Posting the FHIR specification on hl7.org

This page provides rough documentation on the overall process for posting the FHIR specification to hl7.org/fhir.

This process is used by the FHIR product director (or designated agent) when a new version of the FHIR specification is posted.

- **1 Things to consider prior to planning this process**
- **2 The actual process**
  - **2.1 Phase 1: The build process**
  - **2.2 Phase 2: post-processing**
  - **2.3 Phase 4: actual release**
- **3 Full release steps**
  - **3.1 Phase 3**
  - **3.2 Phase 5**
  - **3.3 Phase 6**

**Things to consider prior to planning this process**

Before planning this process, the following things must be considered, and the plan that leads to posting a new version of FHIR must allow for them:

- Typically, stable versions are posted 6-8 weeks prior to an HL7 meeting for a connectathon, and/or supporting ballot, if applicable. Full releases mostly at other times.
- Whenever a new version is posted, the process must be co-ordinated with the reference implementation maintainers (at least James Agnew, Brian Postlethwaite, Pascal Pfiffner, Nicola Rhyzikov) in advance.
- FMG must sign off on a QA plan to ensure that the posted release represents a stable consensus from the committers that represent different domains.
- There must be a freeze process and a technical QA prior to the build to ensure that the build and the terminology reference server are in sync.
- For full releases, the TSC must ok the publication - make sure this is part of the process, and the interaction between the TSC and the build is well understood.
- For a full release, press releases must be written prior to the rest of the process occurring, and HQ support (webmaster, press release) must be lined up before starting.

**The actual process**

Once the build is stable, and FMG have signed off on it’s state and the technical build tools are all aligned, start the actual build.

**Phase 1: The build process**

- upgrade the version (see fhir.ini). Rebuild following the instructions in fhir.ini. Then commit to svn.
- then, upgrade the terminology server to the new version, and upgrade the active terminology server (e.g. tx.fhir.org)
- delete the vscache contents (build/vscache), and run the build again. (slow - hours). Check for new errors cropping up. Then commit again.

Note: this is the content that TSC approves from publication. From the time it is submitted to TSC, it is frozen until approved.

- delete the contents of the publish directory
- now, start the build again, but this time with the parameter -web (check)
- check the output for (new) errors

**Phase 2: post-processing**

What is on build.fhir.org is the final copy, and cannot be altered without invoking the TSC technical correction process, except in 3 regards:

- The version information (titles, product names, versions) in the header and footer can be changed by the Product Director.
- A pink or gold box can be added to the page immediately above the header that describes how the page / specification fits into the overall version release of the FHIR product versions.
- URLs for links and images can be rebase for moving from one location to another (i.e. from http://build.fhir.org to hl7.org/fhir/...), and external URLs can be fixed.

This is the rough process for moving the content to the web. In order to use this process, you need an existing directory that contains the same content as the /fhir directory on the HL7 web server, and ftp access to that directory.

- update the fhir-versions code system
- copy the generated output from publish (from the run with the -web parameter) to fhir/[X] where X is the destination directory.
- tag the build to /tags/[X] so that there is a branch in which the release can be maintained as necessary (make it a branch, not a tag).
- increment the version number in trunk, following the fhir.ini process, so implementers can tell them apart (note: this step can delayed until there is change in the trunk).
- if this a full release, then the value of [x] is... don't know (as "STU"N, but not next time). TBA...
Otherwise the value of [X] is YYYYMmm where YYYY is the year and MMM is the month (e.g. Sep) of the official ballot cycle. Note that this is the month “after” you actually publish.

now, using a text editor, do search and replace on the HTML files. For each search/replace, check that the number of replaces is as expected

- Next time: change this in the branch source so it sticks for technical corrections
- replace the title in the header (e.g. "Current build") with the correct title
- replace the stated version in the footer (e.g. "STU3 Candidate (v3.0.0-11828)") with the correct version for publication. The technical version must be correct, including the build number
- replace the url in the 'compare to version X' link (and possibly the stated version) with the correct URL
- search for references to build.fhir.org in the spec, and... resolve... them (good luck with this; a few are legitimately for the current build, but most are not, and need to be changed to either hl7.org/fhir or hl7.org/fhir/[X]
- run the java class "org.hl7.fhir.tools.converters.WebSitePatcherForHL7IIS" on the fhir/[x] directory - this will produce .xml1, .xml2, .json1, .json2 clones of xml and json files. These are needed to support IIS mime-type redirects

Done - the content is now ready. Check everything on it again. and again

Phase 4: actual release

- ftp the content from fhir/[x] on your local directory to fhir/[x] on the server
- edit the local copy of directory.html to note the new release, and ftp that to the fhir directory
- let the community know that a new version of FHIR is available. At least via Zulip, FHIR email list and tweet using #FHIR

Full release steps

Phase 3:

If you are doing a full release, then there's a whole lot of additional steps to take, to replace the existing content

- write a windows batch file to delete the existing spec (all files, all sub-directories that are not a past version, or an affiliate root) and then copy from [y] where [y] is a temporary directory name, and then delete [y]
  - a template for the batch file can be found in build/tools/web/web-update.bat
- organise for the HL7 webmaster to run this on the server later in the process (ensure it is marked read-only before doing that)
- create a fhir/[y] directory in your local copy of the web
- copy the publish content from the build to the fhir/[y] directory, and follow similar instructions as above. Note that the actual urls and names and releases will be a little different for this, which is the main release going to http://hl7.org/fhir

Phase 5

- test the batch file in a temporary copy of the entire fhir subtree, to ensure that it upgrades correctly
- ftp the content from fhir/[x] on your local directory to fhir/[x] on the server
- advise the web master to run the batch file (as already scheduled)
- edit the local copy of directory.html to note the new release, and update the current version information and ftp that to the fhir/[x] directory
- once the batch is run:
  - let the community know that a new version of FHIR is available. At least via Zulip, FHIR email list and tweet using #FHIR. In addition:
    - a press release should be issued
    - a blog should be made on the product director blog detailing what is worth knowing about the new release and recognising key contributors
    - let mr-histalk know
    - other media work may be appropriate.

Phase 6

Now, all the existing posted versions have to be updated for the new version. To do this:

- edit the main[] routine of org.hl7.fhir.tools.publisher.OldVersionRedirector to register any versions of FHIR that have been superceded
- run this on your local copy of the fhir directory.
  - you can run this repeatedly if necessary
- upload all html files in local fhir tree to the FHIR web server (in practice, upload every file, and then tell ftp client to ignore files that haven't changed)