Developers' Guide

Table of Contents

1. Developers' Guide	
1.1. Other Guides	
2. Using an IDE	
2.1. Developing using IntelliJ IDEA	
2.2. Developing using Eclipse	
3. Building Apache Isis	40
3.1. Git	
3.2. Installing Java	
3.3. Installing Maven	
3.4. Building all of Apache Isis	
3.5. Checking for Vulnerabilities	
3.6. Checking for use of internal JDK APIs	
4. AsciiDoc Documentation	
4.1. Where to find the Docs	
4.2. Naming Conventions	
4.3. Writing the docs	
4.4. Build and Review (using Maven)	
4.5. Instant Rebuild (using Ruby)	
4.6. Publish procedure	
5. Contributing	
5.1. Recommended Workflow (github)	
5.2. Alternative Workflow (JIRA patches)	
5.3. Setting up your fork/clone	
5.4. Commit messages	
5.5. Creating the patch file	
5.6. Sample Contribution Workflow	
5.7. If your pull request is accepted	
6. Appendix: Git Cookbook	
6.1. Modifying existing files	
6.2. Adding new files	
6.3. Deleting files	
6.4. Renaming or moving files	
6.5. Common Workflows	
6.6. Backing up a local branch	
6.7. Quick change: stashing changes	
6.8. Ignoring files	
6.9. More advanced use cases	
6.10. If you've accidentally worked on master branch	

	6.11. If you've forgotten to prefix your commits (but not pushed)	61
7.	Appendix: Asciidoc Templates	63
	7.1. Callouts	63
	7.2. TODO notes	63
	7.3. Xref to Guides	63
	7.4. Link to Isis Addons	78
	7.5. Source code	79
	7.6. Images	79
	7.7. YouTube (screencasts)	80
	7.8. Tables	80
	7.9. Misc	80
8.	Appendix: Project Lombok	82
	8.1. Future thoughts	82
9.	Appendix: AgileJ	83

Chapter 1. Developers' Guide

This developers' guide is for:

- programmers who want to just use Apache Isis to build applications, and want help setting up their development environment or to build their code from the command line (eg to execute within a continuous integration server such as Jenkins)
- programmers who want to contribute back patches (bug fixes, new features) either to the codebase or the framework's documentation
- committers of Apache Isis itself who want guidance on release process, publishing documents and other related procedures.

1.1. Other Guides

Apache Isis documentation is broken out into a number of user, reference and "supporting procedures" guides.

The user guides available are:

- Fundamentals
- Wicket viewer
- Restful Objects viewer
- Security
- Testing
- Beyond the Basics

The reference guides are:

- Annotations
- Domain Services
- Configuration Properties
- Classes, Methods and Schema
- Apache Isis Maven plugin

The remaining guides are:

- Developers' Guide (this guide)
- Committers' Guide (release procedures and related practices)

Chapter 2. Using an IDE

The vast majority of Java developers use an IDE to assist with developing their code, and we highly recommend that you do like wise as you develop your Apache Isis applications using an IDE. Apache Isis is built with Maven, and all modern IDEs can import Maven projects.

This chapter shows how to setup and use two of the most popular IDEs, IntelliJ IDEA and Eclipse.

2.1. Developing using IntelliJ IDEA



This material does not constitute an endorsement; JetBrains is not affiliated to Apache Software Foundation in any way. JetBrains does however provide complimentary copies of the IntelliJ IDE to Apache committers.

This section describes how to install and setup JetBrains' IntelliJ IDEA, then how to import an application into IntelliJ and run it.

2.1.1. Installing and Setting up

This section covers installation and setup. These notes/screenshots were prepared using IntelliJ Community Edition 14.1.x, but are believed to be compatible with more recent versions/other editions of the IDE.

Download and Install

Download latest version of IntelliJ Community Edition, and install:

Start the wizard, click through the welcome page:



Figure 1. IntelliJ Installation Wizard - Welcome page

Choose the location to install the IDE:

💷 IntelliJ II	DEA Community Edition Setup 🛛 🗕 🗖 🗙
זו	Choose Install Location Choose the folder in which to install IntelliJ IDEA Community Edition.
Setup will install Intel different folder, click	liJ IDEA Community Edition in the following folder. To install in a Browse and select another folder. Click Next to continue.
Destination Folder	tBrains\IntelliJ IDEA Community Edition 14.1.1 Browse
Space required: 553. Space available: 30.6	6MB 5GB
	< Back Next > Cancel

Figure 2. IntelliJ Installation Wizard - Choose Location

Adjust any installation options as you prefer:

I	IntelliJ IDEA Community Edition Setup 🛛 🗕 💌
키	Installation Options Configure your IntelliJ IDEA Community Edition installation
	eate Desktop shortcut
Creat	te associations ava
	< <u>Back</u> <u>Next</u> > Cancel

Figure 3. IntelliJ Installation Wizard - Installation Options

and the start menu:

)	IntelliJ IDEA Community Edition Setup 🛛 🗖	×
키	Choose Start Menu Folder Choose a Start Menu folder for the IntelliJ IDEA Communi Edition shortcuts.	ty
Select t can als	the Start Menu folder in which you would like to create the program's shortcuts. so enter a name to create a new folder.	You
7-Zip Access Admin Agent Apach Atlassi Beyond Chocol Chrom Dolby Dropbo	ssibility sories nistrative Tools : Ransack (64-bit) ne Tomcat 7.0 Tomcat7 sian d Compare 3 JohateyGUI me Apps	~
	< Back Institut Car	icel

Figure 4. IntelliJ Installation Wizard - Start Menu Folder

and finish up the wizard:

Intellij IDEA C	ommunity Edition Setup 🛛 🗕 🔍	
	Completing the IntelliJ IDEA Community Edition Setup Wizard IntelliJ IDEA Community Edition has been installed on your computer. Click Finish to close this wizard.	
< <u>B</u> ack <u>F</u> inish Cancel		

Figure 5. IntelliJ Installation Wizard - Completing the Wizard

Later on we'll specify the Apache Isis/ASF code style settings, so for now select I do not want to import settings:



Figure 6. IntelliJ Installation Wizard - Import Settings

Finally, if you are young and trendy, set the UI theme to Darcula:

Si Custom	ize IDEA ×
UI Themes → Default plugins → Featured plugins Set UI theme import java.swing.*; public class HelloWorld { public HelloWorld { public HelloWorld { public HelloWorld() { JFrame frame = new JFrame ("Hello wor JLabel label = new JLabel(); label.setFont (new Font ("Serif", Font label frame frame	Darcula Darcula HelloWorld.java × import javax.swing.*; import java.awt.*; public class HelloWorld { public HelloWorld() { JFrame frame = new JFrame("Hello wor JLabel label = new JLabel(); label.setFont(new Font("Serif", Font label. frame. frame. frame. frame. frame. frame. } public ste }
UI theme can be changed later in Settings Appearance & Behavi Skip All and Set Defaults	or Appearance Next: Default plugins

Figure 7. IntelliJ Installation Wizard Set UI Theme

New Project

In IntelliJ a project can contain multiple modules; these need not be physically located together. (If you are previously an Eclipse user, you can think of it as similar to an Eclipse workspace).

Start off by creating a new project:

J	Welcome to IntelliJ IDEA 🛛 🗕 🗖 🗙
	J
	IntelliJ IDEA Version 14.1.1
	★ Create New Project 🖓
	I Import Project
	D Open
	Check out from Version Control +
	🏶 Configure 🗸 Get Help 🗸

Figure 8. IntelliJ Create New Project

We want to create a new **Java** project:

J I	New Project	×
Lava	Project <u>S</u> DK: <none></none>	Ne <u>w</u>
	Additional Libraries and Frameworks:	Set up Project SDK
Li Java FX	G Groover	🕆 JDK
IntelliJ Platform Plugin	Globby	👆 坑 IntelliJ Platform Plugin SDk
<i>m</i> Maven		🖷 Android SDK
Gradle		
© Groovy		
🗐 Griffon		
ta Empty Project	Use library: [No library selected]	Create
	Error: library is not specified Previous Mext Cancel	Help

Figure 9. IntelliJ Create New Project - Create a Java project

We therefore need to specify the JDK. Apache Isis supports both Java 7 and Java 8.



Figure 10. IntelliJ Create New Java Project - Select the JDK

Specify the directory containing the JDK:



Figure 11. IntelliJ Create New Project - Select the JDK location

Finally allow IntelliJ to create the directory for the new project:



Figure 12. IntelliJ Create New Project

File templates

Next we recommend you import a set of standard file templates. These are used to create new classes or supporting files:



Figure 13. File templates

The file templates are provided as a settings JAR file, namely **isis-settings-file-templates.jar**. Download this file and import using File > Import Settings, specifying the directory that you have downloaded the file to:



Figure 14. IntelliJ Import Settings - Specify JAR file

Select all the categories (there should just be one), and hit OK. then hit restart:

D I	Restart Needed ×
?	Settings imported successfully. You have to restart IDEA to reload the settings. Restart IntelliJ IDEA?
	OK

Figure 15. IntelliJ Import Settings - Restart

Live templates

We also recommend you import a set of live templates. These are used to add new methods to existing classes:

public	class	Customer0rder	implements	Comparable <customerorder></customerorder>	{
--------	-------	---------------	------------	--	---

is	
isc-jdo-1m-b-fk	Apache Isis Collection (JDO, 1:m parent bidirectional to foreign key) 💡
isobj-events	Apache Isis Object: abstract domain events declarations
isp-jdo	Apache Isis Property (JDO)
isa	Apache Isis Action
isa-dis	Apache Isis Action disablement
isa-event	Apache Isis Action (with domain event)
isa-event-decl	Apache Isis Action (domain event declaration only)
isa-hid	Apache Isis Action visibility
isa-p-auto	Apache Isis Action parameter auto-complete
isa-p-cho	Apache Isis Action parameter choices
isa-p-def	Apache Isis Action parameter defaults
isa-p-val	Apache Isis Action parameter validation
isa-val	Apache Isis Action validation
isc-dis	Apache Isis Collection disablement
isc-event-decl	Apache Isis Collection (domain event declaration only)
isc-hid	Apache Isis Collection visibility
isc-jdo-1m-b-jt	Apache Isis Collection (JDO, 1:m parent bidirectional to join table)
isc-jdo-1m-u-fk	Apache Isis JDO Collection (1:m parent unidirectional)
isc-jdo-1m-u-jt	Apache Isis Collection (JDO, 1:m parent unidirectional to join table)
isc-jdo-mn-ub-c	Apache Isis Collection (JDO, m:n child)
Ctrl+Down and Ctrl+Up will move of	caret down and up in the editor >> π

Figure 16. Live templates

The live templates have a prefix of prefixed either:

- is : for Apache Isis domain objects
- ju : for JUnit tests
- jm: for JMock mocks or libraries

• ad : for Asciidoc documentation; a full list can be found in the appendix.

The live templates are also provided as a settings JAR file, namely **isis-settings-live-templates.jar**. Download and import (as for the previous settings JAR files).

Coding Standards

Next, we suggest you recommend you import settings for standard ASF/Apache Isis coding conventions. This file is also provided as a settings file, namely **isis-settings-code-style.jar**. Download and import (as for the above settings JAR files).

Other Settings (Compiler)

There are also some other settings that influence the compiler. We highly recommend you set these.

