BIOLOGY RELATED INFORMATION STORAGE KIT

Developer's Manual

SLIMS

tvanrossum

SLIMS Developer Manual v1

1 Environment

SLIMS was developed using Java 1.6.0, JSPs, Hibernate 3.3.1.GA, DB2 and mySQL (at different times), Apache Tomcat 6.0.18, NetBeans IDE 6.5, Jasper Reports 3.5.1 and JasperSoft's iReport 3.5.1 all on Microsoft Windows XP. SLIMS was tested on Mozilla Firefox 3 and Microsoft Internet Explorer 7.

2 Database

2.1 Table and Field Descriptions

2.1.1 Common Fields

Field	Description
COMMENTS	Any user-generated comments for this entry
CREATED	When was this entry created
CREATED_BY	Who created this entry
MODIFIED	When was this entry last modified
MODIFIED_BY	Who last modified this entry

2.1.2 Meta_Data Tables

SLIMS uses these tables to store information on how to display their parent tables (tblcontainers is the "parent table" of tblcontainers_meta_data). They all have the same fields and work the same way except tblcontainercontents_meta_data. Each entry in a meta_data table corresponds to one field of its parent database (except tblcontainercontents_meta_data) and describes how to display that field.

In General

Field	Description
SYSID	Primary key
PROPERTY_NAME	The name of the field this entry corresponds to as it appears in SLIMS (not the database)
SHORT_NAME	A user-readable short version of this field's name
LONG_NAME	A user-readable full version of this field's name
VIEW_COLUMN_NUMBER	When viewing data for the parent table, which column will this field be shown in (1 is for the database ID and is only shown to admin users, so for most users the left-most column shown will be the one with a value of 2). More than once field can have the same value for this field (in which case the order of those fields will be alphabetical). Values need not be a complete sequence (gaps are fine).
IS_SORTABLE	Indicates whether the user can sort by this column (0=no,1=yes)
IS_EDITABLE	Indicates whether the user can edit this column (0=no,1=yes)
SHOW_IN_REPORTS	Not used in SLIMS
COMMENTARY	Comments

Tblcontainercontents_meta_data Specifically

Which rows and the order in which they are displayed for tblcontainercontents varies by user. Therefore, this table is slightly more complex. Each user has their own set of entries defining their view scheme, which are indicated by the USERID column. For example, all the entries in this table with USERID = 4 define how the user with ID=4 will view the tblcontainercontents table. Any entries in this table with USERID not = 4 will be ignored when displaying table tblcontainercontents for the user with ID=4.
When viewing the 'samples' of the database (actually viewing the contents, not the DB samples) information from other tables than just TBLCONTAINERCONTENTS can be displayed. This means TBLCONTAINERCONTENTS_META_DATA needs entries corresponding to fields in tables other than its parent table.

2.1.3 Log Tables

These tables are populated by update and delete triggers on the main four tables (tblcontainercontents, tblcontainers, tblsamples, tblsubject). Whenever an update or delete is made on one of these tables, the 'pre-change' version of the entry being changed is stored in the log table, along with a timestamp of when the change was being made (field UPDATED).

2.1.4 tblcontainercontents

Field	**Description**
CONTAINERID	Primary key of tblcontainers, the container the content is in
"ROW"	The row in the container the content is in
"COLUMN"	The column in the container the content is in
SAMPLEID	Primary key of tblsamples, the sample of which the content is an instance
VOLUME	The volume of the content in ul, -1 for unknown
CONCENTRATION	The concentration of the content in ng/ul, -1 for unknown
CONTAINERCONTENTSID	Primary key
PARENTID	The content this content came from (ie the tube material was taken from to fill a well) (-2= tubes have no parents; -3 = no data available)
CONTAMINATED	If the tube/well is contaminated (0 no, 1 yes, 2 unknown)
DILUTION	The dilution of the content (ex 1:100), 1:1 for stock and blank for unknown or N/A
QUANTIFIED	The date this content was inventoried
MATERIALTYPEID	What type of material this content is (WGA, genomic, unextracted, etc)
AMPLIFICATIONDATE	What date this sample was amplified, will be null if non WGA
BARCODE	The barcode on a tube or plate, if the content has one

