CVS Repositories, Branching and TAGS

CVS no more

We used to use CVS to manage our source code. See Git

Please do NOT check in any files under the build directory in our projects e.g. build/report, build/classes or build/deploy. In general, any files that are generated as part of the normal build process are not checked into CVS.

The exceptions to the "generated files not checked in" rule are configuration files such as the hibernate files (generated by XDoclet or Hibernate), web.xml, struts-config.xml (generated by XDoclet).

See Building LAMS for information on accessing the repositories.

Working on a Branch

When you are working with a LAMS branch for a while, the easiest is probably to run a different workspace. Within Eclipse, do "File, Switch Workspace" then do "Browse". From the directory listing, create a new directory such as Eclipse<Branchname>Workspace. Switch to this directory and then check out all LAMS projects, using the appropriate branch and/or tag.

You will need to set up the Shared Libraries again - they are configured on a workspace level, rather than Eclipse wide.

When you check in, the code will automatically be checked back into the branch. Please see the details below on each branch for whether or not you should merge the changes.

Then when you need to switch back to the head to work, you can just change workspaces.

Switching between tag/branches

If you are just checking a file on a branch, or merging, it is easy to switch branches/tag. In Eclipse, right mouse click on the project name, then select "Team", "Switch to another Branch or Version" then select the appropriate version.

When you switch between branches or tags, you can do more than one project at a time. But always do lams_central and jsmath by themselves - the projects are so large you often get CVS timeout errors while you are doing it - and you don't always get a decent error message when switching tags so you could end up with half a project changed.

It helps if you make the following changes in Eclipse. In preferences, find Team, CVS.

- Increase the connection timeout to at least 360 (on the Connection tab in Team, CVS)
- Turn on "Show CVS console automatically when command run" on the Team, CVS, Console entry.

It pays to do a synchronise first, so that you get the CVS console to appear. You may have to synchronise a couple of projects to make it work. You need to get the console with entries like:

```
cvs update \-C \-d "windows.properties" "liblist.txt" "build.xml" "conf/j2ee/application.xml" "lib/lams/lams.
jar"       U windows.properties       U liblist.txt       U build.xml       U conf/j2ee/application.xml       U
lib/lams/lams.jar ok (took 0:05.406)
```

Understanding Branching/Merge in Eclipse

Branching with Eclipse and CVS gives a good introduction to branching or merging. Just be careful of the jsmath and lam_central projects - they tend to time out before a tag or branch can be completed, and will leave part of the code not tagged/branched. It is best to do a Synchronize first and that should display the CVS log in the console window. Once you have the log showing, then do your tag/branch. Otherwise the log may not be shown and you will not be sure if the branch/tag worked.

If possible, create a branch on a tag. If you do not, Eclipse will create a root tag based on the branch name. This tag is needed for merging.

Branching and Flash Source

When you are working on a branch and a module's flash movie is required to be compiled following changes to the Flash AS (ActionScript) source files, be aware that the source files the Flash FLA is pointing to may not be correct.

Typically when working on a branch you will have a different workspace. The source files for the Flash FLA need to point to the source files in this workspace.

To change which source files the Flash FLA points to:

- Select File -> Publish Settings... from the menu.
- On the Flash tab click Settings next to the ActionScript version field.
- Adjust the classpath entries to point to the correct source files. Click OK.
- On the Format tab also adjust the Flash .swf export path.
- Click OK to close the dialog and your ready to publish.
  Also be aware when using an ActionScript editor that the source files you are editing are from the correct workspace.

The HEAD and Branches

HEAD

Usage: The "normal" development location. At present it is being used for LAMS 2.4+ development.

LAMS language files (from the I18N website) should be checked into the HEAD and to a branch before release.

