Sample Tracking

Issue Creator

JIRA CLI Interface

https://studio.plugins.atlassian.com/wiki/display/JCLI/JIRA+Command+Line+Interface

Command Line:

/usr/local/java/1.6.0_23/bin/java -jar ../lib/jira-cli-2.1.0.jar -a runFromCSV --server http://jira/ --user pedworth --password A@BTaxis1 --file samples.csv --common "--action createIssue" --continue

Explanation of Command Line options:

- **/usr/local/java/1.6.0_23/bin/java** Java 1.6+ is required
- **-jar ../lib/jira-cli-2.1.0.jar** The actual program
- **--action runFromCSV** A CSV file will be used to construct the commands. The format of the file is described below.
- **--server http://jira/** The URL to connect to
- **--user pedworth** The user to carryout the actions as, the user must be an administrator.
- **--password A@BTaxis1** Nasty hard coded password. A method of using a password from stdin is planned, which will remove the password from ps and command history. There must be something better though.
- **--file samples.csv** The file containing the information about the sample to create. The format is described below.
- **--common "--action createIssue"** The option in quotes is passed to the invocations made when processing the CSV file. It indicates that the contents of the CSV file should be treated as parameters to the action 'createIssue'.
- **--continue** If one line of the CSV data fails continue with the rest of the file

CSV File:

```
Project,type,summary,customfield_10153,customfield_10161,customfield_10152
Sample Tracking,Sample,giv3_NHRCA_36031,36031,36031,36031
```

CSV File's Format

The file contains two areas, the header row and the body. The header row describes the contents of the body. The body contains one row per sub-invocation of the CLI.

The header consists of a comma separated list of 'column names'. The 'column names' are used to transform a row into a command line. The 'column names' are used to form the names of the

command line options, while the corresponding body row entry is used for the value. e.g.

```
option1, option2
row1valueForOption1,row1valueForOption2
row2valueForOption1,row2valueForOption2
```

Produces results equivalent to running

```
jira-cli --option1 "row1valueforOption1" --option2 "row1valueForOption2" <contents of common>
jira-cli --option1 "row2valueforOption1" --option2 "row2valueForOption2" <contents of common>
```

Custom fields are handled specially, as they are set with a single option (custom) which encodes the multiple name value pairs as a comma separated list of of key:value pairs. e.g.

```
option1, customField1, customField2
row1valueForOption1,row1valueForCustomField1,row1valueForCustomField2
row2valueForOption1,row2valueForCustomField1,row2valueForCustomField2
```

Produces results equivalent to running

```
jira-cli --option1 "row1valueforOption1" <contents of common>
       --custom "customField1:row1valueForCustomField1,customField2:row1valueForCustomField2"
jira-cli --option1 "row2valueforOption1" <contents of common>
       --custom "customField1:row2valueForCustomField1,customField2:row2valueForCustomField2"
```

Fields that are of type 'Database Values Plugin' are set to the Extent ID. This may change when multiple databases are being supported to a compound key of database and Extent ID.

Full equivalent command

```
Run:   --action createIssue
       --summary "giv3_NHRCA_36032"
       --project "Sample Tracking"
       --type "Sample"
           --custom
           "'customfield_10153:36032','customfield_10161:36032','customfield_10152:36032','customfield_10155:No','customfield_10156
           :RNA'"
```

Output

From

```
<blank line>
Run: --action createIssue --summary "giv3_NHRCA_36031" --project "Sample Tracking" ...
Issue ST-718 created.
<blank line>
Run: --action createIssue --summary "giv3_NHRCA_36032" --project "Sample Tracking" ...
Issue ST-719 created.
<blank line>
```

Run completed successfully. 2 actions were successful from file: /local/devel/VIRIFX/users/pedworth/workspace/atlassian-jira-cli-2.1.0/tests/samples.csv

Flurp/Rurp integration

Flurp/Rurp summary

Flurp / Rurp are the initiating point for storing information about samples in the GLK. They use the collaborator spreadsheet to:

- if necessary, add a collection row to Extent Table with;
  - ref_id set to the collection's id (e.g. MALA)
  - parent_id set to point to the 'Genome' record for the project.
  - Extent_Type_id set to the value for type 'COLLECTION' looked up in the Extent_Type table.
- add a lot row to Extent Table with;
  - ref_id set to the lot's id (e.g. MALA001)
  - parent_id set to point to the collection record
  - Extent_Type_id set to the value for type 'LOT' looked up in the Extent_Type table.
- add a row per Sample to Extent Table with;
  - ref_id set to the sample's BAC id (35531)
  - parent_id set to point to the lot record
  - Extent_Type_id set to the value for type 'SAMPLE' looked up in the Extent_Type table.
- add rows of Meta data about the samples to the Extent Attribute table.