On the **Compiler** Settings page, ensure that build automatically is enabled (and optionally compile independent modules in parallel):

J		Settings			X
	Build, Execution, D	Build, Execution, Deployment > Compiler @ For current project			
 Appearance & Behavior Keymap Editor Plugins Version Control Build, Execution, Deployment Build Tools Maven Gradle Gradle Gant Excludes Java Compiler Annotation Processors Validation RMI Compiler Groovy Compiler 	Resource patterns: !?*,java;!?*,form;!?*,class;!?*,groovy;!?*,scala;!?*,flex;!?*,kt;!?*,clj;!?*,aj Use ; to separate patterns and ! to negate a pattern. Accepted wildcards: ? — exactly one symbol; * — more symbols; / — path separator; /**/ — any number of directories; <dir_name>:pattern> — restrissource roots with the specified name Clear output directory on rebuild Add @NotNull assertions Automatically show first error in editor Display notification on build completion Make project automatically (only works while not running / debugging) Compile independent modules in parallel (may require larger heap size) Rebuild module on dependency change 700 Additional build process VM options: 700</dir_name>				ero or
Android Compiler Android Compilers Coverage © Debugger Path Variables					
			ок	Cancel <u>A</u> pply He	elp

Figure 17. IntelliJ Compiler Settings

On the Annotation Processors page, enable and adjust for the 'default' setting:

I Settings						×	
Q	\supset	Build, Execution, Deployment > Compiler > Anno	tation Processors 🐵 For current proje	ct			
Editor		+ - 5	Enable annotation processing				
File Types		▼ Default	 Obtain processors from project (classpath			
Copyright	e	tev					
Emmet		🔁 isis	O Processor path:				
Images		🔁 isis-app-kitchensink	Store generated sources relative to:	O Module output direct	ony O Module content root		
Intentions		🕞 isis-app-kitchensink-app	Store generated sources relative to.				
Language Injections	•	🕞 isis-app-kitchensink-dom	Production sources directory:	target/generated-source	s/annotations		
Spelling	•	🖬 isis-app-kitchensink-fixture					
TODO		🖬 isis-app-kitchensink-integtests	lest sources directory:	target/generated-test-so	ources/test-annotations		
Plugins		🖬 isis-app-kitchensink-webapp	Annotation Processors				
Version Control		🕞 isis-core-security					
Build, Execution, Deployment		🕞 isis-documentation		Processor FQ	Name	+	
Build Tools Compiler		🕞 isis-viewer-wicket-applib	Compiler will run all automatically discovered processors				
		🕞 isis-viewer-wicket-impl	Complier will run all automatically discovered processors				
Excludes		🕞 isis-viewer-wicket-model					
Java Compiler		🕞 isis-viewer-wicket-ui					
Annotation Processors		🖬 simpleapp	Apportation Processor options				
Validation		🕞 simpleapp-app	Annotation Processor options				
RMI Compiler		🔁 simpleapp-dom	Option Name		Value	+	
Groovy Compiler		🕞 simpleapp-fixture				-	
Coverage	•	simpleapp-integtests		No processor-specific op	otions configured		
Debugger		🕞 simpleapp-webapp					
Path Variables		Maven default annotation processors profile					
Languages & Frameworks		Annotation profile for asciidoctor-maven-plugin	WARNING				
▶ Tools		Annotation profile for estatio-dom	If option 'Clear output directory	on rebuild' is enabled, th	e entire contents of directories where genera	ated	
Other Settings			sources are stored WILL BE CLEAR	RED on rebuild.			
-				2			
					OK Cancel Apply	Help	

Figure 18. IntelliJ Annotation Processor Settings

This setting enables the generation of the Q* classes for DataNucleus type-safe queries, as well as being required for frameworks such as Project Lombok.



IntelliJ may also have inferred these settings for specific projects/modules when importing; review the list on the left to see if the default is overridden and fix/delete as required.

Other Settings (Maven)

There are also some other settings for Maven that we recommend you adjust (though these are less critical):

First, specify an up-to-date Maven installation, using File > Settings (or Intellij > Preferences if on MacOS):

J i		Settings
Q maven 😵	Build, Execution, Deploymen	t > Build Tools > Maven 🐵 For current project
Appearance & Behavior	Work <u>o</u> ffline	
Notifications	Use plugin <u>r</u> egistry	
Keymap	Z Execute goals recursively	
▼ Editor	Print exception stack trace	s
File and Code Templates	Always update <u>s</u> napshots	
Live Templates	Output <u>l</u> evel:	Info
Plugins • Build, Execution, Deployment	<u>C</u> hecksum policy:	No Global Policy
▼ Build Tools @	Multiproject build fail policy:	Default
▼ Maven @	Plugin undate policy:	Default ignored by Mayen 3+
Importing	Threads (Testien):	
Runner	Mayen home directory	Rundled (Maven 3)
Running Tests	waven <u>n</u> ome directory.	Pundled (Mayon 2)
Repositories	User <u>s</u> ettings file:	Bundled (Maven 2) Bundled (Maven 3) C:/bin/apache-maven-3.2.3
	Local repository:	C:\Users\Dan\.m2\repository
		OK Cancel Apply Help

Figure 19. IntelliJ Maven Settings - Installation

Still on the Maven settings page, configure as follows:



Figure 20. IntelliJ Maven Settings - Configuration

Other Settings (Misc)

These settings are optional but also recommended.

On the **auto import** page, check the optimize imports on the fly and add unambiguous imports on the fly

ฮเ		Settings	×
Q	\supset	Editor > General > Auto Import	Reset
▶ Appearance & Behavior		XML	
Keymap		Show import popup	
▼ Editor			
▼ General		Java	
Smart Keys		Insert imports on paste: Ask	
Appearance			
Editor Tabs		Show import <u>p</u> opup	
Code Folding		Optimize imports on the fly	
Code Completion		Add unambiguous imports on the fly	
Auto Import		Exclude from Import and Completion	
Postfix Completion			
Console Folding			+
Colors & Fonts		No exclude patterns	-
Code Style	Ē		
Java			
Groovy			
HTML			
NOSL			
Properties			
XML		Ν	
Inspections	Ē	νζ	
File and Code Templates	Ē		
File Encodings	ē		
Line Terroleter			
			OK Cancel <u>Apply</u> Help

Figure 21. IntelliJ Maven Settings - Auto Import

2.1.2. Importing Maven Modules

Let's load in some actual code! We do this by importing the Maven modules.

First up, open up the Maven tool window (View > Tool Windows > Maven Projects). You can then use the 'plus' button to add Maven modules. In the screenshot you can see we've loaded in Apache Isis core; the modules are listed in the *Maven Projects* window and corresponding (IntelliJ) modules are shown in the *Projects* window:



Figure 22. IntelliJ Maven Module Management - Importing Maven modules

We can then import another module (from some other directory). For example, here we are importing the Isis Addons' todoapp example:



Figure 23. IntelliJ Maven Module Management - Importing another Module

You should then see the new Maven module loaded in the Projects window and also the Maven

Projects window:



Figure 24. IntelliJ Maven Module Management -

If any dependencies are already loaded in the project, then IntelliJ will automatically update the CLASSPATH to resolve to locally held modules (rather from .m2/repository folder). So, for example (assuming that the <version> is correct, of course), the Isis todoapp will have local dependencies on the Apache Isis core.

You can press F4 (or use File > Project Structure) to see the resolved classpath for any of the modules loaded into the project.

If you want to focus on one set of code (eg the Isis todoapp but not Apache Isis core) then you *could* remove the module; but better is to ignore those modules. This will remove from the the *Projects* window but keep them available in the *Maven Projects* window for when you next want to work on them:



Figure 25. IntelliJ Maven Module Management - Ignoring Modules

Confirm that it's ok to ignore these modules:



Figure 26. IntelliJ Maven Module Management - Ignoring Modules (ctd)

All being well you should see that the *Projects* window now only contains the code you are working on. Its classpath dependencies will be adjusted (eg to resolve to Apache Isis core from .m2/repository):



Figure 27. IntelliJ Maven Module Management - Updated Projects Window

2.1.3. Running

Let's see how to run both the app and the tests.

Running the App

Once you've imported your Isis application, we should run it. We do this by creating a Run configuration, using Run > Edit Configurations.

Set up the details as follows:

ฮเ	Run/Debug Configurations ×					
+ - ① % ↑ ↓ D ↓2 ▼ C Application	Name: todoapp Configuration Code Coverage Logs Startup/Connection					
 ► IUnit ► % Defaults 	Main glass: org.apache.isis.WebServer VM options: Image: Comparent set of the set					
	Before launch: Make, Maven Goal + - / ↑ ↓ Make MRun Maven Goal 'Isis Addons ToDoApp DOM: datanucleus:enhance -o' OK Cancel Apply Help					

Figure 28. IntelliJ Running the App - Run Configuration

We specify the Main class to be org.apache.isis.WebServer; this is a wrapper around Jetty. It's possible to pass program arguments to this (eg to automatically install fixtures), but for now leave this blank.

Also note that Use classpath of module is the webapp module for your app, and that the working directory is \$MODULE_DIR\$.

Next, and most importantly, configure the DataNucleus enhancer to run for your dom goal. This can be done by defining a Maven goal to run before the app:

I	Select Maven Goal	×
Working <u>d</u> irectory	C:/GITHUB/isisaddons/isis-app-todoapp/dom	🖬
<u>C</u> ommand line	datanucleus:enhance -o	
	ок	Cancel

Figure 29. IntelliJ Running the App - Datanucleus Enhancer Goal

The -o flag in the goal means run off-line; this will run faster.



if you forget to set up the enhancer goal, or don't run it on the correct (dom) module, then you will get all sorts of errors when you startup. These usually manifest themselves as class cast exception in DataNucleus.

You should now be able to run the app using Run > Run Configuration. The same configuration can also be used to debug the app if you so need.

Running the Unit Tests

The easiest way to run the unit tests is just to right click on the dom module in the *Project Window*, and choose run unit tests. Hopefully your tests will pass (!).

🔻 🗖 isis-app-todoapp [todoapp	Find Usages	۸l++ E7
I dom [todoapp-dom]		
Fixture Itadeann fixture	Find in <u>P</u> ath	Ctrl+Shift+F
	Repl <u>a</u> ce in Path	Ctrl+Shift+R
Images	Analy <u>z</u> e	•
Integtests [todoapp-int]	Defector	
🗖 logs	Refactor	· · · · ·
🕨 🖿 webapp [todoapp-weba	Add to F <u>a</u> vorites	•
.gitattributes	Show Image Thumbnails	Ctrl+Shift+T
🕸 .gitignore	<u>R</u> eformat Code	Ctrl+Alt+L
🗈 .travis.yml	Optimi <u>z</u> e Imports	Ctrl+Alt+O
bumpver.sh	Remove Module	Delete
🗈 datanucleus.log	🌣 Run Maven	•
LICENSE	Debug Maven	•
<i>m</i> pom.xml		
pom.xml.versionsBackup	Make Module todoapp-dom	
M README.md	Compil <u>e</u> Module 'todoapp-dom'	Ctrl+Shift+F9
	📧 Create 'All Tests'	
₽ todoapp.imi	Run 'All Tests'	Ctrl+Shift+F10
🕨 🖬 isis-module-sessionlogge		

Figure 30. IntelliJ Running the App - Unit Tests Run Configuration

As a side-effect, this will create a run configuration, very similar to the one we manually created for the main app:

וש	Run/Debug	g Configurations	×
+ - 🕅 🗄 🐓 🛧 🕨 ե 🎼	Name: All in todoapp-do	om	□ <u>S</u> hare □ Single <u>i</u> nstance only
 Application Unit 	Configuration Code Co	verage Logs Startup/Connection	
📼 All in todoapp-dom	Test kind: All in	package 🔽	<u>F</u> ork mode: none
All in todoapp-integtests Se Defaults	Package:		
	Search for tests: O In y	<u>w</u> hole project	
	o In s	s <u>i</u> ngle module	
	O Acr	oss modu <u>l</u> e dependencies	
	<u>V</u> M options:	-ea	
	Program a <u>rg</u> uments:		
	Working directory:	\$MODULE_DIR\$	
	Environment variables:		
	Use classpath of mod	todoapp-dom	
	Use alternative JRF:		
		L	
	<u>B</u> efore launch: Make		
		ок	Cancel Apply Help

Figure 31. IntelliJ Running the App - Unit Tests Run Configuration

Thereafter, you should run units by selecting this configuration (if you use the right click approach you'll end up with lots of run configurations, all similar).

Running the Integration Tests

Integration tests can be run in the same way as unit tests, however the dom module must also have been enhanced.