2.1.5 tblcontainers

Field	**Description**
CONTAINERID	Primary key of container
CONTAINERTYPEID	What type of container this is (96 well plate, box of tubes, etc)
CONTAINERNAME	The systematic name of this container
FREEZERID	Which freezer this container is in
SHELF	The shelf the container is on in the freezer
DISCARDED	Whether this container has been discarded or not (0=no,1=yes)

SHIPPEDOUT	Whether this container has been/will be shipped out or not (0=no,1=yes)
SHIPPEDDATE	When this container was/will be shipped out
SHIPPEDTOID	Where this container was/will be shipped out to
STOCK	Whether the wet lab considers this container only to have stock contents (like a box of extracted DNA tubes or a stock (1:1 dilution) WGA plate)
MATERIALTYPEID	What kind of material is in this container (all WGA, all genomic, a mix of these etc)
VALID	Whether the container is valid (explanation of this should be in comments field) (0=no,1=yes)
CONTAINERALIAS	Non systematic name for a container. What is actually written on the container if it differs from the systematic name, or what the container was once known as
"DATE"	The date written on the container
INITIALS	The initials of the plate maker(s)
LOT	
LOCATION	The specifics of where the container is on the shelf in the freezer

2.1.6 tblcontainertypes

Field	Description
CONTAINERTYPEID	Primary key
DESCRIPTION	The description of what kind of container this is (ie a 96 well plate, a 384 well plate, a box of tubes etc)
"ROWS"	The number of rows in this container
COLUMNS	The number of columns in this container
SORTORDER	The order in which the entries should appear (not used)

2.1.7 tblcontrol

Field	Description
CONTROLID	Primary key
DESCRIPTION	The name of this kind of control
TYPE	Whether it’s a negative or positive control (0 is negative control, 1 is positive control)

2.1.8 tblcontrollayoutwells

These are standard control layouts that are used to prompt users to create their actual control wells for a container (see TBLCONTROLWELLS)

Field	Description
CONTROLLAYOUTID	Primary key
ROW	The row this default control well is in
COLUMN	The column this default control well is in

DEFAULTCONTROL	The id of the default control that is in this well
LAYOUTNAME	The name of this layout of controls

2.1.9 tblcontrolwells

These are the entries for the actual controls in plates, like a ‘tblcontainercontents’ for controls.

Field	Description
CONTROLWELLID	Primary key
CONTAINERID	The container this control well is in
ROW	The row this control well is in
COLUMN	The column this control well is in
CONTROLID	The id of the control that is in this well
VOLUME	The volume of the control that is in this well

2.1.10 tblethnicitylookup

2.1.11 tblfreezers

2.1.12 tblgenotypingruncontainers

2.1.13 tblgenotypingruns

2.1.14 tblgenotypingrunsamplestatus

2.1.15 tblgenotypingrunsnpstatus

2.1.16 tblmaterialtype

2.1.17 tblphenotypelookup

Field	Description
PHENOTYPEID	Primary key
NAME	The name of the phenotype
TYPE	The type of phenotype: 1=subject either affected or unaffected (binary). 2=subject’s status requires number or text for classification (not binary).
DESCRIPTION	Description of phenotype

ABREVIATON	Phenotype’s abbreviation

2.1.18 tblphenotypes

Field	Description
SUBJECTID	Subject in question (part of primary key)
PHENOTYPEID	Phenotype in question (part of primary key)
VALUE	This subject’s affectation status for this phenotype (either 0/1 for type 1 phenotypes or a number/words for a type 2)
ABREVIATON	Mistaken column, should be removed

