exchange or any database) and display information on the caller. Basically it is like having a screen pop application directly on the phone.

Call Center

As for the Screen pop, the incoming URI can be used in a call center environment to display CRM or queue information on the caller. The on-hook URI can also be used to collect the wrap-up code at the end of a call.

And many more…

7 XML CONFIGURATION

After creating an XML application to use on the IP phone, the application can be accessed as a Service or a Key.

7.1 CONFIGURING A SOFT OR PROGRAMMABLE KEY USING THE PHONE WEB UI

In addition to linking an XML application to a custom service, an application can be also be linked to a softkey and/or a programmable key depending on the phone and on the phone configuration (expansion modules).

Select “Operation” => “Softkeys and XML” (6867i/6869i/6873i/6920/6930/6940

Or

Select “Operation” => “Programmable Keys” (6863i/6865i)

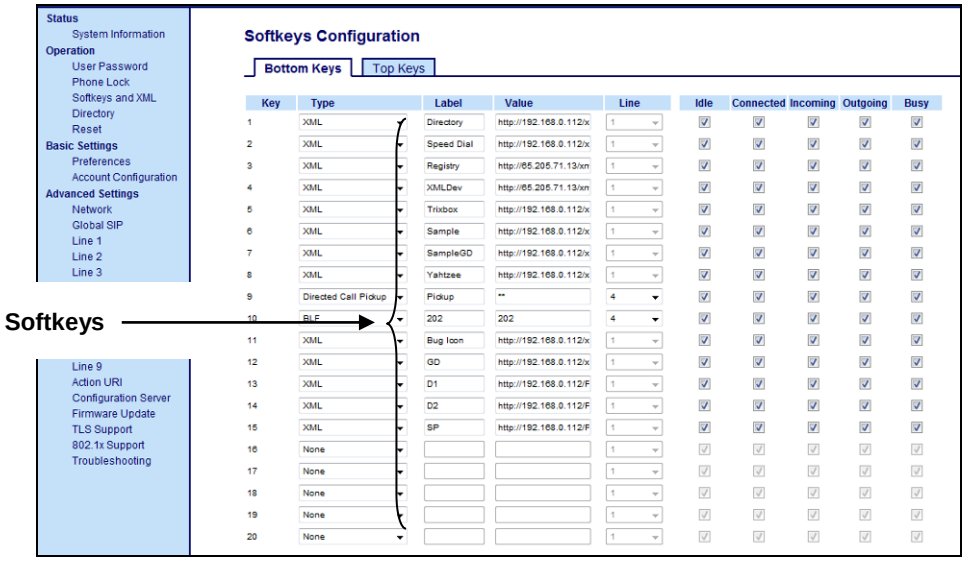
Or

Select “Operation” => “Expansion Module X” (6865i/6867i/6869i/6873i/6920/6930/6940 with expansion module(s))

Choose type “XML” for the desired key.

Enter the URI in the value field.

Click Save Settings.



Note: Reboot is not required.

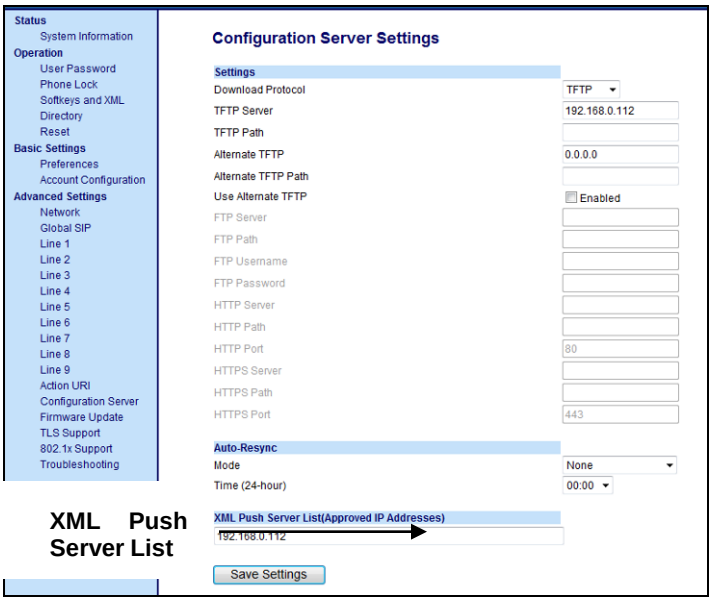
7.2 CONFIGURING THE XML PUSH SERVER LIST USING THE PHONE WEB UI

The IP phone will only accept HTTP POSTs from the IP addresses set in the **XML Push Server List**.

Select Advanced Settings=>Configuration Server.

Enter a comma-separated list of IP addresses or domain names in the **XML Push Server List** field.

Click Save Settings and reboot the phone.



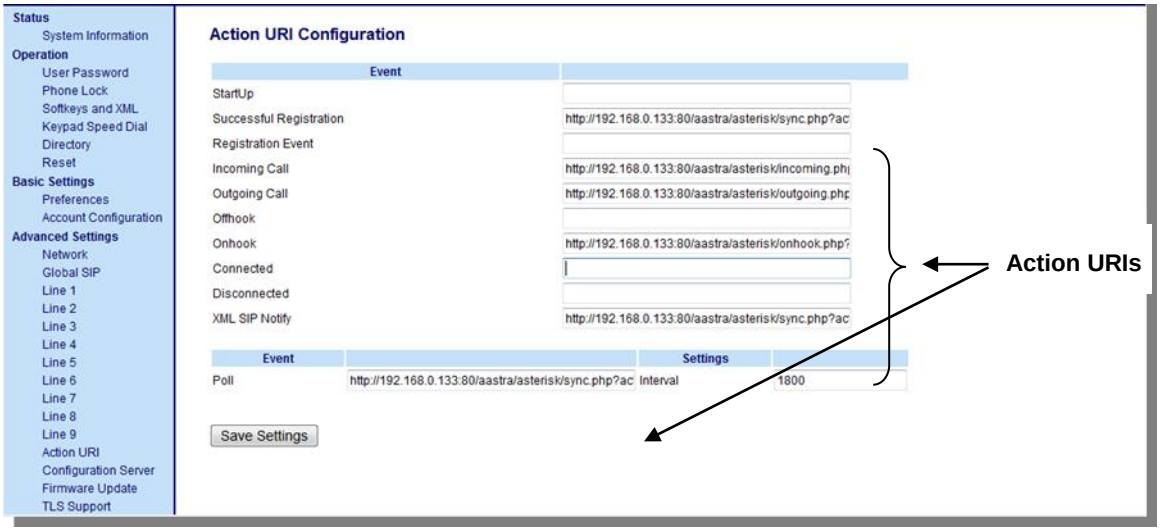
7.3 CONFIGURING THE ACTION URIS USING THE PHONE WEB UI

You can configure the URI to be called for each type of event supported by the phone from the Web UI...

Select Advanced Settings=>Action URI.

Enter the URI for each type of event.

Click Save Settings.



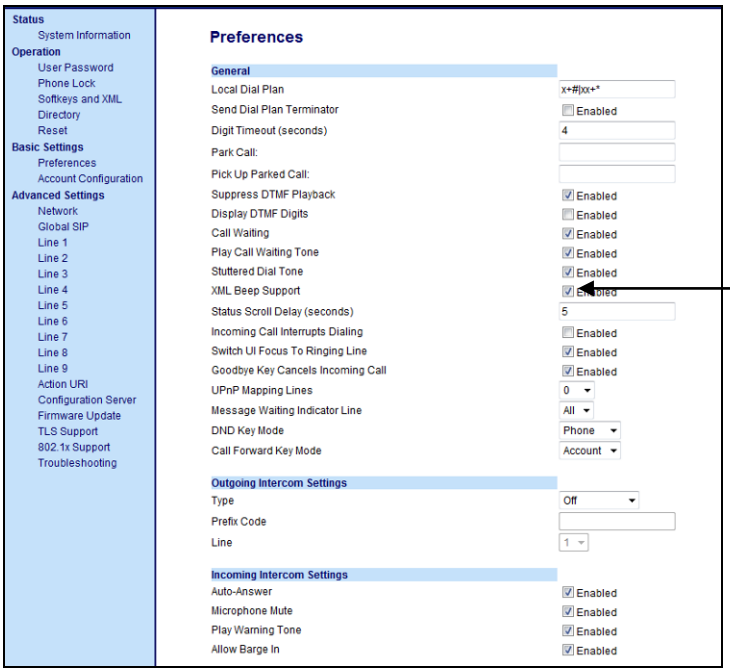
7.4 CONFIGURING THE XML BEEP SUPPORT USING THE PHONE WEB UI

You can configure the **XML Beep Support** (enabled or disabled) from the Web UI. It impacts the behavior of the `AastraIPPhoneStatus` object regarding the phone notification.

Select Basic Settings=>Preferences.

Enable or disable the parameter.

Click Save Settings.



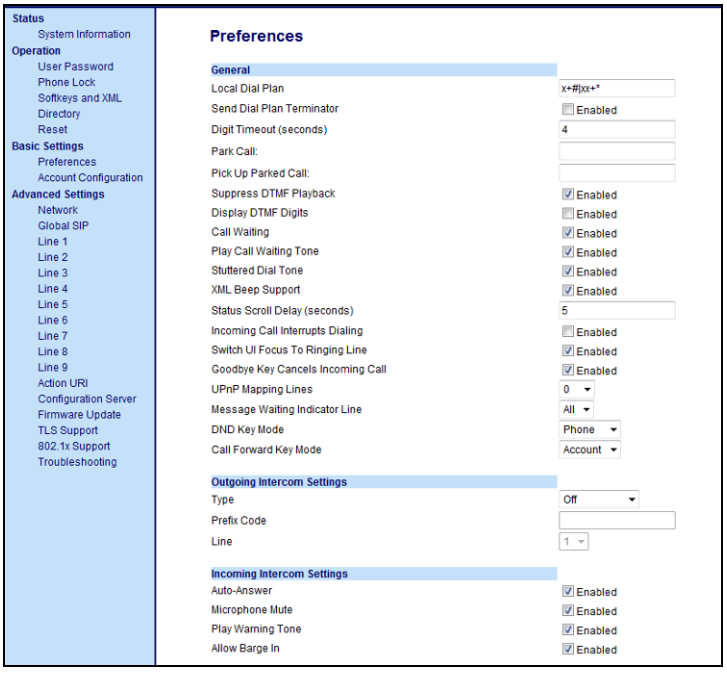
7.5 CONFIGURING THE STATUS SCROLL DELAY USING THE PHONE WEB UI

You can configure the **Status Scroll Delay** (delay in seconds, default 5) from the Web UI. It impacts the behavior of the `AastraIPPhoneStatus` object defining the delay between each message.

Select Basic Settings=>Preferences.

Enter the value in seconds.

Click Save Settings.



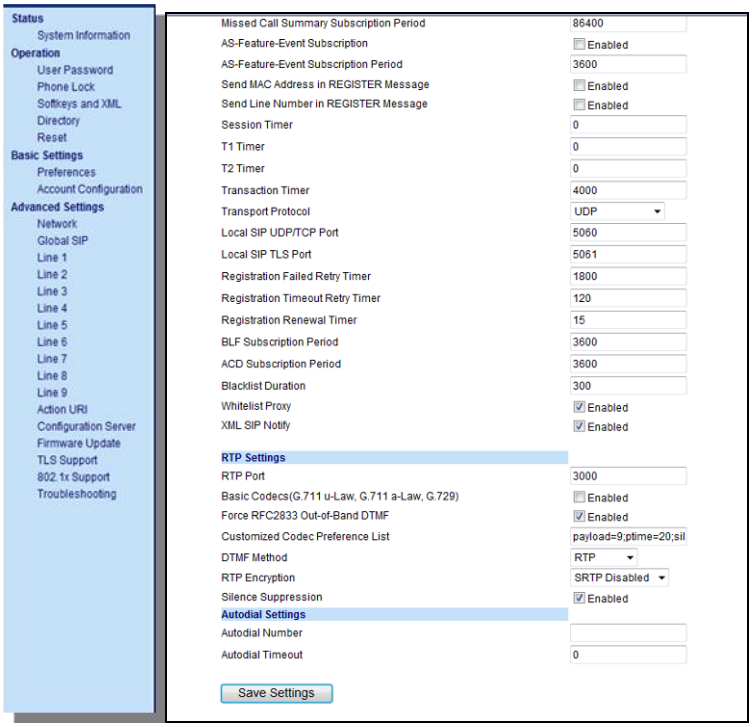
7.6 CONFIGURING THE XML SIP NOTIFY USING THE PHONE WEB UI

You can configure the **XML SIP Notify** from the Web UI. It will enable or disable SIP Notify aastra-xml to be processed by the phone.

Select Advanced Settings=>Global SIP.

Enable or disable the parameter.

Click Save Settings.



7.7 XML CONFIGURATION USING THE CONFIGURATION FILES

The *aastra.cfg* and *<mac>.cfg* Configuration File contains all the configuration parameters for the phone. Please refer to the phone administration guide for more details.

7.7.1 GENERAL XML PARAMETERS

You can configure the XML applications in the *aastra.cfg* or *<mac>.cfg* file using the following parameters:

```
xml application URI[1]
xml application title[1]
xml application post list
xml beep notification
xml status scroll delay
xml get timeout
xml lock override
services script[2]
callers list script
directory script
redial script
xfer script
conf script
icom script
voicemail script
options script
auto offhook
```

Parameter – xml application URI	Configuration Files aastra.cfg, <mac>.cfg
Description	This is the XML application you are loading into the IP phone configuration.
Format	HTTP server path or fully qualified Domain Name
Default Value	Not Applicable
Range	Not Applicable
Example	xml application URI: http://172.16.96.63/aastra/internet.php

Parameter – xml application title	Configuration Files aastra.cfg, <mac>.cfg
Description	This parameter allows you to rename the XML application in the IP phone UI (Services->4. Custom Feature). By default, when you load an XML application to the IP phone, the XML application title is called "Custom Feature". The "xml application title" parameter allows you to change that title. For example, if you are loading a traffic report XML application, you could change this parameter title to "Traffic Reports", and that title will display in the IP phone UI as Services->4. Traffic Reports.
Format	Alphanumeric characters
Default Value	Not Applicable

[1] Not supported on Mitel 53i and Mitel 9143i

[2] Not supported on Mitel 51i, Mitel 53i and Mitel 9143i

Range	Not Applicable
Example	xml application title: Traffic Reports

Parameter – xml application post list	Configuration Files aastra.cfg, <mac>.cfg
Description	The HTTP server that is pushing XML applications to the IP phone.
Format	IP address in dotted decimal format and/or Domain name address
Default Value	Not Applicable
Range	Not Applicable
Example	xml application post list: 10.50.10.53, dhcp10-53.ana.aastra.com

Parameter – xml beep notification	Configuration Files aastra.cfg, <mac>.cfg
Description	Enables or disables a BEEP notification on the phone when an AastraIPPhoneStatus object containing a “beep” attribute arrives to the phone.
Format	Boolean
Default Value	Value 1 (ON)
Range	0 (OFF) No beep is audible even if the beep attribute is present in the XML object. 1 (ON) The phone beeps when an XML object with the “beep” attribute arrives to the phone.
Example	xml beep notification: 0

Parameter – xml status scroll delay	Configuration Files aastra.cfg, <mac>.cfg
Description	Specifies the length of time, in seconds, that each XML status message displays on the phone.
Format	Integer
Default Value	5

Range	1 to 25
Example	xml status scroll delay: 3

Parameter – xml get timeout	Configuration Files aastra.cfg, <mac>.cfg
Description	Specifies the length of time, in seconds, that the phone will wait for a HTTP GET answer called by an XML.
Format	Integer
Default Value	0 (no timeout)
Range	0 to 4294967295
Example	xml get timeout: 10

Parameter – xml lock override	Configuration Files aastra.cfg, <mac>.cfg
Description	Specifies the method to use for overriding a locked phone when XML applications are sent to the phone. There are three settings for this parameter: Phone prevents XML POSTs and XML GETs from being received or sent. Phone allows XML POSTs; however, XML GETs by pressing the XML keys (softkeys/programmable keys/extension module keys) are not allowed. Phone allows XML POSTs to the phone as well as XML GETs to/from the phone by pressing the XML keys (softkeys/programmable keys/extension module keys).
Format	Integer
Default Value	0
Range	0 to 2
Example	xml lock override: 1, xml lock override: 2

Parameter – services script	Configuration Files aastra.cfg, <mac>.cfg
Description	Specifies the XML URI to call when a “Services” key (prgkey or softkey) is pressed. The URI overrides native behavior of the

	“Services” key.
Format	HTTP server path or fully qualified Domain Name
Default Value	Not Applicable
Range	Not Applicable
Example	services script: http://172.16.96.63/services.php

Parameter – callers list script	Configuration Files aastra.cfg, <mac>.cfg
Description	Specifies the XML URI to call when a “Callers List” key (prgkey or softkey) is pressed. The URI overrides native behavior of the “Callers List” key.
Format	HTTP server path or fully qualified Domain Name
Default Value	Not Applicable
Range	Not Applicable
Example	callers list script: http://172.16.96.63/callers.php

Parameter – directory script	Configuration Files aastra.cfg, <mac>.cfg
Description	Specifies the XML URI to call when a “Directory” key (prgkey or softkey) is pressed. The URI overrides native behavior of the “Directory” key.
Format	HTTP server path or fully qualified Domain Name
Default Value	Not Applicable
Range	Not Applicable
Example	directory script: http://172.16.96.63/directory.php

Parameter – redial script	Configuration Files aastra.cfg, <mac>.cfg
Description	Specifies the XML URI to call when a “Redial” key (prgkey or softkey) is pressed. The URI overrides native behavior of the “Redial” key.
Format	HTTP server path or fully qualified Domain Name

Default Value	Not Applicable
Range	Not Applicable
Example	redial script: http://172.16.96.63/redial.php

Parameter – xfer script	Configuration Files aastra.cfg, <mac>.cfg
Description	Specifies the XML URI to call when a “Xfer” key (prgkey or softkey) is pressed. The URI overrides native behavior of the “Xfer” key.
Format	HTTP server path or fully qualified Domain Name
Default Value	Not Applicable
Range	Not Applicable
Example	xfer script: http://172.16.96.63/xfer.php



Note: The Xfer key is redirected only in the “connected” and “dialing” states.

Parameter – conf script	Configuration Files aastra.cfg, <mac>.cfg
Description	Specifies the XML URI to call when a “Conference” key is pressed. The URI overrides native behavior of the “Conference” key.
Format	HTTP server path or fully qualified Domain Name
Default Value	Not Applicable
Range	Not Applicable
Example	conf script: http://172.16.96.63/conf.php



Note: The Conference key is redirected only in the “connected” state.

Parameter – icom script	Configuration Files aastra.cfg, <mac>.cfg

Description	Specifies the XML URI to call when a “Icom” key (prgkey or softkey) is pressed. The URI overrides native behavior of the “Icom” key.
Format	HTTP server path or fully qualified Domain Name
Default Value	Not Applicable
Range	Not Applicable
Example	icom script: http://172.16.96.63/icom.php

Parameter – voicemail script	Configuration Files aastra.cfg, <mac>.cfg
Description	Specifies the XML URI to call when a “Voicemail” key (prgkey or softkey) is pressed. The URI overrides native behavior of the “Voicemail” key.
Format	HTTP server path or fully qualified Domain Name
Default Value	Not Applicable
Range	Not Applicable
Example	voicemail script: http://172.16.96.63/vm.php

Parameter – options script	Configuration Files aastra.cfg, <mac>.cfg
Description	Specifies the XML URI to call when the “Options” key is pressed. The URI overrides native behavior of the “Options” key. On a normal key press of the “Options” key the XML application set in the URI is displayed On a long key press the normal/local options menu is displayed
Format	HTTP server path or fully qualified Domain Name
Default Value	Not Applicable
Range	Not Applicable
Example	options script: http://172.16.96.63/opt.php

Notes:

- If no Options URI script is configured, the local Options Menu on the phone displays as normal.
- If you configure password access to the Options Menu, this password is required when

accessing the local Option Menu, but is not required for the Options Key redirection feature.

- Pressing the Options Menu for redirection from the server does not interfere with normal operations of the phone (for example, pressing the options menu when on a call does not affect the call).
- If the phone is locked, you must unlock the phone before accessing the Options Menu redirect feature. After pressing the Options Key, the phone displays a screen that allows you to unlock the phone before continuing.

Parameter – auto offhook	Configuration Files aastra.cfg, <mac>.cfg
Description	Specifies whether or not the phone is prevented from entering the off-hook/dialing state, if the handset is off-hook for more than 2 seconds, and the call ends.
Format	Boolean
Default Value	0 (disabled)
Range	0 (disabled) 1 (enabled)
Example	auto offhook: 1

7.7.2 ACTION URI PARAMETERS

These URIs will be configurable via the configuration files using the following parameters.