One approach is to initially run the tests use the right click on the integtests module; the tests will fail because the code won't have been enhanced, but we can then go and update the run configuration to run the datanucleus enhancer goal (same as when running the application):

วเ	Run/Debug	g Configurations	×
+ - 🛱 🖁 🌮 ↑ ∔ 🗅 ↓2	Name: All in todoapp-in	tegtests	□ <u>S</u> hare □ Single <u>i</u> nstance only
Application JUnit	Configuration Code Co	verage Logs Startup/Connection	
💷 All in todoapp-dom	Test kind: All in p	oackage 🔽	Eork mode: none
All in todoapp-integtests Separate	Package:		
	Search for tests: O In y	<u>v</u> hole project	
	O In s	ingle module	
	O Acr	oss modu <u>l</u> e dependencies	
	<u>∨</u> M options:	-ea	
	Program a <u>rg</u> uments:		
	Working directory:	\$MODULE_DIR\$	
	Environment variables:		
	Use classpath of mod	todoapp-integtests	
	Use alternative JRE:		
		L	
	<u>B</u> efore launch: Make, May	en Goal	
	+ F F F F F F F F F F F F F F F F F F F		
	<i>m</i> Run Maven Goal 'Isis A	ddons ToDoApp DOM: datanucleus:enhar	nce -o'
	•	ОК	Cancel Apply Help

Figure 32. IntelliJ Running the App - Integration Tests Run Configuration

2.1.4. Hints and Tips

Keyboard Cheat Sheets

You can download 1-page PDFs cheat sheets for IntelliJ's keyboard shortcuts: * for Windows * for MacOS

Probably the most important shortcut on them is for Find Action: - ctrl-shift-A on Windows - cmd-shift-A on MacOS.

This will let you search for any action just by typing its name.

Switch between Tools & Editors

The Tool Windows are the views around the editor (to left, bottom and right). It's possible to move these around to your preferred locations.

- Use alt-1 through alt-9 (or cmd-1 through alt-9) to select the tool windows
 - Press it twice and the tool window will hide itself; so can use to toggle
- If in the *Project Window* (say) and hit enter on a file, then it will be shown in the editor, but (conveniently) the focus remains in the tool window. To switch to the editor, just press Esc.
 - If in the *Terminal Window*, you'll need to press Shift-Esc.
- If on the editor and want to locate the file in (say) the *Project Window*, use alt-F1.
- To change the size of any tool window, use ctrl-shift-arrow

Using these shortcuts you can easily toggle between the tool windows and the editor, without using the mouse. Peachy!

Navigating Around

For all of the following, you don't need to type every letter, typing "ab" will actually search for ".a.*b.".

- to open classes or files or methods that you know the name of:
 - ctrl-N to open class
 - ctrl-shift-N to open a file
 - (bit fiddly this) ctrl-shift-alt-N to search for any symbol.
- open up dialog of recent files: ctrl-E
- search for any file: shift-shift

Navigating around: * find callers of a method (the call hierarchy): ctrl-alt-H * find subclasses or overrides: ctrl-alt-B * find superclasses/interface/declaration: ctrl-B

Viewing the structure (ie outline) of a class * ctrl-F12 will pop-up a dialog showing all members ** hit ctrl-F12 again to also see inherited members

Editing

- Extend selection using ctrl-W
 - and contract it down again using ctrl-shift-W
- to duplicate a line, it's ctrl-D
 - if you have some text selected (or even some lines), it'll actually duplicate the entire selection
- to delete a line, it's ctrl-X
- to move a line up or down: shift-alt-up and shift-alt-down
 - if you have selected several lines, it'll move them all togethe
- ctrl-shift-J can be handy for joining lines together
 - just hit enter to split them apart (even in string quotes; IntelliJ will "do the right thing")

Intentions and Code Completion

Massively useful is the "Intentions" popup; IntelliJ tries to guess what you might want to do. You can activate this using `alt-enter`, whenever you see a lightbulb/tooltip in the margin of the current line.

Code completion usually happens whenever you type '.'. You can also use ctrl-space to bring these up.

In certain circumstances (eg in methods0) you can also type ctrl-shift-space to get a smart list of methods etc that you might want to call. Can be useful.

Last, when invoking a method, use ctrl-P to see the parameter types.

Refactoring

Loads of good stuff on the Refactor menu; most used are:

- Rename (shift-F6)
- Extract
 - method: ctrl-alt-M
 - variable: ctrl-alt-V
- Inline method/variable: ctrl-alt-N
- Change signature

If you can't remember all those shortcuts, just use ctrl-shift-alt-T (might want to rebind that to something else!) and get a context-sensitive list of refactorings available for the currently selected object

Plugins

You might want to set up some additional plugins. You can do this using File > Settings > Plugins (or equivalently File > Other Settings > Configure Plugins).

Recommended are:

• Maven Helper plugin

More on this below.

• AsciiDoctor plugin

Useful if you are doing any authoring of documents.

Some others you might like to explore are:



Figure 33. IntelliJ Plugins

Maven Helper Plugin

This plugin provides a couple of great features. One is better visualization of dependency trees (similar to Eclipse).

If you open a pom.xml file, you'll see an additional "Dependencies" tab:



Clicking on this gives a graphical tree representation of the dependencies, similar to that obtained by mvn dependency:tree, but filterable.

m simpleapp-webapp	
○ Conflicts	Refresh
O All Dependencies as List	
• All Dependencies as Tree	
Q* Show GroupId 😤 😤	
geronimo-servlet_3.0_spec : 1.0 (compile)	
hsqldb : 2.3.1 (compile)	
▼ isis-core-runtime : 1.9.0-SNAPSHOT (compile)	
activation : 1.1.1 (compile)	
▼ axon-core : 2.4 (compile)	
cglib-nodep : 2.2.2 (compile)	
commons-collections : 3.2.1 (compile)	
commons-io : 2.4 (compile)	
disruptor : 3.2.0 (compile)	
joda-time : 2.3 (compile)	
slf4j-api : 1.7.10 (compile)	Nothing to show
▼ xstream : 1.4.7 (compile)	
xmlpull : 1.1.3.1 (compile)	
xpp3_min : 1.1.4c (compile)	
▼ commons-email : 1.3.3 (compile)	
activation : 1.1.1 (compile)	
▼ dom4j : 1.6.1 (compile)	
xml-apis : 1.0.b2 (compile)	
guava : 16.0.1 (compile)	
hamcrest-library : 1.3 (compile)	
isis-core-log4j : 1.9.0-SNAPSHOT (compile)	
guava : 16.0.1 (compile)	
hamcrest-library : 1.3 (compile)	
log4j : 1.2.17 (compile)	
sif4j-api : 1.7.10 (compile)	
slf4j-log4j12 : 1.7.10 (compile)	
▼ isis-core-metamodel : 1.9.0-SNAPSHOT (compile)	
commons-cli : 1.2 (compile)	
commons-codec : 1.9 (compile)	
datanucleus-api-jdo : 4.0.5 (compile)	
datanucleus-core : 4.0.6 (compile)	
datanucleus-ido-queny: 4.0.4 (compile)	
Test Desenderer Archere	

The plugin also provides the ability to easily run a Maven goal on a project:

J				New	•	·0	jects\dev] - simpleapp
<u>F</u> ile		<u>E</u> dit <u>V</u> iew <u>N</u> avigate <u>C</u> ode Analy <u>z</u> e <u>R</u> efactor <u>B</u> uild R <u>u</u> n		Add Framework Support			
C:	si	impleapp 🔪 🖬 webapp 👌		AsciiDoc	•		
ť		Project 👻	Ж	Cu <u>t</u>	Ctrl+X		
Proje	•	incode-camel-coda and modules	ß	<u>С</u> ору	Ctrl+C	F	
÷	Þ	isis-tck and modules		C <u>o</u> py Path	Ctrl+Shift+C	ас	che.org/licenses/LICENSE
~	Þ	and modules		Copy as Plain Text		00	licable law or agreed t
ture	Þ	Calorg.apache.isis.core and modules		Copy Reference 0	Ctrl+Alt+Shift+C	u	nder the License is dis
Stru	v	🖬 simpleapp and modules	đ	<u>P</u> aste	Ctrl+V		WARRANTIES OR CONDITIO
۷		Image: Complete Co		Find Usages	Alt+F7	ve	erning permissions and l
		Fixture [simpleapp-fixture]		Find in Path	Ctrl+Shift+F		
		Integrests [simpleapp-integrests]		Replace in Path	Ctrl+Shift+R	/m	aven.apache.org/POM/4.0
		Image: Simpleapp		Analyze	,	/m • • <	aven.apache.org/maven-v
		🔻 🔁 webapp [simpleapp-webapp]		Refactor	•		, mode ever bronk
		Ide		Add to Foundton			che isis example applic
		▶ 🗖 lib		Add to Favorites	Charles Chiffles T	np	leapp
				Show image inumbhalis	Ctri+Shilt+1	-S	NAPSHOT
		▶ □ src		<u>R</u> eformat Code	Ctrl+Alt+L		
		m pom.xml		Optimize Imports	Ctrl+Alt+O	ap	<pre>p-webapp pape(/artifactId>)</pre>
		Li todoapp and modules		Remove Module	Delete	Ē	app / mames
		Content-OLDSITE (C:\APACHE\Isis-site\content-OLDSITE)	¢	Run Maven	Þ	m	1 Test file ck
		Componentation linin documentation (COMPACHE) isis ait	¢	Debug Maven	•	0	clean generate-sources
		Liova assortiant (danhayawaad java assortiant) (C. (AFACHE (ISIS-GIL		Make <u>M</u> odule 'simpleapp-webapp'		0	help:effective-pom
		simpleann [simpleann_archetyne] (C:\APACHE\isis-ait-n		Compil <u>e</u> e 'simpleapp-webapp'	Ctrl+Shift+F9	0	dependency:tree
		External Libraries		Create 'All Tests'		0	clean install
			Þ	R <u>u</u> n 'All Tests'	Ctrl+Shift+F10	0	clean
			<u>)</u> (<u>D</u> ebug 'All Tests'		0	validate
			G	Profile 'All Tests'		0	compile
			8	Run 'All Tests' with Co <u>v</u> erage		0	test ro
				Local <u>H</u> istory	•	0	package n<
				Git	•	•	verify
			Ø	Synchronize 'webapp'		0	install
				Show in Explorer	Alt+X	0	deploy ty
				File Path	Ctrl+Alt+F12	0	site
			ß	Compare With	Ctrl+D	0	jetty:run
				External Tools			Plugins Ie
				Open Medule Settings	Г <i>А</i>	<u>c</u>	Reimport
				Move Module to Group	r-4		New Goal \${
avorites				Mark Directory As	,		<pre><destinationfile>\${ nackage</destinationfile></pre>

This menu can also be bound to a keystroke so that it is available as a pop-up:



Troubleshooting

When a Maven module is imported, IntelliJ generates its own project files (suffix .ipr), and the application is actually built from that.

Occasionally these don't keep in sync (even if auto-import of Maven modules has been enabled).

To fix the issue, try: * reimport module * rebuild selected modules/entire project * remove and then re-add the project * restart, invalidating caches * hit StackOverflow (!)

One thing worth knowing; IntelliJ actively scans the filesystem all the time. It's therefore (almost always) fine to build the app from the Maven command line; IntelliJ will detect the changes and keep in sync. If you want to force that, use File > Synchronize, ctrl-alt-Y.

If you hit an error of "duplicate classes":



then make sure you have correctly configured the annotation processor settings. Pay attention in particular to the "Production sources directory" and "Test sources directory", that these are set up correctly.

2.1.5. Running Integration Tests

When running integration tests from within IntelliJ, make sure that the search for tests radio button is set to In single module:

I Run/Debug Configurations				×
+ - 🗊 🗄 🛠 🕈 + 🖿 12	<u>N</u> ame:	All in incode-est	tatio-ecp-migration-integte	Single instance only
 	Config	guration Code C	overage Logs Startup/Conn	ection
All in estatio-integtests Property Manu Tast\$ NewProperty, pewProperty	<u>T</u> est k	tind: All in	package 🔻	Eork mode: none
LeaseTest\$NewMandate.whenSecondaryPartyIsKnownAndHasBan	Packa	ge:		
InvoiceCalculationServiceTest_normalRun All in incode-estatio-ecp-migration-integtests	Searc	h for tests: O li	n <u>w</u> hole project	
▶ 🖗 Defaults		A ()	cross module dependencies	
	<u>V</u> M o	ptions:	-ea	
	Progr	am a <u>rg</u> uments:		T
	<u>W</u> ork	ing directory:	\$MODULE_DIR\$	
	<u>E</u> nviro	onment variables:		
	Use c	lasspath of m <u>o</u> d	🔁 incode-estatio-ecp-migr	ation-integtests 🔹
		lse alternative <u>J</u> RE	:	• •••
	▼ B <u>e</u> for	e launch: Make, M	aven Goal	
	+ -	ke		
	m Rur	n Maven Goal 'Est	atio (ECP) DOM - Italy: datanuc	cleus:enhance -o'
	Sho	ow this page	Ν	
			43	
			ок	Cancel <u>A</u> pply Help

If this radio button is set to one of the other options then you may obtain class loading issues; these result from IntelliJ attempting to run unit tests of the dom project that depend on test classes in that module, but using the classpath of the integtests module whereby the dom test-classes (test-jar artifact) are not exposed on the Maven classpath.