2.1.19 tblsamples

Field	Description
SAMPLEID	Primary key
SUBJECTID	Subject this sample belongs to
"NAME"	The name of this sample (referred to in SLIMS as the sample ID) ex: Az1001-1b (not: the root of this ID does not always reflect the subject ID)
VALID	Whether this sample is valid
NUMCALLS	Not used by SLIMS
LOCATION	Not used by SLIMS
NUMNOCALLS	Not used by SLIMS
FREEZERLOCATION	Not used by SLIMS
CONCENTRATION	Not used by SLIMS
PARENT	Not used by SLIMS
SAMPLETYPEID	The type of biological material this sample is or was extracted from (ie blood, epithelial etc)
SAMPLEPROCESSID	Not used by SLIMS
PARENTID	The source of this sample (if it was created from another sample)
COLLECTIONDATE	The date this sample (the unextracted version) was collected
SAMPLETYPEYEAR1ID	If we have type information for a yr1 sample, but aren’t sure if the lab has the yr1 sample or the yr7, the yr1 sample type goes here
EXTRACTIONDATEYEAR1	If we have an extraction date for a yr1 sample, but aren’t sure if the lab has the yr1 sample or the yr7, the yr1 extraction date goes here
COLLECTIONDATEYEAR1	If we have a collection date for a yr1 sample, but aren’t sure if the lab has the yr1 sample or the yr7, the yr1 collection date goes here
EXTRACTIONDATE	The date DNA was extracted from the biological material of this sample

2.1.20 tblsampletypes

2.1.21 tblshippedto

2.1.22 tblshoppinglistcontainercontents

2.1.23 tblshoppinglistcontainers

2.1.24 tblshoppinglists

Field	Description
LISTID	Primary key
LISTNAME	The name of the ‘shopping cart’ list
INUSEBY	If a user is currently editing this list or using it to populate a container etc (anything other than viewing it) then this field will have the user’s id as a value. Other users are only allowed to use a list if this field is null, though they can still view it

2.1.25 tblshoppinglistsamples

2.1.26 tblshoppinglistsubjects

2.1.27 tblsimsusers

Field	Description
SYSID	Primary key
FULLNAME	User’s full name
LOGIN	User’s login
PASSW	User’s password
RIGHTS	What level of permissions does this user have (a = admin (no restrictions), eu= expert user (no restrictions), dl = dry lab limited user (can only view data and make lists), wl = wet lab limited user (can only view data, make lists and change volumes))
COMMENTARY	Comments field
PARTNER	Not used in SLIMS
USERTYPEID	What type of user is this ()
INITIALS	The user’s initials (especially useful for wet lab because this is what is written on plates)

2.1.28 tblsubject

Field	Description
SUBJECTID	Primary key
COHORTID	The cohort this subject belongs to
ID	The subject’s ID within its cohort (ex 1001-1) called the “subject ID” in SLIMS
FAMILYID	The subject’s family ID within its cohort (ex 1001)
GENDER	The subject’s gender (0 unknown; 1 male; 2 female)
CODE	Not used in SLIMS
HASCONSENT	Whether we have consent from the subject (0 no, 1 yes, 2 unknown)
MOTHERID	The subject’s mother’s “ID” field (a string representing the subject ID like 1001-4, *not* the database ID of the subject’s mother (ie not a tblsubject.subjectID value))
FATHERID	The subject’s father’s “ID” field (a string representing the subject ID like 1001-5 *not* the database ID of the subject’s mother (ie not a tblsubject.subjectID value))
ETHNICITYID	The subject’s ethnicity
PASSEDQC	Not used in SLIMS
NUMSAMPLESCOLLECTED	Not used in SLIMS (though it could be if the fields were populated, it represents the number of samples collected from a subject)
"COMMENT"	Comments

2.1.29 tblusertypes

2.2 Triggers

There are update and delete triggers on tblsubject, tblsample, tblcontainercontents and tblcontainers that log ‘before’ snapshots of entries when one is updated or deleted. Previous versions of entries are stored in log tables <table name>log (see Log Tables).

2.3 Diagram

For a UML style diagram of the database, see file SLIMS_databaseDiagram.jpg.