Sample Tracking issue creation requirements

The following fields are initialized when an issue is created:

- Database (e.g. giv)
- Collection Code (e.g. MALA)
- BAC Id (e.g. 35531)
- Summary / Sample Id (e.g. giv_MALA_35531)
- Blinded Number (e.g. NIGSP_MALA_00003)
- Computed Subtype (Set to the sample's Extent ID to look-up its value dynamically)

Implementation

Manual Issue Creation Using the output from Flurp/Rurp

Flurp and Rurp return a table of the samples inserted and IDs associated with them.

FluRP : The Flu Receiving Program

Lot EUID is: 1130342620556

Blinded Number	BAC ID	Library ID	Cat#	Sample Name	Extent ID	Species Code
NIGSP_MALA_00003	37645	JQUM	T42449	MALA00003	1130342620557	Influenza A virus (A/Malaysia/07145/1995(H3N2))
NIGSP_MALA_00004	37646	JQUN	T42450	MALA00004	1130342620558	Influenza A virus (A/Malaysia/07831/1995(H3N2))
NIGSP_MALA_00005	37647	JQUO	T42451	MALA00005	1130342620559	Influenza A virus (A/Malaysia/07832/1995(H3N2))
NIGSP_MALA_00008	37648	JQUP	T42452	MALA00008	1130342620561	Influenza A virus (A/Malaysia/10081/1996(H3N2))
NIGSP_MALA_00009	37649	JQUQ	T42453	MALA00009	1130342620562	Influenza A virus (A/Malaysia/10135/1996(H3N2))
NIGSP_MALA_00010	37650	JQUR	T42454	MALA00010	1130342620563	Influenza A virus (A/Malaysia/10111/1996(H3N2))
NIGSP_MALA_00011	37651	JQUS	T42455	MALA00011	1130342620564	Influenza A virus (A/Malaysia/10360/1996(H3N2))
NIGSP_MALA_00012	37652	JQUT	T42456	MALA00012	1130342620565	Influenza A virus (A/Malaysia/10370/1996(H3N2))
NIGSP_MALA_00013	37653	JQUU	T42457	MALA00013	1130342620566	Influenza A virus (A/Malaysia/10807/1996(H3N2))
NIGSP_MALA_00014	37654	JQUV	T42458	MALA00014	1130342620567	Influenza A virus (A/Malaysia/10675/1996(H3N2))

Illustration 1: Output of Flurp after samples have been inserted.

The JIRA fields are filled in the following way:

- Summary / Sample Id; a combination of Database, Collection Code and BAC Id where
  - Database is provided separately
  - Collection Code is parsed from Blinded Number
  - BAC id is in the output
- Database; see summary
- Collection Code ; see summary or set to the sample's Extent ID (from the output). Extent ID can be used to dynamically looked up the value later.
- BAC Id; see summary
- Blinded Number; from the output
- Computed Subtype; Extent ID (from the output). Extent ID can be used to dynamically looked up the value later.

Manually initiated, Automatic Issue Creation

Nadia's spec defines the format for the files as:

<db>,<collection code>,<bac_id>

The JIRA fields are filled in the following way:

- Summary / Sample Id; a combination of Database, Collection Code and BAC Id all of which are in the input file.
- Database; in the input file
- Collection Code ; in the input file
- BAC Id; in the input file
- Blinded Number; Would have to be read from the GLK by JIRA using the BAC Id and database.
- Computed Subtype; Would have to be read from the GLK by JIRA using the BAC Id and database.

Automatically Issue Creation from within Flurp/Rurp

Flurp/Rurp a convenient place for adding the JIRA sample initialization code is in load_data(&$idata) in load.php. The actual GLK Extents are created during getSample($bac_id) which works as get or create Extent of type SAMPLE and ref_id = $bac_id. The JIRA call could take place just after the getSample call completes.

The JIRA fields are filled from the following sources:

- Summary / Sample Id; a combination of Database, Collection Code and BAC Id where
  - Database comes from the post ($_POST['db_name'])
  - Collection Code comes from the opt_data item (opt_data['collection_code'])
  - BAC id comes from a local variable ($bac_id)
- Database; see summary
- Collection Code ; see summary or set to the sample's Extent ID ($sample_euid). Extent ID can be used to dynamically looked up the value later.
- BAC Id; see summary
- Blinded Number; from the output
- Computed Subtype; Extent ID ($sample_euid). Extent ID can be used to dynamically looked up the value later.

