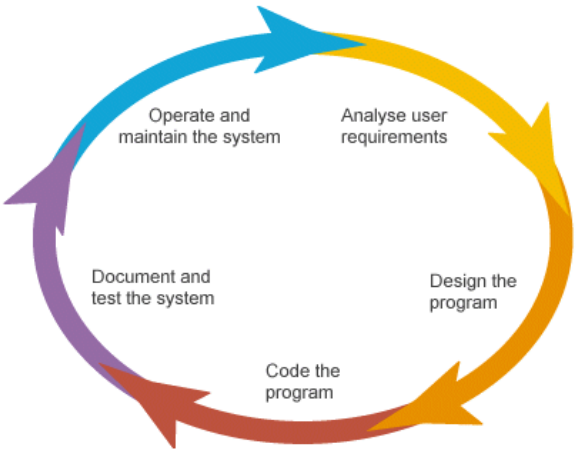
TESTING MUSEUM WEBAPPS

jb lowe 3/20/2014

Introduction

Thank you for helping to test the latest and greatest version of the PAHMA webapps! Hopefully this will not take too long nor be too painful. There are 13 webapps to test, and some of them are pretty specialized and difficult to verify.[1] And, alas, few of those details are yet elaborated here: you are just supposed to know!

Typically, the "software development lifecycle" goes something like this:



We are at the purple "document and test stage".

At the moment, the lifecyle for these applications is managed a bit more informally here at the Museum, something like the following:

- We plug away with the software we have, noting bugs, and dreaming up enhancements. These get filed as "JIRA issues" (i.e. bug reports) at http://issues.collectionspace.org.
- The issues are triaged, prioritized, worried about, elaborated, verified, and eventually addressed in the form of changes to program code.
- When a critical mass of changes accumulates, or the need for a particular fix or enhancement becomes critical, an informal "release" is made of a new version. This is usually called the "NV version" and is installed alongside the existing production code on the same server. Both the production and test versions of the code can operate with the production or test version of the database – one need only point the webapp to another config file in the URL (i.e. the bit that follows "webapp=" in the URL). Of course, we want to use the test code with the test database to start with!
- We test and try to break the NV version, checking especially that all the advertised stuff works and that nothing has broken in the repair process. And that new bugs have not been introduced. *This is the step we are doing now.*
- When we are satisfied that the NV version is working properly, the code is "pushed to production", and the NV version disappears until the next iteration. The bug reports are marked *Resolved* and a new cycle of bug finding and fixing begins.

Testing software is a bit of a black art, especially testing interactive software like this: there are usually many possible inputs and actions that are possible, and the program really ought to do the right thing in all cases. In particular, we need to test the following things:

---

[1]For example, we want to ensure that the *Barcode Label Generator* works, but we obviously don't want to print more than a couple labels to test it. Therefore, the Dev version of this webapp only creates the files for the printer, but nothing prints. To check that the label printing part of the webapp *really* works, we carefully print one or two interesting labels using the Prod version.

- That the bugs that were fixed really are fixed.
- That the new features and enhancements that have been added really work as intended.
- That we have not broken anything in the process of making changes. This is called *regression testing* – when something that used to work becomes broken, it is called a *regression*.

You can see that this can get pretty complicated after a few cycles: not only do we have to test the new stuff, we have to go back and re-test all the previous bug fixes and features, to make sure they all still work as expected.

To do this most efficiently and rigorously, we should have a *test plan* that lists all the things that need to be tested, and instructions how to do the testing. There would be a test plan for each webapp, and an overall plan describing how the whole testing and deploymenet apparatus works.

Alas, you are looking at what we have for a test plan. It is not much!

So, here is what I would like to suggest:

- Everyone picks an app or two from the list below.
- You exercise the "General Testing Procedure" as described below.
- As you test, you make up a short test plan – that is, you record your inputs and actions, and what you observe as a result. The test plan is simply a short narrative of your experience, a list of steps, really – you can keep it as a open document on your computer as you do the testing.
- When we're done, please send me (jblowe@berkeley.edu) the document or paste it in an email.

I will, or Michael and I will, undertake to merge all the texts into one coherent version of this document. It will make testing in the future go much faster and be much more reliable.