3 Key Application Files:

3.1 icapture.com

3.1.1 ApplicationListener.java

This class controls opening and closing of hibernate mappings for the web application. It should be included in the listener section of the configuration file for the application (see listener section in web.xml)

3.1.2 ColumnData.java

This file enables the adding and removing of columns from a user’s view of container contents data. It stores the default views for each user type and an array for every possible column to be added.

3.1.3 DataExport.java

When a user wants to export a table (from search or browse or list of containers etc), this file makes appropriate headers for the columns of data in the file and prepared the data to be exported (method: getHeader(…)). After the data is prepared, it can be accesses by an iterator and an overridden next() method that returns data line by line in tab delimited format.

3.1.4 FileUploading.java

Enables files to be uploaded to create new subjects, samples, containers and shopping lists. Has methods to check a file for upload and return any lines that have problems. Also has methods to create elements.

3.1.5 FilterObject.java

Sets up power searches. Checks every request field for input and if it is found, adds that field to the query (method: setFilter(HttpServletRequest request)). Prepares strings to add to queries that return search results (get<Object>String()).

3.1.6 MetadataManager.java

This file builds the profiles describing what data a user will view for each object (which is based on the Meta_Data Tables). It gets which fields need to be viewed in what order from the MetaData tables and the actually values for these fields from the Object.java’s getValueArrayReadable(). It then consolidates these two information arrays to build an array that will be shown to the user.

3.1.7 Plater.java

Keeps track of plating operations: what samples need to be plated, what samples have been plated, order of samples based on phenotypes, what controls need to be plated, what controls have been plated, how much space there is on a plate after adding controls etc.
This is used for make 96 well plates from a list of contents as well as making 384 well plates from a list of 96 well plates.

3.1.8 SampleSelector.java

Manages and tracks the decision tree structure being made by the user. Prepares and runs queries to fetch the 'best-match' or 'next-best-match' contents. Manages accepting and rejecting of contents and storage of those accepted.

3.1.9 SearchObject.java

Sets up simple searches. Checks the request field for input and what to search and uses those fields to set up the the query (method: setUpSearch (HttpServletRequest request)). Prepares strings to add to queries that return search results (get<Object>String()). This file is a simplified version of FilerObject.java

3.1.10 SessionListener.java

This class supports session management for the web application. It should be included in the listener section of the configuration file for the application (see listener section in web.xml)

3.1.11 UserHttpSess.java

Manages setting up the application and contains nearly all the database querying methods. This includes methods to add, edit and delete entries in the database, fetch all objects of a certain type (and potentially only those that satisfy search criteria) and get an object by its database ID. It also stores 'current' variables shared throughout the application such as power and simple search objects, a user's ID, type, and permission status, a plating object, the active shopping list, and active objects (like containers, subjects etc—though these can be unreliable).

3.1.12 View<Object>Manager.java

Manages table view screens, particularly multi-page navigation of tables with more entries than the set view limit per page (usually 1000).
ViewContainerContentManager.java also manages the processing, adding and removing steps involved when a user customizes or resets their columns for that screen.

3.2 icapture.hibernate

3.2.1 <Table>.hbm.xml

Maps database table to java object.
This file tells SLIMS what each column of a table should be called within SLIMS, what type of variable it is (optional) and whether it can be null. If a table has foreign keys, these are handled specially, see object links

3.2.2 <Table>.java

Java class for the object representation of a table.
This file has getter/setter methods for every field and a getValueArrayReadable() method to return an array of values for all the fields that are to be displayed to the user. Which fields are to be displayed to users are defined in the '_meta_data' tables (see Meta_Data Tables) and icapture.com.MetadataManager.java (see INSERT LINK)

3.2.3 Persistent.java

This is the parent class of objects that represent tables mapped by Hibernate. Its common fields are for id, visibleName, creator, modifier, createDate, modifDate and comment plus it has getters and setters for each.

3.2.4 TempMetaData.java

Object class for meta data tables.