2.1.6. Advanced

In this section are a couple of options that will reduce the length of the change code/build/deploy/review feedback loop.

Setting up Dynamic Reloading

DCEVM enhances the JVM with true hot-swap adding/removing of methods as well as more reliable hot swapping of the implementation of existing methods.

In the context of Apache Isis, this is very useful for contributed actions and mixins and also view models; you should then be able to write these actions and have them be picked up without restarting the application.

Changing persisting domain entities is more problematic, for two reasons: the JDO/DataNucleus enhancer needs to run on domain entities, and also at runtime JDO/DataNucleus would need to rebuild its own metamodel. You may find that adding actions will work, but adding new properties or collections is much less likely to.

To set up DCEVM, download the appropriate JAR from the <u>github page</u>, and run the installer. For example:

```
java -jar DCEVM-light-8u51-installer.jar
```



Be sure to run with appropriate privileges to be able to write to the installation directories of the JDK. If running on Windows, that means running as Administrator.

After a few seconds this will display a dialog listing all installations of JDK that have been found:

				- 🗆	×
Dynamic Code E A modification of the Java HotSpot(TM)	Volutic VM that allows	on VM	s redefinition at runtime.	Object A B C A A A A B New Version	
Enhance current Java (JRE/JDK) installation You can either replace current Java VM or i	s with DCEVM	(http://git as alternat	chub.com/dcevm/dcevm). tive JVM (run with -XXeltjvm=d	cevm command-line option).	^
This program is free software; you can rediversion 2 only, as published by the Free So This code is distributed in the hope that i Please choose installation directory:	stribute it a ftware Found t will be use	and/or modif ation. eful, but WI	Ty it under the terms of the G	NU General Public License ven the implied warranty of	*
Directory	Java Version	Type	Replaced by DCEVM?	Installed altivm?	
C:\Program Files\lava\idk1 7 0 79	17079	1DK (64 Bit)	No	No	_
cryrograinn ies pava gaktinio_75	1.8.0.45	1DK (64 Bit)	No	No	
IC:\Program Files\lava\idk1.8.0_45	10.010_10				
C:\Program Files\Java\jdk1.8.0_45 C:\Program Files\Java\jdk1.8.0_51	1.8.0 51	JDK (64 BIT)	NO		
C:\Program Files\Java\jdk1.8.0_45 C:\Program Files\Java\jdk1.8.0_51 C:\Program Files\Java\jre1.8.0_65	1.8.0_51	JDK (64 Bit) JRE (64 Bit)	No	No	
C: \Program Files\Java\jdk1.8.0_45 C:\Program Files\Java\jdk1.8.0_51 C:\Program Files\Java\jre1.8.0_65 C:\Program Files\Java\jre7	1.8.0_51 1.8.0_65 1.7.0_79	JRE (64 Bit) JRE (64 Bit) JRE (64 Bit)	No No	No No No	
C: \Program Files\Java\jdk1.8.0_45 C:\Program Files\Java\jdk1.8.0_51 C:\Program Files\Java\jre1.8.0_65 C:\Program Files\Java\jre7 C:\java\jprofiler8	1.8.0_51 1.8.0_65 1.7.0_79 Could not get	JRE (64 Bit) JRE (64 Bit) JRE (64 Bit) JDK	No No No No	No No No	

Select the corresponding installation, and select Replace by DCEVM.

Dynamic Code Evolution VM Installer				— п x					
Dynamic Code E A modification of the Java HotSpot(TM,	Volutic	on VM	ss redefinition at runtime.	Object A B C New Version					
Enhance current Java (JRE/JDK) installation	ns with DCEVM	(http://gi	thub.com/dcevm/dcevm).	^					
You can either replace current Java VM or install DCEVM as alternative JVM (run with -XXaltjvm=dcevm command-line option). This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License version 2 only, as published by the Free Software Foundation.									
Please choose installation directory:									
Directory	Java Version	Type	Replaced by DCEVM2	Installed altivm?					
C:\Program Files\lava\idk1 7.0.79	1 7 0 79	IDK (64 Bit)	No	No					
C:\Program Files\Java\idk1.8.0.45	1.8.0.45	1DK (64 Bit)	No	No					
C:\Program Files\Java\idk1.8.0_51	1.8.0 51	1DK (64 Bit)	Yes (25.51-b03-dcevmlight-3)	No					
C:\Program Files\Java\ire1.8.0 65	1.8.0 65	JRE (64 Bit)	No	No					
C:\Program Files\Java\jre7	1.7.0_79	JRE (64 Bit)	No	No					
C:\java\jprofiler8	Could not get	JDK	No	No					
Add installation directory									

In IntelliJ, register the JDK in File > Project Structure dialog:



Finally, in the run configuration, select the patched JDK:

I Run/Debug Configurations			×		
+ - 🗈 🛠 🕈 💺 🗖 🐙	Name: incode-module-c	ommchannel-webapp (with fixtures)	ce only		
V 🖶 Application	Configuration Code Coverage Loss Startum/Connection				
isisaddons	Conigaration Code Co	verage Logs startup/connection			
Incode	Main <u>c</u> lass:	org.apache.isis.WebServer			
other	VM options:		B		
 Isisaddons-app-quickstart-webapp (with fixtures) 	Brogram argumente	m ara incode module commohannel ann CommChannelModuleAnnManifectWithFirsturer			
incode-module-commchannel-webapp (with fixtures)	Program arguments.	-morg.incode.module.commonanne.app.commonannelwioddieAppWannestWitnPixtures			
Cincode-module-note-webapp (with fixtures)	Working directory:	\$MODULE_DIR\$			
incode-module-note-webapp (no fixtures)	Environment variables:				
a simple app-web app					
simpleapp-webapp (with fixtures)	Use classpath of mod	incode-module-commchannel-webapp			
™IsisCli	☑ Use alternative JRE:	1.8.0_51 (DCEVM patched)			
▶ ■ JUnit					
Maven	Enable capturing for	m snapshots			
▶ % Defaults	■ Before launch: Make, Maw + - / / ↑ ↓ ↓ ## Make m Run Maven Goal 'Incoc Show this page	ren Goal			
		OK Cancel Apply F	telp		

Setting up JRebel

See the repo for the (non-ASF) Isis JRebel plugin. With some modification, this should work for IntelliJ too.

Note that JRebel is a commercial product, requiring a license. At the time of writing there is also

currently a non-commercial free license (though note this comes with some usage conditions).

2.2. Developing using Eclipse



This material does not constitute an endorsement; Eclipse foundation is not affiliated to Apache Software Foundation in any way.

If you are an Eclipse user, then we recommend you download the "Eclipse JEE package" configuration.

When running an Apache Isis application, it's necessary to setup the development environment so that the Java bytecode can be enhanced by the DataNucleus enhancer. If working in Eclipse, then JDO enhancement is most easily done by installing the DataNucleus' Eclipse plugin. This hooks the bytecode enhancement of your domain objects into Eclipse's normal incremental compilation.

This plugin needs to be configured for each of your domain modules (usually just one in any given app). The steps are therefore:

- import the project into Eclipse
- configure the DataNucleus enhancer
- run the app from the .launch file

2.2.1. Screencast

This screencast shows how to import an Apache Isis maven-based application into Eclipse and configure to use with the JDO Objectstore.

2.2.2. Editor Templates

We provide a set of editor templates. These are used to add new methods to existing classes. (These are equivalent to the IntelliJ live templates):

- is (Apache Isis domain objects). Download
- ju (for JUnit tests) Download
- jm (for JMock mocks or libraries) Download

To install, download each XML file, then go to Windows > Preferences > Java > Editor > Templates and choose Import.

2.2.3. Importing the Project

Use File > Import, then Maven > Existing Maven Projects.

2.2.4. Add DataNucleus support


In Eclipse, for the *domain object model* project, first add DataNucleus support:

⊿ 🔛 > com.eur ▷ 🚑 > src/n		Copy Qualified Name Paste	Ctrl+V	 pendencyManagement>
▷ ∰ src/test	×	<u>D</u> elete	Delete	and and and
 ▷ ➡ JRE Sys ▷ ➡ Maven ▷ ➡ .setting ▷ ➡ ide 	<u>.</u>	Remove from Context <u>B</u> uild Path <u>S</u> ource Refactor	Ctrl+Alt+Shift+Down Alt+Shift+S Alt+Shift+T	<pre>endencies> <dependency></dependency></pre>
 ▷ □ IIb □ IIb □	è Z	Import Exp <u>o</u> rt		<pre><groupid>org.apache.isis.run <artifactid>jdo-applib <dependency> <groupid>org.apache.isis.run</groupid></dependency></artifactid></groupid></pre>
▷ Construction of the second seco	6g	Re <u>f</u> resh Clo <u>s</u> e Project Close <u>U</u> nrelated Projects <u>A</u> ssign Working Sets	F5	<pre><groupid>org.xnap.commons </groupid>org.xnap.commons</pre>
Image: Point of the second	-	<u>R</u> un As <u>D</u> ebug As <u>P</u> rofile As Mark as Deployable	> >	encies Dependency Hierarchy Effective P
<pre></pre>		Validate T <u>e</u> am Compare With Rep <u>l</u> ace With Restore from Local History	> >	ar constraints (DTD or XML schema) detect ar constraints (DTD or XML schema) detect ar constraints (DTD or XML schema) detect java.math.BigDecimal is never used javax.jdo.annotations.IdentityType is neve
		DataNucleus	•	Add DataNucleus Support
		<u>Ivi</u> aven	•	liavavido annotations IdGeneratorStrategy

Then turn on Auto-Enhancement:

⊿ 🛗 > com.eu	Ð	Copy Qualified Name		
▷ 2 > src/	B	Paste	Ctrl+V	
⊳ 👍 src/m	~	- Delete	Delete	<pre>ependencyManagement></pre>
⊳ 🔠 src/te	^	Delete	Delete	nendencies)
JRE Sy	<u>.</u>	Remove from Context	Ctrl+Alt+Shift+Down	<dependency></dependency>
Maver		Build Path	•	<pre><groupid>org.apache.isis</groupid></pre>
👂 🗁 .settin		<u>Course</u>	Alta Chitta C A	<pre><artifactid>applib</artifactid></pre>
> 🗁 ide		Source	Alt+Shift+S	
⊳ 🚰 lib		Refac <u>t</u> or	Alt+Shift+T ►	<pre><groupid>org apache isis runtimes</groupid></pre>
> 🚠 > src	5.	Import		<pre><artifactid>ido-applib</artifactid></pre>
b 📂 target				
🗁 target	ک	Export		<dependency></dependency>
⊳ 🗁 uml	, (shi	Refresh	F5	<pre> <groupid>org.apache.isis.runtimes.</groupid></pre>
x .classp	× .	Close Project		
📑 .gitigr		Clo <u>s</u> e Project		() dependency?
🛐 > .pro		Close Unrelated Projects		<dependency></dependency>
📄 datanı		Assign Working Sets		<pre><groupid>org.xnap.commons</groupid></pre>
🛃 > pon		Due Ar		<pre></pre>
b 🛃 com.euro		<u>K</u> un As	,	dencies Dependency Hierarchy Effective POM po
b 🕌 com.euro		<u>D</u> ebug As	•	
N 🔛 com euro		Profile As	►	
	e .	Mark as Deployable		gs, 0 others
Unit 🏇 Debug	-	Validate		
		Teens		Enable Auto-Enhancement
			· · · · ·	Run Enhancer Tool
		Compare With	•	Pun Schema Tool
		Rep <u>l</u> ace With	►	
		Restore from Local History		Create persistence.xml for project
		DataNucleus	+	Remove DataNucleus Support
		Mayen	•	Tr Javax.juo.annotations.tuoeneratorotrategy is neve

Update the classpath

DataNucleus' enhancer uses the domain object model's own classpath to reference DataNucleus JARs. So, even though your domain objects are unlikely to depend on DataNucleus, these references must still be present.