- **action uri startup**: <URI to GET on startup>
- **action uri registered**: <URI to GET on successful registration>
- **action uri registration event**: <URI to GET on any registration event>
- **action uri incoming**: <URI to GET on an incoming call>
- **action uri outgoing**: <URI to GET on outgoing call>
- **action uri offhook**: <URI to GET on an off-hook event>
- **action uri onhook**: <URI to get on an on-hook event>
- **action uri poll**: <URI to get when configured timer expires>
- **action uri poll interval**: <polling interval in seconds for action uri poll>
- **action uri xml sip notify**: <URI to get when a aastra-xml SIP notify without content is received by the phone>
- **sip xml notify event**: <Enables or disables the phone to accept an aastra-xml SIP Notify event>
- **action uri connected**: <URI to get on a connect event>
- **action uri disconnected**: <URI to get on a disconnect event>

Parameter – action uri startup	**Mitel Web UI**: Advanced Settings->Action URI **Configuration Files** aastra.cfg, <mac>.cfg
Description	URI to be called at then end of the boot sequence.
Format	HTTP(s) server path or fully qualified Domain Name
Default Value	Not applicable
Range	Not applicable
Example	action uri startup: http://myserver.com/myappli.xml

Parameter – action uri registered	**Mitel Web UI**: Advanced Settings->Action URI **Configuration Files** aastra.cfg, <mac>.cfg
Description	URI to be called the first time the phone successfully registers.
Format	HTTP(s) server path or fully qualified Domain Name
Default Value	Not applicable
Range	Not applicable
Example	action uri registered: http://myserver.com/myappli.xml

Parameter – action uri registration event	**Mitel Web UI**: Advanced Settings->Action URI **Configuration Files** aastra.cfg, <mac>.cfg
Description	URI to be called for every registration event.
Format	HTTP(s) server path or fully qualified Domain Name
Default Value	Not applicable
Range	Not applicable
Example	action uri registration event: http://myserver.com/myappli.xml

Parameter – action uri incoming	**Mitel Web UI**: Advanced Settings->Action URI **Configuration Files** aastra.cfg, <mac>.cfg
Description	URI to be called every time the phone receives an incoming call.
Format	HTTP(s) server path or fully qualified Domain Name
Default Value	Not applicable
Range	Not applicable
Example	action uri incoming: http://myserver.com/myappli.xml

Parameter – action uri outgoing	**Mitel Web UI**: Advanced Settings->Action URI **Configuration Files** aastra.cfg, <mac>.cfg
Description	URI to be called every time the phone makes a valid outgoing call.
Format	HTTP(s) server path or fully qualified Domain Name
Default Value	Not applicable
Range	Not applicable
Example	action uri outgoing: http://myserver.com/myappli.xml

Parameter – action uri offhook	**Mitel Web UI**: Advanced Settings->Action URI **Configuration Files** aastra.cfg, <mac>.cfg
Description	URI to be called every time the phone goes offhook.
Format	HTTP(s) server path or fully qualified Domain Name
Default Value	Not applicable
Range	Not applicable
Example	action uri offhook: http://myserver.com/myappli.xml

Parameter – action uri onhook	**Mitel Web UI**: Advanced Settings->Action URI **Configuration Files** aastra.cfg, <mac>.cfg
Description	URI to be called every time the phone goes back on hook after an active call.
Format	HTTP(s) server path or fully qualified Domain Name
Default Value	Not applicable
Range	Not applicable
Example	action uri onhook: http://myserver.com/myappli.xml

Parameter – action uri poll	**Mitel Web UI**: Advanced Settings->Action URI **Configuration Files** aastra.cfg, <mac>.cfg
Description	URI to be called every "action uri pool interval" seconds.
Format	HTTP(s) server path or fully qualified Domain Name
Default Value	Not applicable
Range	Not applicable
Example	action uri poll: http://myserver.com/myappli.xml

Parameter – action uri poll interval	**Mitel Web UI**: Advanced Settings->Action URI **Configuration Files** aastra.cfg, <mac>.cfg
Description	Specifies the interval, in seconds, between calls from the phone to the "action uri poll".
Format	Integer
Default Value	0 (disabled)
Range	Not applicable
Example	action uri poll interval:60

Parameter – action uri xml sip notify	**Mitel Web UI**: Advanced Settings->Action URI **Configuration Files** aastra.cfg, <mac>.cfg
Description	URI to be called when an empty XML SIP NOTIFY is received ny the phone (sip xml notify event must be enabled).
Format	HTTP(s) server path or fully qualified Domain Name
Default Value	Not applicable
Range	Not applicable
Example	action uri xml sip notify: http://myserver.com/myappli.xml

Parameter – sip xml notify event	**Mitel Web UI**: Advanced Settings->Global SIP-> Advanced SIP Settings **Configuration Files** aastra.cfg, <mac>.cfg
Description	Enables or disables the phone to accept an aastra-xml SIP NOTIFYmessage.
Format	Boolean
Default Value	0
Range	0 - disabled 1 – enabled
Example	sip xml notify event:1

Parameter – action uri connected	**Mitel Web UI**: Advanced Settings->Action URI **Configuration Files** aastra.cfg, <mac>.cfg
Description	URI to be called every time the phone goes to the connected state.
Format	HTTP(s) server path or fully qualified Domain Name
Default Value	Not applicable
Range	Not applicable
Example	action uri connected: http://myserver.com/co.php

Parameter – action uri disconnected	**Mitel Web UI**: Advanced Settings->Action URI **Configuration Files** aastra.cfg, <mac>.cfg
Description	URI to be called every time the phone goes back to the idle state.
Format	HTTP(s) server path or fully qualified Domain Name
Default Value	Not applicable
Range	Not applicable
Example	action uri disconnected: http://myserver.com/di.php

7.7.3 PROGRAMMABLE AND SOFT KEYS

You can configure keys calling an XML application using the following parameters.

```
softkeyX/prgkeyX/topsoftkeyX/expmodT keyX type: xml
softkeyX/prgkeyX/topsoftkeyX/expmodT keyX value: http://someapp.xml
```

As described in chapter 5.2, system variables can be used in the URI to be called by pressing a key.

7.7.4 EXAMPLES

Example (6863i/6865i)

```
# XML configuration
xml application URI: http://172.16.96.63/aastra/internet.php
xml application post list: 10.10.50.53, xmlserver.aastra.com
xml beep notification: 1
xml status scroll delay: 5

# Key 1
prgkey1 type: xml
prgkey1 value: http://172.16.96.63/aastra/internet.php

# Key 2
prgkey2 type: xml
prgkey2 value: http://myserver.com/login.php?user=$$SIPUSERNAME$$
```

Example (6867i/6869i/6873i/6920/6930/6940)

```
# XML configuration
xml beep notification: 1
xml status scroll delay: 5

# Softkey 1
softkey1 type: xml
softkey1 label: My XML
softkey1 value: http://172.16.96.63/aastra/internet.php

# Softkey 2
softkey2 type: xml
softkey2 label: Login
softkey2 value: http://myserver.com/login.php?user=$$SIPUSERNAME$$
```

8 TROUBLESHOOTING XML APPLICATIONS

8.1 INTRODUCTION

The following figure shows the HTTP call flows when the phone is performing an XML operation.

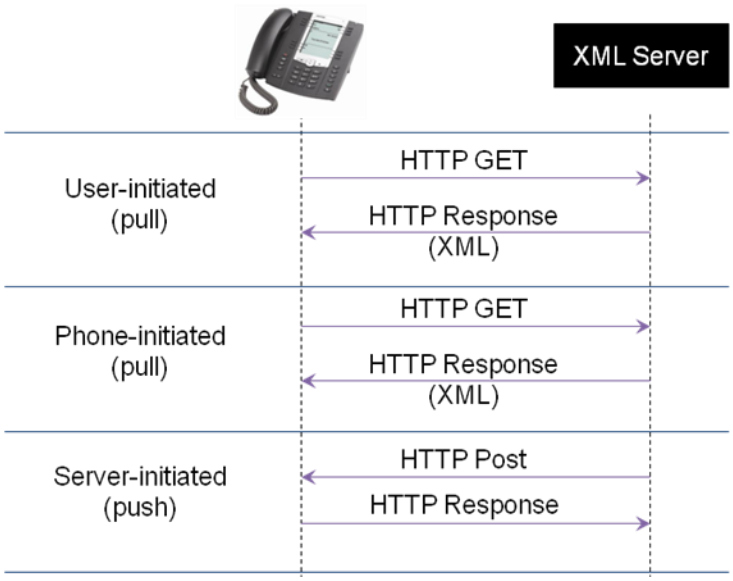


Figure 92: HTTP flow for XML applications

When the phone performs an HTTP GET there are many potential sources of error such as:

Server is not reachable	Answer
Network is down	Cannot display
Server is down	Cannot display
HTTP server application is down	Cannot display
URI can not be resolved (DNS issue)	Cannot display
Server is reachable	**Answer**
Wrong file name on the server	File not found
Parsing error of the XML answer	Cannot display
Timeout reached before getting an answer	Connect Timeout
URI can not be parsed (wrong format)	Invalid URI

Most of the time, an error will translate into a “Cannot display” which is not very helpful to find the origin of the problem as well as very frustrating for the XML developer.

The most common error is a XML parsing error and often the most difficult one to fix.

Note: Using the php classes provided in the toolkit is a good way to limit the number of parsing errors as the object are built following the XSD scheme.

When the server performs a PUSH to the phone, the common errors are:

Nothing happens, most of the time the problem comes from a wrong configuration of the XML push list.

“Cannot display”, if the XML object pushed to the phone creates a parsing error (object not properly formatted).

8.2 TROUBLESHOOTING TOOLS

The following tools will help you troubleshoot problems with the XML services.

Standard web browser (Microsoft Internet Explorer 6.0 or a later version)

- Verify the connectivity
- Verify the validity of the URI called by the phone
- Verify if the XML answer has a correct syntax
- Network packet analyzer such as Wireshark
- Verify what is exchanged between the phone and the server
- HTTP Server log
- Verify if the HTTP GET reached the server
- Verify the parameters of the HTTP GET
- XML validator tools
- Verify the XML syntax and the compliance with the XSD model
- Phone log (syslog)
- Verify how the phone processes a XML request

This last tool is the most powerful as it allows you to read phone internal traces and understand why something went wrong. It is the best way to debug a parsing error.

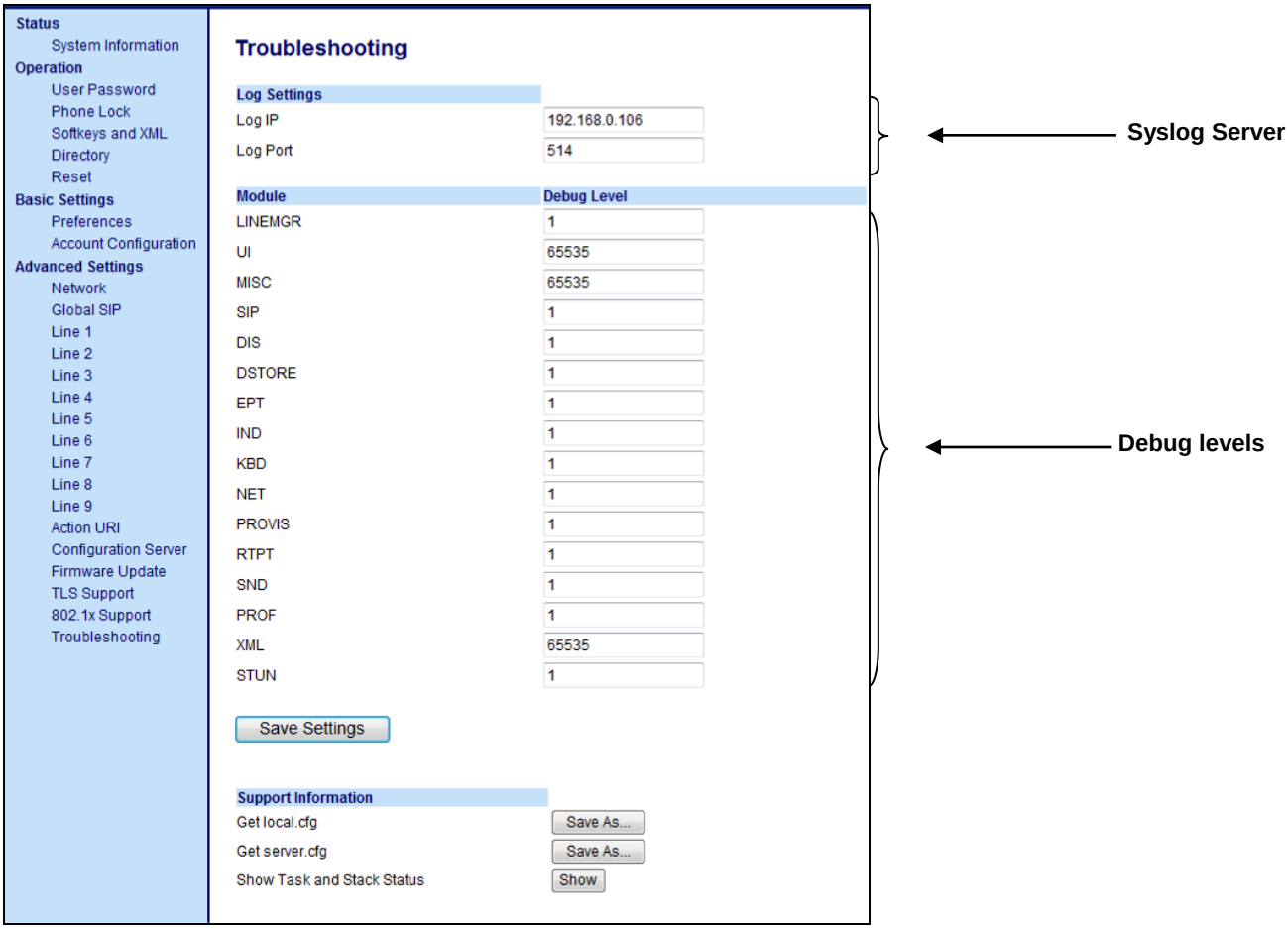
The phone can be configured to send internal traces to a Syslog server (such as Kiwi Syslog Deamon) either from the WebUI or from the configuration files.

8.3 CONFIGURING THE SYSLOG SERVER USING THE WEBUI

You can configure the **Syslog server and activate the traces** from the Web UI.

- Select Advanced Settings=>Troubleshooting.
- Set Log IP (IP address of the Syslog server) and Log port (usually 514)
- Set XML to 65535
- Click 

.



8.4 CONFIGURING THE SYSLOG SERVER USING THE CONFIGURATION FILES

The traces can also be activated from the configuration files (aastra.cfg and/or <MAC>.cfg), using the following parameters:

Syslog server

- log server ip IP address of the Syslog server
- log server port Port of the Syslog server

Traces level

- log module xml debug level for xml module

Example

```
# Syslog Server
log server ip: 192.168.0.106
log server port: 514

# Debug list
log module xml: 65535
```

8.5 PARSING ERROR DEBUG EXAMPLE

In this example, the phone is making an XML GET and receives the following XML object as an answer. The answer has an obvious parsing error, the phone displays ‘Cannot display’.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<AastraIPPhoneTextMenu>
<Title>Graphic Apps Demo</Title>
<MenuItem>
<Prompt>Clock Digital GMT+1</Prompt>
<URI>http://192.168.0.112/xml/lucaGD/demo.php?type=dclock</URI>
</MenuItem>
<MenuItem>
<Prompt>Clock Analog GMT+1</Prompt>
<URI>http://192.168.0.112/xml/lucaGD/demo.php?type=aclock</URI>
</MenuItem>
<MenuItem>
<Prompt>Different Fonts</Prompt>
<URI>http://192.168.0.112/xml/lucaGD/demo.php?type=font</URI>
</MenuItem>
<AastraIPPhoneTextMenu>
```

The error is on the last line, it should be `</AastraIPPhoneTextMenu>`.

The following Syslog traces tell us where the problem is.

```
CreateApp:    (UI)    INFO:    XML=|<?xml    version="1.0"    encoding="ISO-8859-
1"?><010><AastraIPPhoneTextMenu><010><Title>Graphic                    Apps
Demo</Title><010><MenuItem><010><Prompt>Clock                       Digital
GMT+1</Prompt><010><URI>http://192.168.0.112/xml/lucaGD/demo.php?type=dclock</U
RI><010></MenuItem><010><MenuItem><010><Prompt>Clock                    Analog
GMT+1</Prompt><010><URI>http://192.168.0.112/xml/lucaGD/demo.php?type=aclock</U
RI><010></MenuItem><010><MenuItem><010><Prompt>Different
Fonts</Prompt><010><URI>http://192.168.0.112/xml/lucaGD/demo.php?type=font</URI
><010></MenuItem><010><AastraIPPhoneTextMenu><010>|<010>mac:00-08-5D-1A-3C-54

ParserData: (XML) FUNC: ParserData ctor<010>mac:00-08-5D-1A-3C-54

TextMenuData: (XML) FUNC: TextMenuData ctor<010>mac:00-08-5D-1A-3C-54

startTagHandler: (XML) ERROR: Invalid tag: <AastraIPPhoneTextMenu> in the Root
state<010>mac:00-08-5D-1A-3C-54

TextMenuData: (XML) FUNC: TextMenuData dtor<010>mac:00-08-5D-1A-3C-54

ParserData: (XML) FUNC: ParserData dtor<010>mac:00-08-5D-1A-3C-54

CreateApp:  (UI)  ERROR:  !!!!!!!!!!!!!!!!!!!  XML  parsing  error:  no  element
found, Line 16, Column 23 !!!!!!!!!!!!!!!!!!!<010>mac:00-08-5D-1A-3C-54