3.3 Reports

3.3.1 <ReportName>.jasper

This is the xml file that defines how the report will be filled and will look. It must be compiled before the report can be accessed.

3.3.2 < ReportName>.jrxml

This is the compiled version of the .jasper file of the same name, it is this file that the user will access when they request a report.

3.4 Web Application Files

3.4.1 JSPs

File	Description
Add<Object>.jsp	JSP for generating add/edit page for the relevant object and handling validation and execution of add or edit.
Add<Object>E.jsp	Form section of Add<Object>.jsp. Can be included in Add<Object>.jsp as many times as necessary in different logic cases.
Add96WellPlates.jsp	Customized 'add container' page for 96 well plates. Used for plating tool.
Add96WellPlatesE.jsp	Input form section for Add96WellPlates.jsp
AddControlWellsToContainer.jsp	Allows user to add control wells to a 96-well plate that they are making using an existing control layout. Used when making a new 96-well plate from a list of samples.
AddControlWellsToContainerE.jsp	Input form section for AddControlWellsToContainer.jsp
AddControlWellsToContainerManual.jsp	Allows user to design a new control layout and add their controls to a 96-well plate that they are making. Used when making a new 96-well plate from a list of samples. Linked to from AddControlWellsToContainer.jsp
AddControlWellsToContainerManualE.jsp	Input form section for AddControlWellsToContainerManual.jsp
AddListContentsToContainer.jsp	Allows user to add samples to their new plate, specifying how much source material is to be used and what the final volumes and concentrations will be of the samples in their new plate.
AddListContentsToContainerE.jsp	Input form section for AddListContentsToContainer.jsp
AdminMenu.jsp	Dropdown navigation element for all ‘supporting data’ view pages.
AdminPage.jsp	Front page for ‘Supporting Data’
Browse.jsp	Front page for browsing data, has links to view pages for subjects, contents and containers
CheckFile.jsp	When making a list from a file, this jsp checks the file for formatting, validity of entries etc and shows the users the results of its checks. It also allows the user to accept the file given the results of the check and proceed in creating the list. Linked to from ChooseFileToLoad.jsp
ChooseFileToLoad.jsp	When making a list from a file, this jsp shows the user what kind of files that can upload and what format is needed for each. The user can choose and attach a file to the request sent to CheckFile.jsp
CloneContainer.jsp	Allows the user to clone a plate. The user must enter the details of the new plate and how much material is to be taken from the source plate.
CloneContainerE.jsp	Input form section for CloneContainer.jsp
DefineFilter.jsp	Power search interface, sets the filter object in UserHttpSess
EditBulkContents.jsp	Allows the user to update the volumes and comments of all the contents in a container

EditBulkContentsE.jsp	Input form section for EditBulkContents.jsp
Export.jsp	Prepares a text file version of a search result or browse for download
ExportList.jsp	Prepares a text file version of a list's search result or list browse for download
Footer.jsp	Chunk of html and jsp coding included at the bottom of every page
Header.jsp	Chunk of html and jsp coding included at the top of every page
HowManyPlates.jsp	Used in the tool that makes a plate from a list of samples. When first using this, it lets the user decide on whether phenotype information should be taken into account during the laying out of samples on plates. It also tells the user how many plates they will need to make (at least, can increase depending on how many controls are used). User is brought back to this screen after each plate is made during the plating job (except for the last plate made which brings them to PlatingSummary.jsp).
Index.jsp	Home page for SLIMS
ListTrimTool.jsp	Allows the user to search for items that either should be kept or remove from their list and lets them either edit their own list to reflect the results or use the results to make a new list.
ListVolumeUpdate.jsp	Allows the user to update the volumes and comments of all the containerContents in a list
LoadNewContainersCheckFile.jsp	When bulk loading containers from a file, this jsp checks the file for formatting, validity of entries etc and shows the users the results of its checks. It also allows the user to accept the file given the results of the check and proceed in creating the containers. Linked to from LoadNewContainersChooseFile.jsp
LoadNewContainersChooseFile.jsp	When bulk loading containers from a file, this jsp shows the user what kind of file they can upload and what format is needed. The user can choose and attach a file to the request sent to LoadNewContainersCheckFile.jsp
LoadNewContentsCheckFile.jsp	When bulk loading contents from a file, this jsp checks the file for formatting, validity of entries etc and shows the users the results of its checks. It also allows the user to accept the file given the results of the check and proceed in creating the contents. Linked to from LoadNewContentsChooseFile.jsp.
LoadNewContentsChooseFile.jsp	When bulk loading "samples" from files, this jsp shows the user what kinds of files they need to upload and what format is needed. This jsp allows them to load the second of two needed files, the one that creates containerContents. The user can choose and attach a file to the request sent to LoadNewContentsCheckFile.jsp
LoadNewSamplesCheckFile.jsp	When bulk loading samples from a file, this jsp checks the file for formatting, validity of entries etc and shows the users the results of its checks. It also allows the user to accept the file given the results of the check and proceed in creating the samples. Linked to from LoadNewSamplesChooseFile.jsp. When the user accepts their file and proceeds to create the samples, they are directed to the second step in creating "samples", creating

