Getting the Most from Lasso Studio for Eclipse

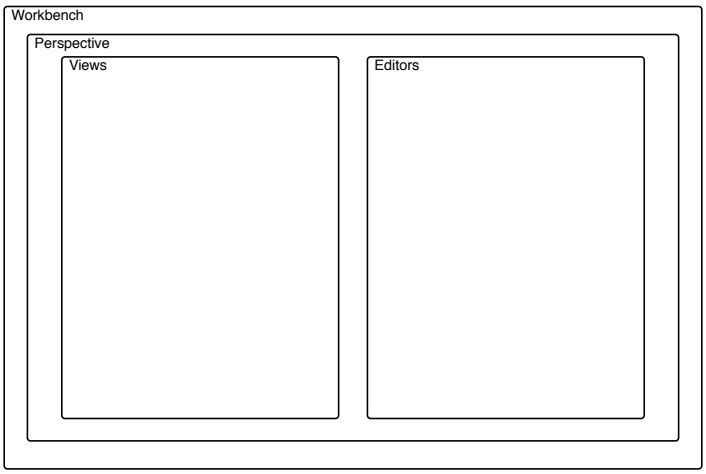
Kyle Jessup & Tom Wiebe

Using the Eclipse IDE

Understanding Eclipse

The Workbench

"The term Workbench refers to the desktop development environment. The Workbench aims to achieve seamless tool integration and controlled openness by providing a common paradigm for the creation, management, and navigation of workspace resources.
Each Workbench window contains one or more perspectives. Perspectives contain views and editors and control what appears in certain menus and tool bars. More than one Workbench window can exist on the desktop at any given time."



Perspectives

"Each Workbench window contains one or more perspectives. A perspective defines the initial set and layout of views in the Workbench window. Within the window, each perspective shares the same set of editors. Each perspective provides a set of functionality aimed at accomplishing a specific type of task or works with specific types of resources. For example, the Java perspective combines views that you would commonly use while editing Java source files, while the Debug perspective contains the views that you would use while debugging Java programs. As you work in the Workbench, you will probably switch perspectives frequently.
Perspectives control what appears in certain menus and toolbars. They define visible action sets, which you can change to customize a perspective. You can save a perspective that you build in this manner, making your own custom perspective that you can open again later.
You can set your General preferences to open perspectives in the same window or in a new window."

- Lasso Studio for Eclipse provides the Lasso Perspective
- Open perspectives from the "Window > Open Perspective..." menu.

View

"Views support editors and provide alternative presentations as well as ways to navigate the information in your Workbench. For example, the Navigator and other navigation views display projects and other resources that you are working with.
Views also have their own menus. To open the menu for a view, click the icon at the left end of the view's title bar. Some views also have their own toolbars. The actions represented by buttons on view toolbars only affect the items within that view.
A view might appear by itself, or stacked with other views in a tabbed notebook. You can change the layout of a perspective by opening and closing views and by docking them in different positions in the Workbench window."

- Also, Fast Views, hidden pop-up views. Handy for seldom used Views.
- Lasso Studio for Eclipse provides the Lasso Script HTML Result View.
- Views may be associated with a particular perspective by default but, can generally be opened in any perspective.

Editor

- Most perspectives consist of an editor and one or more views
- Editors can be associated with a particular File type, if there is no associated editor, Eclipse uses the OS to launch the file in it's default system editor (i.e. bbedit for CSS files)
- The left margin contains error flags, bookmarks, breakpoints for debugging or todo's

Project

Repeat 3 times:
Eclipse is not just a text editor
Eclipse is not just a text editor
Eclipse is not just a text editor

- All your work files must be contained within a project. A lasso project in the case of Lasso Projects.
- The project defined build settings (i.e. integration with Lasso Developer)
- You can open and close projects to save system resources and 'Go into' projects to focus on only the task at hand in the navigator

Installation and Configuration

Thoroughly covered in the Lasso Studio for Eclipse manual.

The Lasso Perspective

Coding

Lasso Project

Associates the file with the Lasso Builder and enables syntax checking, running and debugging of lasso files.