ParserData: (XML) FUNC: ParserData ctor<010>mac:00-08-5D-1A-3C-54
```

9 WHY XML APPLICATIONS FOR AN IP PHONE?

XML applications can be split in 3 categories:

- Telephony applications, integration with the call processing, Voice Mail server, Conference server, Call Center application…
- Media and information
- Vertical applications

9.1 TELEPHONY APPLICATIONS

This chapter details potential XML telephony applications which could be developed to enhance integration of an IP phone with the other telecom applications.

9.1.1 DIRECTORY

The first obvious applications that can be developed are the directory application, it includes:

- PBX directory
- Corporate directory (Global list from a Microsoft Exchange server)
- Personal contacts (My contacts in Outlook)
- Any LDAP directory (public or private)

9.1.2 CALL PROCESSING

XML applications can also be used to develop interactions between the call processing and the phone:

- DND
- Call Forward
- Parked calls
- Call pickup
- PBX configuration

9.1.3 VOICE-MAIL

- Voice mail messages management (play, skip, delete, …)
- Display message envelopes
- Presence management

9.1.4 CONFERENCE BRIDGE

- Conference booking
- Conference reminder
- Audio console

9.1.5 CONTACT CENTER

- Agent Login/Logout
- Access to call center reports
- Account codes
- Wrap codes

9.2 MEDIA AND INFORMATION

These are the applications getting information from the Web.

- Weather Alerts
- Stock Alerts
- Stock Prices
- Worldwide Time/Temperature
- Server Alarms and Notifications
- Server Status
- Account Balances
- Current Gas Prices
- Local Movie Times
- Upcoming Concerts-By Category
- Order Flowers-by Category
- Send Order to Starbucks
- Send SMS Messages
- Track FedEx Package (or Airborne, UPS, etc.)
- Calling Card Minutes Remaining
- Reserve Meeting Rooms
- Contact center Metrics (Calls Waiting, Longest Hold, Performance against Service Level, etc.)
- Pro Sports Scores, Vegas Betting Lines
- Multitude of Banking apps: Balances, Transfers, etc.
- Language Translation
- Daily Horoscope
- Broadcast Joke Of The Day/Inspirational Quote of the Day

9.3 VERTICAL APPLICATIONS

This chapter details potential vertical applications that can be developed as an XML application for the Mitel IP phones. The list is far from being exhaustive.

9.3.1 HUMAN RESOURCES

- Available Vacation Days
- Available Personal Days
- 401K balance
- Clock-In/Clock-Out

9.3.2 TRAVEL/HOTEL

- Current Balance
- In-Room Dining Ordering
- Delivery Dining Options
- Extend Stay
- Schedule Airport Shuttle
- Request Housekeeping/Engineering
- Leave Feedback

- Wake-Up Call
- Book Corporate Travel
- Do Not Disturb

9.3.3 HEALTH CARE

- Test Results
- Manage Appointment
- Appointment Reminders
- Take-Your-Medicine Reminders
- Order Hospital Meals
- Check Pharmacy Inventory
- Schedule Blood Donation

9.3.4 EDUCATION

- Attendance
- Request Substitute Teacher (used by primary teacher)
- Review Open Requests for Substitute Teacher (used by potential subs)
- Schedule Classes
- Request Dorm Room Change
- School Closing Notification/Status
- Parent Contact Info

9.3.5 LAW ENFORCEMENT

- Amber Alerts
- Traffic Ticket Plead By Phone
- Fugitive Alerts

10 PHONE SELF-CONFIGURATION USING XML

10.1 INTRODUCTION

The deployment of a SIP phone is not a simple task; you have to face 2 challenges:

- Provide the address of the configuration server to the phone
- Link the MAC address of the phone with a SIP extension

The first challenge is usually solved by using DHCP option 66 (bootp) to provision the phone with the IP address/name of the configuration server (TFTP, FTP, HTTP or HTTPS).

The second challenge is more difficult as you have to know the MAC address of the phone in advance in order to prepare the specific configuration file the phone will use. Usually, each phone is identified (scan of the MAC address) and then linked to an extension.

It is possible to have a complete self-configuration of the phone using an XML application called by the action uri startup at the end of the boot sequence.

10.2 MESSAGE FLOW

It is possible for a third-party to develop an automatic configuration process. The following is a description of how this can be done using existing phone features.

The aastra.cfg file sets the startup action uri configuration parameter to point to the configuration script and configuration download information.

Phone downloads the aastra.cfg file, ignores missing <MAC>.cfg file and continues boot process.

Phone executes startup uri, running the configuration script. The MAC address of the phone and the phone model are in the HTTP headers of the request.

The script uses XML to gather required configuration information and creates <mac>.cfg file. The <mac>.cfg file must reset the startup action uri to avoid the configuration script being called on subsequent boots.

The script reboots the phone via XML reboot command or via SIP check-sync message.

Phone reboots, directly downloads both aastra.cfg and newly created <MAC>.cfg file

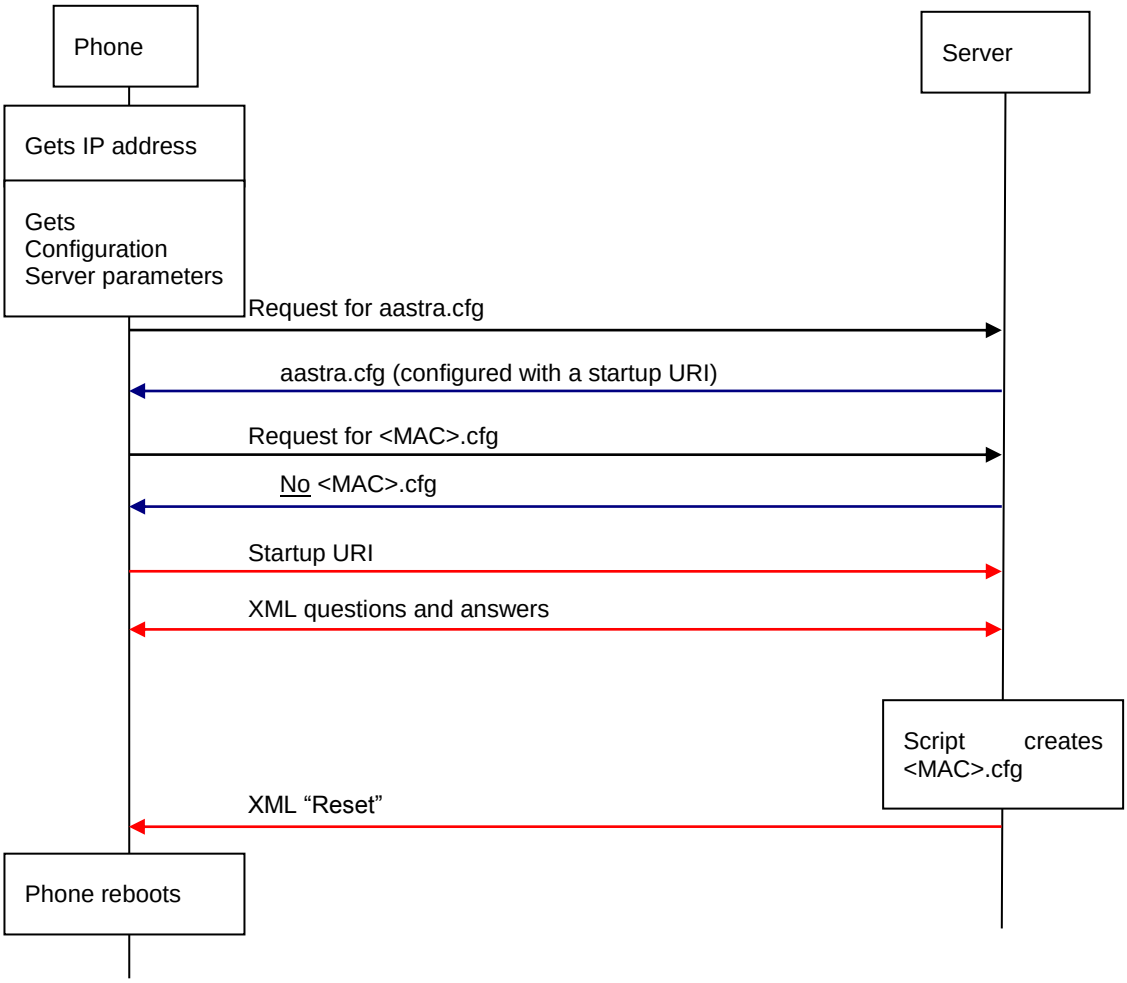


Figure 93: Auto-configuration message flow

10.3 AUTO-CONFIGURATION POLICY

The auto-configuration policy which includes the flow of questions asked to the user and the script to generate the <MAC>.cfg file is totally open with this mechanism.

Multiple options are available.

Extension is already provisioned in the IP-PBX database.

One way to implement this feature might be to have all the extensions already provisioned on the switch side and the XML flow will be used to identify the user (extension number and voicemail password for instance). The script must then control if the extension is not already assigned and create the <MAC>.cfg based on the extension(s) configuration.

Another way would be to display the list of available extensions and let the user select his extension; of course password protection is needed to avoid any hacking of the platform.

Extension is not provisioned in the IP-PBX database

A second option is to have the script to provision the extension in the database. To do so, the script can ask for user general parameters (name…) and automatically creates an extension in the switch database and then creates the <MAC>.cfg, the extension number can be either automatically assigned or the user can select it in a list of available extensions.

10.4 ARCHITECTURE

The following diagram represents the architecture of what needs to be developed to implement the Mitel self-configuration mechanism.

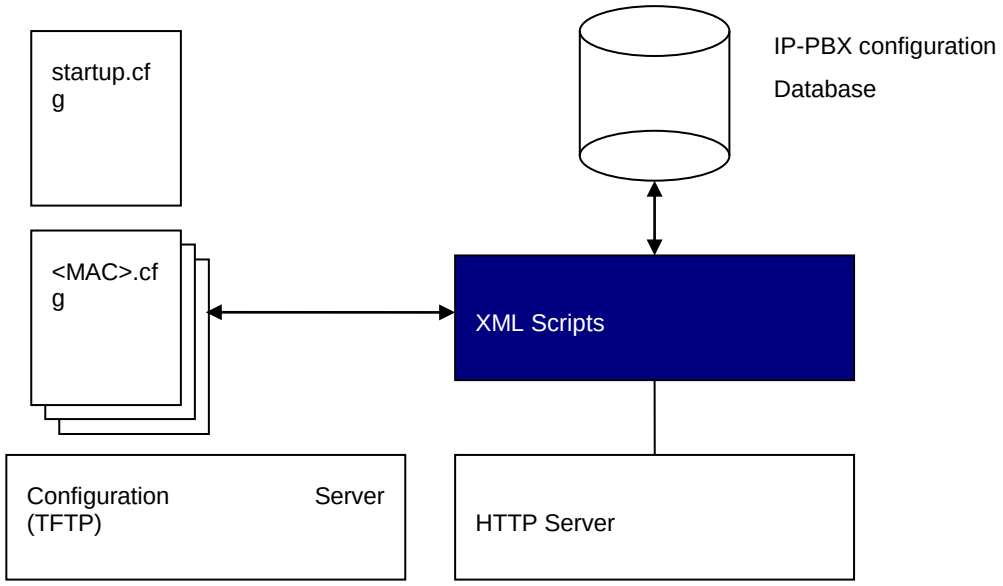


Figure 94: Self-configuration implementation architecture

The development effort to implement the self-configuration is straight forward but the complexity depends on the policy to attribute extensions on the phone system.

11 SAMPLE XML APPLICATIONS

Here is the list of XML applications provided by Mitel, these XML applications are available either through an Internet server hosted by Mitel or as source code (part of the Mitel XML API SDK) to be executed from a local PC running XAMPP for example.

- Area code (US)
- Biorhythms
- CNN News
- Currency Converter
- ESPN News
- Fox News
- Horoscope
- IP Geolocation
- Local Weather (US)
- Movies
- Netflix
- Stock
- Today…
- World Clock
- Yahtzee
- All the above

11.1 ACCESS FROM THE INTERNET

Mitel has made available, **for demonstration purpose only**, a list of XML applications on the Internet.

Notes:

Mitel does not guarantee the availability of these applications. The applications can change at any time without notice.

These applications should not be used commercially; any abusive use of these applications will be detected and the phone will be automatically banned from the applications.

your Mitel SIP phone must have Internet access to use these applications.

The XML applications can be configured individually on a phone key or as a global menu.

11.2 LOCAL SERVER USING XAMPP

XAMPP is a free, cross-platform, easy-to-use web server capable of serving dynamic pages. XAMPP consists mainly of the Apache HTTP Server, MySQL database and interpreters for scripts written in the PHP and Perl programming languages. The program is released under the GNU General Public License.

Notes: your server must have Internet access to use these applications.

XAMPP is an acronym for:

- **X** (any of the four operating systems Windows, Linux, Sun Solaris and Mac OS X)
- **A**pache
- **M**ySQL
- **P**HP
- **P**erl

The XAMPP 1.6.6 Basic Package includes the following main components:

- Apache HTTPD 2.2.8 + OpenSSL 0.9.8g
- MySQL 5.0.51
- PHP 5.2.5 + PHP 4.4.8 rc2 dev + PEAR
- SQLite 2.8.15

- phpMyAdmin 2.11.4
- Mercury Mail Transport System v4.52
- FileZilla FTP Server 0.9.25

Note: the XML scripts provided by Mitel use only Apache and PHP.

11.2.1 XAMPP INSTALLATION

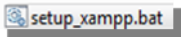
The nice thing with XAMPP is that it does not need to be installed. Simply extracting the archive to the root folder of your disk is enough. No registry keys are written, no files are copied to the Windows directory. You can even put XAMPP on a memory stick and run it from there. This makes it very portable and easy to use.

XAMPP can be downloaded from http://www.apachefriends.org/en/xampp.html, where you can select which Operating System you want to use XAMPP.

Please refer to the XAMPP web site for detailed information on how to install the software.

Notes:

- Make sure you extract XAMPP to the root directory of your disk or memory stick (e.g. C:\), otherwise you may run into troubles later
- If you have an existing PHP installation on your computer, please uninstall it or remove the Windows environment variable PHPRC (open the Windows Control Panel, go to System, Advanced, Environment Variables and remove the PHPRC variable). Otherwise XAMPP might use the wrong PHP settings.

Once XAMPP is installed, just run 

which is located in the xampp directory. This configures the installation of XAMPP on your computer.

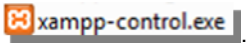
To uninstall, just remove the xampp directory.

11.2.2 XML SCRIPTS INSTALLATION

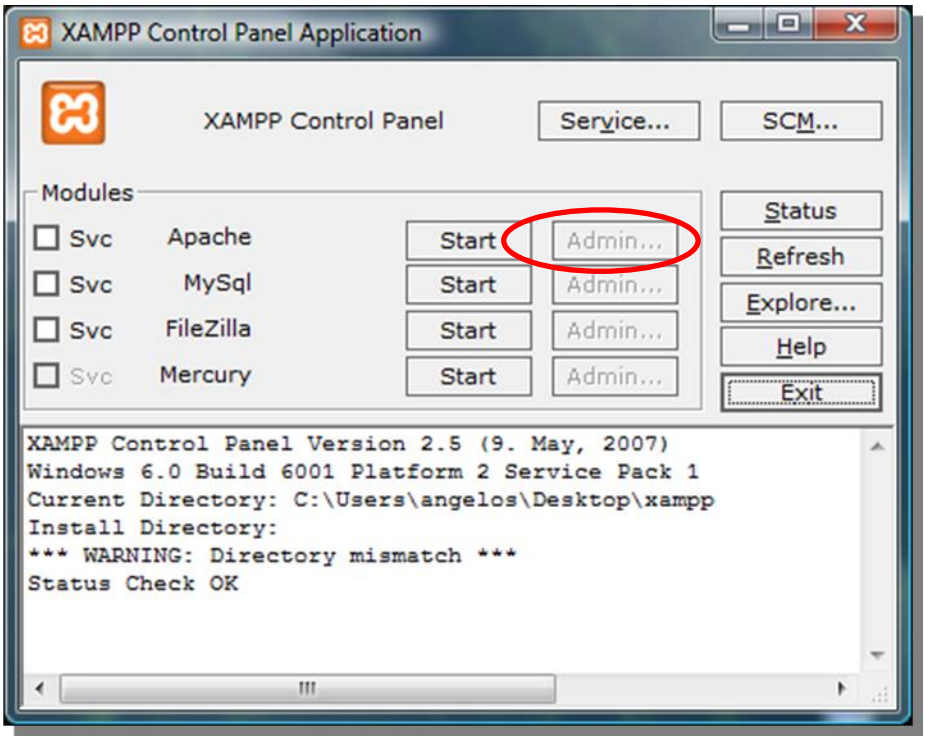
Just create a directory named “xml” under “xampp/htdocs” directory and unzip the xml-xampp.zip file provided in the Mitel XML API SDK in this directory.

Also create a “cache” directory under the xampp directory. This directory will be used to cache data on the server.

11.2.3 XAMPP START AND STOP

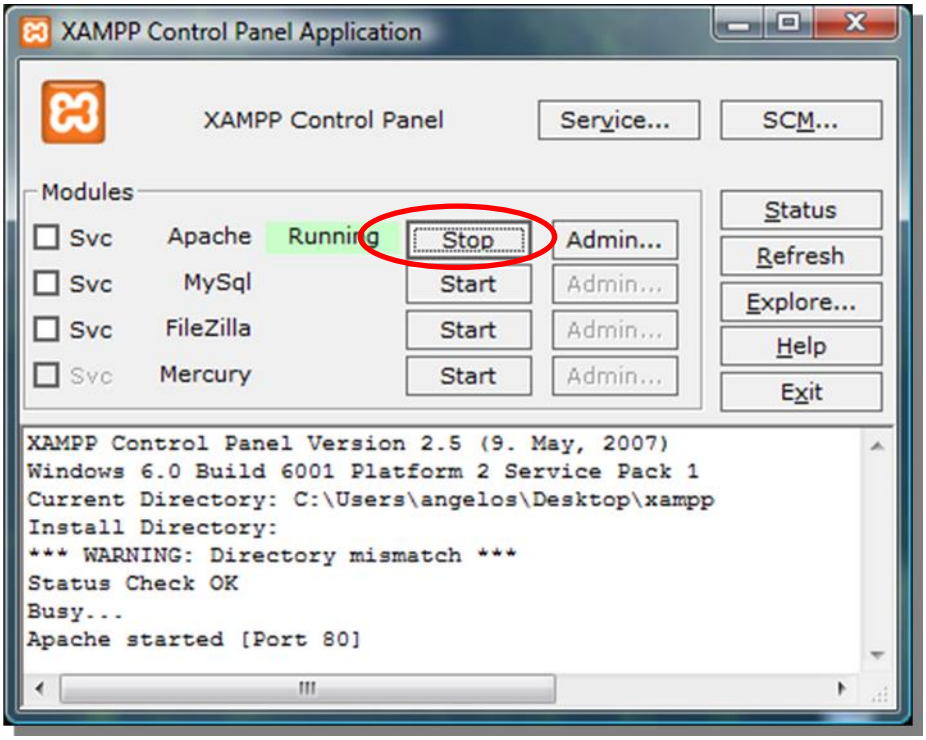
In the xampp directory run 

. To start the Apache HTTP server, just click on the “Start” button next to the Apache module.



Apache is now running.

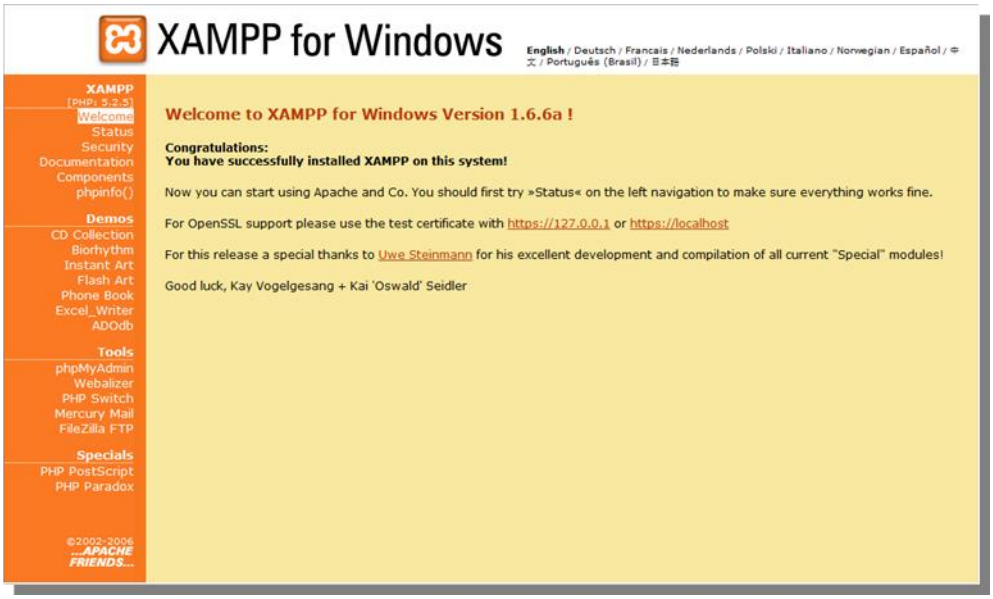
Note: In case you have a firewall running (native for Windows XP or Vista), you will be prompted to authorize Apache HTTP Server.



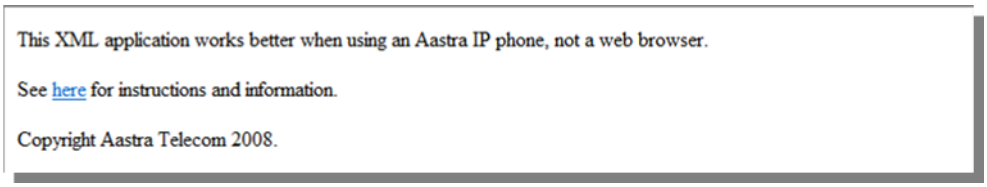
To stop Apache, simply click on the Stop-button. To close the XAMPP Control Panel, click on the Exit button.

11.2.4 TEST YOUR INSTALLATION

To test the HTTP server, direct your Web browser to http://localhost/xampp/. You should see the following page:



To test the XML scripts, direct your Web browser to http://localhost/xml/area/area.php. You should see the following page:



11.2.5 TROUBLESHOOTING APACHE

Most common problem is that another program is already using TCP server port 80 (HTTP) and port 443 (HTTPS).

Run the program xampp-portcheck.exe in the xampp directory.

Check which program is using server port 80 and/or 443. Shutdown that program.

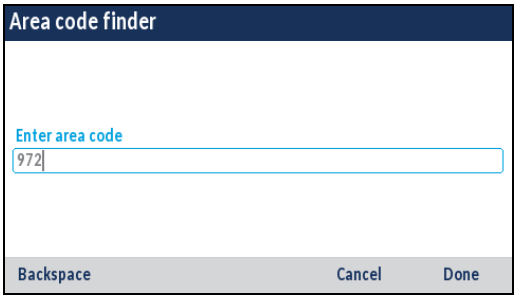
11.3 APPLICATIONS

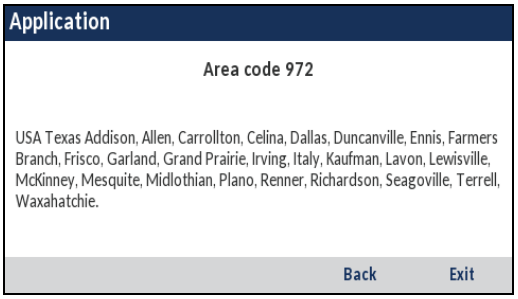
This chapter describes all the sample applications available either from the Internet or from XAMPP running on a server (represented below as myserver.com).

11.3.1 AREA CODE LOOKUP (US)

Description			
This application allows the user to lookup for the State/Cities of any given US area code.			
Phone compatibility			
✓	Mitel 6863i/6865i	✓	Mitel 6867i/6869i/6873i/6920/6930/6940
Configuration (Internet)			
uri=http://65.36.55.137/xml/area/area.php			

Configuration (XAMPP)
uri=http://myserver.com/xml/area/area.php
Screenshots



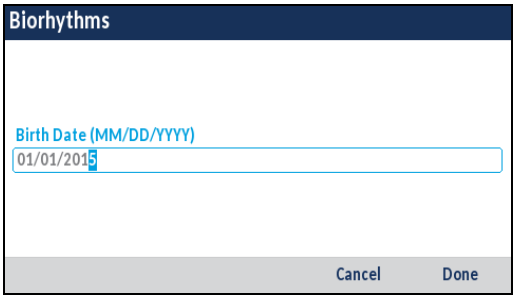


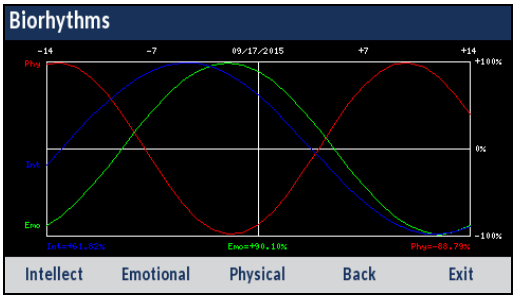
11.3.2 BIORHYTHMS

Description
Check your Biorhythms.

Phone compatibility			
✓	Mitel 6863i/6865i	✓	Mitel 6867i/6869i/6873i/6920/6930/6940

Configuration (Internet)
uri=http://65.36.55.137/xml/games/biorhythms.php
Configuration (XAMPP)
uri=http://myserver.com/xml/games/biorhythms.php
Screenshots (6867i/6969i/6873i)





Comments
User's birth date is stored on the server. Only phones with graphical display (6867i, 6869i, 6873i, 6920, 6930 and 6940) will display the charts PHP-GD extension is needed for this XML script.

11.3.3 CNN NEWS

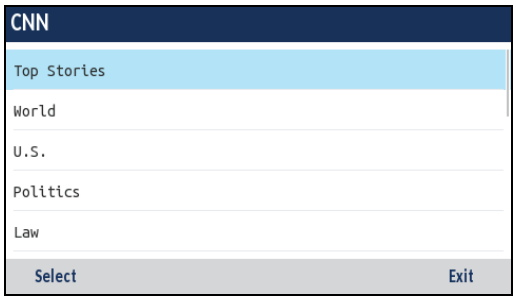
Description
RSS feed from CNN.com including the following topics: Top Stories World U.S. Politics Law Technology Science and Space Health Entertainment Travel Education Video Offbeat Most Popular Most Recent

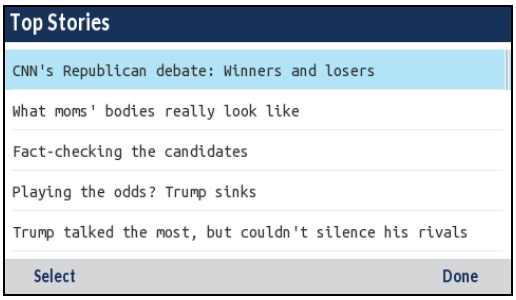
Phone compatibility			
✓	Mitel 6863i/6865i	✓	Mitel 6867i/6869i/6873i/6920/6930/6940

Configuration (Internet)
uri=http://65.36.55.137/xml/rss/rss.php?feed=cnn

Configuration (XAMPP)
uri=http://myserver.com/xml/rss/rss.php?feed=cnn

Screenshots





11.3.4 CURRENCY CONVERTER

Description
This application uses www.yahoo.com to convert any currency in to another currency. It also allows the user to setup a list of favorite conversions.

Phone compatibility			
×	Mitel 6863i/6865i	✓	Mitel 6867i/6869i/6873i/6920/6930/6940

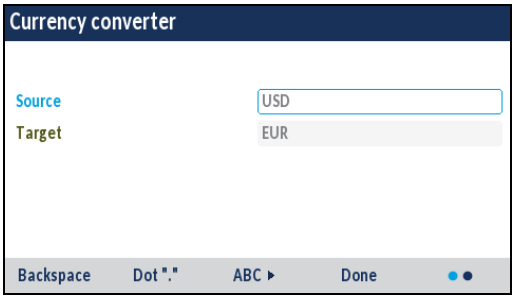
Configuration (Internet)

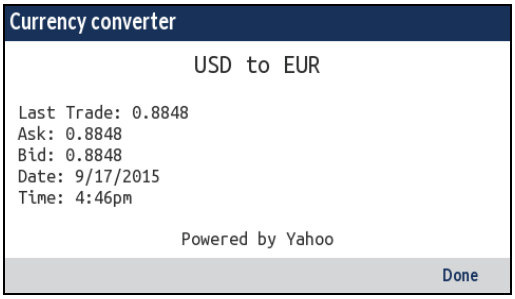
uri=http://65.36.55.137/xml/stock/currency.php

Configuration (XAMPP)

uri=http://myserver.com/xml/stock/currency.php

Screenshots





Comments

User's last request and favorites are stored on the server.

11.3.5 ESPN NEWS

Description

RSS feed from ESPN.com bringing news for the most popular sports in North America.

Top Headlines	Motorsports
NFL	Soccer
NBA	College Basketball
MLB	College Football
NHL	

Phone compatibility

✓	Mitel 6863i/6865i	✓	Mitel 6867i/6869i/6873i/6920/6930/6940

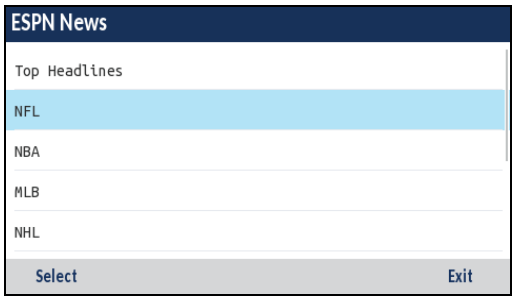
Configuration (Internet)

uri=http://65.36.55.137/xml/rss/rss.php?feed=espn

Configuration (XAMPP)

uri=http://myserver.com/xml/rss/rss.php?feed=espn

Screenshots



11.3.6 FOX NEWS

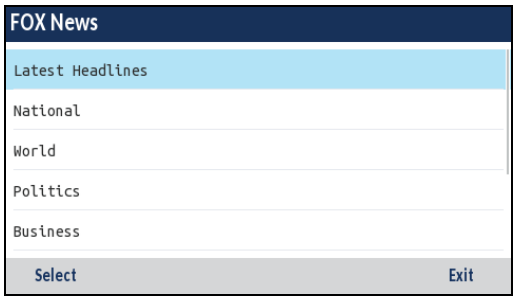
Description
RSS feed from foxnews.com including the following topics: Latest Headlines — Business National — SciTech World — Health Politics — Entertainment

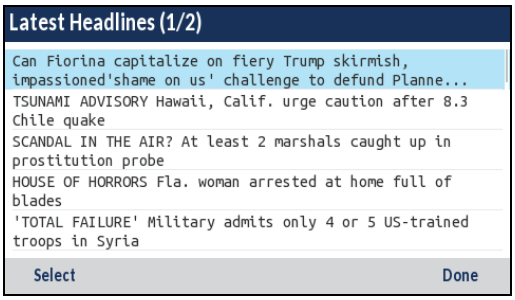
Phone compatibility			
✓	Mitel 6863i/6865i	✓	Mitel 6867i/6869i/6873i/6920/6930/6940

Configuration (Internet)
uri=http://65.36.55.137/xml/rss/rss.php?feed=fox

Configuration (XAMPP)
uri=http://myserver.com/xml/rss/rss.php?feed=fox

Screenshots





11.3.7 HOROSCOPE

Description
RSS feed from dailyhoroscopes.com.

Phone compatibility

✓	Mitel 6863i/6865i	✓	Mitel 6867i/6869i/6873i/6920/6930/6940

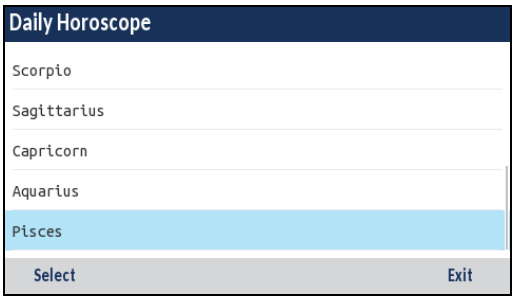
Configuration (Internet)

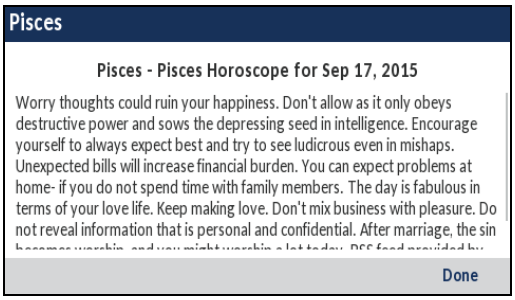
uri=http://65.36.55.137/xml/rss/rss.php?feed=horoscope

Configuration (XAMPP)

uri=http://myserver.com/xml/rss/rss.php?feed=horoscope

Screenshots





11.3.8 IP GEOLOCATION

Description

Identify the location (Country, City…) of any given public IP address. This application uses an API provided by http://www.geoplugin.net.

Phone compatibility

✓	Mitel 6863i/6865i	✓	Mitel 6867i/6869i/6873i/6920/6930/6940

Configuration (Internet)

uri=http://65.36.55.137/xml/area/ip.php

Configuration (XAMPP)

uri=http://myserver.com/xml/area/ip.php

11.3.9 MOVIES

Description

RSS feed from yahoo.com including the following topics:

Top 10 Box Office

Opening This Week

Coming Soon

Phone compatibility

✓	Mitel 6863i/6865i	✓	Mitel 6867i/6869i/6873i/6920/6930/6940

Configuration (Internet)

uri=http://65.36.55.137/xml/rss/rss.php?feed=movies

Configuration (XAMPP)

uri=http://myserver.com/xml/rss/rss.php?feed=movies

11.3.10 NETFLIX

Description

RSS feed from netflix.com including the following topics:

Top 100	Horror Top 25
New Releases*	Sci-Fi Top 25
Action/adventure Top 25	Television Top 25
Comedy Top 25	Thrillers Top 25
Drama Top 25	

Phone compatibility

✓	Mitel 6863i/6865i	✓	Mitel 6867i/6869i/6873i/6920/6930/6940

Configuration (Internet)

uri=http://65.36.55.137/xml/rss/rss.php?feed=netflix

Configuration (XAMPP)

uri=http://myserver.com/xml/rss/rss.php?feed=netflix

Comments

*Not available if you are using XAMPP under Microsoft Windows due to memory limitation for PHP.

11.3.11 STOCK QUOTES

Description

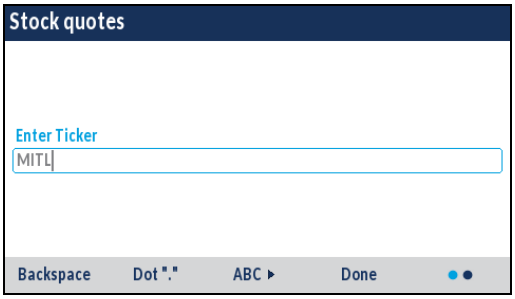
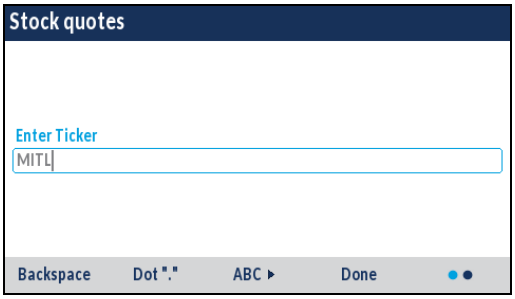
This application uses www.yahoo.com to get the value of any given stock. Please refer to yahoo.com for the syntax of the stock ticker.

Phone compatibility

✓	Mitel 6863i/6865i	✓	Mitel 6867i/6869i/6873i/6920/6930/6940

Configuration (Internet)

uri=http://65.36.55.137/xml/stock/stock.php

Configuration (XAMPP)
uri=http://myserver.com/xml/stock/stock.php
Screenshots
 
Comments
User's last request and favorites are stored on the server. Only large screen phones have access to the favorites.

11.3.12 TODAY…

Description
RSS feed from answers.com including the following topics: Word of the Day Birthdays today This day in History Quote of the Day

Phone compatibility			
✓	Mitel 6863i/6865i	✓	Mitel 6867i/6869i/6873i/6920/6930/6940

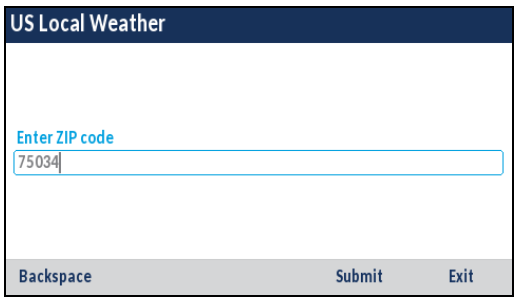
Configuration (Internet)
uri=http://65.205.17.13/xml/rss/rss.php?feed=day
Configuration (XAMPP)
uri=http://myserver.com/xml/rss/rss.php?feed=day

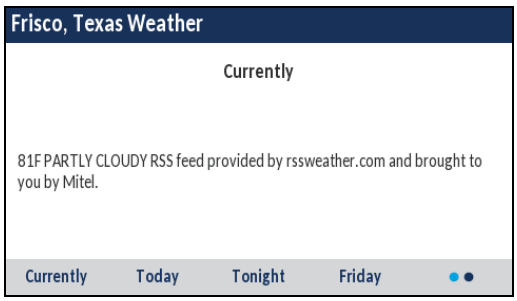
11.3.13 LOCAL WEATHER

Description
RSS feed from rssweather.com providing weather forecast for any given US ZIP code

Phone compatibility			
✓	Mitel 6863i/6865i	✓	Mitel 6867i/6869i/6873i/6920/6930/6940

Configuration (Internet)
uri=http://65.36.55.137/xml/weather/weather.php
Configuration (XAMPP)
uri=http://myserver.com/xml/weather/weather.php
Screenshots





Comments
User's last request is stored on the server

11.3.14 YAHTZEE

Description			
A Yahtzee game…			
Phone compatibility			
×	Mitel 6863i/6865i	✓	Mitel 6867i/6869i/6873i/6920/6930/6940
Configuration (Internet)			
uri=http://65.36.55.137/xml/games/yahtzee.php			
Configuration (XAMPP)			
uri=http://myserver.com/xml/games/yahtzee.php			

11.3.15 GLOBAL MENU

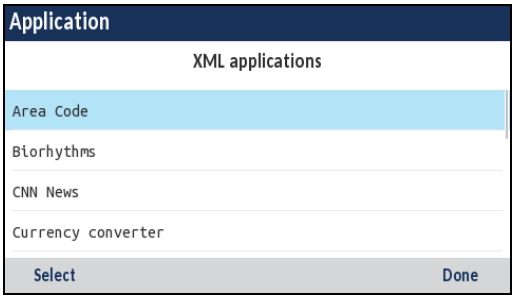
Description			
This application is an XML application agragator. It can be configured as an XML key or as the XML custom menu behind the "Services" key (not supported on 9143i and 6753i).			
Phone compatibility			
✓	Mitel 6863i/6865i	✓	Mitel 6867i/6869i/6873i/6920/6930/6940
Configuration (Internet)			

uri=http://65.36.55.137/xml/menu/mymenu.php

Configuration (XAMPP)

uri=http://myserver.com/xml/menu/mymenu.php

Screenshots



Comments

Only Mitel phones with softkeys are able to customize the list of applications.

12 APPENDIX A: XSL MODEL

The xsl model is provided as an xsd file in this SDK.

13 APPENDIX B: OBJECT ORIENTED PHP CLASSES

Mitel also provides an object oriented API to develop XML applications which is included in the XML SDK.

Note: The PHP objects are taking care of the XML escape encoding when they are needed.

13.1 AASTRAIPPHONECALLLOG()

This class allows you to create a XML PhoneCallLog object.

Include

AastraIPPhoneCallLog.class.php

Methods

- setTopTitle(title,color,icon_index) to set the Top Title of the XML screen
  - title string
  - color string, "red", "blue", ... (optional)
  - icon_index integer, icon number
- setCancelAction(uri) to set the cancel parameter with the URI to be called on Cancel (optional)
  - uri string
- setDestroyOnExit() to set DestroyonExit parameter to 'yes', 'no' by default (optional)
- setBeep() to enable a notification beep with the object (optional)
- setLockIn(uri) to set the Lock-in tag to 'yes' and the GoodbyeLockInURI (optional)
  - uri string, GoodByeLockInURI
- setAllowAnswer() to set the allowAnswer tag to 'yes' (optional only for non softkey phones)
- setAllowDrop() to set the allowDrop tag to 'yes' (optional only for non softkey phones)
- setAllowXfer() to set the allowXfer tag to 'yes' (optional only for non softkey phones)
- setAllowConf() to set the allowConf tag to 'yes' (optional only for non softkey phones)
- setTimeout(timeout) to define a specific timeout for the XML object (optional)
  - timeout integer (seconds)
- setRefresh(timeout,URL) to add Refresh parameters to the object (optional)
  - timeout integer (seconds)
  - URL string
- setEncodingUTF8() to change encoding from default ISO-8859-1 to UTF-8 (optional)
- addIcon(index,icon) to add custom icons to the object (optional)
  - index integer, icon index
  - icon string, icon name or definition
- generate() to return the generated XML for the object
- output(flush) to display the object
  - flush boolean optional, output buffer to be flushed out or not.
- addEntry(name,number,date,time,selection,duration,type,terminal,count,line)

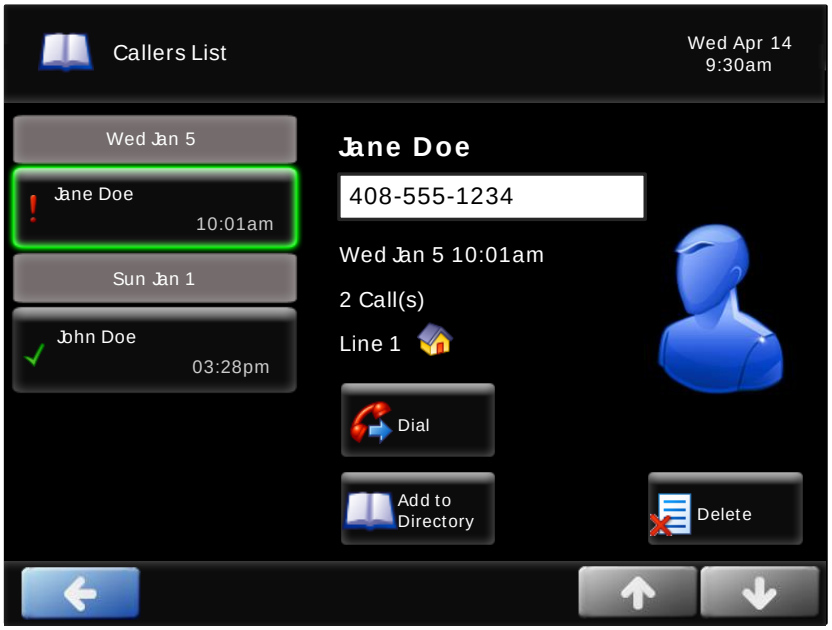
    - name string (optional)
    - number string
    - date string MM-DD-YYYY
    - time string HH:MM (military time)
    - selectionstring (optional)
    - duration integer call duration in seconds (optional)
    - type string call type (incoming/outgoung/missed) (optional)
    - terminal string terminal type (office/mobile/home) (optional)
    - count integer number of calls (optional)
    - line integer line used (1-9) (optional)
- setScrollConstrain() to avoid the list to wrap
- setScrollUp(uri) to set the URI to be called when the user presses the Up arrow (optional)
    - uri string
- setScrollDown(uri) to set the URI to be called when the user presses the Down arrow (optional)
    - uri string
- setDeleteUri(uri) to configure the uri called by the "Delete" button (optional)
    - uri string
- setDeleteAllUri(uri) to configure the uri called by the "Delete ALL" button (optional)
    - uri string
- setDialUri(uri) to configure the uri called by the "Dial" button (optional)
    - uri string
- setAddUri(uri) to configure the uri called by the "Add to directory" button(optional)
    - uri string

Example