	contents (LoadNewContentsChooseFile.jsp).
LoadNewSamplesChooseFile.jsp	When bulk loading “samples” from files, this jsp shows the user what kinds of files they need to upload and what format is needed. This jsp allows them to load the first of two needed files, the one that creates samples. The user can choose and attach a file to the request sent to LoadNewSamplesCheckFile.jsp
LoadNewSubjectsCheckFile.jsp	When bulk loading subjects from a file, this jsp checks the file for formatting, validity of entries etc and shows the users the results of its checks. It also allows the user to accept the file given the results of the check and proceed in creating the subjects. Linked to from LoadNewSubjectsChooseFile.jsp
LoadNewSubjectsChooseFile.jsp	When bulk loading subjects from a file, this jsp shows the user what kind of file they can upload and what format is needed. The user can choose and attach a file to the request sent to LoadNewSubjectsCheckFile.jsp
LogIn.jsp	First page users sees when visiting SLIMS, allows them to log in
Make384From96.jsp	This file lets the user make a 384 well plate from the 96 well plates in a list of containers. It lets them name the new plates and set the layout for the 384 well plate. When the user is done, this jsp creates the new plates, updates the source volumes and directs the user to PlatingSummary.jsp
Make384From96E.jsp	Input form section for Make384From96.jsp
MakeSDS.jsp	Code to make an SDS import file for a plate. Writes the file to the user’s machine. This is accessed through a button in the container viewing tables
PlatingSummary.jsp	Shows a summary of the plates just made, for each plate it shows: name, layout and links to reports for them. When the user is finished, they are directed to SLIMS home page and their active list is closed.
QueryExport.jsp	Code to write the results of an SQL query to the user’s machine
QuerySearch.jsp	JSP for user to input and run an SQL query
SampleSelectorCrtieriaChoice.jsp	Sample selector: allows the user to set up their selection criteria tree
SampleSelectorCrtieriaSubChoice.jsp	Sample selector: allows the user to specify details for a criterion selected from SampleSelectorCrtieriaChoice.jsp, if necessary.
SampleSelectorCrtieriaChoiceProfile.jsp	Unused ‘profile-based’ version of SampleSelectorCrtieriaChoice.jsp (as opposed to the tree version)
SampleSelectorFetch,jsp	Sample selector: fetches the best match container contents for the current selection profile
SampleSelectorSummary.jsp	Sample selector: displays a summary of the selection tree the user has set up and asks if the user wants to continue the process with this tree.
SampleSelectorViewResults.jsp	Sample selector: shows the user the next batch of best-match sample and lets the user accept or reject them. Subjects for which no samples have been found/accepted are also listed
Search.jsp	Front page for searching data, has links to simple search, power search and SQL query search
SelectReport.jsp	Shows the user what reports are available

