mobydick 7.12 (de) Dokumentation

Author: pascom Netzwerktechnik GmbH & Co. KG
Date: 15.03.2017 09:30

Table of Contents

ThemaA compete list of all available FlexPanel widgets

See also

- FlexPanel erstellen

Inhalt dieser Seite:

1 Overview

Every administrator can create and assign flexpanel scripts to one or more users. These scripts are transferred to the users client application and invoked automatically or on demand.

1.1 Scripting

Look at the provided example script to learn about the necessary imports. The scripts programming language is groovy, which is a dialect of the java.

By calling OpPane pane = frame.getMainPane() you get the main panel. This panel will hold your widgets. Please take care that every widget has **unique name** in the script.

2 Existing widgets

2.1 UserItem

This widget represents one user from the roster. It shows the user's current state, phone state, user's full name and current status

Usage

```
// just give the users bare jid
def jflores = new UserItem("jflores@mobydick")
// Alternatively use the extension number.
def extension320 = new UserItem("320", true)

//This method will change appearance of the widget. It will show just name and phone state.
jflores.setAppearance(WidgetAppearance.Compact)
// this will set the default appearance
extension320.setAppearance(WidgetAppearance.Default)


// Default Sizes: Compact (w:200,h:30), DefaultAppearance (w:200,h:50);
```

2.2 MeItem

Show informations about the currently logged in user including presence, phone state, location.

Usage

```
def jflores = new MeItem()
```

2.3 PhonebookItem

This widget represents one user from the phonebook. It's not read from MobyDick database. You can put custom name and phone number in this widget

2.3.1 Available constructors

- PhonebookItem(name,phoneNumber) - *example PhonebookItem phonebookItem1 = new PhonebookItem("Stefan Tosic","123123");*

2.3.2 Available methods

- setAppearance(WidgetAppearance.Compact)
    - This method will change appearance of the widget. It will show just the name. - *example phonebookItem1 .setAppearance(WidgetAppearance.Compact)*
- setAppearance(WidgetAppearance.Default)
    - Widget appearance will be as in initial description - *example phonebookItem1 .setAppearance(WidgetAppearance.Default)*

2.3.3 Default size

Compact(w:200,h:30), Default(w:200,h:50)

2.4 TeamWidget

This widget represents one queue. It can show agents or waiting calls in the queue

Usage

```
// TeamWidget(queueName,numberOfItems,numberOfColumns,teamWidgetContent)
// Shows the team "Developer" with 5 call items in 1 column.
// It will be vertical and calls be not ordered by time
TeamWidget developers = new TeamWidget("Developers",5,1,TeamWidgetContent.Calls);


// TeamWidget(queueName,numberOfItems,numberOfColumns,timeOrdered,teamWidgetContent)
// Here you'll get the calls ordered by waiting time.
TeamWidget developers2 = new TeamWidget("Developers",5,1,true,TeamWidgetContent.Calls);

// TeamWidget(queueName,numberOfItems,numberOfColumns,widgetOrientation,teamWidgetContent)
// You can also have vertical orientation and logged in Agents instead of Calls
TeamWidget developers3 = new
TeamWidget("Developers",5,1,WidgetOrientation.Vertical,TeamWidgetContent.LoggedAgents);

// TeamWidget(queueName,numberOfItems,numberOfColumns,widgetOrientation,timeOrdered,teamWidgetContent)
// And finally vertical oriented and time ordered
TeamWidget developers4 = new
TeamWidget("Developers",5,1,WidgetOrientation.Vertical,true,TeamWidgetContent.Calls);

// you can modify the child items size: (for now only AgentItem supports Compact!)
developers3.setChildrenAppearance(WidgetAppearance.Compact)


// get back default size:
developers4.setChildrenAppearance(WidgetAppearance.Default)
```

2.4.1 Parameter description

- - queueName - queue name
  - numberOfItems - number of items (agents or waiting calls) visible on screen (slots). If number is less then number of waiting calls or agents, some items will be invisible
  - numberOfColumns - in how many columns/rows items will be arranged
  - widgetOrientation - orientation of widget, aligning of the items. Can be WidgetOrientation.Horizontal or WidgetOrientation.Vertical
  - timeOrdered - if team shows waiting calls this parameter determinates how waiting calls will be ordered. If true items will be ordered as they came in the queue. If false waiting call finds the first available slot
  - teamWidgetContent - what widget contains. Can be TeamWidgetContent.Calls - shows waiting calls, TeamWidgetContent.AllAgents - show all agents, TeamWidgetContent.LoggedAgents - show only logged agents in the queue