```
require_once('AastraIPPhoneCallLog.class.php');
$object = new AastraIPPhoneCallLog();
$object->setTopTitle('Callers List','','1');
$object->setDestroyOnExit();
$object->setCancelAction($XML_SERVER);
$object->addEntry('John                              Doe','972-555-2345','01-01-
2012','15:28','1','60','incoming','mobile','','1');
$object->addEntry('Jane                              Doe','408-555-1234','01-05-
2011','10:01','2','','missed','home','2','1');
$object->setAddUri('http://myserver/myscript.php?action=add');
$object->addIcon('1', 'Icon:Book');
$object->output();
```

In this example, the label and the type of the softkey 1 are changed.

Output



13.2 AASTRAIPPHONECONFIGURATION()

This class allows you to create a XML PhoneConfiguration object.

Include

- AastraIPPhoneConfiguration.class.php

Methods

- setEncodingUTF8() to change encoding from default ISO-8859-1 to UTF-8 (optional)
- generate() to return the generated XML for the object
- output(flush) to display the object
  - flush boolean optional, output buffer to be flushed out or not.
- setType(type) to set the type of configuration object (optional)
  - type string, configuration change type
- addEntry(parameter,value,type) to add a configuration change
  - parameter string, parameter name
  - value string, parameter value
  - type string, conmfiguration change type (optional)
- setTriggerDestroyOnExit() to set the triggerDestroyOnExit tag to "yes" (optional)

Example

```
require_once('AastraIPPhoneConfiguration.class.php');
$configuration = new AastraIPPhoneConfiguration();
$configuration->addEntry('softkey1 label','Test');
$configuration->addEntry('softkey1 type','xml');
$configuration->setTriggerDestroyOnExit();
$configuration->setBeep();
$configuration->output();
```

In this example, the label and the type of the softkey 1 are changed.

13.3 AASTRAIPPHONEEXECUTE()

This class allows you to create a XML PhoneExecute object.

Include

- AastraIPPhoneExecute.class.php

Methods

- setBeep() to enable a notification beep with the object (optional)
- setEncodingUTF8() to change encoding from default ISO-8859-1 to UTF-8 (optional)
- generate() to return the generated XML for the object
- output(flush) to display the object
    - flush boolean optional, output buffer to be flushed out or not.
- setTriggerDestroyOnExit() to set the triggerDestroyOnExit tag to "yes" (optional)
    - addEntry(url,interruptCall,title) to add an action to be executed.
    - url string
    - interruptCall string, optional, "yes" or "no"
    - title string, optional, 6867i/6869i and 6973i only, title to be displayed

Example

```
require_once('AastraIPPhoneExecute.class.php');
$execute = new AastraIPPhoneExecute();
$execute->addEntry('http://myserver.com/script.php?choice=2');
$execute->addEntry('Command: Reset');
$execute->output();
```

13.4 AASTRAIPPHONEFORMATTEDTEXTSCREEN()

This class allows you to create a XML FormattedTextScreen object.

Include

- AastraIPPhoneFormattedTextScreen.class.php

Methods

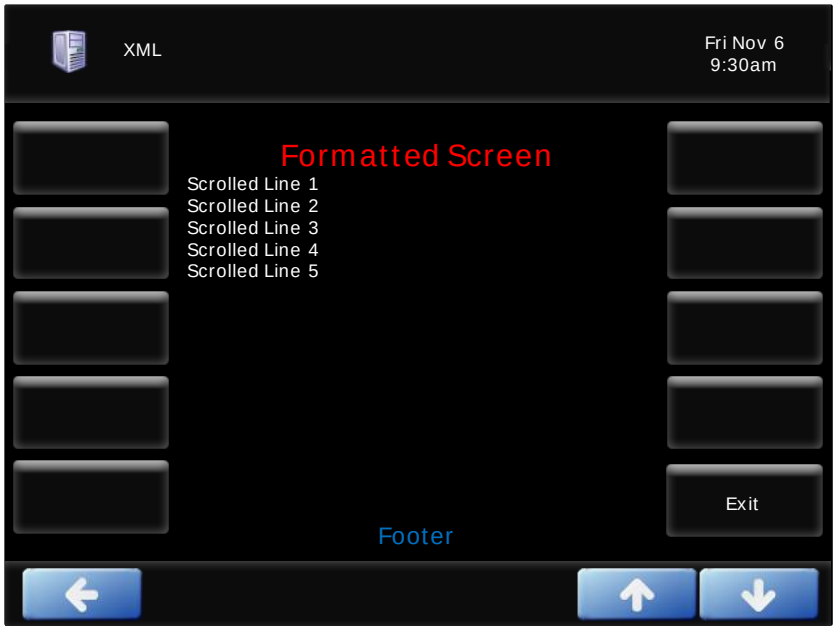
- setTopTitle(title,color,icon_index) to set the Top Title of the XML screen
    - title string
    - color string, "red", "blue", ... (optional)
    - icon_index integer, icon number
- setCancelAction(uri) to set the cancel parameter with the URI to be called on Cancel (optional)

    - uri string
- setDestroyOnExit() to set DestroyonExit parameter to 'yes', 'no' by default (optional)
- setLockIn(uri) to set the Lock-in tag to 'yes' and the GoodbyeLockInURI (optional)
    - uri string, GoodByeLockInURI
- setCallProtection(notif) to protect the XML object against incoming calls
    - notif enable/disable the display of an incoming call notification (optional)
- setAllowAnswer() to set the allowAnswer tag to 'yes' (optional)
- setAllowDrop() to set the allowDrop tag to 'yes' (optional)
- setAllowXfer() to set the allowXfer tag to 'yes' (optional)
- setAllowConf() to set the allowConf tag to 'yes' (optional)
- setTimeout(timeout) to define a specific timeout for the XML object (optional)
    - timeout integer (seconds)
- setBackgroundColor(color) to change the XML object background color (optional)
    - color string, "red", "blue", ...
- addSoftkey(index,label,uri) to add custom soktkeys to the object (optional)
    - index integer, softkey number
    - label string
    - uri string
- setRefresh(timeout,URL) to add Refresh parameters to the object (optional)
    - timeout integer (seconds)
    - URL string
- setEncodingUTF8() to change encoding from default ISO-8859-1 to UTF-8 (optional)
- generate() to return the generated XML for the object
- output(flush) to display the object
    - flush boolean optional, output buffer to be flushed out or not.
- addLine(text,size,align,color,wrap, blink) to add a formatted line
    - text string
    - size string, optional, "double"
    - align string, optional, "left", "right" or "center"
    - color string, optional
    - wrap string, optional, “yes”, “no” (default)
    - blink string, optional, "slow", "fast" or "no" (default)
- setScrollStart() to define the beginning of the scrolling section
- setScrollEnd() to define the end of the scrolling section
- setAllowDTMF() to allow DTMF passthrough on the object
- setDial(number,line) to set the number to be dialed as well as the line to use when going off-hook or with the custom softkey Softkey::Dial2
    - number string
    - line integer (optional)

Example

```
require_once('AastraIPPhoneFormattedTextScreen.class.php');
$ftext = new AastraIPPhoneFormattedTextScreen();
$ftext->setDestroyOnExit();
$ftext->addLine('Formatted Screen','double','center',’red’);
$ftext->setScrollStart();
$ftext->addLine('Scrolled text1');
$ftext->addLine('Scrolled text2');
$ftext->addLine('Scrolled text3');
$ftext->addLine('Scrolled text4');
$ftext->addLine('Scrolled text5');
$ftext->setScrollEnd();
$ftext->addLine('Footer',NULL,'center',’blue’);
$ftext->addSoftkey('10', 'Exit', 'SoftKey:Exit');
$ftext->output();
```

Output



13.5 AASTRAIPPHONEICONMENU()

This class allows you to create a XML IconMenu object.

Include

- AastraIPPhoneIconMenu.class.php

Methods

- setDestroyOnExit() to set DestroyonExit parameter to 'yes', 'no' by default (optional)
- setBeep() to enable a notification beep with the object (optional)
- setLockIn(uri) to set the Lock-in tag to 'yes' and the GoodbyeLockInURI (optional)
    - uri string, GoodByeLockInURI
- setCallProtection(notif) to protect the XML object against incoming calls
    - notif enable/disable the display of an incoming call notification (optional)
- setTimeout(timeout) to define a specific timeout for the XML object (optional)

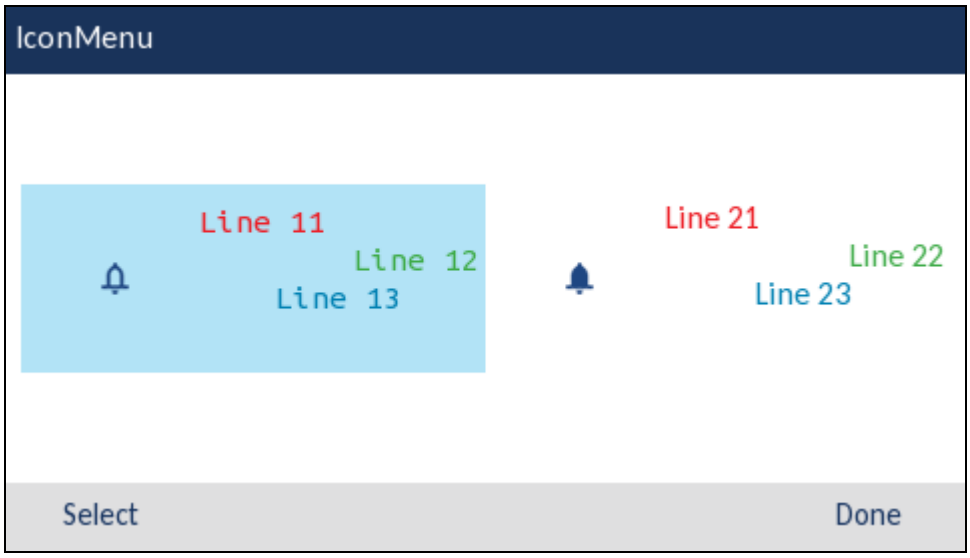
    - timeout integer (seconds)
- setBackgroundColor(color) to change the XML object background color (optional)
    - color string, "red", "blue", ...
- addSoftkey(index,label,uri) to add custom soktkeys to the object (optional)
    - index integer, softkey number
    - label string
    - uri string
- setRefresh(timeout,URL) to add Refresh parameters to the object (optional)
    - timeout integer (seconds)
    - URL string
- setEncodingUTF8() to change encoding from default ISO-8859-1 to UTF-8 (optional)
- generate() to return the generated XML for the object
- output(flush) to display the object
    - flush boolean optional, output buffer to be flushed out or not.
- setLayout(layout) to set the desired layout
    - layout 1 or 2
- setMode(mode) to set the screen mode
    - mode 'regular' or 'extended'
- setDefaultIndex(index) to set the default selection in the list (optional)
    - index index (1-24)
- setFontMono(fontMono) to decide the use of the monotype font
    - fontMono boolean ("yes"/"no")
- addEntry(url,selection,icon,fontMono,dial,line) to add an element in the list to be displayed
    - url string
    - selection string (optional)
    - icon integer (optional)
    - fontMono boolean (optional)
    - dial string, phone number to dial (optional)
    - line integer, SIP line to use (optional)
- addLine(text,align,color) to add a line in the current item
    - text string
    - align string, optional, "left", "right" or "center"
    - color string, optional, "red", "black", ...

Example