See the earlier section on DataNucleus enhancer for details of the contents of the pom.xml. Chances are it is already set up from running the SimpleApp archetype.

Then, tell DataNucleus to use the project classpath:

Preferences			
type filter text		DataNucleus	⇐ ▾ ⇔ ▾
b General			
⊳ Ant		Settings that are applied across the DataNucleus Enhancer/SchemaTool	
b Data Management		Persistence API JDO 🗸	
⊿ DataNucleus		Puntime Classrath Entries	
Enhancer		Kunume Classpath Entries	
SchemaTool			
GlassFish Preferences			
⊳ Help			
HQL editor			
Install/Update	=		
⊳ Java			
Java EE			
Java Persistence			
JavaScript			
JBoss Tools			
b Maven			
⊳ Mylyn		Add JARs Remove	
Plug-in Development			
Project Archives		Ulse project classpath when rupping tools	
Remote Systems			
Run/Debug		Logging Configuration: C:\java\eclipse-jee-indigo-SR2-wir Browse	
KunJettykun			
> Jeiver	-		Restore <u>D</u> efaults <u>Apply</u>
?			OK Cancel
-			

When the enhancer runs, it will print out to the console:

📮 Console 🕱 🛛 🔳	🗙 💥 📑 🛃 🚍 🚝 🛃 📼 🖛 🗂 🗖
<terminated> DataNucleus Enhancer [Java Application</terminated>	n] C:\Program Files\Java\jre6\bin\javaw.exe (28 Nov 20
DataNucleus Enhancer (version 3.1.1) : E	hancement of classes
DataNucleus Enhancer completed with succ	ess for 30 classes. Timings : input=132
	-
< III	•

Workaround for path limits (the DN plugin to use the persistence.xml)

If running on Windows then the DataNucleus plugin is very likely to hit the Windows path limit.

To fix this, we configure the enhancer to read from the persistence.xml file.

As a prerequisite, first make sure that your domain object model has a persistence.xml file. Then specify the persistence-unit in the project properties:

Properties for quickstart_wicket	et_restful_jdo-dom	
type filter text	Enhancer	⇔ • ⇔ • •
Resource Builders	☑ Enable project specific settings	Configure Workspace Settings
DataNucleus	Please set your preferences for use of the DataNucleus Enhancer. These will be th	e defaults when enhancing all projects
Enhancer	Input File Extensions	
SchemaTool		
Git	class	Add
Hibernate Settings		
Java Build Path		Remove
Java Code Style		
Java Compiler		
Java Editor		
Javadoc Location	📝 Run in Verbose Mode	
Maven	Persistence-Unit name quickstart	
Project Archives		
Project Facets		
Project References		
Run/Debug Settings		
Task Penesiten/		
Task Tags		
Templates		
Validation		
WikiText		
		Restore <u>D</u> efaults <u>Apply</u>
?		OK Cancel

Workaround: If the enhancer fails

On occasion it appears that Eclipse can attempt to run two instances of the DataNucleus enhancer. This is probably due to multiple Eclipse builders being defined; we've noticed multiple entries in the Eclipse's Debug view:



At any rate, you'll know you've encountered this error if you see the following in the console:



The best solution is to remove DataNucleus support and then to re-add it:

⊿ 🔐 > com.eu		Cop <u>y</u> Qualified Name	CHUN	
⊳ @# src/m		Paste	Ctrl+V	ependencyManagement>
⊳ ﷺ src/te	×	Delete	Delete	
JRE Sy	<u>e</u>	Remove from Context	Ctrl+Alt+Shift+Down	<dependency></dependency>
▷ ➡ Maver ▷ ➡ .settin ▷ ➡ ide ▷ ➡ lib		<u>B</u> uild Path <u>S</u> ource Refac <u>t</u> or	► Alt+Shift+S ► Alt+Shift+T ►	<proupid>org.apache.isis</proupid>
> 200 mB > 200 > src > 200 target 200 € target	2 2	Import Exp <u>o</u> rt		<pre><groupid>org.apache.isis.runtimes. <artifactid>jdo-applib <dependency></dependency></artifactid></groupid></pre>
▷ 👉 uml Image: class of the clas	4 C	Re <u>f</u> resh Clo <u>s</u> e Project Close <u>U</u> nrelated Projects <u>A</u> ssign Working Sets	F5	<pre><group1d>org.apache.lsis.runtimes. <artifactid>jdo-datanucleus <dependency> <groupid>org.xnap.commons<artifactid>gettext-commons</artifactid></groupid></dependency></artifactid></group1d></pre>
Image: Point State Image:	* J	<u>R</u> un As <u>D</u> ebug As <u>P</u> rofile As Mark as Deployable <u>V</u> alidate T <u>e</u> am	> > >	dencies Dependency Hierarchy Effective POM po gs, 0 others Chable Auto-Enhancement Bun Enhancer Tool
sterminated		Compare With Replace With Restore from Local History	•	Run Schema Tool Create persistence.xml for project
		DataNucleus	•	Remove DataNucleus Support
		Mayon	h l	per anguer annotation state of the terror of a tegy is never

If you consistently hit problems, then the final recourse is to disable the automatic enhancement and to remember to manually enhance your domain object model before each run.

Not ideal, we know. Please feel free to contribute a better solution :-)

2.2.5. Running the App

The simpleapp archetype automatically provides a **.launch** configurations in the webapp module. You can therefore very simply run the application by right-clicking on one of these files, and choosing "Run As..." or "Debug As...".



The screencast above shows this in action.

2.2.6. Other domain projects.

There is nothing to prevent you having multiple domain projects. You might want to do such that each domain project corresponds to a DDD module, thus guaranteeing that there are no cyclic dependencies between your modules.

If you do this, make sure that each project has its own persistence.xml file.

And, remember also to configure Eclipse's DataNucleus plugin for these other domain projects.

2.2.7. Advanced

In this section are a couple of options that will reduce the length of the change code/build/deploy/review feedback loop.

Setting up Dynamic Reloading

DCEVM enhances the JVM with true hot-swap adding/removing of methods as well as more reliable hot swapping of the implementation of existing methods.

In the context of Apache Isis, this is very useful for contributed actions and mixins and also view models; you should then be able to write these actions and have them be picked up without restarting the application.

Changing persisting domain entities is more problematic, for two reasons: the JDO/DataNucleus enhancer needs to run on domain entities, and also at runtime JDO/DataNucleus would need to rebuild its own metamodel. You may find that adding actions will work, but adding new properties or collections is much less likely to.

For details of setting up DCEVM, see the corresponding section in the IntelliJ documentation.

Chapter 3. Building Apache Isis

3.1. Git

The Apache Isis source code lives in a git repo.

3.1.1. Installation

The easiest place to get hold of command-line git is probably the github download page.

On Windows, this also installs the rather good mSysGit Unix shell. We recommend that you enable git for both the mSysgit and the Windows command prompt:



Once git is installed, the two main command line tools to note are:

- git command line tool
- gitk for viewing the commit history

If using Windows, note that github also have a dedicated Windows client. With a little hacking around, it can also be made to work with non-github repositories.

If using Mac, you might also want to check out Atlassian's Sourcetree.

Cloning the Apache Isis repo

First, clone the Apache Isis repo.

If you are a **committer**, then clone from the Apache read/write repo:

git clone https://git-wip-us.apache.org/repos/asf/isis.git

If you are **not a committer**, please see the <u>contributing</u> page for details on which repo to clone from.

Configuring Git

Next up is to configure your user name and password; see also Apache's git docs:

```
git config user.name "<i>My Name Here</i>"
git config user.email <i>myusername@apache.org</i>
```

Next, configure the core.autocrlf so that line endings are normalized to LF (Unix style) in the rep; again see Apache's git page:

• on Windows, use:

git config core.autocrlf true

• on Mac/Linux, use:

```
git config core.autocrlf input
```

The Windows setting means that files are converted back to CRLF on checkout; the Mac/Linux setting means that the file is left as LF on checkout.

We also recommend setting core.safecrlf, which aims to ensure that any line ending conversion is repeatable. Do this on all platforms:

```
git config core.safecrlf true
```

Note that these settings are supplemented in the repo by the .gitattributes file and that explicitly specifies line handling treatment for most of the common file types that we have.

Next, we recommend you setup this a refspec so that you can distinguish remote tags from local ones. To do that, locate the [remote "origin"] section in your .git/config and add the third entry shown below:

```
[remote "origin"]
    url = ... whatever ...
    fetch = ... whatever ...
    fetch = +refs/tags/*:refs/tags/origin/*
```

This will ensure that a git fetch or git pull places any remote tags under origin/xxx. For example, the `isis-1.0.0`tag on the origin will appear under `origin/isis-1.0.0.

If you don't use git outside of Apache, you can add the --global flag so that the above settings apply for all repos managed by git on your PC.

3.1.2. Getting help

Three commands of git that in particular worth knowing:

• git help command

will open the man page in your web browser

• git gui

will open up a basic GUI client to staging changes and making commits.

• gitk --all

will open the commit history for all branches. In particular, you should be able to see the local master, which branch you are working on (the HEAD), and also the last known position of the master branch from the central repo, called origin/master.

You might also want to explore using a freely available equivalent such as Atlassian SourceTree.

For further reading, see:

- git config man page
- .gitattributes man page
- .gitattributes git-scm.com docs

3.2. Installing Java

Apache Isis is compatible with Java 7 and Java 8. For every-day use, the framework is usually compiled against Java 8.

Releases however are cut using Java 7, leveraging the link :http://maven.apache.org/plugins/maven-toolchains-plugin/[Maven toolchains plugin]).

Therefore install either/both of Java 7 JDK and Java 8 JDK. Note that the JRE is not sufficient.



If you intend to contribute back patches to Apache Isis, note that while you can develop using Java 8 within your IDE, be sure not to use any Java 8 APIs.

3.2.1. Configure Maven toolchains plugin

If you are a committer that will be performing releases of Apache Isis, then you *must* configure the toolchains plugin so that releases can be built using Java 7.

This is done by placing the toolchains.xml file in ~/.m2 directory. Use the following file as a template, adjusting paths for your platform:

```
<?xml version="1.0" encoding="UTF8"?>
<toolchains>
    <toolchain>
        <type>jdk</type>
        <provides>
            <version>1.8</version>
            <vendor>oracle</vendor>
        </provides>
        <configuration>
            <jdkHome>/usr/lib64/jvm/jdk1.8.0_65</jdkHome>
            <!--
            <jdkHome>c:\Program Files\Java\jdk1.8.0_65</jdkHome>
            -->
        </configuration>
    </toolchain>
    <toolchain>
        <type>jdk</type>
        <provides>
            <version>1.7</version>
                                      (1)
            <vendor>oracle</vendor>
        </provides>
        <configuration>
            <jdkHome>/usr/lib64/jvm/jdk1.7.0_79</jdkHome>
            <!--
            <jdkHome>c:\Program Files\Java\jdk1.7.0_79</jdkHome>
            -->
        </configuration>
    </toolchain>
</toolchains>
```

① The Apache Isis build is configured to search for the (1.7, oracle) JDK toolchain.

The Apache Isis parent pom.xml activates this plugin whenever the apache-release profile is enabled.

3.3. Installing Maven

Install Maven 3.0.x, downloadable here.

Set MAVEN_OPTS environment variable:

```
export MAVEN_OPTS="-Xms512m -Xmx1024m"
```



Previously we suggested -XX:MaxPermSize=256m, but this option has been removed in Java 8. (As of 1.9.0, Apache Isis is built using Java 8 but with source and target set to JDK 1.7).

3.4. Building all of Apache Isis

To build the source code from the command line, simply go to the root directory and type:

mvn clean install

The first time you do this, you'll find it takes a while since Maven needs to download all of the Apache Isis prerequisites.

Thereafter you can speed up the build by adding the -o (offline flag). To save more time still, we also recommend that you build in parallel. (Per this blog post), you could also experiment with a number of JDK parameters that we've found also speed up Maven:

```
export MAVEN_OPTS="-Xms512m -Xmx1024m -XX:+TieredCompilation -XX:TieredStopAtLevel=1"
mvn clean install -o -T1C
```

For the most part, though, you may want to rely on an IDE such as Eclipse to build the codebase for you. Both Eclipse and Idea (12.0+) support incremental background compilation.