2.4.2 Available methods

- setAppearance(WidgetAppearance.Compact)
  - This method will change appearance of the widget. It will show just name and extension of the queue. - *example developers.setAppearance(WidgetAppearance.Compact)*
- setAppearance(WidgetAppearance.Default)
  - Widget appearance will be as in initial description - *example developers .setAppearance(WidgetAppearance.Default)*

2.4.3 Default size

Depends of number of elements:

- WidgetOrientation.Vertical
  Gap between elements is 5. Vertical padding in total 39(title+margin). Horizontal padding in total 32(margin).
- WidgetOrientation.Horizontal
  Gap between elements is 5. Vertical padding in total 29(title+margin). Horizontal padding in total 32(margin).

2.5 InboundQueueChannelItem

Subitem of the TeamWidget. Represents one waiting call. Can't be created out of the TeamWidget

2.5.1 Default size

(w:200,h:50)

2.6 AgentItem

Subitem of the TeamWidget. Represents one agent in the queue. Can't be created out of the TeamWidget

2.6.1 Default size

Compact (w:200,h:30), DefaultAppearance (w:200,h:50);

2.7 InboundChannelItem

This widget represents a incoming call to my phone. If the logged in user does't have incoming call this widget will show no data. When a call comes in, the widget will show the number/name and the color of the border reflects the call state (ringing, busy...)

Usage

```
def inboundChannel = new InboundChannelItem()

// to make it a bit more visible you can do:
inboundChannel.setNoticeable(true, true) // first parameter is "setBiggerFont", second "showAnimation" to
let the widget shake in ringing state
// Default Size: (w:200,h:70)
```

2.8 HangupWidget

This widget is used to hangup the incoming call. It just shows the icon. User need to drop the chosen call to this widget and this call will be hanged up

2.8.1 Available constructors

- HangupWidget() - *example HangupWidget hangupWidget = new HangupWidget();*

2.8.2 Default size

(w:50,h:50)

2.9 HoldWidget

This widget is used to hold the call. User need to drop the chosen call to this widget and this call will be holden

2.9.1 Available constructors

- HoldWidget() - *example HoldWidget holdWidget = new HoldWidget();*

2.9.2 Default size

(w:200,h:70)

2.10 OffhookWidget

This widget is used to answer the call. It just shows the icon. User need to drop the chosen call to this widget and this call will be answered

2.10.1 Available constructors

- OffhookWidget() - *example OffhookWidget offhookWidget = new OffhookWidget();*

2.10.2 Default size

(w:50,h:50)

2.11 SearchWidget

This widget is used to search for some phonebook entry in mobydick database. It just shows the icon. Depending on the trigger action widget can perform multiple actions

2.11.1 Available constructors

- SearchWidget() - *example SearchWidget searchWidget = new SearchWidget();*

2.11.2 Default size

(w:50,h:50)

2.12 ExternalUrlItem

This widget is used to forward call information to some url. It appends the call parameters to the URL and opens it in the defualt browser.

2.12.1 Available constructors

- ExternalUrlItem(name,url) - *example ExternalUrlItem externalUrlItem = new ExternalUrlItem("OTRS"," www.otrs.com");*

2.12.2 Default size

(w:200,h:50)

2.13 WebViewWidget

This widget is used to show some content in internal web browser (webview).

2.13.1 Available constructors

- WebViewWidget(url,refreshInterval,width,height) - *example WebViewWidget webViewWidget = new WebViewWidget("www.google.com",5,300,300);*
- WebViewWidget(url,refreshInterval) - *example WebViewWidget webViewWidget = new WebViewWidget("www.google.com",5);*
    - *creates widget with predefined size*

2.13.2 Parameter description:

-
    - refreshInterval - if 0 webViewWidget will not refresh, if >0 then widget will refresh content each refreshInterval seconds

2.13.3 Default size

(w:200,h:150)

2.14 StatusWidget

This widget shows the failure messages received from the server. User can track what happens with sent commands

2.14.1 Available constructors

- StatusWidget() - *example StatusWidget statusWidget = new StatusWidget();*

2.14.2 Default size

(w:200,h:55)

2.15 StatisticWidget

This widget shows the counters of one team. It shows number of busy,ringing,paused, free and not active agents in the queue

2.15.1 Available constructors

- StatisticWidget(queueName,showInactive,showNumbers,width,height) - *example StatisticWidget statisticWidget = new StatisticWidget("Developers",true,true,300,400);*
  - *Creates the statistic widget with given dimensions. showInactive can be true or false, if true inactive counters will be included otherwise not. showNumbers can be true or false, if true the numbers of the agents will be shown in the label*
- StatisticWidget(queueName,showInactive) - *example StatisticWidget statisticWidget = new StatisticWidget(" Developers",true);*
  - Creates the statistic widget with default size 300x300 and does't show numbers in legend

