# JFXPanel Bean

Servoy 7.2 offers integration with JavaFX in the Smart Client, a new UI Toolkit for the Java platform.

The integration comes in the form of a low level JFXPanel bean, to which JavaFX UI components can be added through code. The API of the bean is inherited directly from JavaFX's JFXPanel class. A convenient .isJavaFXAvailable() method is added to check whether JavaFX is available or not.

Detailed information on the integration of JavaFX through the JFXPanel into Java Swing applications (which the Servoy Smart Client is), can be found in the JavaFX for Swing Developers tutorial of Oracle.

The most important thing to take into account when integrating JavaFX is that all interactions with the JavaFX components need to take place on the JavaFX User thread, while all interactions with the Servoy scripting layer need to place on Swing's Event Dispatch Thread (EDT). This means that working with JavaFX in Servoy solutions requires knowledge of Java and requires inline Java code.

**Requirements**

- JavaFX is available on Windows, OSX and Linux as of Java 7 update 6, thus in order to use JavaFX the Smart Client or Servoy Developer (see notes) must be launched with Java 7 update 6 or higher.
- In order to enable JavaFX in a Smart Client running from a Servoy Application Server, the setting *servoy.client.javafx* on the Servoy Admin page must be set to true (Default is false)

**Notes**

- The JFXPanel bean is Smart Client only, as JavaFX is a Java UI toolkit, thus it has no place in the Web Client or Mobile Client. In the Web Client the bean will render an empty panel
- Running Servoy Developer on Java 7 on OSX is not as straight forward as it should be. See Running Servoy Developer on Java 7 on MAC OSX for more details

## Simple sample

A small "Hello World" example using the bean:

**Hello World example**

```
if (elements.myfxpanel.isJavaFXAvailable()) {
   var jsRunnable = {
          run: function () {
                 var text = new Packages.javafx.scene.text.Text("Hello World");
                 text.setFont(new Packages.javafx.scene.text.Font(24));
                 var pane = new Packages.javafx.scene.layout.BorderPane(); pane.setCenter(text);
                 var scene = new Packages.javafx.scene.Scene(pane);
                 elements.myfxpanel.setScene(scene);
          }
   }
   var runnable = new Packages.java.lang.Runnable(jsRunnable);
   Packages.com.sun.javafx.application.PlatformImpl.runLater(runnable);
}
```