```
require_once('AastraIPPhoneIconMenu.class.php');
$menu = new AastraIPPhoneIconMenu();
$menu->setDestroyOnExit();
$menu->setTopTitle('IconMenu');
$menu->setDeFaultIndex('3');
$menu->setLayout('1');
$menu->setMode('regular');
```

```
$menu->setFontMono('yes');
$menu->addEntry(    'http://myserver.com/script.php?choice=1',
                    'Value=1',
                    'Icon:Alarm',
                    'yes'
                );
$menu->addLine('Line 11','left','red');
$menu->addLine('Line 12','right','green');
$menu->addLine('Line 13','center','blue');
$menu->addEntry(    'http://myserver.com/script.php?choice=2',
                    'Value=2',
                    'Icon:AlarmFilled',
                    'no'
                );
$menu->addLine('Line 21','left','red');
$menu->addLine('Line 22','right','green');
$menu->addLine('Line 23','center','blue');
$menu->addSoftkey('1', 'My Select', 'http://myserver.com/script.php?action=1');
$menu->addSoftkey('6', 'Exit', 'SoftKey:Exit');
$menu->output();
```

Output



13.6 AASTRAIPPHONEIMAGEMENU()

This class allows you to create a XML ImageMenu object.

Include

- AastraIPPhoneImageMenu.class.php

Methods

- setTopTitle(title,color,icon_index) to set the Top Title of the XML screen
  - title string
  - color string, "red", "blue", ... (optional)
  - icon_index integer, icon number
- setCancelAction(uri) to set the cancel parameter with the URI to be called on Cancel (optional)
  - uri string

- setDestroyOnExit() to set DestroyonExit parameter to 'yes', 'no' by default (optional)
- setBeep() to enable a notification beep with the object (optional)
- setLockIn(uri) to set the Lock-in tag to 'yes' and the GoodbyeLockInURI (optional)
    - uri string, GoodByeLockInURI
- setCallProtection(notif) to protect the XML object against incoming calls
    - notif enable/disable the display of an incoming call notification (optional)
- setAllowAnswer() to set the allowAnswer tag to 'yes' (optional)
- setAllowDrop() to set the allowDrop tag to 'yes' (optional)
- setAllowXfer() to set the allowXfer tag to 'yes' (optional)
- setAllowConf() to set the allowConf tag to 'yes' (optional)
- setTimeout(timeout) to define a specific timeout for the XML object (optional)
    - timeout integer (seconds)
- setBackgroundColor(color) to change the XML object background color (optional)
    - color string, "red", "blue", ...
- addSoftkey(index,label,uri,icon_index) to add custom soktkeys to the object (optional)
    - index integer, softkey number
    - label string
    - uri string
    - icon_index integer, icon number
- setRefresh(timeout,URL) to add Refresh parameters to the object (optional)
    - timeout integer (seconds)
    - URL string
- setEncodingUTF8() to change encoding from default ISO-8859-1 to UTF-8 (optional)
- addIcon(index,icon) to add custom icons to the object (optional)
    - index integer, icon index
    - icon string, icon name or definition
- generate() to return the generated XML for the object
- output(flush) to display the object
    - flush boolean optional, output buffer to be flushed out or not.
- setImage(image)to define the image to be displayed
    - image string
- setGDImage(GDImage) to use a GDImage for display, the size is forced to 40x144
    - GDImage GDImage
- setAlignment(vertical,horizontal) to define image alignment
    - vertical string, "top", "middle", "bottom"
    - horizontal string, "left", "middle", "right"
- setSize(height,width) to define image size
    - height integer (pixels)
    - width integer (pixels)

- setURIBase(uriBase) to define the base URI for the selections
    - uriBase string
- addURI(key,uri) to add a selection key with its URI
    - key string (1-9, * and #)
    - uri string
- setMode(mode) to define the image mode to be displayed (normal, extended, fullscreen) (optional)
    - mode string enum normal, extended, fullscreen
- setDoneAction(uri) to set the URI to be called when the user selects the default "Done" key (optional)
    - uri string
- setImageAction(uri) to set the imageAction parameter with the URI to be called when user presses on the displayed image (optional)
    - uri string

Example

```
require_once('AastraIPPhoneImageMenu.class.php');
$imagem = new AastraIPPhoneImageMenu();
$imagem->setDestroyOnExit();
$imagem->setSize(40,144);
$imagem-
>setImage('ffffffc02ffffffee4ffffbfffc05fffe7ff7a7ffffffffeffeebd7fffffea6bcf
ffffe796f3feff6fa289f0a86f4866fa20df42414595dd0134f8037ed1637f0e2522b2dd003b6eb
936f05fffbd4f4107bba6eb0080e93715000010b754001281271408c640252081b1b22500013c5c
66201368004e04467520dc11067152b82094d418e1002472058054947801050026015300201314 0
0020a05c91088b002b08c21c0000c200000001fe80000000000000001c0000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000020041000004008300000ff08500000000c90000000071000
000000000001401400000140140000014014000001401400000140140000000000000007c0ff00
000c308800000810880000081088000000c307000000620000000000003f000001e02000000330200
000021000000003301e000001e033000000021000003f033000002001e000002000000000001e
000c03fc33003c013021007c02101201f00330ff03f001e000039000003e039001e00103f003300
101f8021003007c03303f003c01e000000c00001e001c03f033007802002100f002002103e00000
1203c401702003cc0290ff039c02902101fc02b000007c03f01a003c020039018c0ff02d03c4021
02703c400001203ec01e000026402b0000264029000026c029000027c01a000038000000033800
000003100000000300000000030003f00003fc03000003fc02000003fc020000030001f00003000
00000030001e000030002b000030002900003fc02900003fc01a00003f000000031003000031c
01e000031f003000033f81e00003f383000001e081e000008c003000003c01e00000fc03000001f
000000003d001a000039003900039002d00003f002700000f8012000007c000000001c00000000
040000000000000000000000000000');
$imagem->addURI('1','http://myserver.com?choice=1');
$imagem->addURI('2','http://myserver.com?Choice=2');
$imagem->addSoftkey('1',                                                    'Label',
'http://myserver.com/script.php?action=1','1');
$imagem->addSoftkey('6', 'Exit', 'SoftKey:Exit');
$imagem->addIcon('1', 'Icon:Envelope');
$imagem->addIcon('2', 'FFFF0000FFFF0000');
$imagem->output();
```

Output



13.7 AASTRAIPPHONEIMAGESCREEN()

This class allows you to create a XML ImageScreen object.

Include

- AastraIPPhoneImageScreen.class.php

Methods

- setTopTitle(title,color,icon_index) to set the Top Title of the XML screen
  - title string
  - color string, "red", "blue", ... (optional)
  - icon_index integer, icon number
- setCancelAction(uri) to set the cancel parameter with the URI to be called on Cancel (optional)
  - uri string
- setDestroyOnExit() to set DestroyonExit parameter to 'yes', 'no' by default (optional)
- setBeep() to enable a notification beep with the object (optional)
- setLockIn(uri) to set the Lock-in tag to 'yes' and the GoodbyeLockInURI (optional)
  - uri string, GoodByeLockInURI
- setCallProtection(notif) to protect the XML object against incoming calls
  - notif enable/disable the display of an incoming call notification (optional)
- setAllowAnswer() to set the allowAnswer tag to 'yes' (optional)
- setAllowDrop() to set the allowDrop tag to 'yes' (optional)
- setAllowXfer() to set the allowXfer tag to 'yes' (optional)
- setAllowConf() to set the allowConf tag to 'yes' (optional)
- setTimeout(timeout) to define a specific timeout for the XML object (optional)
  - timeout integer (seconds)
- setBackgroundColor(color) to change the XML object background color (optional)
  - color string, "red", "blue", ...
- addSoftkey(index,label,uri,icon_index) to add custom soktkeys to the object (optional)
  - index integer, softkey number
  - label string
  - uri string
  - icon_index integer, icon number
- setRefresh(timeout,URL) to add Refresh parameters to the object (optional)
  - timeout integer (seconds)
  - URL string
- setEncodingUTF8() to change encoding from default ISO-8859-1 to UTF-8 (optional)
- addIcon(index,icon) to add custom icons to the object (optional)
  - index integer, icon index
  - icon string, icon name or definition
- generate() to return the generated XML for the object
- output(flush) to display the object
  - flush boolean optional, output buffer to be flushed out or not.
- setImage(image) to define the image to be displayed
  - image string
- setGDImage(GDImage) to use a GDImage for display, the size is forced to 40x144

    - GDImage GDImage
- setAlignment(vertical,horizontal) to define image alignment
    - vertical string, "top", "middle", "bottom"
    - horizontal string, "left", "middle", "right"
- setSize(height,width) to define image size
    - height integer (pixels)
    - width integer (pixels)
- setAllowDTMF() to allow DTMF passthrough on the object
- setScrollUp(uri) to set the URI to be called when the user presses the Up arrow (optional)
    - uri string
- setScrollDown(uri) to set the URI to be called when the user presses the Down arrow (optional)
    - uri string
- setScrollLeft(uri) to set the URI to be called when the user presses the Left arrow (optional)
    - uri string
- setScrollRight(uri) to set the URI to be called when the user presses the Right arrow (optional)
    - uri string
- setMode(mode) to define the image mode to be displayed (normal, extended, fullscreen) (optional)
    - mode string enum normal, extended, fullscreen
- setDoneAction(uri) to set the URI to be called when the user selects the default "Done" key (optional)
    - uri string
- setImageAction(uri) to set the imageAction parameter with the URI to be called when user presses on the displayed image (optional)
    - uri string

Example

```
require_once('AastraIPPhoneImageScreen.class.php');
$images = new AastraIPPhoneImageScreen();
$images->setDestroyOnExit();
$images->setSize(40,40);
$images-
>setImage('ffffffc02ffffffee4ffffbfffc05fffe7ff7a7fffffffeffeebd7fffffea6bcf
ffffe796f3feff6fa289f0a86f4866fa20df42414595dd0134f8037ed1637f0e2522b2dd003b6eb
936f05fffbd4f4107bba6eb0080e93715000010b754001281271408c640252081b1b22500013c5c
66201368004e04467520dc11067152b82094d418e10024720580549478010500260153002093140
0020ac5c91088b0f2b08c21c07d0c2006009fdfe81f80efe0107fe0fb1c3ffff8ffc3fffef8f7fe
bffbfcf87ffbff64');
$images->addSoftkey('1',                                                    'Mail',
'http://myserver.com/script.php?action=1','1');
$images->addSoftkey('6', 'Exit', 'SoftKey:Exit');
$images->addIcon('1', 'Icon:Envelope');
$images->output();
```

Output



13.8 AASTRAIPPHONEINPUTSCREEN() – SINGLE INPUT FIELD

This class allows you to create a XML InputScreen object.

Include

- AastraIPPhoneInputScreen.class.php

Methods

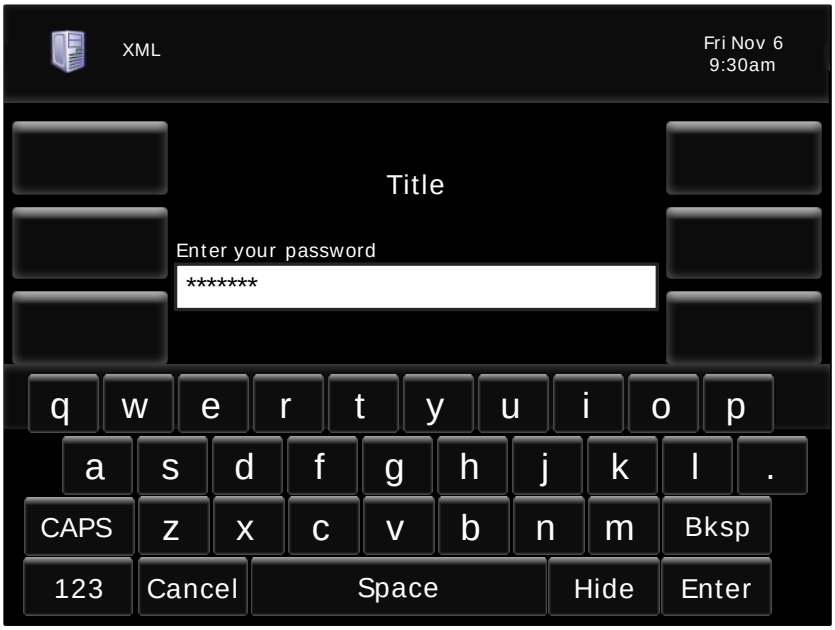
- setTitle(title) to setup the title of an object (optional)
  - title string
- setTopTitle(title,color,icon_index) to set the Top Title of the XML screen
  - title string
  - color string, "red", "blue", ... (optional)
  - icon_index integer, icon number
- setCancelAction(uri) to set the cancel parameter with the URI to be called on Cancel (optional)
  - uri string
- setDestroyOnExit() to set DestroyonExit parameter to 'yes', 'no' by default (optional)
- setLockIn(uri) to set the Lock-in tag to 'yes' and the GoodbyeLockInURI (optional)
  - uri string, GoodByeLockInURI
- setCallProtection(notif) to protect the XML object against incoming calls
  - notif enable/disable the display of an incoming call notification (optional)
- setAllowAnswer() to set the allowAnswer tag to 'yes' (optional)
- setAllowDrop() to set the allowDrop tag to 'yes' (optional)
- setAllowXfer() to set the allowXfer tag to 'yes' (optional)
- setAllowConf() to set the allowConf tag to 'yes' (optional)
- setTimeout(timeout) to define a specific timeout for the XML object (optional)
  - timeout integer (seconds)
- setBackgroundColor(color) to change the XML object background color (optional)
  - color string, "red", "blue", ...

- addSoftkey(index,label,uri) to add custom soktkeys to the object (optional)
    - index integer, softkey number
    - label string
    - uri string
- setRefresh(timeout,URL) to add Refresh parameters to the object (optional)
    - timeout integer (seconds)
    - URL string
- setEncodingUTF8() to change encoding from default ISO-8859-1 to UTF-8 (optional)
- generate() to return the generated XML for the object
- output(flush) to display the object
    - flush boolean optional, output buffer to be flushed out or not.
- setURL(url) to set the URL to called after the input
    - url string
- setType(type) to set type of input, 'string' by default
    - type enum ('IP', 'string', ‘stringN’, 'number', 'dateUS'...)
- setDefault(default) to set default value for the input (optional)
    - default string
- setParameter(param) to set the parameter name to be parsed after the input
    - param string
- setInputLanguage(language) to set the language of the input (optional)
    - language enum ("English", "French"....)
- setPassword() to set the Password parameter to 'yes', 'no' by default (optional)
- setNotEditable() to set the editable parameter to 'no', 'yes' by default (optional)
- setEditable() is now replaced by setNotEditable but kept for compatibility reasons (optional)
- setPrompt(prompt) to set the prompt to be displayed for the input.
    - prompt string

Example

```
require_once('AastraIPPhoneInputScreen.class.php');
$input = new AastraIPPhoneInputScreen();
$input->setTitle('Title');
$input->setPrompt('Enter your password');
$input->setParameter('param');
$input->setType('string');
$input->setURL('http://myserver.com/script.php');
$input->setPassword();
$input->setDestroyOnExit();
$input->setDefault('Default');
$input->output();
```

Output



13.9 AASTRAIPPHONEINPUTSCREEN() – MULTIPLE INPUT FIELDS

This class allows you to create a XML InputScreen object with multiple input fields.

Include

- AastraIPPhoneInputScreen.class.php

Methods

- setTitle(title) to setup the title of an object (optional)
  - title string
- setTopTitle(title,color,icon_index) to set the Top Title of the XML screen
  - title string
  - color string, "red", "blue", ... (optional)
  - icon_index integer, icon number
- setCancelAction(uri) to set the cancel parameter with the URI to be called on Cancel (optional)
  - uri string
- setDestroyOnExit() to set DestroyonExit parameter to 'yes', 'no' by default (optional)
- setLockIn(uri) to set the Lock-in tag to 'yes' and the GoodbyeLockInURI (optional)
  - uri string, GoodByeLockInURI
- setCallProtection(notif) to protect the XML object against incoming calls
  - notif enable/disable the display of an incoming call notification (optional)
- setAllowAnswer() to set the allowAnswer tag to 'yes' (optional)
- setAllowDrop() to set the allowDrop tag to 'yes' (optional)
- setAllowXfer() to set the allowXfer tag to 'yes' (optional)
- setAllowConf() to set the allowConf tag to 'yes' (optional)

- setTimeout(timeout) to define a specific timeout for the XML object (optional)
    - timeout integer (seconds)
- setBackgroundColor(color) to change the XML object background color (optional)
    - color string, "red", "blue", ...
- addSoftkey(index,label,uri) to add custom soktkeys to the object (optional)
    - index integer, softkey number
    - label string
    - uri string
- setRefresh(timeout,URL) to add Refresh parameters to the object (optional)
    - timeout integer (seconds)
    - URL string
- setEncodingUTF8() to change encoding from default ISO-8859-1 to UTF-8 (optional)
- generate() to return the generated XML for the object
- output(flush) to display the object
    - flush boolean optional, output buffer to be flushed out or not.
- setURL(url) to set the URL to called after the input
    - url string
- setType(type) to set the default type of input 'string' by default
    - type enum ('IP', 'string', 'number', 'dateUS'...)
- setDefault(default) to set default default value for the input (optional)
    - default string
- setParameter(param) to set the default parameter name to be parsed after the input
    - param string
- setPassword() to set the default Password parameter to 'yes', 'no' by default (optional)
- setNotEditable() to set the default editable parameter to 'no', 'yes' by default (optional)
- setEditable() is now replaced by setNotEditable but kept for compatibility reasons (optional)
- setPrompt(prompt) to set the default prompt to be displayed for the input.
    - prompt string
- setDefaultIndex(index) to define the field index the object will use to start (optional)
    - index integer, optional, default is 1
- setDisplayMode(display) to define the aspect of the display, normal/condensed (optional)
    - display enum ("normal, "condensed"), default is "normal".
- setInputLanguage(language) to set the language of the input (optional)
    - language enum ("English", "French"....)
- addField(type) to add an input field and setting its type
    - type (IP, string, stringN, number, dateUS, timeUS,dateInt, timeInt or empty) if the type is an empty string then the type is inherited from the main object.
- setFieldPassword(password) to set the password mode for the input field, overrides the value set by setPassword for the field

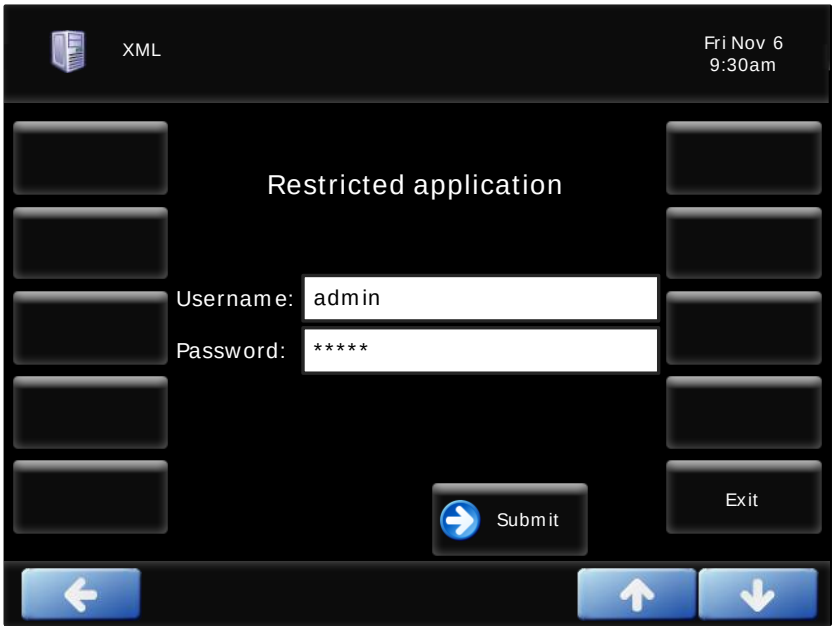
- password enum ("yes", "no")

- setFieldEditable(editable) to set the input field editable mode ('yes', no'), overrides the value set by setEditable or setNotEditable for the field
  - editable enum ("yes", "no")
- setFieldParameter(parameter) to set the parameter name to be parsed after the global input, overrides the value set by setParameter for the field
  - parameter string
- setFieldPrompt(prompt)to set the prompt to be displayed for the input field, overrides the value set by setPrompt for the field
  - prompt string
- setFieldSelection(selection) to set the Selection tag for the field
  - selectionstring
- setFieldDefault(default) to set default value for the input field, overrides the value set by setDefault for the field
  - default string

Example

```
require_once('AastraIPPhoneInputScreen.class.php');
$input = new AastraIPPhoneInputScreen();
$input->setTitle('Restricted application');
$input->setDisplayMode('condensed');
$input->setURL($XML_SERVER);
$input->setDestroyOnExit();
$input->addField('empty');
$input->addField('string');
$input->setFieldSelection('1');
$input->setFieldPrompt('Username:');
$input->setFieldParameter('user');
$input->setFieldSelection('1');
$input->addField('number');
$input->setFieldPassword('yes');
$input->setFieldPrompt('Password:');
$input->setFieldParameter('password');
$input->setFieldSelection('2');
$input->addSoftkey('10', 'Exit', 'SoftKey:Exit');
$input->output();
```

Output



13.10 AASTRAIPPHONESOFTKEY()

This class allows you to create a XML PhoneSoftkey object.

Include

- AastraIPPhoneSoftkey.class.php

Methods

- setEncodingUTF8() to change encoding from default ISO-8859-1 to UTF-8 (optional)
- generate() to return the generated XML for the object
- output(flush) to display the object
  - flush boolean optional, output buffer to be flushed out or not.
- addEntry(parameter,value) to add a configuration change for a dynamic attribute
  - parameter string, parameter name
  - value string, parameter value
- setTriggerDestroyOnExit() to set the triggerDestroyOnExit tag to "yes" (optional)

Example