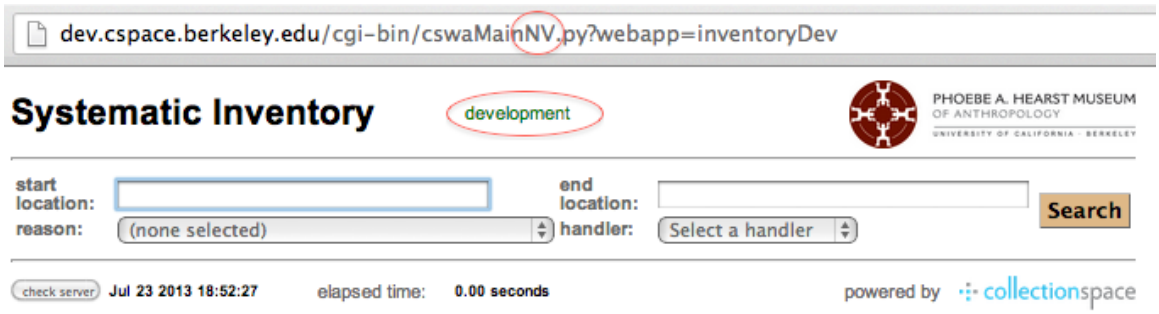
Thank you in advance for all your help!

Webapps to test

		Types of Input Accepted			
Webapp	*Updates DB?*	*Locations*	*Objects*	*Other fields*	*Other controls*
Barcode Label Generator	No	x	x		x
Barcode Scan File Upload	Yes				x
Bulk Object Edit	Yes	x			x
Collection Stats	Yes	x			x
Crate Move	Yes	x			x
Government Holdings	No			x	x
Hierarchy Viewer	No				x
Key Information Review	Yes	x		x	x
Move Object Range	Yes	x	x		x
Object Details	No		x		x
Object Info Review	Yes		x	x	x
Packing List Report	No	x		x	x
Systematic Inventory	Yes	x			x

General testing procedure

1. Log in the usual way (at https://dev.cspace.berkeley.edu/cgi-bin/cswaMainNV.py), and ensure you are using the correct URL (i.e. the "NV" webapp), and ***pointing to the Dev server! (green letters in header!)***



2. Have a pencil and paper, or perhaps a text file for notes open on your computer.

3. Verify that each control works as expected ("control" = input box, button, check box). That is, verify that the normal workflow works correctly. Someday there will be a checklist of specific actions or recipes to try…

4. Once that is done, spend a bit of time trying to "break" the webapp: enter inappropriate values, esp. blank or wildly inappropriate values, and see if you can get it to demonstrate some bizarre behavior. If you can, write down the "steps to reproduce", in some detail, please!

5. Review the list of changes (the "changelog") below, and ensure that the bugs mentioned are fixed, and the new features work as expected. Rarely is the title of the JIRA issue alone enough to determine what problem or request is being addressed: you'll need to look up the issue details in at http://issues.collectionspace.org.

6. If and when you find problems, please email details to both John (jblowe@berkeley.edu) and Michael (mtblack@berkeley.edu). Cut-and-paste error messages or interesting tidbits and include them in the email where possible.

7. If you have questions about how a webapp is supposed to work (e.g. whether the behavior is a bug or a feature), please email either of us.

Interesting Facts About the Webapps You May Not Know

The webapps have a number of foibles you should know about:

- Term matching for authority fields, e.g. Field Collection Place or Ethnographic Group may include terms from other vocabularies and non-preferred terms (however, deleted terms no longer show up.) In fact, the selection algorithm is a bit complicated.
- Term matching for museum numbers is on the literal value of the object number, but the range is calculated on the basis of the *museum number sort field* (which you cannot see, except when viewing the catalog record).
- For locations, however, both matching and ranges are based on the literal value (Locations do not have a sort field, they sort "as themselves".)
- You do not have to enter an End Location. If you don't the app will use the Start Location and fill it in for you.
- Most of the webapps have a limit of 500 locations in a range, and 500 objects in a location. The Packing List is the exception: it will try to render the whole museum if you ask it. This value, incidentally, is based solely on the limitations of the "average web browser".

Sample Test Plan (initial draft)

Webapp:	**Bulk Object Editor**
URL:	https://dev.cspace.berkeley.edu/cgi-bin/cswaMainNV.py?webapp=bulkeditDev
Tester:	John L.
Date:	3/20/2014

Step	Action	Result	Pass?
1	Login with CSpace credentials	Expected screen	
2	Press Search Button, without entering any data	Red error text	
3	Enter location “		
4			
5			
6			
7			
8			
9			

List of specific things to test for this session (3/20/2014)

Bug fixes and enhancements:

PAHMA-1065	CGI webapps should use user's CSpace credentials for update
PAHMA-1074	The new webapp authentication system only admits tenant admins
PAHMA-1060	New KIR/OIR/BOE fieldset for HSR
PAHMA-1068	Add "Taxonomy" to options for hierarchy viewer
PAHMA-1067	A new button should be added below table.png on collectionstats
PAHMA-1066	Add button graphics to collectionstats web app
PAHMA-1040	Deal with LMI records that have a NULL currentlocation
PAHMA-1013	Modify LocHandlers dropdown list for webapps
PAHMA-1011	Create a basic monitoring system for changes to selected CSpace data
PAHMA-999	Monitor and ensure ongoing functionality of Bulk Box Movement system
PAHMA-992	Web app landing page links direct to broken URLs referencing .pyc files
PAHMA-980	Please add X to the Loc Handlers list for web apps (and Talend scripts)
PAHMA-986	Please X from location handler list on web apps
PAHMA-972	Disable validation of Tricorder data
PAHMA-1050	Inconsistent formatting of location+box in webapp vs. iReport
PAHMA-1012	New report: Researcher summary + Accession number + preferred alternate number
PAHMA-1049	Deleted objects appear in Systematic Inventory iReport

New webapps and reports:

PAHMA-912	New webapp: Bulk Object Edit (“BOE”)
PAHMA-1061	Report to show all previous locations for objects in a location
PAHMA-943	iReport/csv file of all objects from sites owned by government agencies
PAHMA-1059	Need customized systematic inventory report for HSR

In progress or Awaiting further details:

PAHMA-1069	Systematic Inventory (and perhaps other webapps) fail when confronted with duplicate object numbers
PAHMA-981	KIR fails due to None being returned by database function cswaDB.getCSIDDetail
PAHMA-1073	Add new field set to OIR/KIR/Bulk Object Editor: “taxonomy”
PAHMA-1063	Add new field set to OIR/KIR/BOE: “Object Type/CM”

Known bugs:

PAHMA-927 Unicode characters not saving correctly in KIR/OIR Brief Description field

Past Fixes and Enhancements, please check for regressions!

Bug fixes and enhancements:

PAHMA-811	Allow user to select different field sets in KIR/OIR webapps
PAHMA-834	Consider more detailed responses re: updated values in KIR/OIR webapps
PAHMA-852	Enable the printing of single object bar code labels
PAHMA-875	Reorder search term matching to account for trailing letters in web apps
PAHMA-879	Please add XXXX as handler for the Systematic Inventory web app
PAHMA-897	Webapps need a "in progress" indicator (now a butterfly)
PAHMA-898	Barcode Print does not print a single object label, on Production
PAHMA-914	Unexpected error msgs from KIR/OIR (memory tickler: "South Pacific Island", non-preferred terms)
PAHMA-910	Add an object count to the pre-print summary text on *Barcode Label Generator* web app
PAHMA-915	Hyperlink URLs for objects on *Key Info* web app (Dev) point to old URL version
PAHMA-917	Double-quotes in object name result in truncated or missing object name in KIR/OIR
PAHMA-921	Web app term matching for place and location should be case-insensitive (i.e., use ILIKE)
PAHMA-926	Remove XXX from handler list
PAHMA-928	Incorrect KIR update behavior, error checking
PAHMA-929	Incorrect KIR update behavior, error checking, type 2
PAHMA-930	Count not updating in OIR (NV version)
PAHMA-931	Illegal vals AltNumType (now a dropdown)
PAHMA-937	Collectionstats: EFC labels, etc.
PAHMA-939	KIR update fails on occasion. Suspect data dependency.
PAHMA-959	Bar code printing of single objects does not ignore deleted objects
PAHMA-960	Enable the printing of bar code labels for a range of object numbers

Known bugs:

PAHMA-927	Unicode characters not saving correctly in KIR/OIR Brief Description field

Awaiting further details:

PAHMA-843 Enable box movement via scanner upload files

New webapps and reports:

PAHMA-954	Create a webapp to report on "Government Holdings"
PAHMA-943	Government Holdings iReport

Previous work: check for "regressions" (i.e. if these fixes have gotten broken)

Bug fixes

• KIR no longer creates duplicate entries of Site and File Code values (PAHMA-878)

• Unicode characters (i.e. "special characters" such as ê) now print properly on barcode labels (PAHMA-857)

• Computed current location can now handle multiple moves on same day (PAHMA-839)

New webapps and reports

• Hierarchical authority viewer (PAHMA-876)

• 24 new reports—4 each for Accession, In Loan, Out Loan, Object Entry, Object Exit, and NAGPRA Claim

Enhancements

• Disk speed has been given a performance upgrade (PAHMA-872)

• KIR web app will now inform user if any values did not 'stick' on update (PAHMA-834)

• Single barcoded labels for objects can be printed (PAHMA-852)

• Location need be entered only once in web apps for single-location range searches

• The cursor is now automatically placed in the entry field (PAHMA-850)