Integration of RESTWS

General Info

Topic: Integration of RESTWS

Type of Project (Spike, Sprint, Epic): Sprint

Lead (Product Owner): Unknown User (r.friedman)

Developer Lead: Unknown User (raff)

Known Developers Assigned (amount of effort in hrs dedicated to this work if possible): Unknown User (patandrea) , Wyclif ,

Sprint Dashboard: https://tickets.openmrs.org/secure/RapidBoard.jspa?rapidView=25

Source code at: https://github.com/openmrs/openmrs-module-webservices.rest/tree/sprint-201302 (the sprint-201302 branch)

CI: https://ci.openmrs.org/browse/BUNDLED-RESTWS0

Date: 2/7/13

How to Participate

Add your name to the list on this wiki page (with any comments about your availability). If you want to join after the sprint has started just join the IRC channel mentioned above and say hello.

The general process:

1. New to OpenMRS sprints? Want help getting started? Join the IRC channel and say "???": I'd like to participate in the sprint!"
2. Checkout code:
    a. Fork https://github.com/openmrs/openmrs-module-webservices.rest
    b. Clone your fork: git clone https://github.com/YOUR_USERNAME/openmrs-module-webservices.rest.git
    c. Add upstream: git remote add upstream https://github.com/openmrs/openmrs-module-webservices.rest.git
    d. Fetch all branches: git fetch --all
    e. Checkout the sprint branch: git checkout **sprint-201302**
3. Compile code:
    a. Run: mvn clean install
4. Setup OpenMRS. Get the standalone: http://sourceforge.net/projects/openmrs/files/releases/OpenMRS_1.9.2/openmrs-standalone-1.9.2.zip/download
    a. Unpack and start openmrs-standalone.jar. Choose the demo database.
    b. Go to http://localhost:8081/openmrs-standalone/admin/modules/module.list and upload a compiled webservices.rest omod, which you can find in webservices.rest/omod/target.
5. Pick a ticket from the available tickets in the top-left of the sprint dashboard page (listed below). Make sure it does not depend on a ticket that is incomplete. If you have any questions about the ticket, ask on the group chat
6. Create a local branch for your ticket (see also our HOWTO for git, use '**sprint-201302' instead of 'master'**)
    a. git checkout -b REPORT-123 sprint-201302
    b. git push origin REPORT-123
7. Do the ticket.
    a. Stage changes: git add -i
    b. Commit: git commit -m "REPORT-123: ticket title"
    c. Push changes: git push -f
8. Issue a pull request for the '**sprint-201302'** branch (see github help) or if you have push rights: whatever works for you! If you don't like pull requests, don't send them. Commit and push directly to the upstream repo. If you do like pull requests, fork the repo and send pull requests, but merge them right after. My favorite way is to work on the repo, but create local branches (without pushing them to the repo). Merge branches locally to the sprint branch and push to the repo.
9. If you push directly to the repo check if tests for the sprint branch in reporting are passing: https://ci.openmrs.org/browse/BUNDLED-RESTWS0 (F avorite the branch to be notified about failures: Login into Bamboo, Choose Actions -> Favorite Branch)
10. Join the daily scrum to share your updates

Participants

- Unknown User (r.friedman)
- Unknown User (darius)
- Unknown User (raff)
- Wyclif Luyima
- Unknown User (kavuri)
- Unknown User (surangak)
- Unknown User (harshadura)
- Unknown User (lluismf)
- Unknown User (jeffblack360)
- Unknown User (patandrea)

Unknown macro: {composition-setup} cloak.toggle.type=wiki

Goals

Unknown macro: {toggle-cloak}

Need to know more about Goals? Click here.

Unknown macro: {cloak}

Goals: *Must Have, Should Have, Could Have and Won't Have*

Must Have means we must have this as part of the solution. It is a requirement or we should rethink doing this work. **This is what we will deliver before the end of the work period** this also doubles as our DOD "Definition Of Done"

Should have means we would be embarrassed not to have it. We could technically produce the solution without it but people would be surprised.