```
require_once('AastraIPPhoneSoftkey.class.php');
$configuration = new AastraIPPhoneSoftkey();
$configuration->addEntry('topsoftkey1 icon left','Icon:Home');
$configuration->addEntry('topsoftkey1 label','Demo Text');
$configuration->output();
```

In this example, the left icon and the label of the topsoftkey 1 are updated.

13.11 AASTRAIPPHONESTATUS()

This class allows you to create a XML PhoneStatus object.

Include

- AastraIPPhoneStatus.class.php

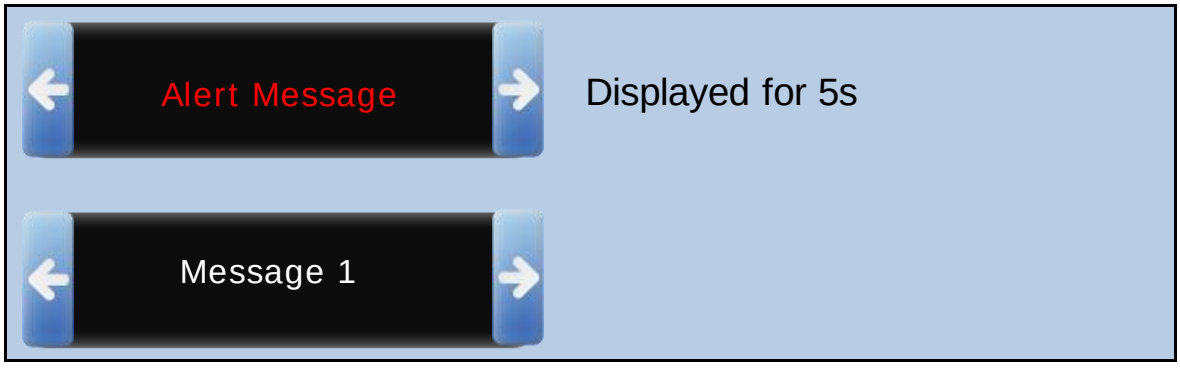
Methods

- setEncodingUTF8() to change encoding from default ISO-8859-1 to UTF-8 (optional)
- generate() to return the generated XML for the object
- output(flush) to display the object
    - flush boolean optional, output buffer to be flushed out or not.
- setSession(session) to setup the session ID
    - session string
- setTriggerDestroyOnExit() to set the triggerDestroyOnExit tag to "yes" (optional)
- addEntry(index,message,type,timeout,uri,icon) to add a message to be displayed on the idle screen.
    - index integer
    - message string
    - type enum ("alert") optional
    - timeout integer (seconds) optional
    - uri string (unused)
    - icon integer (graphical phones only, optional)

Example

```
require_once('AastraIPPhoneStatus.class.php');
$status = new AastraIPPhoneStatus();
$status->setSession('Session');
$status->addEntry('1','Message 1');
$status->addEntry('2','Alert Message',’alert’,5);
$status->output();
```

Output



13.12 AASTRAIPPHONETEXTMENU()

This class allows you to create a XML TextMenu object.

Include

- AastraIPPhoneTextMenu.class.php

Methods

- setTitle(title) to setup the title of an object (optional)
    - title string
- setCancelAction(uri) to set the cancel parameter with the URI to be called on Cancel (optional)
    - uri string
- setDestroyOnExit() to set DestroyonExit parameter to 'yes', 'no' by default (optional)
- setBeep() to enable a notification beep with the object (optional)
- setLockIn(uri) to set the Lock-in tag to 'yes' and the GoodbyeLockInURI (optional)
    - uri string, GoodByeLockInURI
- setAllowAnswer() to set the allowAnswer tag to 'yes' (optional)
- setAllowDrop() to set the allowDrop tag to 'yes' (optional)
- setAllowXfer() to set the allowXfer tag to 'yes' (optional)
- setAllowConf() to set the allowConf tag to 'yes' (optional)
- setTimeout(timeout) to define a specific timeout for the XML object (optional)
    - timeout integer (seconds)
- setBackgroundColor(color) to change the XML object background color (optional)
    - color string, "red", "blue", ...
- addSoftkey(index,label,uri) to add custom soktkeys to the object (optional)
    - index integer, softkey number
    - label string
    - uri string
- setRefresh(timeout,URL) to add Refresh parameters to the object (optional)
    - timeout integer (seconds)
    - URL string
- setEncodingUTF8() to change encoding from default ISO-8859-1 to UTF-8 (optional)
- generate() to return the generated XML for the object
- output(flush) to display the object
    - flush boolean optional, output buffer to be flushed out or not.
- setDefaultIndex(index) to set the default selection in the list (optional)
    - index index (1-30)
- setBase(base) to configure the menuItem base URI
    - base string
- resetBase() to reset the menuItem base URI
- setScrollConstrain() to avoid the list to wrap
- setNumberLaunch() to allow keypad selection
- setTouchLaunch() to allow quick item selection
- addEntry(name,url,selection,icon,dial,line,color,split,iconr1,iconr2,iconr3,iconr4) to add an element in the list to be displayed
    - name string or array(0=>Line1,1=>Line2,2=>Offset,3=>Char,4=>Mode)

    - url string
    - selection string
    - icon integer
    - dial string, phone number to dial
    - line integer, SIP line to use (optional)
    - color string, "red", "black", ... (optional)
    - split integer, position of the split between line 1 and line 2 (optional)
    - iconr1 integer (optional)
    - iconr2 integer (optional)
    - iconr3 integer (optional)
    - iconr4 integer (optional)
- setScrollUp(uri) to set the URI to be called when the user presses the Up arrow (optional)
    - uri string
- setScrollDown(uri) to set the URI to be called when the user presses the Down arrow (optional)
    - uri string
- natsortbyname() to order the list, must not be used in conjunction with setBase or resetBase

Example

```
require_once('AastraIPPhoneTextMenu.class.php');
$menu = new AastraIPPhoneTextMenu();
$menu->setTitle('Title');
$menu->setDestroyOnExit();
$menu->setDeFaultIndex('3');
$menu->addEntry('Choice     2',      'http://myserver.com/script.php?choice=2',
'Value=2');
$menu->addEntry('Choice     1',      'http://myserver.com/script.php?choice=1',
'Value=1');
$menu->addEntry('Choice     3',      'http://myserver.com/script.php?choice=3',
'Value=3');
$menu->natsortByName();
$menu->addSoftkey('1', 'My Select', 'http://myserver.com/script.php?action=1');
$menu->addSoftkey('10', 'Exit', 'SoftKey:Exit');
$menu->output();
```

Output



13.13 AASTRAIPPHONETEXTSCREEN()

This class allows you to create a XML TextScreen object.

Include

- AastraIPPhoneTextScreen.class.php

Methods

- setTitle(title) to setup the title of an object (optional)
  - title string
- setTopTitle(title,color,icon_index) to set the Top Title of the XML screen
  - title string
  - color string, "red", "blue", ... (optional)
  - icon_index integer, icon number
- setCancelAction(uri) to set the cancel parameter with the URI to be called on Cancel (optional)
  - uri string
- setDestroyOnExit() to set DestroyonExit parameter to 'yes', 'no' by default (optional)
- setBeep() to enable a notification beep with the object (optional)
- setLockIn(uri) to set the Lock-in tag to 'yes' and the GoodbyeLockInURI (optional)
  - uri string, GoodByeLockInURI
- setAllowAnswer() to set the allowAnswer tag to 'yes' (optional)
- setAllowDrop() to set the allowDrop tag to 'yes' (optional)
- setAllowXfer() to set the allowXfer tag to 'yes' (optional)
- setAllowConf() to set the allowConf tag to 'yes' (optional)
- setTimeout(timeout) to define a specific timeout for the XML object (optional)

    - timeout integer (seconds)
- setBackgroundColor(color) to change the XML object background color (optional)
    - color string, "red", "blue", ...
- addSoftkey(index,label,uri) to add custom soktkeys to the object (optional)
    - index integer, softkey number
    - label string
    - uri string
- setRefresh(timeout,URL) to add Refresh parameters to the object (optional)
    - timeout integer (seconds)
    - URL string
- setEncodingUTF8() to change encoding from default ISO-8859-1 to UTF-8 (optional)
- addIcon(index,icon) to add custom icons to the object (optional)
- generate() to return the generated XML for the object
- output(flush) to display the object
    - flush boolean optional, output buffer to be flushed out or not.
- setText(text) to set the text to be displayed.
    - text string
- setAllowDTMF() to allow DTMF passthrough on the object
- setDial(number,line) to set the number to be dialed as well as the line to use when going off-hook or with the custom softkey Softkey::Dial2
    - number string
    - line integer (optional)

Example

```
require_once('AastraIPPhoneTextScreen.class.php');
$text = new AastraIPPhoneTextScreen();
$text->setTitle('Title');
$text->setText('Text to be displayed and this text can be really long…');
$text->setDestroyOnExit();
$text->addSoftkey('1', 'Mail', 'http://myserver.com/script.php?action=1');
$text->addSoftkey('10', 'Exit', 'SoftKey:Exit');
$text->output();
```

Output



13.14 AASTRAIPPHONESCROLLABLETEXTMENU()

This class allows you to create TextMenu supporting more than 30 items, all the next/previous page management is handled by the object.

Include

- AastraIPPhoneScrollableTextMenu.class.php

Methods

- setEntries(entries) Set entries of the list by 2 dim array.
  - entries Inner array field names: 'name', 'url', 'selection', 'icon', 'dial'
- verifyCookie(cookie) Verifies if the cookie of the HTTP requests matches the cookie of the saved context.
- setBackURI(URI) Set the cancel parameter with the URI to be called on Cancel or Back Softkey (optional)
- setBackKeyPosition(position) Set position of Back Softkey. Default is 3.
- setNextKeyPosition(position) Set position of Next Softkey. Default is 4.
- setPreviousKeyPosition(position) Set position of Previous Softkey. Default is 5.
- setExitKeyPosition(position) Set position of Back Softkey. Default is 6.
- setSelectKeyPosition(position) Set position of Back Softkey. Default is 1.
- setNextKeyIcon(icon) Set icon of Next Softkey. Default is Icon:TailArrowDown. Set NULL to disable icon.
- setPreviousKeyIcon(icon) Set icon of Previous Softkey. Default is Icon:TailArrowUp. Set NULL to disable icon.
- disableExitKey() Disable the Exit Softkey
- setSelectKeyLabel(label) Set the label of the Select Softkey. Default is 'Select'. Make sure the string is in language.ini.
- setCancelAction(uri) to set the cancel parameter with the URI to be called on Cancel or Back Softkey (optional)
- output() to display the object

- addEntry(name,url,selection,icon,dial) to add an element in the list to be displayed
- natsortbyname() to order the list
- setTitle(Title) to setup the title of an object (optional)
- setTitleWrap() to set the title to be wrapped on 2 lines (optional)
- setTopTitle(title,color,icon_index) to set the Top Title of the XML screen
    - title string
    - color string, "red", "blue", ... (optional)
    - icon_index integer, icon number
- setDestroyOnExit() to set DestroyOnExit parameter to "yes" (optional)
- setBeep() to enable a notification beep with the object (optional)
- setLockIn(uri) to set the Lock-in tag to 'yes' and the GoodbyeLockInURI (optional)
    - uri string, GoodByeLockInURI
- setAllowAnswer() to set the allowAnswer tag to 'yes' (optional)
- setTimeout(timeout) to define a specific timeout for the XML object (optional)
- addSoftkey(index,label,uri,icon_index) to add custom softkeys to the object (optional)
- setRefresh(timeout,URL) to add Refresh parameters to the object (optional)
- generate() to return the object content
- setDefaultIndex(index) to set the default selection in the list (optional)
- setStyle(style) to set the style of the list numbered/none/radio (optional)
- setWrapList() to allow 2 lines items (optional)

Examples

```
require_once('AastraIPPhoneScrollableTextMenu.class.php')
$menu = new AastraIPPhoneScrollableTextMenu();
$menu->setTitle('My Menu');
$menu->addEntry('Choice 1', $XML_SERVER."?choice=1", '1');
# ... add as many entries you want
$menu->addEntry('Choice 100', $XML_SERVER."?choice=100", '100');
$menu->output();
```

And

```
require_once('AastraIPPhoneScrollableTextMenu.class.php')
$entries[0]['name'] = "Choice 1";
$entries[0]['url'] = $XML_SERVER."?choice=1";
$entries[0]['selection'] = "1";
# ... add as many entries you want
$entries[99]['name'] = "Choice 100";
$entries[99]['url'] = $XML_SERVER."?choice=100";
$entries[99]['selection'] = "100";
$menu = new AastraIPPhoneScrollableTextMenu();
$menu->setTitle('My Menu');
$menu->setEntries($entries):
$menu->output();
```

13.15 AASTRAIPPHONESCROLLABLEDIRECTORY()

This class allows you to create a complete directory application by jusr providing the results of a directory query as an array.

Include

- AastraIPPhoneScrollableDirectory.class.php

Methods

- setDialKeyPosition(position)setNameDisplayFormat(format) Set position of Dial Softkey in list view. Default is 2.
- setNameDisplayFormat(format) Set name display format. 0="Firstname Lastname", 1="Lastname, Firstname"
- natsortByLastname() Sort by lastname (same as natsortByName in case firstname is not provided)
- natsortByFirstname() Sort by firstname (same as natsortByName in case firstname is not provided)

Overwritten methods from AastraIPPhoneScrollableTextMenu

- setEntries(records) Set directory entries by 2 dim array. Inner array fields: See addEntry(record)
- addEntry(record) Add directory entry

Array fields: (name is mandatory, rest optional)

- name: Lastname or name
- firstname: Firstname (optional)
- title: Title
- department: Department
- company: Company
- icon: Icon
- office: Office number display format
- officeDigits: Office number digits / extension to be dialed. Optional
- mobile: Cell number display format
- mobileDigits: Cell number digits / extension to be dialed. Optional
- home: Home number display format
- homeDigits: Home number digits / extension to be dialed. Optional
- office2: Alternative / 2nd office number display format
- office2Digits: 2nd office number digits / extension to be dialed. Optional
- speedURL: If this field is present, a "+Speed" Softkey will be shown in zoom mode. Selected number will be passed in $selection variable.

Example: speedURL=http://xmlserver/xml/speed/speed.php?action=add&name=Peter

Inherited from AastraIPPhoneScrollableTextMenu

- setTopTitle(title,color,icon_index) to set the Top Title of the XML screen
    - title string
    - color string, "red", "blue", ... (optional)
    - icon_index integer, icon number
- setEntries(entries) Set entries of the list by an 2 dim array. Inner array field names: 'name', 'url', 'selection', 'icon', 'dial'
- verifyCookie(cookie) Verifies if the cookie of the HTTP requests matches the cookie of the saved context.
- setBackURI(URI) Set the cancel parameter with the URI to be called on Cancel or Back Softkey (optional)

- setBackKeyPosition(position) Set position of Back Softkey. Default is 3.
- setNextKeyPosition(position) Set position of Back Softkey. Default is 4.
- setPreviousKeyPosition(position) Set position of Back Softkey. Default is 5.
- setExitKeyPosition(position) Set position of Back Softkey. Default is 6.
- setSelectKeyPosition(position) Set position of Back Softkey. Default is 1.
- setNextKeyIcon(icon) Set icon of Next Softkey. Default is Icon:TailArrowDown. Set NULL to disable icon.
- setPreviousKeyIcon(icon) Set icon of Previous Softkey. Default is Icon:TailArrowUp. Set NULL to disable icon.
- disableExitKey() Disable the Exit Softkey
- setSelectKeyLabel(label) Set the label of the Select Softkey. Default is 'Select'. Make sure the string is in language.ini.
- setCancelAction(uri) to set the cancel parameter with the URI to be called on Cancel or Back Softkey (optional)
- addIcon(index,icon) to add custom icons to the object (optional)
- output() to display the object
- addEntry(name,url,selection,icon,dial) to add an element in the list to be displayed
- natsortbyname() to order the list

Examples

```
require_once('AastraIPPhoneScrollableDirectory.class.php')
$records[0]['name'] = "Smith";
$records[0]['firstname'] = "Lisa";
$records[0]['office'] = "+1 (0) 555-123-4321";
$records[0]['officeDigits'] = "4321";
$records[0]['mobile'] = "079 555 12 34";
# ... add as many entries you want
$records[99]['name'] = "Miller";
$records[99]['firstname'] = "Bob";
$records[99]['office'] = "+1 (0) 555-123-1234";
$records[99]['officeDigits'] = "1234";
$records[99]['home'] = "044 555 22 33";
$records[99]['company'] = "Mitel";
$directory = new AastraIPPhoneScrollableDirectory();
$directory->setTitle('Directory');
$directory->setBackURI($XML_SERVER."?action=start");
$directory->setEntries($records):
$directory->output();
```

13.16 EXAMPLES PROVIDED WITH THE PHP API

With the PHP API, Mitel provides some source code on how to use it.

sample.php for the regular XML objects

The whole directory “php_classes” must be installed on a HTTP server anywhere behind the root directory

sample.php

On any type of phone, create a XML softkey/prgkey to call the sample.php script using the following uri:

http://myserver.com/mydirectory/sample.php

It includes examples for

- PhoneConfiguration

- PhoneExecute
- PhoneInputScreen (3)
- PhoneTextMenu (2)
- PhoneTextScreen
- PhoneFormattedTextScreen
- PhoneStatus (2)

14 APPENDIX C: DYNAMIC PARAMETERS

The following table lists the configuration parameters that can be modified by a PhoneConfiguration object without a reboot of the SIP phone.

Parameter	Comments
ALL action uri	
ALL SIP parameters	
admin password	
alternate tftp server	
audio mode	
auto offhook	
auto resync days	
auto resync max delay	
auto resync mode	
auto resync time	
background image	
call forward key mode	
call transfer disabled	
callers list disabled	This parameter is dynamic so a user can't access it or add to it. However, you need to reboot the phone to clear the list.
conf script	
conference disabled	
date format	
directed call pickup	
directory 1	You need to reboot the phone to download new directories.
directory 2	
directory disabled	This parameter is dynamic so a user can't access it or add to it. However, you need to reboot the phone to clear the list.

Parameter	Comments
directory script	
dnd key mode	
download protocol	
dst config	
dst end day	
dst end hour	
dst end month	
dst end relative date	
dst end week	
dst minutes	
dst start day	
dst start hour	
dst start month	
dst start relative date	
dst start week	
feature key selection list	
ftp password	
ftp server	
ftp username	
handset sidetone gain	
handset tx gain	
handsfree tx gain	
headset sidetone gain	
headset tx gain	
http path	
http server	
https validate certificates	

Parameter	Comments
https validate expires	
https validate hostname	
icom script	
inactivity brightness level	
language	
line1 ring tone	
line2 ring tone	
line3 ring tone	
line4 ring tone	
line5 ring tone	
line6 ring tone	
line7 ring tone	
line8 ring tone	
line9 ring tone	
live dialpad	
log module lldp	
map conf key to	
map redial key to	
mwi missed calls	
options password enabled	
options script	
play a ring splash	
preferred line	
preferred line timeout	
prgkeyN line	Changes to subscriptions (eg BLF or BLA) require a reboot.
prgkeyN name	
prgkeyN type	

Parameter	Comments
prgkeyN value	
redial disabled	This parameter is dynamic so a user can't access it or add to it. However, you need to reboot the phone to clear the list.
redial script	
ring tone	
ringback timeout	
softkeyN label	Changes to subscriptions (eg BLF or BLA) require a reboot.
softkeyN line	
softkeyN states	
softkeyN type	
softkeyN value	
softkey selection list	
speeddial edit	
suppress dtmf playback	
switch ui focus to ringing line	
tftp server	
time format	
time reserved	
time server disabled	
time server1	
time server2	
time server3	
time zone code	
time zone minutes	
time zone name	
tone set	

Parameter	Comments
use alternate tftp server	
user password	
voicemail script	
xfer script	
xml application title	
xml application URI	
xml beep notification	

15 APPENDIX D: LOCALIZED INPUT CHARACTER SET

15.1 ENGLISH

Key	Upper case	Lower case
1	1.:;=_,-'&()	1.:;=_,-'&()
2	ABC2	abc2
3	DEF3	def3
4	GHI4	ghi4
5	JKL5	jkl5
6	MNO6	mno6
7	PQRS7	pqrs7
8	TUV8	tuv8
9	WXYZ9	wxyz9
0	0+	0+
*	* <SPACE>	* <SPACE>
#	#/\@	#/\@

15.2 FRENCH/FRANÇAIS

Key	Upper case	Lower case
1	1.:;=_,-'&()	1.:;=_,-'&()
2	ABC2ÀÂÇÁÅÆ	abc2àâçáåæ
3	DEF3ÉÈÊË	def3éèêë
4	GHI4ÎÏ	ghi4îï
5	JKL5	jkl5
6	MNO6ÑÓÒÔÖ	mno6ñóòôö
7	PQRS7	pqrs7
8	TUV8ÚÙÛÜ	tuv8úùûü
9	WXYZ9	wxyz9
0	0+	0+
*	* <SPACE>	* <SPACE>
#	#/\@	#/\@

15.3 SPANISH/ESPAÑOL

Key	Upper case	Lower case
1	1.:;=_,-'&()	1.:;=_,-'&()
2	ABC2ÁÀÇ	abc2áàç
3	DEF3ÉÈ	def3éè
4	GHI4ÏÍ	Ghi4ïí
5	JKL5	Jkl5
6	MNO6ÑÓÒ	mno6ñóò

Key	Upper case	Lower case
7	PQRS7	pqrs7
8	TUV8ÚÜ	tuv8úü
9	WXYZ9	wxyz9
0	0+	0+
*	* <SPACE>	* <SPACE>
#	#/\@	#/\@

15.4 GERMAN/DEUTSCH

Key	Upper case	Lower case
1	1.:;=_,-'&()	1.:;=_,-'&()
2	ABC2ÄÀ	abc2äà
3	DEF3É	def3é
4	GHI4	ghi4
5	JKL5	jkl5
6	MNO6Ö	mno6ö
7	PQRS7ß	pqrs7ß
8	TUV8Ü	tuv8ü
9	WXYZ9	wxyz9
0	0+	0+
*	* <SPACE>	* <SPACE>
#	#/\@	#/\@

15.5 ITALIAN/ITALIANO

Key	Upper case	Lower case
1	1.:;=_,-'&()	1.:;=_,-'&()
2	ABC2ÀCÇ	abc2àcç
3	DEF3ÉÈË	def3éèë
4	GHI4	ghi4
5	JKL5	jkl5
6	MNO6ÓÒ	mno6óò
7	PQRS7	pqrs7
8	TUV8Ù	tuv8ù
9	WXYZ9	wxyz9
0	0+	0+
*	* <SPACE>	* <SPACE>
#	#/\@	#/\@

15.6 PORTUGUESE/PORTUGÊS

Key	Upper case	Lower case
1	1.:;=_,-'&()	1.:;=_,-'&()
2	ABC2ÁÀÂÃÇ	abc2áàâãç
3	DEF3ÉÊ	def3éê
4	GHI4Í	ghi4í
5	JKL5	jkl5
6	MNO6ÓÔÕ	mno6óôõ
7	PQRS7	pqrs7
8	TUV8ÚÜ	tuv8úü
9	WXYZ9	wxyz9
0	0+	0+
*	* <SPACE>	* <SPACE>
#	#/\@	#/\@

16 APPENDIX E: XML SELF-CONFIGURATION

Mitel provides here a possible implementation for the self-configuration, this is just an example source code not supported by Mitel.

In this implementation targeted for Trixbox CE using FreePBX, the chosen policy is to have all the extensions pre-configured in the Asterisk database and use

- the extension
- the voice mail password

for the credentials to authenticate the user and link the MAC address of the phone to an extension.

The script checks the credentials but also uses a log file to trace the extensions that are already used (startup.cfg).

On top of the XML self-configuration we also provide here a “hot desking” capability using the “logout.php”, the user can logout his extension of his current phone and login somewhere else.

For instance the logout key can be configured as a softkey

```
softkey1 type: xml
softkey1 label: logout
softkey1                                                                value:
http://192.168.0.110/startup/logout.php?extension=$$SIPUSERNAME$$
softkey1 states: idle
```

Notes:

- TFTP server root directory is located at ‘/tftpboot’
- Asterisk directory is located at ‘/etc/asterisk’
- XML script and all the related files are located at the root directory of the HTTP server (typically /var/www/html) under the directory “startup“.
- The “startup” directory must have read/write for the HTTP server user.

aastra.cfg (typically located at /tftpboot)

Note: In this example, the Trixbox CE server is located at 192.168.0.110 and is also the TFTP server.

