Tools

This chapter covers the following topics:

- Development Workplace versus Eclipse Plug-in
- Important Tools
- Java Development Environment

Development Workplace versus Eclipse Plug-in

- **Development Workplace**
  The development workplace is a browser-based development environment. In the workplace, you can create, edit and run layouts using the different tools which help you maintain these layout definitions. From within the workplace, you can also access demo applications and documentation. For more information, see the *Development Workplace* documentation.

- **Eclipse Plug-in**
  For the development of layouts in an Eclipse environment, the Eclipse plug-in Ajax Developer is available. It supports the major features from the development workplace in an Eclipse environment. For more information, see the *Ajax Developer* documentation.

Important Tools

The most important tools that are used for the development tasks are briefly described below.

- **Layout Painter**
  The Layout Painter is an Application Designer server application, which means that the editor runs in the browser or Eclipse and the application logic runs inside the Application Designer server. In the Layout Painter, you specify the layout of your pages using an XML tree. An active WYSIWYG preview is provided in which you can test your applications.

  Detailed information is provided in:

  - *Layout Painter* in the *Development Workplace* documentation.
  - *Layout Painter* in the *Ajax Developer* documentation.

- **Layout Manager**
  While the Layout Painter allows you to edit one layout definition, the Layout Manager allows you to manage all layout definitions of a project. Mass operations like regeneration of all intelligent HTML pages of a selected project can be done with this tool.

  For detailed information, see *Layout Manager* in the *Development Workplace* documentation.

- **Code Assistant**
  The Code Assistant - like the Layout Painter - is an Application Designer server application and runs inside the browser or Eclipse. You use its code generator to speed up development of Java adapter classes. All set/get methods of referenced properties can be generated easily using this tool. The

Code Assistant is integrated in the Layout Painter.

Detailed information is provided in:

- *Using the Code Assistant* in the *Development Workplace* documentation.
- *Using the Code Assistant* in the *Ajax Developer* documentation.

- **Literal Assistant**
  Optionally, you can define all literals inside a page layout - the "text IDs". The text for a text ID is derived from the multi language management at runtime. The Literal Assistant is a tool for editing and translating these literals. It is integrated in the Layout Painter.

  Detailed information is provided in:

  - *Using the Literal Assistant* in the *Development Workplace* documentation.
  - *Using the Literal Assistant* in the *Ajax Developer* documentation.

- **Style Sheet Editor**
  All controls that are contained inside the Application Designer control library are rendered using a style sheet. Application Designer delivers a variety of predefined styles but also allows you to generate custom styles sheets. To simplify the creation of custom style sheets, the Style Sheet Editor is available. With this tool, you can both define the very basic style elements (main colors to be used) as well as change the style definitions of the controls on the lowest level.

  Detailed information is provided in:

  - *Style Sheet Editor* in the *Development Workplace* documentation.
  - *Style Sheet Editor* in the *Ajax Developer* documentation.

- **WAR Packager**
  To deliver your web applications built with Application Designer, you need to create a web archive (*.war* file). This file is either created by tools you might select from your development environment or by the WAR Packager: You can create the *.war* file automatically, using an easy-to-use graphical user interface.

  For detailed information, see *WAR Packager* in the *Development Workplace* documentation.

Java Development Environment

You can use a standard Java development environment to develop the adapter classes for your layouts. The Ajax Developer plug-in provides extended support for Eclipse (open source), which is one of the most popular Java IDEs. For detailed information, see the *Ajax Developer* documentation.

Alternatively, you can also use different Java development environments such NetBeans (open source) and JBuilder. However, Application Designer does not provide any enhanced support for these environments as it does for Eclipse.

Depending on the functional capabilities, you can also use the development environment for remote debugging. If your development environment does not provide a remote debugger, you can use other debugging tools such as the JSwat debugger. See *Appendix F - Using JSwat for Debugging*.