Important Tags

- lams2_4: LAMS 2.4 release tag (13 April 2012) -**latest stable release**
- lams2_3_5: LAMS 2.3.5 release tag (28 December 2010)
- lams2_3_4: LAMS 2.3.4 release tag (05 March 2010)
- lams2_3_3: LAMS 2.3.3 release tag
- lams2_3: tag point for branch of same name
- lams2_3_spring_jboss_tune: tag point for branch of same name
- lams2_2_final: LAMS 2.2 release
- lams2_2_rc1: Initial tag for LAMS 2.2.
- lams2_1_1: LAMS 2.1.1 (14/8/08 chat fix + 5/9/08 language files)
- rams1_1: RAMS 1.1
- lams2_1: Final cut for LAMS 2.1.
- lams2_1_rc1: LAMS 2.1 RC1.
- lams2_1_alpha: LAMS 2.1 Prerelease version of LAMS 2.1. Tag taken when demonstration box built for the LAMS conference.
- lams2_0_4: LAMS 2.0.3 release (21 June 2007) Minor release required to fix bug in Windows upgrader.
- lams2_0_3: LAMS 2.0.3 release (1 June 2007)
- lams2_0_2: LAMS 2.0.2 release (19 March 2007)
- lams2_0_1: LAMS 2.0.1 release.
- lams2_0_stable: Final cut of LAMS 2.0. Includes changes to lams_central, lams_common, lams_tool_mcq.
- lams2_0: Initial cut of LAMS 2.0. was replaced by lams2_0_stable.

Branch lams2_3_release

Usage: LAMS 2.3 development, while new features for 2.4 continue on head.

Projects branched:

```
lams_admin              lams_learning     lams_tool_gmap    lams_tool_notebook     lams_tool_vote
lams_build              lams_monitoring   lams_tool_images  lams_tool_sbmt         lams_tool_wiki
lams_central            lams_tool_chat    lams_tool_lamc    lams_tool_scribe       lams_www
lams_common             lams_tool_daco    lams_tool_laqa    lams_tool_spreadsheet
lams_contentrepository  lams_tool_deploy  lams_tool_larsrc  lams_tool_survey
lams_gradebook          lams_tool_forum   lams_tool_nb      lams_tool_task

lams_tool_assessment
lams_tool_dimdim
lams_tool_mindmap
lams_tool_pixlr
```

Branch lams2_3_spring_jboss_tune

Usage: testing major rejig of spring configuration before merging into head.

Projects branched:

```
lams_admin              lams_monitoring       lams_tool_example  lams_tool_larsrc    lams_tool_spreadsheet
lams_www
lams_build              lams_tool_assessment  lams_tool_forum    lams_tool_nb        lams_tool_survey
lams_central            lams_tool_chat        lams_tool_gmap     lams_tool_notebook  lams_tool_task
lams_common             lams_tool_daco        lams_tool_images   lams_tool_pixlr     lams_tool_videorecorder
lams_contentrepository  lams_tool_deploy      lams_tool_lamc     lams_tool_sbmt      lams_tool_vote
lams_learning           lams_tool_dimdim      lams_tool_laqa     lams_tool_scribe    lams_tool_wiki
```

Branch lams2_2

Usage: Fixes and improvements made on HEAD will be backported to this branch in order to create minor version updates to 2.2 (This differs to the purpose of the historic branches below which were where 'new features' were developed with the intention of eventually merging back to HEAD). Branch point is the lams2_2_final tag.

Projects branched: lams_admin, lams_build, lams_central, lams_common, lams_contentrepository, lams_flash, lams_learning, lams_monitoring, lams_tool_chat, lams_tool_daco, lams_tool_deploy, lams_tool_dimdim, lams_tool_forum, lams_tool_gmap, lams_tool_images, lams_tool_lamc, lams_tool_laqa, lams_tool_larsrc, lams_tool_nb, lams_tool_notebook, lams_tool_sbmt, lams_tool_scribe, lams_tool_spreadsheet, lams_tool_survey, lams_tool_task, lams_tool_vote, lams_tool_wiki, lams_www

Historic Branches

The following branches are in the code base but are no longer used.

- editonfly_branching. Used to develop Edit On The Fly, now known as Live Edit. Functionality merged back to the head for the 2.0.3 release.
- lams2_0_fixes. Emergency releases based on the 2.0 codebase.

Branch ldap2_0_4

Usage: Development of sourcing the user details from an LDAP server and to integration WebCT. Done on a branch so we can release prior to 2.1. The LDAP server changes were released and tagges as lams2_0_5. Initial release to CFL for WebCT Integration tagged as lams2_0_6.

Projects branched: lams_admin (from lams2_0_3 tag), lams_build (from lams2_0_3 tag), lams_central (from lams2_0_3 tag), lams_common (from lams2_0_4 tag), lams_monitoring (from lams2_0_3 tag), lams_www (from lams2_0_4 tag).

For all other projects check out the lams2_0_3 tag.

Branch shibboleth_2_0

Usage: Development of the Shibboleth'd LAMS.

Projects branched: lams_admin, lams_build, lams_central, lams_common, lams_monitoring.
For all other projects, check out the lams2_0_3 tag.

Started at tag: Mostly lams2_0, merged with lams2_0_2 tag, merged with lams2_0_3 tag (7/6/07). lams_monitoring branched from lams2_0_3 tag (3/9/7) .

Merge strategy: All critical bugs on the head will be merged into the shibboleth_2_0 so that life isn't too hard for the shibboleth_2_0 developers! The shibboleth_2_0 changes will be merged back into the head at the end of each major stage of the project (e.g. getting LAMS to authenticate based on SAML). Why this strategy? Only a limited number of projects will be affected and we don't want any partial work to be rolled out in any emergency releases.

Branch lams2_0_3_fixes

Usage: To do a 2.0.4 release which fixes an error in the update scripts for 2.0.3

Projects branched: lams_www, lams_common.
For all other projects, check out the lams2_0_3 tag.

Started at tag: lams2_0_3

Merge strategy: None

How To Merge

Merging must be done project by project - it cannot be done on the whole codebase in one go.

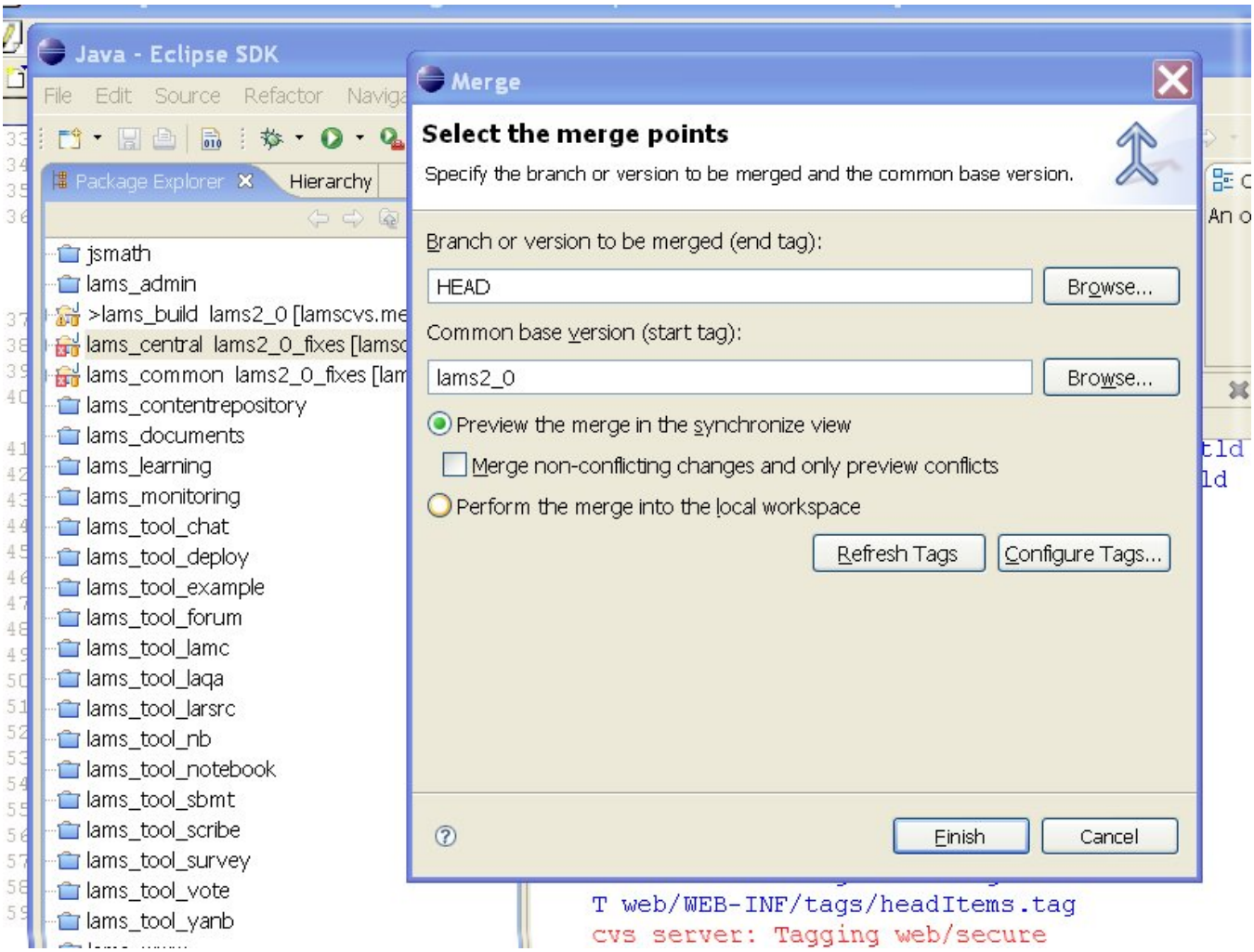
To merge change from one branch to another you need to know

- The branch (or HEAD) that **will** contain the updated file. That is, the branch that will be changed.
- The branch (or HEAD) that contains the new file. That is, the branch that already has the change.
- The tag on which both branches are based. So if you are dealing with the HEAD and a branch, then this is the tag on which the branch was taken. If you are dealing with two branches, then it is the latest tag that applies to both branches. Check "Started at tag" above to tell you which tag.

If you are doing a very large merge of a branch to the head (say for Shibboleth development), consider merging the changes from the HEAD to the branch and testing, and then merging the changes from the branch to the head once you are happy you have worked out any conflicts.

To merge:

- Set up your local copy of the project to contain the branch (or HEAD) that **will** contain the updated file. That is, the branch that will be changed.
- Right mouse click on the project name and to "Team", "Merge".
- In "Branch or version to be merged", enter the branch (or HEAD) that contains the new file. That is, the branch that already has the change.
- In the "Common Base Version" enter the tag on which both branches are based.
- Leave "Preview the merge in the synchronize view" turned on.
- The picture below shows the Merge window set up to merge any outstanding changes on the HEAD onto the lams2_0_fixes branch, working from the lams2_0 tag. The lams2_0_fixes branch was made based on the lams2_0 tag.



- Click Finish.
- The Synchronize view will be displayed, showing any files that need merging. Go through the files and merge the appropriate file (right mouse click on files/folders and select merge). This will merge them in your local workspace. *Check what the changes are and deal with any conflicts. You will have to think about this process - don't just blindly merge the files.*
- The changes will now be "on the branch" in your local workspace, but not in CVS.
- Depending on what the changes are like, you might want to build and deploy the project and test the changes. If it is something trivial and you are working on a branch, then you probably can put off testing until you have other changes to test. If it is a major batch of changes or you are updating the head then please be more careful and if in doubt, build and test.
- When you are happy that the changes won't break the build, check in the changes as usual (using Team Synchronize). Now the branch is updated in CVS, and you can check FishEye to make sure the merge was done okay.

Graphical View of Branches

Completely confused about which branch is on what code? Perfectly understandable. If you can get WinCVS working, then you can get a tree view of all the branches. There is also an Eclipse CVS plugin that does this - see http://versiontree.sourceforge.net/ for a screenshot.