```
# Setup DHCP mode
dhcp: 1

# Setup TFTP server address
tftp server: 192.168.0.110

# Time server
time server disabled: 0
time server1: pool.ntp.org
time server2: 192.168.0.110

# Startup URI
action uri startup: http://192.168.0.110/startup/startup.php
```

gen-aastra script shell to generate the aastra.cfg file

```
#!/bin/sh
# To put on /usr/local/sbin
echo ""
echo "-----------------------------------------------------------------"
echo "Creating a default config file for Mitel phones"
echo "-----------------------------------------------------------------"
echo ""
echo "Creating /tftpboot/aastra.cfg..."
echo ""

IFCONFIG=`which ifconfig 2>/dev/null||echo /sbin/ifconfig`
IPADDR=`$IFCONFIG eth0|gawk '/inet addr/{print $2}'|gawk -F: '{print $2}'`

cat > /tftpboot/aastra.cfg <<EOF
# Setup DHCP mode
dhcp: 1

# Setup TFTP server address
tftp server: $IPADDR

# Time server
time server disabled: 0
time server1: pool.ntp.org
time server2: $IPADDR

# Startup URI
action uri startup: http://$IPADDR/xml/startup/startup.php

EOF

chmod 666 /tftpboot/aastra.cfg

echo "Created /tftpboot/aastra.cfg using $IPADDR for the proxy. If the"
echo "IP address of your Asterisk system changes run this script again and
reboot."
echo "Reboot your Mitel phones by disconnecting the power to the phone."
echo ""
```

Template files for the phones (typically located at /var/www/html/startup)

The following file is a template used to create the MAC.cfg file, there must be one file for each Mitel phone.

- Aastra6863i.cfg for the 6863i
- Aastra6865i.cfg for the 6865i
- Aastra6867i.cfg for the 6867i
- Aastra6867i.cfg for the 6869i
- Aastra6873i.cfg for the 6873i
- Mitel6920.cfg for the 6920
- Mitel6930.cfg for the 6930
- Mitel6940.cfg for the 6940

In these template files, you can use variables that will be replaced with the corresponding value by the script.

- $$AA_SIPAUTHNAME_AA$$ for the user SIP authentication name
- $$AA_SIPSECRET_AA$$ for the user SIP secret
- $$AA_SIPUSERNAME_AA$$ for the user SIP username
- $$AA_SIPCALLERID_AA$$ for the user SIP caller-id

- $$AA_PROXY_SERVER_AA$$ for the name/IP address of the proxy server
- $$AA_REGISTRAR_SERVER_AA$$ for the name/IP address of the registrar server

The template files can of course be completed with extra parameters (timezone, softkeys …).

```
# SIP Lines
sip line1 auth name: $$AA_SIPAUTHNAME_AA$$
sip line1 password: $$AA_SIPSECRET_AA$$
sip line1 user name: $$AA_SIPUSERNAME_AA$$
sip line1 display name: $$AA_SIPCALLERID_AA$$
sip line1 screen name: $$AA_SIPCALLERID_AA$$
sip line1 proxy ip: $$AA_PROXY_SERVER_AA$$
sip line1 proxy port: 5060
sip line1 registrar ip: $$AA_REGISTRAR_SERVER_AA$$
sip line1 registrar port: 5060
sip line1 vmail: *98
sip line1 mode: 0

# Action URI
action uri startup:
```

startup.php (located at /var/www/html/startup)

```
<?php
########################################################################
# Sample script for self-configuration with Trixbox CE/Asterisk
# Mitel SIP Phones R2.3.0 or better
#
# php source code
# Provided by Mitel 2008
#
# Phone supported
#    Mitel 68xx and 69xx series
########################################################################

########################################################################
# Includes
########################################################################
require_once('include/config.inc.php');
require_once('include/backend.inc.php');
require_once('phpagi/misc.php');
require_once('phpagi/phpagi-asmanager.php');

########################################################################
# Private functions
########################################################################

########################################################################
# Aastra_decode_HTTP_header()
#
# Returns an array
#   0 Phone Type
#   1 Phone MAC Address
#   2 Phone firmware version
########################################################################
function Aastra_decode_HTTP_header()
{
$user_agent=$_SERVER["HTTP_USER_AGENT"];
if(stristr($user_agent,"Aastra"))
       {
       $value=preg_split("/ MAC:/",$user_agent);
       $fin=preg_split("/ /",$value[1]);
       $value[1]=preg_replace("/\-/","",$fin[0]);
       $value[2]=preg_replace("/V:/","",$fin[1]);
       }
else
       {
       $value[0]="MSIE";
       $value[1]="NA";
       $value[2]="NA";
       }

$value[3]=$_SERVER["REMOTE_ADDR"];

return($value);
}

########################################################################
# lookup_config_file(extension)
# Checks if extension is already in use.
#
# Parameters
#     extension     extension t check
#
# Returns 1 if extension already in use
########################################################################
```

```
function lookup_config_file($extension)
{
$config="startup.cfg";

# Init return
$return=0;

# Read config file
$array = @parse_ini_file($config, true);

# Test MAC address
if($array[$extension]['mac']!="") $return=1;

return($return);
}

########################################################################
# update_config_file(extension,mac,ip,model)
# Update the config file with the new extension parameters
#
# Parameters
#     extension     user extension
#     mac           MAC address of the phone
#     ip            IP address of the phone
#     model         Phone model
########################################################################
function update_config_file($extension,$mac,$ip,$model)
{
$config="startup.cfg";

# Read config file
$array = @parse_ini_file($config, true);
if($array==NULL) $array=array();

# Update value
$array[$extension]['mac']=$mac;
$array[$extension]['ip']=$ip;
$array[$extension]['model']=$model;

# Update config file
reset($array);
$handle = @fopen($config, "w");
if($handle)
       {
       while ($v = current($array))
              {
              fputs($handle,"[".key($array)."]"."\n");
              fputs($handle,"mac=".$v['mac']."\n");
              fputs($handle,"ip=".$v['ip']."\n");
              fputs($handle,"model=".$v['model']."\n\n");
              next($array);
              }
       fclose($handle);
       }
}

########################################################################
# create_mac(extension,mac,username,secret,callerid,model)
# Creates the MAC.cfg file for the user.
#
# Parameters
#     extension     user extension
#     mac           MAC address of the phone
```

```
#     username      SIP authname
#     secret        SIP secret
#     callerid      User CallerID
#     model         Phone model
#
####################################################################
function create_mac($mac,$extension,$username,$secret,$callerid,$model)
{
Global $AA_PROXY_SERVER,$AA_REGISTRAR_SERVER;

$value=preg_split("/ /",$callerid);
$result=$value[0]." ".$value[1];
$result=preg_replace("/</","(",$result);
$result=preg_replace("/>/",")",$result);

# Prepare replace strings
$search=array('/\$\$AA_SIPAUTHNAME_AA\$\$/','/\$\$AA_SIPSECRET_AA\$\$/','/\$\$A
A_SIPUSERNAME_AA\$\$/','/\$\$AA_SIPCALLERID_AA\$\$/','/\$\$AA_PROXY_SERVER_AA\$
\$/','/\$\$AA_REGISTRAR_SERVER_AA\$\$/');
$replace=array($username,$secret,$extension,$result,$AA_PROXY_SERVER,$AA_REGIST
RAR_SERVER);

$read = @fopen($model.".cfg", "r");
if($read)
       {
       $write = @fopen("/tftpboot/".$mac.".cfg", "w");
       if($write)
              {
              # Create file header
              while($line=fgets($read,200))
                     {
                     $line = preg_replace($search, $replace, $line);
                     fputs($write,$line);
                     }
              fputs($write,"\n");
              fclose($write);
              }
       fclose($read);
       }
}

####################################################################
# get_callerid(user)
#
# This function retrieves the user callerID of a user in the Asterisk
# registry (FreePBX 2.3)
#
# Parameters
#   @user           user ID
#
# Returns
#   CallerID as a string
####################################################################
function get_callerid($user)
{
Global $ASTERISK_LOCATION;

# Try in the config file first
$sip_array = parse_ini_file($ASTERISK_LOCATION."sip_additional.conf", true);

# Extension exists?
if ($sip_array[$user]==NULL) $callerid="Unknown";
else
```

```
	{
	# FreePBX ?
	if(strstr($sip_array[$user]['callerid'],'device'))
		{
		# Connect to AGI
		$as = new AGI_AsteriskManager();
		$res = $as->connect();

		# Get value in the database
		$res = $as->Command('database get AMPUSER '.$user.'\/cidname');
		$line=split("\n", $res['data']);
		$cid=split(" ", $line[1]);
		$callerid= $cid[1].' '.$cid[2];

		# Disconnect properly
		$as->disconnect();
		}
	else
		{
		# Retrieve the value
		$value=explode(" <",$sip_array[$user]['callerid'],2);
		$callerid=$value[0];
		}
	}

# Return Caller ID
return($callerid);
}

########################################################################
# get_username(user)
#
# This function retrieves the username of a user in the Asterisk
# registry (FreePBX 2.3)
#
# Parameters
#   @user           user ID
#
# Returns
#   username as a string
########################################################################
function get_username($user)
{
Global $ASTERISK_LOCATION;

# Get all the user data
$sip_array = @parse_ini_file($ASTERISK_LOCATION."sip_additional.conf", true);

# Collect data
if($sip_array[$user]['username']!='') $username=$sip_array[$user]['username'];
else
	{
	# Connect to AGI
	$as = new AGI_AsteriskManager();
	$res = $as->connect();

	# Get value in the database
	$res = $as->Command('database get DEVICE '.$user.'\/user');
	$line=split("\n", $res['data']);
	$cid=split(" ", $line[1]);
	$username= $cid[1];

	# Disconnect properly
```

```
        $as->disconnect();
        }

# Return answer
return($username);
}

########################################################################
# get_secret(user)
#
# This function retrieves the SIP password (secret) of a user in the
# Asterisk configuration
#
# Parameters
#   @user           user ID
#
# Returns
#   secret as a string
########################################################################
function get_secret($user)
{
Global $ASTERISK_LOCATION;

# Get all the user data
$sip_array = @parse_ini_file($ASTERISK_LOCATION."sip_additional.conf", true);

# Return answer
return($sip_array[$user]['secret']);
}


########################################################################
# Main code
########################################################################
# GLOBAL VARIABLES
$XML_SERVER = "http://".$_SERVER['SERVER_ADDR'].$_SERVER['SCRIPT_NAME'];
$AA_PROXY_SERVER = $_SERVER['SERVER_ADDR'];
$AA_REGISTRAR_SERVER = $_SERVER['SERVER_ADDR'];
$ASTERISK_LOCATION = "/etc/asterisk/";

# Retrieve parameters
$extension=$_GET["extension"];
$password=$_GET["password"];
$action=$_GET["action"];
$step=$_GET["step"];

# Set content type
header("Content-Type: text/xml");

# Reboot
if($action=="reboot")
        {
        $output = "<AastraIPPhoneExecute>\n";
        $output .= "<ExecuteItem URI=\"Command: Reset\"/>\n";
        $output .= "</AastraIPPhoneExecute>\n";
        header("Content-Length: ".strlen($output));
        echo $output;
        exit;
        }

# Input Extension
if ($extension=="")
        {
```

```
	$output = "<AastraIPPhoneInputScreen type=\"number\" LockIn=\"yes\">\n";
	$output .= "<Title>Initial startup</Title>\n";
	$output .= "<Prompt>Enter Extension</Prompt>\n";
	$output .= "<URL>$XML_SERVER</URL>\n";
	$output .= "<Parameter>extension</Parameter>\n";
	$output .= "<Default></Default>\n";
	$output .= "</AastraIPPhoneInputScreen>\n";
	header("Content-Length: ".strlen($output));
	echo $output;
	exit;
	}

# Input Password
if ($password=="")
	{
	$output  =  "<AastraIPPhoneInputScreen  type=\"number\"  password=\"yes\"
LockIn=\"yes\" destroyOnExit=\"yes\">\n";
	$output .= "<Title>Initial startup</Title>\n";
	$output .= "<Prompt>Enter Password</Prompt>\n";
	$output .= "<URL>$XML_SERVER?extension=$extension</URL>\n";
	$output .= "<Parameter>password</Parameter>\n";
	$output .= "<Default></Default>\n";
	$output .= "</AastraIPPhoneInputScreen>\n";
	header("Content-Length: ".strlen($output));
	echo $output;
	exit;
	}

# IF authentication failed
if (!$userinfo = verify_user($extension, $password))
	{
	# Display error
	$output        =        "<AastraIPPhoneTextScreen        LockIn=\"yes\"
destroyOnExit=\"yes\">\n";
	$output .= "<Title>Authentication failed</Title>\n";
	$output .= "<Text>Wrong user and password.</Text>\n";
	$output .= "</AastraIPPhoneTextScreen>\n";
	header("Content-Type: text/xml");
	header("Content-Length: ".strlen($output));
	echo $output;
	exit;
	}

# IF already configured
if(lookup_config_file($extension)==1)
	{
	# Display error
	$output        =        "<AastraIPPhoneTextScreen        LockIn=\"yes\"
destroyOnExit=\"yes\">\n";
	$output .= "<Title>Error</Title>\n";
	$output .= "<Text>Extension already in use.</Text>\n";
	$output .= "</AastraIPPhoneTextScreen>\n";
	header("Content-Type: text/xml");
	header("Content-Length: ".strlen($output));
	echo $output;
	exit;
	}

# Get all the user data
$sip_array = parse_ini_file($ASTERISK_LOCATION."sip_additional.conf", true);

# If user not found
if ($sip_array[$extension]==NULL)
```

```
        {
        # Display error
        $output         =         "<AastraIPPhoneTextScreen         LockIn=\"yes\"
destroyOnExit=\"yes\">\n";
        $output .= "<Title>Internal error</Title>\n";
        $output .= "<Text>Extension is not provisioned.</Text>\n";
        $output .= "</AastraIPPhoneTextScreen>\n";
        header("Content-Type: text/xml");
        header("Content-Length: ".strlen($output));
        echo $output;
        exit;
        }
else
        {
        # Collect data
        $username=get_username($extension);
        $secret=get_secret($extension);
        $callerid=get_callerid($extension);
        }

# Get MAC address and type of phone
$value=Aastra_decode_HTTP_header();

# Create mac.cfg
create_mac($value[1],$extension,$username,$secret,$callerid,$value[0]);

# Update config file
update_config_file($extension,$value[1],$value[3],$value[0]);

# Create Reboot screen
$output = "<AastraIPPhoneTextScreen destroyOnExit=\"yes\">\n";
$output .= "<Title>REBOOT</Title>\n";
$output .= "<Text>Reboot</Text>\n";
$output .= "</AastraIPPhoneTextScreen>\n";
header("Refresh: 1; url=".$XML_SERVER."?action=reboot");

# Display XML Object
header("Content-Type: text/xml");
header("Content-Length: ".strlen($output));
echo $output;
exit;
?>
```

logout.php (located at /var/www/html/startup)

```
<?php
########################################################################
# Sample script for shot desking with Trixbox CE/Asterisk
# Mitel SIP Phones R2.3.0 or better
#
# php source code
# Provided by Mitel 2008
#
# Phone supported
#    Mitel 5i series
########################################################################

########################################################################
# Includes
########################################################################
require_once('include/config.inc.php');
require_once('include/backend.inc.php');

########################################################################
# Private functions
########################################################################

########################################################################
# Aastra_decode_HTTP_header()
#
# Returns an array
#   0 Phone Type
#   1 Phone MAC Address
#   2 Phone firmware version
########################################################################
function Aastra_decode_HTTP_header()
{
$user_agent=$_SERVER["HTTP_USER_AGENT"];
if(stristr($user_agent,"Aastra"))
       {
       $value=preg_split("/ MAC:/",$user_agent);
       $fin=preg_split("/ /",$value[1]);
       $value[1]=preg_replace("/\-/","",$fin[0]);
       $value[2]=preg_replace("/V:/","",$fin[1]);
       }
else
       {
       $value[0]="MSIE";
       $value[1]="NA";
       $value[2]="NA";
       }

$value[3]=$_SERVER["REMOTE_ADDR"];

return($value);
}


########################################################################
# update_config_file(extension)
# Update the config file deleting the current extension
#
# Parameters
#     extension     user extension
########################################################################
function update_config_file($extension)
{
# Config file
```

```
$config="startup.cfg";

# Read config file
$array = @parse_ini_file($config, true);
if($array==NULL) $array=array();

# Update config file
$handle = @fopen($config, "w");
if($handle)
	{
	foreach($array as $key=>$value)
		{
		if($key!=$extension)
			{
			fputs($handle,"[".$key."]"."\n");
			fputs($handle,"mac=".$value['mac']."\n");
			fputs($handle,"ip=".$value['ip']."\n");
			fputs($handle,"model=".$value['model']."\n\n");
			}
		}
	fclose($handle);
	}
}

######################################################################
# delete_mac(mac)
# Deletes the MAC.cfg file for the user.
#
# Parameters
#     mac           MAC address of the phone
#
######################################################################
function delete_mac($mac)
{
# Delete MAC.cfg
@unlink("/tftpboot/".$mac.".cfg");
}

######################################################################
# Main code
######################################################################
# GLOBAL VARIABLES
$XML_SERVER = "http://".$_SERVER['SERVER_ADDR'].$_SERVER['SCRIPT_NAME'];

# Retrieve parameters
$extension=$_GET["extension"];
$password=$_GET["password"];
$action=$_GET["action"];

# Set content type
header("Content-Type: text/xml");

# Process action
switch($action)
	{
	case 'display':
		# Reboot
		$output      =      "<AastraIPPhoneTextScreen      LockIn=\"yes\"
destroyOnExit=\"yes\">\n";
		$output .= "<Title>REBOOT</Title>\n";
		$output .= "<Text>Reboot.</Text>\n";
		$output .= "</AastraIPPhoneTextScreen>\n";
		break;
```

```
      default:
            # Input Password
            if ($password=='')
                  {
                  $output    =    "<AastraIPPhoneInputScreen    type=\"number\"
password=\"yes\" LockIn=\"yes\" destroyOnExit=\"yes\">\n";
                  $output .= "<Title>Logout</Title>\n";
                  $output .= "<Prompt>Enter Password</Prompt>\n";
                  $output .= "<URL>$XML_SERVER?extension=$extension</URL>\n";
                  $output .= "<Parameter>password</Parameter>\n";
                  $output .= "<Default></Default>\n";
                  $output .= "</AastraIPPhoneInputScreen>\n";
                  }
            else
                  {
                  # IF authentication fails
                  if (!verify_user($extension,$password))
                        {
                        # Display error
                        $output  =  "<AastraIPPhoneTextScreen  LockIn=\"yes\"
destroyOnExit=\"yes\">\n";
                        $output .= "<Title>Authentication failed</Title>\n";
                        $output .= "<Text>Wrong credentials.</Text>\n";
                        $output .= "</AastraIPPhoneTextScreen>\n";
                        }
                  else
                        {
                        # Get MAC address and type of phone
                        $value=Aastra_decode_HTTP_header();

                        # Erase mac.cfg
                        delete_mac($value[1]);

                        # Update config file
                        update_config_file($extension);

                        # Reboot needed
                        $output = "<AastraIPPhoneExecute>\n";
                        $output                .=               "<ExecuteItem
URI=\"".$XML_SERVER."?action=display"."\"/>\n";
                        $output      .=      "<ExecuteItem      URI=\"Command:
FastReboot\"/>\n";
                        $output .= "</AastraIPPhoneExecute>\n";
                        }
                  }
            break;
      }


# Display XML Object
header("Content-Type: text/xml");
header("Content-Length: ".strlen($output));
echo $output;
exit;
?>
```

17 APPENDIX F: CSV BASED DIRECTORY

17.1 INTRODUCTION

The application described in this chapter is a generic directory application using a CSV file as data source as well as a server-side speed dial application which can be linked from the directory application.

The directory application allows the user to perform a lookup based on

- The first letters of the first name
- The first letters of the last name
- The first letters of the Company name
- (at least 3) Letters anywhere in the complete name or company name

Also, it allows the user to configure:

- The results display format: ‘first name last name’ or ‘last name first name’
- Sorting index: ‘last name’ or ‘first name’

The directory application can also be password protected (configured in config/directory.conf), the password is requested only for the first use, once authenticated by its MAC address, the phone no longer requires the password check.

The source code is provided in the XML SDK as a zip file called csv_directory_5.0.0.zip

---

Note: the source code is not supported by Mitel, it is provided only as an example.

17.2 PHONE COMPATIBILITY

The directory and speed dial applications are available for

- Mitel 6863i/6865i
- Mitel 6867i/6869i/6873i
- Mitel 6920/6930/6940

But some limitations apply in the directory application when the phone is a non softkey phone a user cannot:

- set a speed dial from the directory application
- change sorting index and display format
- access to record details
- perform a search using first name or last name

17.3 INSTALLATION

The provided scripts must be extracted under the ROOT directory of the HTTP server in a ‘xml’directory, a cache directory (default /var/cache/aastra) must also be created with read/write access for the HTTP server user.

The cache directory can be configured in config/server.conf, see chapter 17.6.1 for more details.

17.4 XML KEY CONFIGURATION

Directory

The uri to use is

http://myserver.com/xml/directory/directory.php?user=USER&source=SOURCE

Where

- myserver.com is the name or IP address of your HTTP server
- USER is the userID, if not provided the script uses the phone MAC address as the userID.
- SOURCE is the name of the csv file located in the 'directory' directory, if not provided, the script uses the file configured as default in config/directory.conf

Speed dial

The uri to use is

http://myserver.com/xml/directory/speed.php?user=USER

Where

- myserver.com is the name or IP address of your HTTP server
- USER is the userID, if not provided the script uses the phone MAC address as the userID.



Note: USER must be the same for both applications in order to have the speed dial integrated in the directory application.

17.5 CSV FILE FORMAT

The CSV file must respect the following format:

```
First name, Last name, Company, Title, Work, Home, Mobile
```

The application is provided with a sample database 'directory/directory.txt' where all the fields have been randomly generated.

17.6 CONFIGURATION FILES

17.6.1 SERVER.CONF

This file includes configuration parameters for the XML server.