When using Eclipse, a Maven profile is configured such that Eclipse compiles to target-ide directory rather than the usual target directory. You can therefore switch between Eclipse and Maven command line without one interfering with the other.

3.5. Checking for Vulnerabilities

Apache Isis configures the OWASP dependency check Maven plugin to determine whether the framework uses libraries that are known to have security vulnerabilities.

To check, run:

```
mvn org.owasp:dependency-check-maven:aggregate -Dowasp
```

This will generate a single report under target/dependency-check-report.html.



The first time this runs can take 10~20 minutes to download the NVD data feeds.

To disable, either run in offline mode (add -o or --offline) or omit the owasp property.

3.6. Checking for use of internal JDK APIs

Apache Isis configures the jdeps maven plugin to check for any usage of internal JDK APIs. This is in preparation for Java 9 module system (Jigsaw) which will prevent such usage of APIs.

To check, run:

This will fail the build on any module that currently uses an internal JDK API.



At the time of writing the isis-core-schema module fails the build.

Chapter 4. AsciiDoc Documentation

Apache Isis' documentation (meaning the website and the users' guide, the reference guide and this contributors' guide) is written using Asciidoc, specifically the Asciidoctor implementation.

The website and guides are created by running build tools (documented below) which create the HTML version of the site and guides. You can therefore easily check the documentation before raising a pull request (as a contributor) or publishing the site (if a committer).

Publishing is performed by copying the generated HTML to a different git repository (isis-site). This is synced by ASF infrastructure over to isis.apache.org.

And to help write the Asciidoc text itself, we provide some templates.

4.1. Where to find the Docs

The (Asciidoc) source code can be found at adocs/documentation (relative to root). Online you'll find it cloned to github here.

4.2. Naming Conventions

For documents with inclusions, use '_' to separate out the logical hierarchy:

xxx-xxx/xxx-xxx.adoc _xxx-xxx_ppp-ppp.adoc _xxx-xxx_qqq-qqq.adoc _xxx-xxx_qqq-qqq_mmm-mmm.adoc _xxx-xxx_qqq-qqq_nnn-nnn.adoc

Any referenced images should be in subdirectories of the *images* directory:

```
xxx-xxx/images/.
/ppp-ppp/.
/qqq-qqq/.
/mmm-mmm
/nnn-nnn
```

And similarly any resources should be in the resources subdirectory:

```
xxx-xxx/resources/.
ppp-ppp/.
qqq-qqq/.
/mmm-mmm/
/nnn-nnn/
```

4.3. Writing the docs

We highly recommend that you install the (IntelliJ) live templates for Asciidoctor, as described in IDE templates. These provide a large number of helper templates.

An appendix lists all the templates available, demonstrating their intended usage and output.

4.4. Build and Review (using Maven)

To (re)build the documentation locally prior to release, change into the adocs/documentation directory and use:

mvn clean compile

The site will be generated at target/site/index.html.

You could then use a web server such as Python's SimpleHTTPServer to preview (so that all Javascript works correctly). However, instead we recommend using instant preview, described next.

4.5. Instant Rebuild (using Ruby)

The ruby script, monitor.rb emulates the mvn compile command, regenerating any changed Asciidoctor files to the relevant target/site directory. Moreover if any included files are changed then it rebuilds the parent (per the above naming convention).

4.5.1. One-time setup

To setup:

- download and install ruby 2.0.0, from http://rubyinstaller.org/downloads/
- download devkit for the Ruby 2.0 installation, also from http://rubyinstaller.org/downloads/. Then follow the installation instructions on their wiki



We use Ruby 2.0 rather than 2.1 because the wdm gem (required to monitor the filesystem if running on Windows) is not currently compatible with Ruby 2.1.

To download the required Ruby dependencies, use:

gem install bundler bundle install

4.5.2. Instant Rebuild

To run, we typically just use:

This script simply runs mvn clean compile && ruby monitor.rb -b. The mvn command performs a clean rebuild of the site, and then the ruby script monitors for any further changes under src/main/asciidoc.

The script also starts up a web server on port 4000 so you can review results. If any .adoc changes, then the appropriate HTML will be regenerated. And, if any new assets (CSS, images etc) are added, they will be copied across. The -b flag passed through means that the script also starts a web browser pointing at the newly generated docs.

The monitor.rb script has a couple of other options, use -h flag for usage:

```
ruby monitor.rb -h
```

which should print:

```
usage: monitor.rb [options]

-p, --port port (default: 4000)

-b, --browser launch browser

-h, --help help
```

4.6. Publish procedure

Only Apache Isis committers can publish to isis.apache.org. We've decided to include these procedures here here (rather than put them in the Committers' Guide), just to keep things together.

4.6.1. One-time setup

The generated site is published by copying into the content/ directory of the isis-site git repo. You therefore need to check this out this repo.

The procedure assumes that two git repos (for isis itself and for isis-site) are checked out into the same parent directory, eg:

```
/APACHE/

isis/ # checkout of isis.git

adocs/

documentation/

README.adoc # this file you are reading right now

...

isis-site/ # checkout of isis-site.git

content/ # the published website
```

If this isn't the case, then it is possible to override the relative directory by passing in a system

property to the mvn goal; see below.

You also need to know that ASF's publishing script work from the 'asf-site' branch, NOT from the 'master' branch. Therefore, in the isis.git repo site:

git checkout asf-site

4.6.2. Publishing (full build)

Back in the adocs/documentation directory of the main isis-git.repo, to copy the generated documents to the isis-site.git repo, run:

mvn clean package

This deletes the entire content of contents, and replaces with the content under target/site. It also fixes up line endings, standardizing on unix-style LFs.



If you have checked out the isis-site.git repo into some other directory (relative to isis.site.git), then this can be overridden by specifying `-Disis-site.dir=… when calling mvn.

To copy and to also commit the generated documents to the isis-site.git repo, run:

sh publish.sh "ISIS-nnnn: a custom commit message"

Behind the scenes this just calls mvn clean install -Dmessage=....

It's also possible to omit the message, in which case publish.sh will reuse the most recent commit message from the main isis.git repo.

Pushing the commits (in the isis-site.git directory, of course) will publishing the changes:

git push

Double check at isis.apache.org.

4.6.3. Publishing (partial build)

If none of the guides have been changed, and if you have run the full rebuild recently, then you can skip the generation of PDFs using:

mvn install -Dskip.pdf -D"ISIS-nnnn: a custom commit message"

The clean goal must not be included though (else all the guides will disappear from the site

content).

Chapter 5. Contributing

This page explains how you can contribute to Apache Isis. You'll probably also want set up your IDE and learn how to build Apache Isis.

Thanks for considering to help out, your contributions are appreciated!

5.1. Recommended Workflow (github)

Apache Isis' source code is hosted in an Apache git repo (https, http), with a clone on github (https, or ssh: git@github.com:apache/isis.git.

As you might imagine, only committers are permitted to push changes to the central git repo. As a contributor, we recommend that you fork the apache/isis repo in github, and then use your fork as a way of publishing your patches for the Apache Isis committers to apply.

The diagram below illustrates the process:



That is:

- 1. as a one-time activity, you fork the github.com/apache/isis repo into your own fork on github.com
- 2. as a one-time activity, you clone your fork to your local computer
- 3. you set the github.com/apache/isis as your upstream branch; this will allow you to keep your local clone up-to-date with new commits

- note the asymmetry here: the upstream repo (the Apache github repo) is **not** the same as the origin repo (your fork).
- 4. you work on your changes locally; when done, you push them to your github fork
- 5. to contribute back a change, raise a JIRA ticket, and ensure your commit message is in the form: ISIS-nnnn: ... so that changes can be tracked (more discussion on this point below). In any case, before you decide to start hacking with Apache Isis, it's always worth creating a ticket in JIRA and then have a discussion about it on the mailing lists.
- 6. Use github to raise a pull request for your feature
- 7. An Apache Isis committer will review your change, and apply it if suitable.

5.2. Alternative Workflow (JIRA patches)

As an alternative, you may decide to clone directly from <u>github.com/apache/isis</u> rather than create your own fork:



In this case your upstream repo is the same as your origin repo, which might seem more straightforward. On the other hand, if you go this route then you'll need create patches locally and attach them to the JIRA ticket.

For the Apache Isis committers it really doesn't matter which route you take, so go with whatever's most comfortable.

5.3. Setting up your fork/clone

If you choose to create your own fork then you'll need an account on <u>github.com</u>. You then fork simply by pressing the "Fork" button:

PUBLIC	apache / isis mirrored from git://git.apache.org/isis.	git	(C Watch	• 1	🖈 Star	0	ဖို Fork	3
	Code	Network	Pull Requests	0			Grap	hs	

An account isn't needed if you just clone straight from the github.com/apache/isis.

Whether you've forked or not, you then need to clone the repo onto your computer. Github makes this very easy to do:

- for Windows users, we suggest you use github's 'Clone in Windows' feature
- for Mac/Linux users, create a clone from the command line:

Again, the info is easily found in the github page:

Code		Network	Pull Requests 0	
Mirror of Apache Isis				
	\sim $-$			_

If you've created your own fork, then you need to add the upstream remote to the github.com/apache/isis. This remote is traditionally called upstream. You should then arrange for your master branch to track the upstream/master remote branch:

If you didn't create your own fork, you can omit the above step. Either way around, you can now fetch new commits using simply:

```
git fetch
```

For more info on tracking branches here and here.

5.4. Commit messages

Although with git your commits are always performed on your local repo, those commit messages become public when the patch is applied by an Apache Isis committer. You should take time to write a meaningful commit message that helps explain what the patch refers to; if you don't then there's a chance that your patch may be rejected and not applied. No-one likes hard work to go to waste!

ISIS-999: Make the example in CONTRIBUTING imperative and concrete

Without this patch applied the example commit message in the CONTRIBUTING document is not a concrete example. This is a problem because the contributor is left to imagine what the commit message should look like based on a description rather than an example. This patch fixes the problem by making the example concrete and imperative.

The first line is a real life imperative statement with a ticket number from our issue tracker. The body describes the behavior without the patch, why this is a problem, and how the patch fixes the problem when applied.

Once your git repo is setup, the next thing you'll most likely want to do is to setup your development environment. See here for more details.

5.5. Creating the patch file

If you are working without a github fork of Apache Isis, then you can create the patches from your own local git repository.

As per this stackoverflow question, create the patch using git format-patch:

git format-patch -10 HEAD --stdout > 0001-last-10-commits.patch

Here -10 is the last 10 commits you have done. You need to change that integer according to the commits you need to apply into the patch.

5.6. Sample Contribution Workflow

Assuming you're development environment is all setup, let's walk through how you might make contribute a patch. In this example, suppose that you've decided to work on JIRA ticket #123, an enhancement to support Blob/Clob datatypes.

5.6.1. Update your master branch

The first thing to do is to make sure your local clone is up-to-date. We do this by retrieving new commits from upstream repo and then merging them as a fast-forward into your local branch.

Irrespective of whether you are using a github fork, the upstream for your local master branch will be tracking the appropriate remote's master branch. So n either case, the same commands work:

Alternatively, you can combine the git fetch and git merge and just use git pull: git checkout master git pull -ff-only

If the merge or pull fails, it means that you must have made commits and there have been changes

meanwhile on the remote master's branch. You can use `gitk --all to confirm. If this fails, see our git cookbook page for a procedure to retrospectively sort out this situation.

5.6.2. Create a topic branch

We recommend you name topic branches by the JIRA ticket, ie <tt>ISIS-nnn-description</tt>. So let's create a new branch based off master and call it "ISIS-123-blobs"

You can confirm the branch is there and is your new HEAD using either gitk --all. Alternatively, use the command line:

```
$ git checkout -b ISIS-123-blobs
```

The command line prompt should also indicate you are on a branch, isolated from any changes that might happen on the master branch.

5.6.3. Make File Changes and Commit

Next, make changes to your files using the usual commands (see also our git cookbook section):

- git add
- git mv
- git rm
- git commit
- git status

and so on.

Continue this way until happy with the change. Remember to run all your tests on the topic branch (including a full mvn clean install).

5.6.4. Rebasing with master

Before you can share your change, you should rebase (in other words replay) your changes on top of the master branch.

