Using the PDF Viewer Component

1 Added by Ken Fyten, last edited by Patrick Corless on Mar 25, 2013

Table of Contents

Search

expand all collapse all

The ICEpdf library can also be used to create a full-featured PDF Viewer component which can be inserted into any Java application. For more information on the viewer component features, see Reference Implementations.

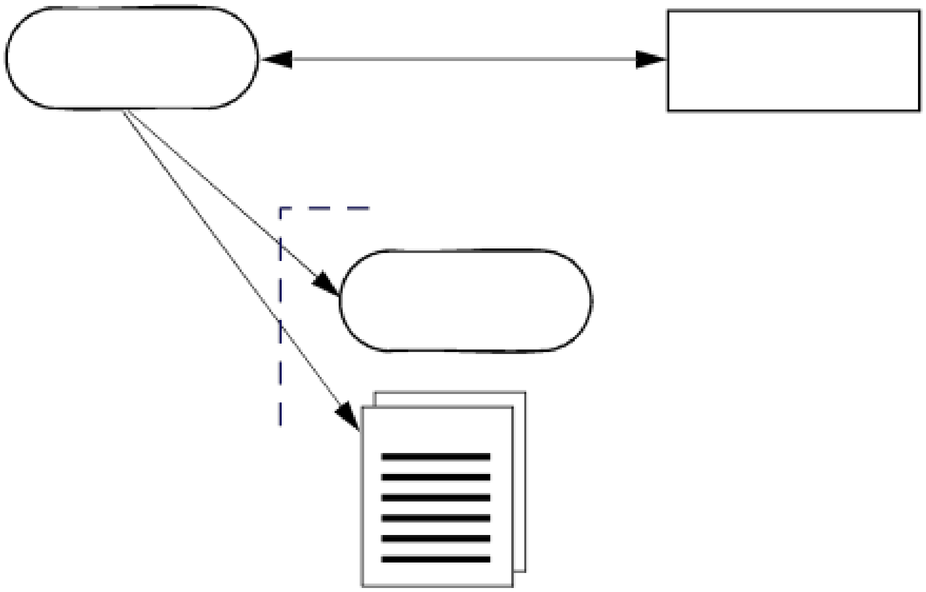
The PDF Viewer application is a reference implementation (RI) application, meaning that all source code used to implement the application is available to developers to modify as required.

The PDF Viewer RI uses the Model-View-Controller (MVC) design pattern for communication between the user, the GUI and the PDF document data. The PDF Viewer's data model is implemented by the ViewModel class. The view, which presents the user interface, is implemented using standard Java Swing components and is constructed by the SwingViewBuilder class. The controller, which interacts between the user, view and data model is represented by the SwingController class.

This relationship can be seen in Figure 1. The combination of the MVC design and the SwingViewBuilder and SwingController classes provides a very powerful and easily adaptable approach to PDF Viewer GUI development. Developers using ICEpdf can readily customize the Viewer user-interface with a very shallow learning curve and minimal coding effort.

Figure 1 - ICEpdf MVC Implementation

SwingController - SwingView Factory



ViewModel
Document

Creating a Viewer Component

The **`org.icepdf.core.ri.common.SwingController`** class provides convenience methods for the most common UI actions, such as rotating the document, setting the zoom level, etc. The

`org.icepdf.core.ri.common.SwingViewBuilder` class is responsible for creating the PDF Viewer component panel populated with Swing components configured to work with the SwingController.

When using the SwingViewBuilder and SwingController classes, it is usually not necessary to use the Document object directly. The SwingController class does this for us.

The following code snippet illustrates how to build a PDF Viewer component:

```
String filePath = "somefilepath/myfile.pdf";

// build a controller
SwingController controller = new SwingController();

// Build a SwingViewFactory configured with the cont
SwingViewBuilder factory = new SwingViewBuilder(cont

// Use the factory to build a JPanel that is pre-con
//with a complete, active Viewer UI.
JPanel viewerComponentPanel = factory.buildViewerPan

// add copy keyboard command
ComponentKeyBinding.install(controller, viewerCompon

// add interactive mouse link annotation support via
controller.getDocumentViewController().setAnnotation
      new org.icepdf.ri.common.MyAnnotationCallback(
             controller.getDocumentViewController())

// Create a JFrame to display the panel in
JFrame window = new JFrame("Using the Viewer Compone
window.getContentPane().add(viewerComponentPanel);
window.pack();
window.setVisible(true);

// Open a PDF document to view
controller.openDocument(filePath);
```

Note
The SwingViewBuilder class provides numerous methods that enable developers to quickly create custom viewer user interfaces (UIs) by including only those UI controls that are required, customizing existing controls, etc.

- Refer to **`org.icepdf.core.ri.common.SwingViewBuilc`** in the JavaDoc API documentation and Customizing the SwingViewBuilder for more information.

- See ICEpdf Viewer Application (RI) for a complete example.