Could have means anything else we could do. But this does not always mean low priority. But our selection of which could haves to include or exclude (or deliver later). Also is considered low hanging fruit (if we are in this code already for *x* reason wecan deliver *y* with little or no extra effort) or can be items or ideas for this module that can be packaged for a later spike or sprint.

Won't Have means this will not happen. It might be technically impossible, a taboo for our organization or out of scope. It does not mean "might happen or would be nice".

Unknown macro: {cloak}

Must Have:

- Just one module for "core OpenMRS REST Web Services" (i.e. get rid of 19ext module)
- the resources need to be separate from the framework (even if this just means that they're in different Java packages)
- the resources defined against 1.8 should be in a different java package from the ones defined against 1.9, etc.
- This is mostly done, but still need to create tickets about fixing tests, and fixing some resources
- Need to have both 1.8 and 1.9 versions of the Concept resource, which support different functionality with regard to mappings
- RESTWS-267

Should Have

- Resources for all "new" 1.9.0 functionality (e.g. Visit, Provider, Concept mapping changes, etc. Some of this may already be done.)
- Have design discussions about "things that are not just CRUD"** example: evaluating a "patient calculation"** example: moving a patient from one queue to another** example: sending an HL7 message

Could Have

- RESTWS-311 - RESTWS should allow modules to add custom searches to existing resources** DJ: I see this as a Could, unless Roger provides a concrete example where this is needed now.** Roger: I think RESTWS-267 is an example of this. (DJ: But that's core...)** Roger: this needs "core design resources"** Decision => During the spring Rafal (or someone with equivalent knowledge and ownership of the RESTWS framework) will take this ticket.
- Support for TestOrder, and new columns in Order, starting in 1.9.2.** Do an example of how to handle out-of-band data model change** Prove this doesn't break things

Won't Have

Design

Unknown macro: {toggle-cloak}

What should go into a Design ? *Click here for more information*

Unknown macro: {cloak}

There should be multiple design meetings. The first starts off with a small group working with the leader to determine the high level scope of the project and then to break down into smaller pieces to eventually place as tickets. Once those meetings havehappened use of the community design meeting time should be used to refine the tickets and if possible assign them to who will be doing the work.
Risks (these should be resolved prior to or during the design meetings):

Enter anything that is still questionable or worrisome that has not been answered: (open issues, potential concerns) Once a decision is made make sure that a summarized answer is included.

Example: Q. How will we handle issues with conflicting userID's? A. All ID's will have an added identifier that is unique.

Effort Accounted For: (At the end of the design call all tickets should have an estimated effort and that total should be balanced against the known available effort of the developers assigned) (Yes/No)

If not in balance in favor of success why and the action plan for remaining items
via IRC on the [#openmrs|IRC:Home] channel on freenode.

Use this channel for **ALL** debugging and random questions having to do with the sprint. Please avoid direct messaging to personal contacts. If you have a question, someone else most likely does too, and our geographically distributed community benefitsfrom public group discussion.

Sprint Break down: If tickets have not already been laid out or explained this is a good place for this.

Unknown macro: {cloak}

Risks: ?

Enter anything that is still questionable or worrisome that has not been answered: ?

Effort Accounted For: ?

Sprint Break down: ?

Kickoff Meeting

Kickoff meeting Date: TBD

Kickoff meeting recording:

(Meeting setup with known developers after the final design meeting, but prior to the start date):

Unknown macro: {iframe}

During Project Notes

Unknown macro: {toggle-cloak}

Notes *(Click here to expand to Etherpad)*

Unknown macro: {cloak}

Unknown macro: {iframe}

Unknown macro: {cloak}

Post Project (Retrospective)

Unknown macro: {toggle-cloak}

Retrospective *(Click here to expand to Etherpad)*

Unknown macro: {cloak}

Unknown macro: {iframe}

Unknown macro: {cloak}

Resources

Additional Information:

Resources

Kickoff meeting recording: ??