Lasso Script File

Just what it sounds like. Default Lasso Script contents can be defined in the "Lasso Studio > Script File Templates" preference pane. Allows you to add copyright info, SCM keywords or boilerplate code

The Lassoscript Editor

Syntax Colouring

- Works for all 3 styles (Square Bracket, Lassoscript and Parenthesis)
- Even works on a mixture of styles in the same document
- Bracket/Parenthesis highlighting
- Highlights the following elements:

  Tags, Types, Constants
  Keywords (attributes)
  String Literals (quoted strings)
  Numeric Literals (integers and decimals)
  Operators (+,-, etc)
  Page Variables ($myVariable)
  Local Variables (#myLocal)
  Comments (Single line // comments or C-style /*...*/ comments

Code Folding

- Allows you to collapse and expand container tags within your document.
- Makes it easier to navigate complex pages, collapse all but the block of code you're working on.
- Makes it easy to ensure you've closed all container tags properly, your source file becomes an outline.

Code Completion

- Typing the first few characters of a tag or variable name presents a list of tags found in the Lasso Reference, or Ctags within your project
- Typing a $ or # will automatically bring up a list of variable name recommendations.
- At the top of the list will be Code Templates, discussed below
- 'Show Definition' (F3) will open include files or ctype/ctag definitions

Tool Tips

- Draws info from the Lasso Reference
- Also works for Custom Types/Tags using the -Description attribute
- Pressing F2 will present an expandable floating window containing the entire description. You can select and copy text within the description, very handy for copying/adapting example code into your existing page

Code Templates

- Tab between insert elements
- Able to show available datasources and tables in an inline

- Use to help remember the syntax for seldom used/complex tags or ctags
- Can be exported to xml format for distribution and sharing

Example Template File

```
<?xmlversion="1.0"encoding="UTF-8"?>
<templates>
<templateautoinsert="true"context="lasso"deleted="false"
description="Page Var"enabled="true"id="var1"name="var">
var('${name}'=${value})${cursor}</template>
</templates>
```

- Template text will be formatted upon insertion to the current formatting preferences

Caveat*: the current version of LSfE (1.5.1) cannot include $variable style elements in a template definition. i.e. inline($params,${database}... will not work, instead, use the long hand form of the variable inline(var('params'),${ database}...*

Code Formatting

- Keeps your code neat and tidy.
- Helps ensure consistent style across multiple developers.
- Lasso has 3 official syntax styles, known commonly as Classic (Square Brackets), Lassoscript (tag:-property) and Parenthesis tag(-property). The Code formatter can switch between these styles with the click of a button, allowing the developer to code in the style most convenient to them and deliver code in the style preferred by their employer/client/ what have you.
- Can set indentation preferences, Tab or up to 5 space characters.
- Text wrapping to avoid long lines.
- quote character preference, single or double quotes.
- Maximum plain text length - How long a string will Lasso include in a lassoscript element?
- Makes working with Version control systems easier as your code is more consistent.

Outline View

- Drag and drop code sections.
- Cut and paste code blocks directly in the outline.
- Useful for navigating your files, double click on any item to navigate to it/select it.
- insert tag/type templates via contextual menu

Custom Tag/Type Template

- Provides a simple interface to help create Tags and Types
- Easily enter tag name, namespace, descriptio, options, and parameters for your tag or type.
- Custom type template allows automatic creation of CallBack tags as well.
- Create member tags within a custom type via contextual click on the type name in the outline.

Automatic Syntax checking and error reporting

- Checks document syntax upon document save and marks errors with a red X in the editor margin along with a notation in the problems window.
- Errors will show for all open projects, close other projects to show only relevant errors.
- Errors can be filtered by severity, type or resource scope (All Projects, This Project, This File etc).