The first thing to do is to pull down any changes made in upstream remote's master since you started your topic branch:

These are the same commands that you would have run before you created your topic branch. If you use gitk --all, there's a good chance that new commits have come in.

Next, we reintegrate our topic branch by rebasing onto master: git checkout ISIS-123-blobs git rebase master

This takes all of the commits in your branch, and applies them on top of the new master branch. When your change is eventually integrated back in, it will result in a nice clear linear history on the public repo. If the rebase fails because of a conflict, then you'll be dumped into REBASE mode. Edit the file that has the conflict, and make the appropriate edits. Once done:

Once the rebase has completed, re-run your tests to confirm that everything is still good.

5.6.5. Raising a pull request

If you have your own fork, you can now simply push the changes you've made locally to your fork:

This will create a corresponding branch in the remote github repo. If you use gitk --all, you'll also see a remotes/origin/ISIS-123-blobs branch.

Then, use github to raise a <u>pull request</u>. Pull requests sent to the Apache GitHub repositories will forward a pull request e-mail to the <u>dev mailing list</u>. You'll probably want to sign up to the dev mailing list first before issuing your first pull request (though that isn't mandatory).

The process to raise the pull request, broadly speaking:

- Open a web browser to your github fork of isis
- Select your topic branch (pushed in the previous step) so that the pull request references the topic branch.
- Click the Pull Request button.
- Check that the Apache Isis mailing list email came through.

5.7. If your pull request is accepted

To double check that your pull request is accepted, update your master branch from the upstream remote:

You can then use gitk --all (or git log if you prefer the command line) to check your contribution has been added.

You can now delete your topic branch and remove the branch in your github:

Finally, you might want to push the latest changes in master back up to your github fork. If so, use:

5.7.1. If your pull request is rejected

If your pull request is rejected, then you'll need to update your branch from the main repository and then address the rejection reason.

You'll probably also want to remove the remote branch on github:

git push origin -delete ISIS-123-blobs

... and continue as before until you are ready to resubmit your change.

[1] inspiration for the recommended commit format comes from the puppet project's contributing

Chapter 6. Appendix: Git Cookbook

This appendix describes the commands often used while working with git. In addition to these basic commands, please make sure you have read:

- building Apache Isis
- Contributing
- Git policy

6.1. Modifying existing files

To modify existing files:

```
git add filename
git commit -m <mark>"ISIS-nnn: yada yada</mark>"
```

The git add command adds the changes to the file(s) to the git index (aka staging area). If you were to make subsequent changes to the file these would not be committed.

The git commit takes all the staged changes and commits them locally. Note that these changes are not shared public with Apache Isis' central git repo.

You can combine these two commands using -am flag to git commit:

```
git commit -am "ISIS-nnn: yada yada"
```

6.2. Adding new files

To add a new file:

```
git add .
git commit -m "ISIS-nnn: yada yada"
```

Note that this sequence of commands is identical to modifying an existing file. However, it isn't possible to combine the two steps using git commit -am; the git add is always needed when adding new files to the repo.

6.3. Deleting files

To delete a file:

```
git rm filename
git commit -m "ISIS-nnn: yada yada"
```

6.4. Renaming or moving files

To rename or move a file:

```
git mv <i>filename</i> <i>newfilename</i>
git commit -m "ISIS-nnn: yada yada"
```

6.5. Common Workflows

The contributing page describes the workflow for non-committers. The Git policy page describes a workflow for Apache Isis **committers**.

6.6. Backing up a local branch

If committing to a local branch, the changes are still just that: local, and run risk of a disk failure or other disaster.

To create a new, similarly named branch on the central repo, use:

git push -u origin <i>branchname</i>

Using gitk --all will show you this new branch, named origin/branchname.

Thereafter, you can push subsequent commits using simply:

git push

Doing this also allows others to collaborate on this branch, just as they would for master.

When, eventually, you have reintegrated this branch, you can delete the remote branch using:

git push origin --delete <i>branchname</i>

For more detail, see these blogs/posts here and here.

6.7. Quick change: stashing changes

If you are working on something but are not ready to commit, then use:

If you use gitk --all then you'll see new commits are made that hold the current state of your working directory and staging area.

You can then, for example, pull down the latest changes using git pull --rebase (see above).

To reapply your stash, then use:

```
git stash pop
```

Note that stashing works even if switching branches

6.8. Ignoring files

Put file patterns into .gitignore. There is one at the root of the git repo, but they can additionally appear in subdirectories (the results are cumulative).

See also:

- github's help page
- man page

6.9. More advanced use cases

6.9.1. If accidentally push to remote

Suppose you committed to master, and then pushed the change, and then decided that you didn't intend to do that:

C1 - C2 - C3 - C4 - C5 - C6 - C7 ^ master ^ origin/master

To go back to an earlier commit, first we wind back the local master:

git reset --hard C5

where C5 is the long sha-id for that commit.

This gets us to:

C1 - C2 - C3 - C4 - C5 - C6 - C7 ^ master ^ origin/master

Then, do a force push:

```
git push origin master --force
```

If this doesn't work, it may be that the remote repo has disabled this feature. There are other hacks to get around this, see for example here.

6.10. If you've accidentally worked on master branch

If at any time the git pull from your upstream fails, it most likely means that you must have made commits on the master branch. You can use gitk --all to confirm; at some point in time both master and origin\master will have a common ancestor.

You can retrospectively create a topic branch for the work you've accidentally done on master.

First, create a branch for your current commit:

```
git branch <i>newbranch</i>
```

Next, make sure you have no outstanding edits. If you do, you should commit them or stash them:

```
git stash
```

Finally, locate the shald of the commit you want to roll back to (easily obtained in gitk -all), and wind master branch back to that commit:

```
git checkout master
git reset --hard <i>shaId</i>  # move master branch shaId of common ancestor
```

6.11. If you've forgotten to prefix your commits (but not pushed)

One of our committers, Alexander Krasnukhin, has put together some git scripts to help his workflow. Using one of these, git prefix, you can just commit with proper message without bothering about prefix and add prefix only in the end **before** the final push.

For example, to prefix all not yet prefixed commits master..isis/666 with ISIS-666 prefix, use:

git prefix ISIS-666 master..isis/666

You can grab this utility, and others, from this repo.

Chapter 7. Appendix: Asciidoc Templates

This appendix lists the (IntelliJ) live templates available for writing documentation using Asciidoc. Instructions for installing the templates can be found here.

In the examples below the text xxx, yyy, zzz are correspond to template variables (ie placeholders).

7.1. Callouts

The Asciidoctor terminology is an "admonition".

Abbrev.	Produces	Example
adadmimporta nt	[IMPORTANT] ==== xxx ====	[IMPORTANT] ==== xxx ====
adadmnote	[NOTE] ==== xxx ====	[NOTE] ==== xxx ====
adadmtip	[TIP] ==== xxx ====	[TIP] ==== xxx ====
adadmwarning	[WARNING] ==== xxx ====	[WARNING] ==== xxx ====

7.2. TODO notes

Add as a placeholder for documentation still to be written or which is work-in-progress.

Abbrev.	Produces	Example
adtodo	<pre>NOTE: TODO</pre>	NOTE: TODO
adwip	<pre>NOTE: WIP - xxx</pre> where: * <code>xxx</code> is additional explanatory text	NOTE: WIP - cool new feature

7.3. Xref to Guides

Cross-references (links) to the various guides

Abbrev.	Produces	Example
adcgcom	<pre>xref:cgcom.adoc#xxx[ttt]</pre> a hyperlink to a bookmark within the committers' guide, where: * <code>xxx</code> is the bookmark’s anchor * <code>ttt</code> is the text to display as the hyperlink for example: <pre>xref:dg.adoc#_cgcom_cutting-a- release[Cutting a release]</pre>	addg
<pre><pre><ref:dg .adoc#xxx[ttt=""]<="" pre=""> a hyperlink to a bookmark within the developers' guide, where: * <code>xxx</code> is the bookmark&# 8217;s anchor * <code>ttt is the text to display as the hyperlink for example: <pre>xref:dg .adoc#_dg_as ciidoc- templates[As ciidoc templates]</pre></code></ref:dg></pre></pre>	Asciidoc templates	adrgant

Abbrev.	Produces	Example
<pre>xref:rg</pre>	Core annotations	adrgcfg
ant.adoc#xx		
x[ttt] a		
hyperlink to		
a bookmark		
within the		
reference		
guide for		
annotations,		
where: *		
<code>xxx<!--</td--><td></td><td></td></code>		
code> is the		
bookmark *		
<code>ttt<td></td><td></td></code>		
de> is the		
text to		
display as		
the		
hyperlink		
for example:		
<pre>xrei:rg </pre>		
ant.adoc#_rg		
ant_aaa_mal		
nicore		

Abbrev.	Produces	Example
<pre>xref:rg</pre>	Configuring Core	adrgcms
cfg.adoc#xxx		
[ttt] a		
hyperlink to		
a bookmark		
within the		
reference		
guide for		
configuratio		
n properties		
guide,		
where: *		
<code>xxx<!--</td--><td></td><td></td></code>		
code> is the		
bookmark *		
<code>ttt<td></td><td></td></code>		
de> is the		
text to		
display as		
the		
hyperlink		
for example:		
<pre>xref:rg</pre>		
cfg.adoc#_rg		
cfg_configuri		
ng-		
core[Configu		
ring		
Core]		

Abbrev.	Produces	Example
<pre>xref:rg</pre>	AbstractService	adrgsvc
cms.adoc#xx		
x[ttt] a		
hyperlink to		
a bookmark		
within the		
reference		
guide for		
classes,		
methods and		
schema,		
where: *		
<code>xxx<!--</td--><td></td><td></td></code>		
code> is the		
bookmark *		
<code>ttt<td></td><td></td></code>		
de> is the		
text to		
display as		
the		
hyperlink		
for example:		
<pre>xref:rg</pre>		
cms.adoc#_r		
gcms_classes		
_super_Abstr		
actService		
AbstractServ		
ice J		

Abbrev.	Produces	Example
<pre>xref:rg</pre>	AppManifest bootstrapping	adrgmvn
svc.adoc#xxx		
[ttt] a		
hyperlink to		
a bookmark		
within the		
reference		
guide for		
domain		
services,		
where: *		
<code>xxx<!--</td--><td></td><td></td></code>		
code> is the		
bookmark *		
<code>ttt<td></td><td></td></code>		
de> is the		
text to		
display as		
the		
hyperlink		
for example:		
<pre>xref:rg</pre>		
svc.adoc#_rg		
cms_classes_		
AppManifest		
-bootstrappi		
ng[`AppMan		
ifest`		
bootstrappin		
g]		

Abbrev.	Produces	Example
<pre>xref:rg mvn.adoc#x xx[ttt]</pre> a hyperlink to a bookmark within the	validate goal	adrgna
reference		
guide for the		
nlugin		
where: *		
<code>xxx<!--</td--><td></td><td></td></code>		
code> is the		
bookmark *		
<code>ttt<td></td><td></td></code>		
de> is the		
text to		
display as		
the		
hyperlink		
for example:		
<pre>xref:rg</pre>		
mvn.adoc#_r		
gmvn_valida		
telvalidate		
goaij		
Abbrev.	Produces	Example
---	----------	---------
<pre>xref:rg</pre>	@Action	adrgnt
ant.adoc#_rg		
ant-		
xxx[`@xxx`		
]		
a hyperlink		
to the "man		
page" for an		
annotation		
within the		
reference		
guide for		
annotations,		
where: *		
<code>xxx<!--</td--><td></td><td></td></code>		
code> is the		
annotation		
type (eg		
<code>@Acti</code>		
on)		
for example:		
<pre>xref:rg</pre>		
ant.adoc#_rg		
ant-		
Action[`@Ac		
tion`]		

Abbrev.	Produces	Example
<pre>xref:rg</pre>	<pre>@Action#semantics()</pre>	adrgsa
ant.adoc#_rg		
ant-		
xxx_ttt[`@xx		
x#ttt()` <td></td> <td></td>		
re>] a		
hyperlink to		
the "man		
page" for the		
specific		
attribute		
(field) of an		
annotation		
within the		
reference		
guide for		
annotations,		
where: *		
<code>xxx<!--</td--><td></td><td></td></code>		
code> is the		
annotation		
type (eg		
<code>@Acti</code>		
on) *		
<code>ttt<td></td><td></td></code>		
de> is the		
attribute (eg		
<code>@sem</code>		
antics <td></td> <td></td>		
>) for		
example:		
<pre>xref:rg</pre>		
ant.adoc#_rg		
ant-		
Action_sema		
ntics[`@Acti		
on#semantic		
s()`]		