SimpleSearch.jsp	Lets the user perform a single-field search of samples, containers and subjects. Clears the filter object in UserHttpSess and sets the search object
Test_JSP_error.jsp	Error page
Tools.jsp	Page to link to loading containers subjects and samples from files
View<Objects>.jsp	View table screen for objects
View<Objects>List.jsp	View table screen for lists of objects
ViewLayout.jsp	Opens in new window from a link on a view containers page. Shows a layout of the containers with sample IDs and volumes
ViewLists.jsp	Jsp for list manager, controls loading of premade lists and closing lists. Has a link to make a new list from a file. Lets the user search for lists by name, creator and modifier.
ViewListsE.jsp	Input form section for ViewLists.jsp
ViewReport.jsp	User doesn't see thing page, it's responsible for loading the requested jasper report and opening it for the user

3.4.2 Other Files

File	**Description**
ddsmoothmenu.css	Css for top menu (dropdown functionality)
ddsmoothmenu.js	Javascript functions for top menu (dropdown functionality)
formstyle.css	Css for web application
utils.js	Common javascript functions for application

3.5 Configuration Files

web.xml, sections of note:

Code	**Description**
`<param-name>viewlimit</param-name>` `<param-value>1000</param-value>`	1000 – number of items viewable per page. (If more than 1000 items in a view table, it will be split among >1 page.)
`<param-name>seconduser</param-name>` `<param-value>username/password</param-value>`	user with username and password will be able to login with administrator rights (and will automatically be added to user) The option can be useful to get access to database, if all information about passwords has been lost
`<listener-class>molpage.com.SessionListener</listener-class>`	see `SessionListener.java`
`listener-class>molpage.com.ApplicationListener</listener-class>`	see `ApplicationListener.java`
`<welcome-file>LogIn.jsp</welcome-file>`	Start page of the application (see `LogIn.jsp`)
`<exception-type>java.lang.Exception</exception-type>` `<location>/Test_JSP_error.jsp</location>`	Error page for "unexpected" runtime exceptions (see `Test_JSP_error.jsp`)

4 Common Tasks

4.1 Adding a table as an object in SLIMS

4.1.1 Using the object

- Create a hibernate mapping file (<Object>.hbm.xml) so that entries in the table can be interacted with like objects
- Create a java class file to define the object (<Object>.java) with all desired fields plus getters and setters
- In constructor UserHttpSess(String logPath), add the class to hibernate configuration in the line that starts:

  "cfg = new Configuration().addClass(User.class)….."
- In method UserHttpSess,openFactory(), add the class to hibernate configuration in the line that starts:

  "cfg = new Configuration().addClass(User.class)….."
- Add new required fields to Fieldname.java
- Add methods to UserHttpSess.java (see other objects' versions for template)
  - getViewUserManager()
  - getCurrentUser()
  - getCurrentUserId()
  - getAllUsers(int startPosition)
  - getAllUsersCount()
  - setUserSortCol(String sortCol)
  - getUserSortId(String sortCol)
  - setCurrentUser(String userID)
- Rebuild SLIMS

4.1.2 Viewing the table

- Do steps in 4.1.1
- Add a meta data table to the database for the table you want to add (see Meta_Data Tables) to control what columns are displayed
- Create a hibernate mapping file for the meta data table (<Object>MetaData.hbm.xml)
- Create a java class file for the meta data table object (<Object>MetaData.java)
- Create a View<Object>Manager.java file to control pagination of table viewing
- Create a View<Object>s.jsp file similar to existing View<Object>s.jsp files
- Add methods to MetaDataManager
- Add to methods of UserHttpSess:
  - UserHttpSess(HttpSession s)
  - buildMetaData()
  - resetCurrents()
  - setSettings()
  - cancelFilter()
  - cancelFilterList()
  - cancelSearch()
  - cancelSearchList()
- Rebuild SLIMS

4.1.3 Adding or updating values

- Create an Add<Object>.jsp file similar to existing Add<Object>.jsp files. Make sure to validate input here.
- Create an Add<Object>E.jsp file similar to existing Add<Object>E.jsp files. This will need to have input elements for every field you want to be able to edit or fill.