```
[General]
# Public IP address (optional), if not provided $_SERVER['HTTP_HOST'] is used
public=
# Path for the Mitel cache directory (optional), if not provided
/var/cache/aastra is used
cache=
# Path for the TFTP root directory (optional) if not provided /tftpboot is used
tftp=
# Path for the XML directory behind the HTTP Server (optional), default is xml
xmldirectory=
# Mode DEBUG (0 or 1), default is 0
debug=
# Mode TRACE (0 or 1), default is 0
trace=
# Forced language (optional), can be en, de, fr...
language=
```

When activated, the traces are stored in a log file located in the 'cache' directory (by default /var/cache/aastra) indexed by the date, a file is generated per day.

17.6.2 DIRECTORY.CONF

This file includes configuration parameters for the Directory application.

```
##############################################################################
# directory.conf
#
# Configuration file for the directory application
##############################################################################

##############################################################################
# Global configuration
#
# [general]
# password          To protect the application with a password
# speeddial         Configure is speedial is enabled (0 or 1)(optional)
# default           Name of the default csv file located in the directory
##############################################################################
[General]
password=
speeddial=
default=directory
##############################################################################
# Dialing profile
#
# [Dialplan]
# country           country code
# long distance     long distance prefix
# international     international prefix
# outgoing          outgoing prefix
# local             list of local phone numbers prefixes separated by a comma
##############################################################################
[Dialplan]
country=1
long distance=1
international=011
outgoing=9
local=
```

17.7 APPLICATION FILES

/	**Root Directory**
language.ini	Language configuration file
license.txt	License file
/directory/	**Directory for the applications**
directory.php	Directory application
directory.txt	Sample CSV file
speed.php	Speed dial application
/config/	**Directory for configuration files**
server.conf	Server configuration

directory.conf	Directory configuration
/include/	**Directory for common includes**
AastraCommon.php	Common Mitel functions
AastraIPPhone.php	
AastraIPPhone.class.php	Mitel XML objects php classes
AastraIPPhoneConfiguration.class.php	
AastraIPPhoneConfigurationEntry.class.php	
AastraIPPhoneExecute.class.php	
AastraIPPhoneExecuteEntry.class.php	
AastraIPPhoneInputScreen.class.php	
AastraIPPhoneInputScreenEntry.class.php	
AastraIPPhoneSoftkey.class.php	
AastraIPPhoneSoftkeyParam.class.php	
AastraIPPhoneSoftkeyEntry.class.php	
AastraIPPhoneStatus.class.php	
AastraIPPhoneStatusEntry.class.php	
AastraIPPhoneTextMenu.class.php	
AastraIPPhoneTextMenuEntry.class.php	
AastraIPPhoneTextScreen.class.php	

18 APPENDIX G: LDAP DIRECTORY

18.1 INTRODUCTION

This appendix describes how to provide access to the Corporate Directory (Active Directory) from the Mitel SIP phones. As LDAP protocol is used to access Active Directory, this document is also valid for any other LDAP enabled directory server. This document is solely about user initiated directory lookup, it does not cover Corporate Directory based caller ID lookup for incoming or outgoing calls.

There is no native LDAP client in the Mitel SIP phone, but the phone's inbuilt XML browser can be used to access a XML Proxy Server which then communicates with the LDAP server. This architecture offers more flexibility than a native LDAP client in the phone since the logic of the LDAP directory application lies in the XML LDAP application (e.g. a PHP script) running on the XML Proxy Server rather than in the LDAP client embedded in the phone's firmware. The following features can be easily controlled by changing the XML LDAP application on the XML Proxy Server:

- Customized search fields on the phone (e.g. first name / last name / any name / department / …)
- Customized LDAP-attributes to be included in the search result (such as title, department, address)
- Number translation (digits to be removed from / added to the numbers found in the LDAP before dialling – e.g. external prefix)

The message flow is given by the diagram below:

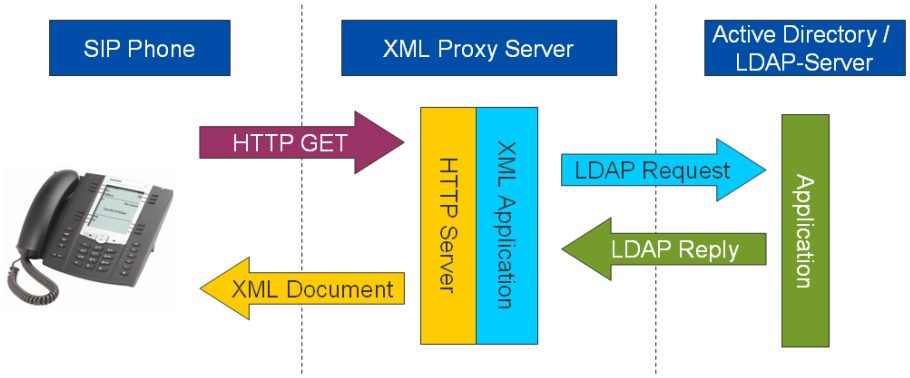


Figure 95: LDAP directory message flow

The directory application allows the user to perform a lookup based on

- The first letters of the first name AND last name
- (at least 3) Letters anywhere in the complete name

The source code is provided in the XML SDK as a zip file called ldap_directory_5.0.0.zip

---

Note: the source code is not supported by Mitel, it is provided only as an example.

18.2 INSTALLATION

The provided scripts must be extracted under the ROOT directory of the HTTP server in a ‘xml’directory, a cache directory (default /var/cache/aastra) must also be created with read/write access for the HTTP server user.

The cache directory can be configured in config/server.conf, see chapter 18.5.1 for more details.

The Web server must support PHP 5.x or higher and the PHP LDAP extension must be available in PHP. Typically, Apache Web server is used. Apache runs on most operating systems.

Windows platforms

To enable the PHP LDAP extension, this line must be inserted into php.ini:

extension=php_ldap.dll

Linux platforms

Just install the php-ldap package (via yum or via rpm)

To check whether PHP 5.x (or higher) and the PHP LDAP extensions are properly installed, create a PHP script with the following content and save it under the Web server's root directory:

<?php phpinfo(); ?>

Then point your Web browser to this script. Check the PHP version and make sure there is a section "ldap" with a table containing the key "LDAP Support" and corresponding value "enabled"

18.3 NETWORK REQUIREMENTS

18.3.1 PHONE -> XML PROXY SERVER

The phone must be able to reach the XML Proxy Server. Typically, the XML Proxy Server will run on port 80/tcp, but any other port can be used as well. The phone can be behind NAT as the TCP requests will always be phone initiated.

In case there is a firewall between the phone and the XML Proxy Server, the firewall must allow phone initiated TCP sessions towards the Web server port on the XML Proxy Server.

18.3.2 XML PROXY SERVER -> LDAP-SERVER

The XML Proxy Server must be able to reach the LDAP-Server on port 389/tcp (default LDAP port) or 3268/tcp (default LDAP port on Active Directory Server) or any other port that is used for LDAP access.

Requests will always be XML Proxy Server initiated.

18.4 ACTIVE DIRECTORY / LDAP-SERVER REQUIREMENTS

The LDAP / Active Directory Server must allow LDAP search requests. Depending on the server's configuration, anonymous access (anonymous bind) is allowed. If not, a username (User DN) and password must be provided. Active Directory typically does not allow anonymous access.

Also the Search Base / Root DN needs to be known. This information should be requested from the LDAP / Active Server administrator.

To test the LDAP server access, use a tool like "Softerra LDAP Browser" (free). If anonymous access is supported, setup a new profile with "Anonymous Bind" selected. Otherwise provide User DN and password. Configure the server's IP address and port plus the "Base DN" and you should be able to browse the LDAP tree.

18.5 CONFIGURATION FILES

18.5.1 SERVER.CONF

This file includes configuration parameters for the XML server.

```
[General]
# Public IP address (optional), if not provided $_SERVER['HTTP_HOST'] is used
public=
# Path for the Mitel cache directory (optional), if not provided
/var/cache/aastra is used
cache=
# Path for the TFTP root directory (optional) if not provided /tftpboot is used
tftp=
# Path for the XML directory behind the HTTP Server (optional), default is xml
xmldirectory=
# Mode DEBUG (0 or 1), default is 0
debug=
# Mode TRACE (0 or 1), default is 0
trace=
# Forced language (optional), can be en, de, fr...
language=
```

When activated, the traces are stored in a log file located in the ‘cache’ directory (by default /var/cache/aastra) indexed by the date, a file is generated per day.

18.5.2 LDAP_DIRECTORY.CONF

This file includes configuration parameters for the LDAP Directory application.

```
################################################################################
################
# ldap_directory.conf
#
# Configuration file for the LDAP directory application
################################################################################
################

################################################################################
################
# Global configuration
#
# [LDAP_Server]
# hostname    Hostname or IP address of LDAP server
# port        TCP server port of LDAP server. Typically this is 389 or 3268
# baseDN      Base / Root Search DN. Example: dc=company, dc=com
# userdomain  User domain to connect to the LDAP server (optional)
# username     User DN to connect to the LDAP server. Leave blank if anonymous
bind is used.
# password    Password for the User DN. Leave blank if anonymous bind is used.

################################################################################
################
[LDAP_Server]
hostname=
port=3268
basedn=
userdomain=
username=
password=

################################################################################
################
# Dialing profile
#
# [Dialplan]
# countrycode   Country code
#                  If phone number starts with "+<countrycode>", this will be
removed
#
# longdistance  Long distance prefix
#                This prefix will be added to the number in case +<countrycode>
has been stripped.
#               Leave blank if not needed.
#
# international International prefix
#               The "+" sign will be replaced with this prefix
#
# outgoing      Outgoing prefix
#               Prefix that needs to be appended for outgoing calls
#               (all call but local calls), e.g. 0 or 9
#
# local           List of local PBX number prefixes separated by a comma (in
national format).
#               Prefix will be removed, no outgoing prefix will be added.
#               Examples: local=905760,978262
#                --> 9057602222 will be replaced by 2222 and 9782623333 will be
replaced by 3333.
#
# localextlen  Numbers with this number of digits or less will be treated as
local PBX extensions
#                 No prefix added (useful in case numbers are stored in local
format in LDAP)
```

```
################################################################################
###############
[Dialplan]
countrycode=1
longdistance=1
international=011
outgoing=
local=
localextlen=4
```

19 APPENDIX H: PICTURECALLERID (6867I, 6869I, 6873I, 6920, 6930 AND 6940)

19.1 INTRODUCTION

This feature, available on the Mitel 6867i, 6869i, 6873i, 6920, 6930 and 6940 allows the display of a picture referenced by the Caller ID number for an incoming and an outgoing call.

When the user makes or receives a call, the phone (by configuration) fetches an image stored on a TFTP/FTP/HTTP/HTTPS server and displays it.

For instance, if the incoming or outgoing number is “9725551234” the phone makes a request for a file named “9725551234.png” which must be

6867i/6920

- Display size is 86x86 pixels (image is rescaled)
- png format
- 24 or 32 bits depth colors

6869i/6930

- Display size is 96x96 pixels (image is rescaled)
- png format
- 24 or 32 bits depth colors

6873i/6940

- Display size is 170x170 pixels (image is rescaled)
- png format
- 24 or 32 bits depth colors

in order to be displayed for consecutive calls. The image is cached in the phone until the next reboot.

If no image matching the request is found on the server or in its cache, the phone displays a default image (below).



This implementation means that the server must host all the pictures in the right format which can be pretty tedious.

19.2 APPLICATION IMPLEMENTATION

To simplify the deployment of the picture Caller ID, Mitel has developed an application which answers to the phone requests but also:

- converts any size picture (png, jpeg, gif) to the right format and cache the converted picture,
- allows number/picture matching (cell phone number, home number…),
- allows phone number pattern matching,
- allows to customize the matching picture by adding a custom label.

With this application, an administrator does not have to convert the pictures to the right format, he just has to put the pictures in any format (jpeg, gif or png) in the “pictures” directory for instance using XXXX.yyy where XXXX is the user extension number.

The administrator can also create number or pattern matching to display all kind of pictures using the configuration file “pictureID.conf” as described in chapter 19.5.2.3.

Here is how the algorithm works (NUMBER is the incoming or outgoing number)

- Checks if NUMBER.png is in the cache directory (if yes sends the image back)
- Remove local prefixes and check the transformed number (if yes sends the image back)
- Check basic mapping and check the transformed number (if yes sends the image back)
- Check advanced pattern matching and check the transformed number (if yes sends the image back)
- Remove external prefix and check the transformed number (if yes sends the image back)
- Remove international prefix and check the transformed number (if yes sends the image back)
- Sends a HTTP 404 (Image Not found) or the configured image

19.3 REQUIREMENTS/COMPATIBILITY

19.3.1 HTTP SERVER

Operating System:

- Microsoft Windows Server 2003/2008
- Microsoft XP/Vista/7
- Most flavors of Linux

HTTP Server:

- Microsoft IIS
- Apache
- Lighttpd
- PHP
    - Release 5.0 or better,
    - Configured to be called by the HTTP server for '.php' files
    - php-gd extension

19.3.2 COMPATIBILITY

The Picture Caller ID application is available only for the following phones:

- Mitel 6867i, 6869i and 6873i
- Mitel 6920, 6930 and 6940

19.4 INSTALLATION

19.4.1 INTRODUCTION

The Picture Caller ID application is provided to allow generic implementation and also for testing purpose as a zip file which includes only the source code to be installed on an existing HTTP Server "pictureCallerID_5.0.0.zip".

19.4.2 HTTP SERVER

The Picture Caller ID application is a single php script which can be installed anywhere behind the root directory of any HTTP Server supporting php scripts.

19.4.3 PACKAGE INSTALLATION

Just untar the provided tarball wherever you want behind the HTTP Server root directory (typically /var/www/html with Apache on Linux).

Just make sure that

- The “cache” directory is accessible in read/write for the HTTP Server user
- The “pictures” directory is accessible in read for the HTTP Server user
- The Picture Caller ID can be relocated under any directory.

19.4.4 TEST YOUR INSTALLATION

To test the HTTP server direct your Web browser (Internet Explorer, Firefox…) to

http://MYSERVERIP/imageServer.php?0000.png if the application has been installed at the HTTP Server root directory

or

http://MYSERVERIP/PATH/imageServer.php?0000.png if the application has been installed at a different directory

Where MYSERVERIP is the IP address/name of your server and PATH the directory behind the HTTP Server root directory.

You should see the following page:



19.4.5 TROUBLESHOOTING

- If the previous test did not work here are a couple of things you may want to check:
- Is the HTTP Server running?
- Is php supported?
- Is php-gd extension installed?
- Are the user rights OK for “pictures” and “cache” directories?

19.4.6 WINDOWS PC USING XAMPP LITE

Please refer to chapter 11.2 on how to use XAMPP as an http server for this application.

19.5 CONFIGURATION

19.5.1 PHONE CONFIGURATION

The Picture Caller ID feature is configured using a parameter called “`image server uri`” in the phone config files (aastra.cfg or MAC.cfg)

image server uri: URL to access the Picture Caller ID application

HTTP Server implementation

image server uri: http://MYSERVERIP/PATH/imageServer.php?

- where MYSERVERIP is the IP address/name of your HTTP server and PATH the directory behind the HTTP Server root directory.

XAMPP Implementation

image server uri: http://MYPC/imageServer.php?

- where MYPC is the IP address/name of the PC where the XAMPP package has been installed.

Note: the "?" at the end of the uri is mandatory, it is not a typo.

19.5.2 APPLICATION CONFIGURATION

The application can be configured by modifying 'pictureID.conf' in the main directory; this configuration file contains 3 sections.

Note: Changes in this file are dynamic.

19.5.2.1 General parameters

This section includes the global configuration parameters:

- Location of the 'pictures' directory
- Location of the 'cache' directory
- Enable or disable upscaling pictures
- Name and customization of the default picture (optional)

```
################################################################################
# Global parameters
#
# [General]
# pictures          Directory for the raw pictures
#                    Default is ./pictures
#                   Must be r for HTTP Server user
#
# cache             Directory for the cached pictures
#                    Default is ./cache
#                   Must be r/w for HTTP Server user
#
# blowup            Enable  (1)  or  disable  (0)  image  scaling  smaller  than
150x200 pixels.
#                   This may/will lead to grainy/pixelized images.
#                   Default is disable
# default           Provides a default image when the number does not match
#                   any picture. A 404 is sent if no number is provided though.
#                   default=mapped_number                                    or
mappedname,LABEL|YPOSITION|ALIGNMENT|COLOR|FONTSIZE
#                    See the [Numbers] section for the format of the line
################################################################################
[General]
pictures=
cache=
blowup=
default=
```

19.5.2.2 Dialing plan

This section contains the PBX dialing plan which will be used to transform the numbers:

- Country code
- Long distance prefix
- International prefix
- Outgoing prefix
- Local prefix(es)

```
################################################################################
# Dialing profile
#
# [Dialplan]
# country           Country code
#                   If phone number starts with "+<country>", this will be
#                   removed
#
# long distance     Long distance prefix
#                   This prefix will be added to the number in case +<country>
#                   has been stripped.
#                   Leave blank if not needed.
#
# international     International prefix
#                   The "+" sign will be replaced with this prefix
#
# outgoing          Outgoing prefix
#                   Prefix that needs to be appended for outgoing calls
#                   (all call but local calls), e.g. 0 or 9
#
# local             List of local PBX number prefixes separated by a comma
#                   (in national format). Prefix will be removed, no outgoing
#                   prefix will be added.
#                   Examples: local=905760,978262
#                   --> 9057602222 will be replaced by 2222 and 9782623333 will
#                   be replaced by 3333.
################################################################################
[Dialplan]
country=1
long distance=1
international=011
outgoing=
local=
```

19.5.2.3 Number transformations

This section describes the number transformations you want the application to use to display a given picture as well as the optional customization of the picture.

Each line is formatted like this

- Number and/or patterns=picture number or name,[Optional customization]
- Pattern format is P followed by:
- any digit including * or #
- X digit between 0 and 9
- Z digit between 1 and 9
- N digit between 2 and 9

The optional customization allows you to add a label on the displayed picture by defining:

- the label itself
- its vertical position in pixels (1 to 200)
- its alignment (left, center, right)
- its color
- its font size

```
#############################################################################
# Number mapping
#
# Line Format
# [Numbers]
#            number   and/or   patterns   comma   separated=mapped_number   or
mappedname,LABEL|YPOSITION|ALIGNMENT|COLOR|FONTSIZE
#    Simple number matching
#        it can be used to map multiple phone numbers (cell, home , other) to a
user
#    Pattern matching
#        it can be used to map open numbers
#        Pattern format is P followed by
#             any digit including * or #
#             X digit between 0 and 9
#             Z digit between 1 and 9
#             N digit between 2 and 9
#              . indicates an open length number, must be positioned at the end
of the pattern
#    The entries in the [Numbers] section are processed based on the order in
#    the file, so it is recommended to put the open patterns at the end.
#
# Where
#    LABEL is the label to be displayed on the picture
#      YPOSITION is the vertical position of the label (between 1 and 200),
default is 100
#    ALIGNMENT is the horizontal alignment left|center|right, default is center
#                              COLOR       is       the       label       color
yellow|orange|pink|purple|black|grey|red|brown|tan|magenta|blue|green|white
(default is white)
#    FONTSIZE is the font size in pixels (between 8 and 24) default is 10
#
# Example
# [Numbers]
# 0795551234=2299
#  Will display the picture "2299" when 0795551234 is dialed or incoming
# P972XXXXXXX,P469XXXXXXX,P214XXXXXXX=default,"DALLAS, TX"|187|center|blue|10
#  Will display the picture "default" with a centered blue label "Dallas, TX"
font size 10 at vertical position 187 when 10 digits numbers starting with 972
or 469 or 214 are dialed or incoming
# P44.=UK
#  Will display the picture "UK" when any number starting with 44 is dialed or
is incoming
#############################################################################
[Numbers]
*97,*98=voicemail
*76,*78,*79,*21,*65=phone
```

19.5.3 NUMBER MATCHING EXAMPLES

The number matching can be used to map external phone numbers (cell phone, home…) to a local extension.

John Doe is extension 2000 on the local PBX, ‘2000.jpg’ is available is the “pictures” directory.

The following line in the configuration file

4085551234,3105551234=2000

Will display the ‘2000.jpeg’ picture when ‘4085551234’ or ‘3105551234’ are calling or being called.

19.5.4 PATTERN MATCHING EXAMPLES

The pattern matching allows you to define hierarchical rules to match a number with a picture; the order in the configuration file is significant.

Example 1

P972XXXXXXX,P469XXXXXXX,P214XXXXXXX =default,"DALLAS, TX"|187|center|red|10

Will display the following picture on the phone

	When a 10 digit number starting with 972, 469 or 214 is calling or being called. .

Example 2

This example demonstrates the relevance of the lines order in the configuration files.

P972555XXXX=default,"ACME Dallas"|187|center|blue|10

P972XXXXXXX,P469XXXXXXX,P214XXXXXXX=default,"DALLAS, TX"|187|center|red|10

Will display:

	When a 10 digit number starting with 972555 is calling or being called.
	When a 10 digit number starting with 972 (except if it is starting with 972555), 469 or 214 is calling or being called.

Example 3

This example demonstrates how to use undefined length patterns.

P44.= default,"United Kingdom"|187|center|red|10

Will display the following picture

	When a number (any number of digits) starting with 44 is calling or being called.

19.6 FILES

This chapter describes the files provided in the package

File	Comment
./	Root directory
imageServer.php	PHP script for the Picture Caller ID application
License.txt	License file
pictureID.conf	Configuration file
./cache	Cache directory
./fonts	Font directory
DejaVuSans-Bold.ttf	True Type font used to customize the labels
./pictures	Picture directory
Aastra.gif	Sample graphic files
conference.png	
default.png	
phone.png	
time.png	
voicemail.png	
Various country flags (US, UK, DE…)	

20 LAST PAGE OF THE DOCUMENT

58015045-00

This page is intentionally left blank