Table of Contents

Sometimes you need additional methods in your UI factory, e.g., create custom controls or preconfigure existing controls.

The UI factory is one of the first classes that is instantiated from a launcher. If you need a custom factory, use an application parameter to configure it.

A custom factory:

CustomSwingFactory.java

```
package apps.firstapp;

import jvx.rad.genui.UIImage;
import jvx.rad.ui.IAlignmentConstants;
import jvx.rad.ui.component.IButton;

import com.sibvisions.rad.ui.swing.impl.SwingFactory;

public class CustomSwingFactory extends SwingFactory
{
   public IButton createButton()
   {
      IButton button = super.createButton();

      button.setBackground(null);
      button.setHorizontalAlignment(IAlignmentConstants.ALIGN_LEFT);
      button.setImage(UIImage.getImage(UIImage.OK_SMALL));
      button.setBorderOnMouseEntered(true);

      return button;
   }
}
```

Our factory extends the default SwingFactory and overwrites createButton. All created buttons have a default icon, no border, and the text is left aligned.

Now we configure our custom factory via command-line:

```
java -cp... apps.firstapp.FirstApplication ""
Launcher.uifactory=apps.firstapp.CustomSwingFactory
```

The second parameter defines a config file, but we don't use one. It is also possible to put the factory parameter in a config file together with other parameters. All configuration options are described in this article.

A preview of an application with our custom factory: