Deprecated Release Process

> ⓘ **About this Page**
>
> This document outlines our process for coordinating releases of Fluid Infusion.

What's in a Release?

We expect that each Fluid Infusion release will contain a combination of framework code, reusable user interface components, and documentation. More specifically:

Technology

- JavaScript code and CSS for the Fluid Infusion framework
- JavaScript, HTML, and CSS for Fluid Infusion components
- Examples and sample code
- API documentation and tutorials

Frequency of Releases

We release versions of the Fluid Infusion framework and components on a regular basis. For more information about the contents of each release, check out the Fluid Community Roadmaps.

Release Version Number

Each release has a unique version number associated with it, e.g. "0.1" or "0.3beta1". This version number must be recorded consistently in a few locations:

- git tags (historically SVN tags)
- Wiki pages and other documentation
- The project `pom.xml` and `project.xml` files
- The perma-comment at the top of the Fluid.js file
- The Infusion source code (each .js file - this is not modified for point releases such as 1.3.1)
- The ImageGallery `web/pom.xml` file, which has a dependency on Infusion (see Tagging the Source Code for instructions)
- The `ant` build scripts properties file, `build-scripts/build.properties`

The following instructions will describe more specifically when, where and how to record the release number.

Process

Task	Who Coordinates with the Community?
Set and announce the release date, code freeze date and bug parade date	Project Manager, Tech Lead, QA Lead
Coordinate the release deliverables and create a roadmap on the wiki	Project Manager, Tech Lead , UX Leads

Review the escalated issues	Whole Team
Create the Release Status wiki page	Release Manager
Ensure known issues in JIRA have been marked with the correct fix version for the release	Whole Team
Create documentation JIRA issues and coordinate with community	Documentation Lead
Work with component design/development teams to produce a test plan for each Fluid component	QA Lead
Recruit QA testers	QA Lead
Create Bug Parade list in JIRA	QA Lead
Coordinate the Bug Parade with the community	QA Lead
Create release related JIRA issues for the main release tasks, and set them as blockers (once Bug Parade is complete)	Release Manager
Coordinate QA and Code Freeze This includes sending emails to the list announcing Code Freeze.	QA Lead
Coordinate Release Bundle QA	QA Lead
Update JIRA	Release Manager
Write release readme and use it as text for announce e-mail	Release Manager
Write release news item	Release Manager
Manage updating the READ Me	Release Manager
Manage updating the website pages (index, downloads, infusion) with new version number	Release Manager
Manage updating the Wiki	Release Manager
Build, test, and release bundles	Release Manager
Manage git project repo and update code for next Release	Release Manager
Send release annoucement email to sakai-ui, jasig-ue, i-staff@ischool.utoronto.ca, fac-all@ischool.utoronto.ca, i-phd@ischool.utoronto.ca	Release Manager

This is a collaborative process, and the community is encouraged to take an active role in defining schedules and coordinating the release process. It is expected that the Release Manager, QA Lead and Documentation Lead roles can be rotating positions based on interest and expertise.

Release Status Wiki Page

Unable to render {include}   The included page could not be found.

Release-related JIRA issues

The real work of creating a release is managed through a number of Blocker JIRA issues. These JIRAs must be created for each release, and any necessary changes to the code must be committed against them.

During Bug Parade

Issue Title	Issue #	Issue Description
Code cleanup for release x.x		Clean up comments, ensure public APIs are fully commented and JSLint new javascript files. Note that this issue should only be in force before Code Freeze. Commits that alter actual code should not be permitted during Code Freeze.
Verify dependencies for components		This issue should be resolved in bug parade. Look at FLUID-2525 for an example of how to create the jira issue
Check licenses and copyrights		Ensure all javascript files have a license and copyright. Copyright should cover only the years since the code file was created and should include only institutions who have made a substantive contribution to the code. Also make sure third party code is compatibly licensed and a license file is included in the same directory as the code. Fluid Licensing
Update and expand on unit test coverage		Increase unit test coverage and ensure that all tests are passing.
screen source files for DOS line ending		Run source tree through an utility like dos2unix to normalize line endings for Unix. // Note: this was not done for 1.1. We're going to do it before reopening the trunk.

| Turn off debug mode | | Update fluid.fail() to throw an error, turn off debug mode for the renderer, etc. //
Note: this was not done for 1.1 because fluid.fail() was not turned back on after 1.0. Adding a task to turn fluid.fail() back on before reopening the trunk. |

Removing DOS CR/LF

For Ubuntu using fromdos command:

```
find . -type -exec fromdos '{}' \;
```

Get fromdos by running `sudo apt-get install tofrodos`.

After Bug Parade is Over

Issue Title	Issue #	Issue Description
Update the `README.txt` file	FLUID-2862	Be sure to update the version number, the zip file names (which include the version number), the "What's New" section with a list of features of the release, folder hierarchy if it has changed at all (e.g. new sample-code folders added), the list of supported browsers if it has changed, and the "Known Issues" section with highlights of the known issues.
Generate and test release bundles	FLUID-2873	Compare the production bundle to the repository and ensure everything is present. Compare the two bundles - the only difference should be that the source bundle does not have a war file, the production bundle has minified javascript and the source bundle does not have minified javascript.
Tag, package and post the release	FLUID-2874	See http://wiki.fluidproject.org/display/fluid/Release+Process#ReleaseProcess-HowtoTagandPackagetheSourceCode for details.
Ensure that component progress indicators and links to demos are up-to-date	FLUID-2875	The progress indicators on each component landing page should be double-checked, to ensure that they are up-to-date. The the links, pointing to the version demos on the website, should be updated to the latest version
Update the demos on the Fluid website	FLUID-2876	Once the release bundle is posted, the links on the website to demos of the latest release must be updated and add/remove any demos as necessary: http://fluidproject.org/index.php/demosFor instructions, seehttp://wiki.fluidproject.org/display/fluid/Release+Process#ReleaseProcess-UpdatingDemosontheFluidProjectwebsite

Try to identify any dependencies between JIRAs (for example, the bundle must be tested before it is posted) and create these links in JIRA.

Documentation JIRA issues

To ensure that the API documentation and tutorials are up-to-date, each document needs to be reviewed. JIRA issues should be created to coordinate this process. The specific pages needed will depend on the current state of the documentation, but the JIRAs should follow the following pattern:

Issue Title	Issue Description

"Tech review of <page> docs"
where <page> may be
 	
- <component> API
- <component> Tutorial
- a framework section, such as Events, Subcomponent, or Renderer
- etc.
	A technical review of <link to actual documentation page>
	
	- Check page for correctness against current code base:
	  - function signatures
	  - options (selectors, supported events, styles, etc.)
	  - dependencies
	  - etc.
	- Check links between pages, add and update where appropriate.

Update the wiki for release	See http://wiki.fluidproject.org/display/fluid/Release+Process#ReleaseProcess-UpdatingtheWikiforRelease for details
Update the website for release	See http://wiki.fluidproject.org/display/fluid/Release+Process#ReleaseProcess-UpdatingDemosontheFluidProjectwebsite for details

NOTE: Technical review should ideally be carried out by someone who did NOT originally create the page!

Updating the Wiki for Release

Updating Wiki: Development

1. Update any development API and integration pages.
2. Update any demos

3. House cleaning - Delete any unnecessary pages and information created in this past iteration.
4. Update download & component pages (see below)

Release download pages

1. Consider putting a disclaimer at the top of any affected pages, with the following text (or something similar):
   - "This page is currently being edited in preparation for the pending X.X release. It's contents should be considered in flux and unreliable until this warning is removed. For the latest stable release, see Fluid Infusion X.X."
2. Duplicate the old Fluid Infusion - Current Release page into a new page called "Fluid Infusion X.X" where X.X is the old release number.
   - _This new page should be a child of Fluid Infusion - Current Release.
3. Update Fluid Infusion - Current Release to the latest release.
   - *Be mindful to follow the excerpt format*. The links to the bundles at the top will be excerpted, and displayed on the Downloads page.

Component pages

1. Go to the Components page
2. For each Component that was updated in this release, ensure that the relevant information page is updated.
3. Update API, Integration, Demo, and Testing sections as necessary.

Updating Wiki: UX

1. House cleaning - Delete any unnecessary design information (pages, notes, etc) added to the Wiki in this past iteration.
2. Update any Components pages to reflect the current state of the design.
   - Wireframes, Storyboards, Design pattern, Functional Req, User Testing, Story Cards, User modeling, Accessibility.
3. Verify that any changes to the Design Handbook is reflected.

How to Tag and Package the Source Code

| Unable to render {include} | The included page could not be found. |

Posting the Distribution

The distribution files must be uploaded to the websites `releases` folder using `ftp`. Links on the website downloads page (http://fluidproject.org/index.php/download-infusion) and the wiki downloads page will link to this location.

Speak to someone on the project core team about `ftp` access.

Updating Demos on the Fluid Project website

> ⊘ these instructions are no longer correct

1. Archive the existing Demos page (http://fluidproject.org/index.php/infusion-demos) as follows:
   - Create a new sub-page of the Demos page and copy and paste the Demos page content there.
2. Extract a copy of the new minified source code and demos into the new releases' directory. (i.e. ~/website/docs/releases/0.999beta1)
3. update the Demos page links to reference the new sample code.
   - add any new Demos
4. Add a link from the Demos page to the Demos just archived in Step 1. Put this link under the "Past Releases" section at the bottom of the page.

Update JIRA

1. Mark the version released and ask the community to update the remaining open issues:
   a. The current assignee should consider the issue and decide whether or not to resolve it, assign to someone else, or change the "Fix for" version to an unreleased version.
2. Do a query for all unresolved issues that affect the previous release and release candidates. Add the new release version as an affected version for each issue in the results.

Managing Unexpected Issues

Bugs happen. When unexpected issues or problems arise, the Release Manager, Technical Lead, or Project Manager will inform the community and work with them to adjust the release schedule accordingly. If you find an issue that you think is a blocker, let the Release Manager know as soon as possible.

If you discover a security issue, follow the Fluid Security Policy and report the issue privately to the security team.