Table of Contents

Extending the Menu 1
Useful Methods 1
Access With ID 2

The standard menu creates and configures the menu and toolbar automatically. It adds default menus like *File*, *Edit*, *Help* and default menu items like *Login/Logout*, *Exit*, *About*. All application workscreens will be added by the application to the menu. Every workscreen has its own menu item and/or toolbar button.

The menu forwards most actions to the application because the application is responsible for the security and workscreen management.

If you don't like the standard menu handling or layout, it's very easy to replace the standard menu with your own implementation.

Extending the Menu

The menu class defines default methods that are used by the application, so it's important to extend the class. The class has a constructor with the application as parameter. The constructor delegates the configuration to the method.

```
protected void configureApplication()
```

Override this method if your application should get a custom style. The default implementation sets an UIBorderLayout for the application pane and adds the content pane to the application pane with CENTER constraint.

It's possible to add the content pane to your hierarchy (e.g., a tabset) and use a different layout for the application pane. There aren't any limits.

The menu class will be created by ProjX via

```
protected Menu createMenu() throws Throwable
```

This method checks the menu class parameter (see Customize an Application) and creates a default menu instance if no user-defined class was set.

Useful Methods

The menu has some relevant methods:

```
public void createStandardMenu()
public void addItem(String pId, String pAction, ...)
```

The createStandardMenu method will be called by the application after menu creation. It recreates the whole menu and adds only standard items. It doesn't add any workscreens. The standard menu contains File, Edit and About menus and toolbar buttons. The standard toolbar contains Exit, Login/Logout and Save, Reload (if connected).

The method delegates the creation of the menubar to the method:

```
protected void createStandardMenuBar()
```

and toolbar creation to

```
protected void createStandardToolBar()
```

Simply override createStandardToolBar and don't call the parent method to hide the toolbar.
The workscreen items will be added by the application after successful login. The method:

```
public void addItem(String pId, ...)
```

will be called for every workscreen. It delegates item creation to

```
protected IMenuItem addMenuItem(String pId, String pGroupId, ...)
```

and

```
protected IButton addToolBarButton(String pId, ...)
```

But there are more useful methods for you. All methods that start with **create**, are helper methods, e.g.,

```
public UIMenuItem createMenuItem(String pAction, ...)
public UIButton createToolBarButton(String pAction, ...)
public UIToggleButton createToolBarToggleButton(String pAction, ...)
```

and there are also some **configure** methods:

```
protected <T extends AbstractUIMenuItem<?>> T configureMenuItem(...)
protected <T extends AbstractUIButton<?>> T configureToolBarButton
```

Sometimes it's enough to override the configure methods instead of create methods because every create method calls a configure method.

Access With ID

Every item should have an ID because the item access is managed with IDs. There are a lot of default IDs, e.g., *Menu.FILE*, *Menu.FILE_LOGIN*.

If you need access to an item, simply use

```
public IComponent[] get(String pId)
```

The method returns all mapped items for the given ID. Sometimes an ID maps multiple items because one item is the menu item and another one is the toolbar button.

If you add custom items to the menu, simply map them with

```
public IComponent[] put(String pId, IComponent... pComponent)
```

The menu class has methods for item control:

```
public void removeItem(String pId)
public void setItemVisible(String pId, boolean pVisible)
public void setItemEnabled(String pId, boolean pEnable)
public boolean isSelected(String pId)
```

From:
https://doc.sibvisions.com/ - **Documentation**

Permanent link:
https://doc.sibvisions.com/applications/replace_menu

Last update: **2020/07/03 12:44**