Bookmarks and Tasks

- Eclipse allows you to set bookmarks in your source files, to facilitate jumping to particular places in your project quickly and easily
- Tasks are useful to note features remaining to implement, code that needs cleaning up or leave notes for other developers
- Create either through right clicking the margin on the line you wish to add the Bookmark/ Task or by simply adding a comment prefixed with “MARK” for bookmarks or “TODO” for tasks
- Better to use the comment method, in case other developers might be editing the project outside of Eclipse. Also handy to keep as much info within your files as possible
- Bookmarks and Tasks for open projects all appear within their respective Views

Running Scripts

- Runs a script and prints output to the Lasso Script HTML Result view
- Speaks to Lasso via the SOAP protocol, can work with either a local or remote Lasso Developer installation.
- Includes files based on the current project definition.

Warning*: This actually runs the script so any database or file tags actions will be executed.*

Run Configuration

- Requires a valid Lasso username/password within the ‘Lasso Studio for Eclipse Users’ group within the current lasso site.
- Edit the Laso Server WSDL parameter to specify a server other than localhost. i.e. a remote server or a locally defined hostname to access a particular lasso site.
- You can define multiple Run configurations for a particular site, each running a specific script or, you can set a ‘Preferred’ configuration along with a default file. A preferred configuration will present you with a selection box to select the script you wish to run on each execution.
- Runtime errors will be reported in the Lasso Script HTML Result View.

Debugging Scripts

- Like running but, interactive. You can view the variables as you step through your file and edit them to test different outcomes
- Use Breakpoints to suspend processing of your file in the debugger
- Tip: break your code up into multiple lines, instead of having a lot of tags on one line of code. The debugger works line by line so, you can’t stop multiple times in the same code

block if it's all on one line, nor can you evaluate values from the middle of a line of code, just the output at the end of the line.

Triggers

- A debugging session can be triggered either directly within Eclipse or via the input of an external browser
- Simulate Request - The request is sent from within Eclipse itself, along with any extra headers configured within the run configuration. Best option if the script doesn't rely on external factors (i.e. form input, cookies, get requests, client type, etc)
- Wait for Web Browser - Awaits a request from an external browser. Allows execution of the script with all environment variables in place.
- Can set a default method in the debugging tab of the current run configuration

Stepping through your program

Resume (F8) - runs the current script to it's end or the next breakpoint.

Suspend - suspends processing of the current script

Terminate - Terminate execution of the current script permanently. ie. 'Abort'

Step Into (F5) - The next expression on the currently selected line is executed and suspends on the next executable expression. i.e. 'run next tag'

Step Over (F6) - Like Step Into but, will run any include or library tags in one step. i.e. 'step over includes'

Step Return (F7) - Allows execution of the current script until it's end. Execution halts when an include or library tag are encountered, after the include is loaded. i.e. 'run to next include'

Debugging Specific Views

- Debug - Control execution of the current script
- Breakpoints - Shows current breakpoints and allows you to turn them on or off, as well as delete them entirely
- Variables - shows the current variables and allows editing of them (via contextual menu. Can also show type names.

Advanced Techniques

Using with Lasso Sites

- If you configure a specific hostname on your machine and within the runtime and debug environments, Lasso will execute the script using the appropriate LassoSite. Likewise for LassoSites defined by path name.
- Remote Debugging
- Edit the Lasso Server WSDL parameter in the run configuration to execute the selected file on a remote Lasso Server
- Note, due to security and performance considerations, it is not wise to debug scripts on a live, production server.
- Integration with other Eclipse Plugins and Command line tools

- Eclipse provides a wide variety of tools for dealing with many different tasks, both open source and commercial. Among these are included:

  Database Editors
  XML Editors
  HTML, Javascript and CSS Editors
  Java development (built in)
  C/C++ development

- Varying levels of support for most scripting languages, i.e. Python, Perl, Ruby, PHP
- This is in and of itself likely one of the strongest features of the Eclipse platform, you can do most of your coding work within one tool, lowering support, upgrade and training costs.

  see http://eclipse-plugins.2y.net/eclipse/index.jsp for more Eclipse Plugins.
- Multiple Platform support — Eclipse Works the same on Mac, Windows and Linux. You can move freely between platforms without notable differences in workflow or features.
- Team Features — Eclipse features excellent built in support for CVS version control and add on tools for other SCM (Source Code Management) tools such as Perforce