Abbrev.	Produces	Example
<pre></pre>	DomainObjectContainer	adrgss
a hyperlink		
to the "man		
page" for an		
(API) domain		
service		
within the		
reference		
guide for		
domain		
services,		
where: *		
<code>xxx<!--</td--><td></td><td></td></code>		
code> is the		
domain		
service (eg		
<code>Doma</code>		
inObjectCont		
ainer		
) for		
example:		
<pre>xref:rg</pre>		
svc.adoc#_rg		
svc_api_Dom		
ainObjectCo		
ntainer[`Do		
mainObjectC		
ontainer`] </td <td></td> <td></td>		
pre>		

Abbrev.	Produces	Example
<pre></pre>	ContentMappingService	adugfun
a hyperlink		
to the "man		
page" for an		
(SPI) domain		
service		
within the		
reference		
guide for		
domain		
services,		
where: *		
<code>xxx<!--</td--><td></td><td></td></code>		
code> is the		
domain		
service (eg		
<code>Conte</code>		
ntMappingSe		
rvice		
) for		
example:		
<pre>xref:rg</pre>		
svc.adoc#_rg		
svc_spi_Cont		
entMappingS		
ervice[`Cont		
entMappingS		
ervice `J <td></td> <td></td>		
e>		

Abbrev.	Produces	Example
<pre>xref:ug</pre>	Core concepts	adugvw
fun.adoc#xx		
x[ttt] a		
hyperlink to		
a bookmark		
within the		
fundamental		
s users'		
guide,		
where: *		
<code>xxx<!--</td--><td></td><td></td></code>		
code> is the		
bookmark&#		
8217;s		
anchor *		
<code>ttt<td></td><td></td></code>		
de> is the		
text to		
display as		
the		
hyperlink		
for example:		
<pre>xref:ug</pre>		
fun.adoc#_u		
gfun_core-		
concepts[Cor		
е		
concepts] <td></td> <td></td>		
re>		

Abbrev.	Produces	Example
<pre>xref:ug</pre>	Customisation	adugvro
vw.adoc#xxx		
[ttt] A		
hyperlink to		
a bookmark		
within the		
Wicket		
viewer		
guide,		
where: *		
<code>xxx<!--</td--><td></td><td></td></code>		
code> is the		
bookmark&#		
8217;s		
anchor *		
<code>ttt<td></td><td></td></code>		
de> is the		
text to		
display as		
the		
hyperlink.		
for example:		
<pre>xref:ug</pre>		
vw.adoc#_ug		
vw_customis		
ation[Custo		
misation] <td></td> <td></td>		
re>		

Abbrev.	Produces	Example
<pre>xref:ug</pre>	RestfulObjects specification	adugsec
vro.adoc#xx		
x[ttt]		
A hyperlink		
to a		
bookmark		
within the		
Restful		
Objects		
viewer		
guide,		
where: *		
<code>xxx<!--</td--><td></td><td></td></code>		
code> is the		
bookmark&#		
8217;s		
anchor *		
<code>ttt<td></td><td></td></code>		
de> is the		
text to		
display as		
the		
hyperlink.		
for example:		
<pre>xref:ug</pre>		
vro.adoc#_ug		
vro_ro-		
spec[Restful		
Objects		
specification		
J		

Abbrev.	Produces	Example
<pre>xref:ug</pre>	Caching and other Shiro Features	adugtst
sec.adoc#xxx		
[ttt] A		
hyperlink to		
a bookmark		
within the		
Secrurity		
guide,		
where: *		
<code>xxx<!--</td--><td></td><td></td></code>		
code> is the		
bookmark&#		
8217;s		
anchor *		
<code>ttt<td></td><td></td></code>		
de> is the		
text to		
display as		
the		
hyperlink.		
for example:		
<pre>xref:ug</pre>		
sec.adoc#_ug		
sec_shiro-		
caching[Cac		
hing and		
other Shiro		
Features] <td></td> <td></td>		
re>		

Abbrev.	Produces	Example
<pre>xref:ug</pre>	BDD Spec Support	adugbtb
tst.adoc#xxx[
ttt] A		
hyperlink to		
a bookmark		
within the		
Testing		
guide,		
where: *		
<code>xxx<!--</td--><td></td><td></td></code>		
code> is the		
bookmark&#		
8217;s		
anchor *		
<code>ttt<td></td><td></td></code>		
de> is the		
text to		
display as		
the		
hyperlink.		
for example:		
<pre>xref:ug</pre>		
tst.adoc#_ugt		
st_bdd-spec-		
support[BDD		
Spec		
Support] <td></td> <td></td>		
e>		

7.4. Link to Isis Addons

Links to (non-ASF) Isis Addons

Abbrev.	Produces	Example
adlinkaddons	<pre>(non-ASF) link:http://isisaddons.org[Isis Addons]</pre> link to the Isis Addons website.	(non-ASF) Isis Addons
adlinkaddons app	<pre>(non-ASF) http://github.com/isisaddons/isis-app- xxx[Isis addons' xxx</pre>] link to the github repo for an example app from the Isis addons; where: * <code>xxx</code> is the name of the example app being linked to for example: <pre>(non-ASF) http://github.com/isisaddons/isis-app-todoapp[Isis addons' todoapp]</pre>	(non-ASF) Isis addons' todoapp

Abbrev.	Produces	Example
adlinkaddons module	<pre></pre> link to the github repo for a module from the Isis addons; where: * <code>xxx</code> is the name of the module being linked to for example: <pre>(non- ASF) http://github.com/isisaddons/isis-module- security[Isis addons' security] module</pre>	(non-ASF) Isis addons' security module
adlinkaddons wicket	<pre></pre> link to the github repo for a wicket UI component from the Isis addons; where: * <code>xxx</code> is the name of the wicket UI component being linked to for example: <pre>(non- ASF) http://github.com/isisaddons/isis-wicket-gmap3[Isis addons' gmap3] wicket extension</pre>	(non-ASF) Isis addons' gmap3 wicket extension

7.5. Source code

Abbrev.	Produces	Example
adsrcjava	[source, java] xxx where: * xxx is the source code snippet.	[source,java] public class Foo { }
adsrcjavac	as for adsrcjava, but with a caption above	
adsrcjavascr ipt	<pre>[source,javascript] xxx where: * xxx is the source code snippet.</pre>	[source,javascript] \$(document).ready(functio n() { });
adsrcjavascr iptc	as for adsrcjavascript, but with a caption above	
adsrcother	<pre>[source,nnn] xxx where: * nnn is the programming language * xxx is the source code snippet.</pre>	
adsrcotherc	as for adsrcother, but with a caption above	
adsrcxml	<pre>[source,javascript] xxx where: * xxx is the source code snippet.</pre>	[source,xml] <html> <title> hello world! </title> </html>
adsrcxmlc	as for adsrcxml, but with a caption above	

7.6. Images

Abbrev.	Produces	Example
adimgfile	<pre>image::{_imagesdir}xxx/yyy.png[width="WWWpx ",link="{_imagesdir}xxx/yyy.png"]</pre> embeds specified image, where: * <code>xxx</code> is the subdirectory under the <code>images/</code> directory * <code>yyy</code> is the image * <code>WWW</code> is the width, in pixels. for example: <pre>image::{_imagesdir}wicket-viewer/layouts/estatio- Lease.png[width="300px",link="{_imagesdir}wicket- viewer/layouts/estatio-Lease.png"]</pre>	image::images/wicket- viewer/layouts/estatio- Lease.png[width="300px",li nk="images/wicket- viewer/layouts/estatio- Lease.png"]
adimgfilec	as for adimgfile, but with a caption above	
adimgurl	<pre>image::xxx[width="WWWpx",link="xxx"]</pre> embeds image from specified URL, where: * <code>xxx</code> is the URL to the image * <code>WWW</code> is the width, in pixels.	
adimgurlc	as for adimgurl, but with a caption above	

7.7. YouTube (screencasts)

Embedded youtube screencasts. (Don't use these in guides, as they cannot be rendered as PDF).

Abbrev.	Produces	Example
adyoutube	<pre>video::xxx[youtube,width="WWWpx",height="HH Hpx"]</pre> where: * <code>xxx</code> is the youtube reference * <code>WWW</code> is the width, in pixels * <code>HHH</code> is the height, in pixels for example: <pre>video::bj8735nBRR4[youtube,width="210px",heig ht="118px"] </pre>	video::bj8735nBRR4[youtu be,width="210px",height=" 118px"]
adyoutubec	as for youtube, but with a caption above	

7.8. Tables

Abbrev.	Produces	Example
adtb13	Table with 3 columns, 3 rows.	

7.9. Misc.

Abbrev.	Produces	Example
adai	<pre>Apache Isis</pre> That is, the literal text "Apache Isis".	Apache Isis

Abbrev.	Produces	Example
adlink	<pre>link:xxx[ttt]</pre> , where: * <code>xxx</code> is * <code>ttt</code> is the text to display as the hyperlink for example: <pre>link:http://isis.apache.org[Apache Isis website]</pre>	Apache Isis website
adanchany	<pre>= anchor:[xxx]</pre> defines an inline anchor to any heading, where: * <code>xxx</code> is the anchor text. For example: <pre>= anchor:[_ugfun_i18n] Internationalization</pre> An alternative (more commonly used in our documentation) is to use the <code>[[…​]]</code> directly above the heading: <pre>[[_ugfun_i18n]] = Internationalization</pre>	
adxrefany	<pre>xref:[xxx]</pre> cross-reference to any document/anchor, where: * <code>xxx</code> is the fully qualified document with optional anchor	
adfootnote	<pre>.footnote:[]</pre> defines a footnote	. [1: this is a footnote]

Chapter 8. Appendix: Project Lombok

Project Lombok is an open source project to reduce the amount of boilerplate in your code.

For example, rather than write:

```
private String name;
public String getName() {
    return name;
}
public void setName(String name) {
    this.name = name;
}
```

you can instead write simply:

@Getter @Setter **private** String name;

Under the covers it is implemented as an annotation processor; it basically hooks into the Java compiler so that it can emit additional bytecode (eg for the getter and setter). See here for details of setting up in IntelliJ (Eclipse has very similar support).

Apache Isis supports Project Lombok, in that the annotations that would normally be placed on the getter (namely Property, @PropertyLayout, @Collection, @CollectionLayout and @MemberOrder) can be placed on the field instead.

There are plugins for Lombok for maven; it's just a matter of adding the required dependency. To compile the code within your IDE (eg so that its compiler "knows" that there is, actually, a getter and setter) will require an Lombok plugin appropriate to that IDE. See the Lombok download page for more information.

8.1. Future thoughts

In the future we might extend/fork Lombok so that it understands Isis' own annotations (ie @Property and @Collection) rather than Lombok's own @Getter and `@Setter.

It might also be possible to use Lombok to generate the domain event classes for each member.

Chapter 9. Appendix: AgileJ



This material does not constitute an endorsement; AgileJ Structure Views is not affiliated to Apache Software Foundation in any way. AgileJ has however provided a complimentary copy of its software to Apache Isis committers.

AgileJ Structure Views is a commercial product to reverse engineer and visualize Java classes from source code.

The key to using the tool is in developing a suitable filter script, a DSL. You can use the following script as a starting point for visualizing Apache Isis domain models:

// use CTRL+SPACE for completion suggestions hide all fields hide setter methods hide private methods hide methods named compareTo hide methods named toString hide methods named inject* hide methods named disable* hide methods named default* hide methods named hide* hide methods named autoComplete* hide methods named choices* hide methods named title hide methods named iconName hide methods named validate* hide methods named modify* hide protected methods hide types annotated as DomainService hide types named Constants hide types named InvoicingInterval hide enums hide constructors hide inner types named *Event hide inner types named *Functions hide inner types named *Predicates show getter methods in green show methods annotated as Programmatic in orange show methods annotated as Action in largest hide dependency lines hide call lines hide method lines

For more information on AgileJ, see Paul Wells' 8-part tutorial series on Youtube; the first can be found here (view the "show more" comments to click through to other parts).