- Add methods to UserHttpSess.java (see other objects' versions for template)
  - checkUserId(String userID)
  - addUser(…)
  - updateUser(…)
- Rebuild SLIMS

4.2 Adding a field to a table in SLIMS

a. add a property or many-to-one tag for the field to <Table>.hbm.xml
b. add the field to <Table>.java in the declarations section, then add getter and setter methods. Finally, add an if statement to the field in method getValueArrayReadable(). How the string is passed here will be how it shows up in the table, so do any formatting at this point. Take not of the array index number you are assigning for your variable.
c. Create an entry for your field in fieldname.java. For example, adding field "location" would mean a line in fieldname.java like:
   public static final String LOCATION = "location";
d. In MetadataManager.java, method build<Table>Data(), add your field to the end of the "m = (key.equals(Fieldname.<…>" statement. Make sure the number associated with your field matches the array index number you took note of in step b.
   Note: if you are update buildContainerContentData(), be sure to also update field ccMetaDataAllColCount.
e. In database table tblcontainers_meta_data, add an entry for your field. See Meta_Data Tables section for column details.
f. Rebuild the application and see your new field!

4.3 Adding a field to containerContents table in SLIMS

a. add a property or many-to-one tag for the field to <Table>.hbm.xml
b. add the field to <Table>.java in the declarations section, then add getter and setter methods. Finally, add an if statement to the field in method getValueArrayReadable(). How the string is passed here will be how it shows up in the table, so do any formatting at this point. Take not of the array index number you are assigning for your variable.
c. Create an entry for your field in fieldname.java. For example, adding field "location" would mean a line in fieldname.java like:
   public static final String LOCATION = "location";
d. In MetadataManager.java, method build<Table>Data(), add your field to the end of the "m = (key.equals(Fieldname.<…>" statement. Make sure the number associated with your field matches the array index number you took note of in step b.
e. Be sure to also update field ccMetaDataAllColCount (increase by the number of fields being added).
f. Add an object[] for your field in the declarations section of ColumnData.java (this is the data that will be used to create entries in tblcontainercontents_meta_data for your user if they add this field to their view scheme). Also, add the field to the default views, where appropriate (if your field is not in tblcontainercontents, make sure to preface it with the object that connects it to ContainerContents, for our example this would be "container.location"). Also, add your field to the hashmap in makeHash()
g. Add 'check' and 'num' entries to EditColumns.jsp and EditColumnsE.jsp
h. To see your new field in SLIMS, add it to your view scheme through SLIMS' interface (see User manual) or update to your default view settings if you added the new column to your user type's default scheme.

4.4 Adding non-containercontents field to containercontents view table (optional column)

a. create the field in fieldname.java with its variable name being: <OBJECT><FIELD>. For example, adding tblcontainer.location would mean a variable name like: “CONTAINERLOCATION”. The value of this variable will be: <object>.<field>, for our example, the line in fieldname.java would be:

   public static final String CONTAINERLOCATION = "container.location";

   in MetadataManager.java, method buildContainerContentData(), add this field to the end of the “m = (key.equals(Fieldname.<…>” statement. Take note of the number associated with your new entry. Be sure to also update field ccMetaDataAllColCount (increase by the number of fields you just added).
b. In ContainerContent.java, method getValueArrayReadable(), add your new field on to the series of if statements. How the string is passed here will be how it shows up in the table, so do any formatting at this point. Make sure your array index here matches the number you took note of in step b.
c. Add an object[] for your field in the declarations section of ColumnData.java (this is the data that will be used to create entries in tblcontainercontents_meta_data for your user if they add this field to their view scheme). Also, add the field to the default views, where appropriate (if your field is not in tblcontainercontents, make sure to preface it with the object that connects it to ContainerContents, for our example this would be “container.location”). Also, add your field to the hashmap in makeHash()
d. Add ‘check’ and ‘num’ entries to EditColumns.jsp and EditColumnsE.jsp