2.15.2 Default size

(w:200,h:150)

2.16 WaitingCounterWidget

This widget shows the number of waiting callers in the queue. It can show the number in different styles when given conditions are satisfied

2.16.1 Available constructors

- WaitingCounterWidget(String queueName) - *example WaitingCounterWidget waitingCounterWidget = new WaitingCounterWidget("Developers");*
- WaitingCounterWidget(String queueName, String title) - *example WaitingCounterWidget waitingCounterWidget = new WaitingCounterWidget("Developers","pascom Developers");*
- WaitingCounterWidget(String queueName, String title, int width, int height) - *example WaitingCounterWidget waitingCounterWidget = new WaitingCounterWidget("Developers","pascom Developers",250,250);*
- WaitingCounterWidget(String queueName, String title, int width, int height, int firstLevelAlert) - *example WaitingCounterWidget waitingCounterWidget = new WaitingCounterWidget("Developers","pascom Developers",250,250,3);*
- WaitingCounterWidget(String queueName, String title, int width, int height, int firstLevelAlert, int secondLevelAlert) - *example WaitingCounterWidget waitingCounterWidget = new WaitingCounterWidget("Developers","pascom Developers",250,250,2,4);*

2.16.2 Parameter description

- queueName - The name of the queue to monitor
- title - Title of the widget
- width - Custom width of the widget
- height - Custom height of the widget
- firstLevelAlert - If number of waiters in the queue is equals or higher then this number but lower then second level, widget will apply first level alert style
- secondLevelAlert - If number of waiters in the queue is equals or higher then this number, widget will apply second level alert style

2.16.3 Default size

(w:150,h:150)

2.17 VariableToggleWidget

Show and Control a Asterisk Database value with a push button. Pressed and unpressed states are mapped to onValue and offValue.

The widget is limited to AstDB Keys with a prefix of "API/" and will only work for xmpp.supervisor users.

2.17.1 Available constructors

- VariableToggleWidget(String label, String key, String offValue, String onValue) - *example VariableToggleWidget("Longer waiting time", "/API/WAITING/TIME", "0", "1");*
- VariableToggleWidget(String label, String key, String offValue, String onValue, int width, int height) - *example VariableToggleWidget("Longer waiting time", "/API/WAITING/TIME", "0", "1",180,40);*

2.17.2 Parameter description

- label - The caption for the button.
- key - The to-be-controlled/monitored Asterisk Database Key
- offValue - Value for "Button is not pressed" (i.E. "0")
- onValue - Value for "Button is pressed" (i.E. "1")
- width - Custom width of the widget
- height - Custom height of the widget

2.17.3 Default size

(w:150,h:30)

2.18 VariableWidget

Show and Control a Asterisk Database value with a slider.

The widget is limited to AstDB Keys with a prefix of "API/" and will only work for xmpp.supervisor users.

2.18.1 Available constructors

- VariableWidget(String label, String key, int minVariableValue, int maxVariableValue) - *example VariableWidget("Queue Priority", "/API/SUPPORT/PRIORITY", 0, 4);*

- VariableWidget(String label, String key, int minVariableValue, int maxVariableValue, int width, int height) - *example VariableWidget("Queue Priority", "/API/SUPPORT/PRIORITY", 0, 4, 300,100);*

2.18.2 Parameter description

- label - The widgets caption.
- key - The to-be-controlled/monitored Asterisk Database Key
- minVariableValue - The sliders "low" value (i.E. 0)
- maxVariableValue - The sliders "high" value (i.E. 10)
- width - Custom width of the widget
- height - Custom height of the widget

2.18.3 Default size

(w:200,h:100)

2.19 ImageWidget

Load a image from a local file or remote URL.

2.19.1 Available constructors

- ImageWidget(int height, int width, boolean preserveRatio, String urlOrLocalPath, boolean isURL) - *example ImageWidget(300,300,true,"http://myserver/mylogo.png",true);*

2.19.2 Parameter description

- width - Custom width of the widget
- height - Custom height of the widget
- preserveRatio: if false then force width and high, otherwise preserve the images aspect ratio.
- urlOrLocalPath: can contain either a url or a local path to the to be loaded image.
- isURL: if true: interpret urlOrLocalPath as url, if false: interpret urlOrLocalPath as local Path