GLK fields populated by Flurp

blinded_number	1515	
center_project	1517	
collection_date		1525
concentration	1532	
country	1533	
date_sent_to_JCVI	1537	
district	1541	
extraction_date	1550	
extraction_method	1554	
host	1561	
library_id	1567	
passage_history	1583	
sample_name	1589	
sample_number	1590	
source_type	1602	
species_code	1604	
subtype	1610	
type	1612	
days_to_hold	1661	
batch_id	1665	
CEIRS_sample_id	1667	
sample_plate_location		1668

Scripts

SQL to create CSV file for JIRA CLI

```
\echo Please enter DB to use lines beginning '\' are sqsh commands and not SQL
\read db read is used to get user input and store it in a variable
\echo Please enter lot code to use
\read  lot
\set file="/tmp/ST-jira.csv" assigns a value to the variable
The next line puts the header in the CSV file
\echo Project,type,summary,customfield_10120,customfield_10121,customfield_10122,customfield_10100,customfield_10126,customfield_10123,customfield_10224 > $file
select The actual query, only one query is used but it contains many sub-queries
  "Sample Tracking,Sample,"+ String construction is used as formatting cannot be easily controlled when multiple columns are selected
  "${db}_" + collection.ref_id + "_" + sample.ref_id +","+
  "${db},"+
  collection.ref_id +","+
  sample.ref_id +","+
  "${db}_" + collection.ref_id + "_" + sample.ref_id +","+
  sub_type +","+
  blinded_number+","+
  CONVERT(VARCHAR, sample.Extent_id) The string is very large, the program calling should use the /w option to set a high line width
from
  ( 1. The first sub-query, gets the sample's details
    select inner_sample.Extent_id, inner_sample.ref_id,
      MAX(CASE WHEN type='subtype' THEN value END) as sub_type, MAX and CASE are used to convert row based data into columns in a single row. See 'ST DB Custom Field'
      MAX(CASE WHEN type='blinded_number' THEN value END) as blinded_number,
      MAX(CASE WHEN type='jira_id' THEN value END) as jira_id,
      MAX(CASE WHEN type='deprecated' THEN value END) as deprecated
    from
      ( 1.1 sub-query to get the sample attributes table with human readable types
        select Extent_id, type, value
        from ${db}..ExtentAttribute ea join ${db}..ExtentAttributeType eat
        on ea.ExtentAttributeType_id = eat.ExtentAttributeType_id
      ) attrib
    join
      ( 1.2 sub-query to select the BAC ids and Extent ids of the samples
        select Extent_id, ref_id from $db..Extent
        where parent_id in (Select Extent_id from ${db}..Extent where ref_id = "$lot") sub-query 1.2.1 finding all of the child Extents (samples) of the lot Extent
      ) inner_sample
    on attrib.Extent_id = inner_sample.Extent_id
    group by inner_sample.Extent_id
  ) sample 1. the join of the sample's id's (1.2) and named attributes (1.1)
join
  ( 2. sub-query to find the collection code, stored in ref_id of the collection extent
    select ref_id from ${db}..Extent e join (Select parent_id from ${db}..Extent where ref_id = "$lot") sub-query 2.1 = 1.2.1
    on e.Extent_id = l.parent_id lot extents are the children of collection extents
  ) collection
on "1" = "1" the on clause appears to be compulsory
where
  jira_id is null and true if the sample hasn't been added before
  deprecated is null don't bother adding deprecated samples
go | sqsh command to terminate the query and execute it. The pipe send the output through the following chain of commands/filters
       sed '/^\W*$/d' | remove blank and empty lines
       sed '/rows affected/d' | remove the summary row
       sed 's/^.*\(Sample [,_A-Za-z0-9]*\).*$/\1/' remove white space before and after
```

```
      >> ${file} save the output to /tmp/ST-jira.csv
\echo file written give some feedback
\quit The script must return to the prompt for the rest of the bash script to executeScript to run SQL scripts and JIRA CLI
#!/usr/local/bin/bash
#create the csv file > /tmp/ST-jira.csv
sqsh -w 255 -S SYBPROD -i /usr/local/devel/VIRIFX/users/pedworth/bin/sql/create_jira_issues.sql
#insert the samples into jira < /tmp/ST-jira.csv > /tmp/ST-jira.created
/usr/local/java/1.6.0_23/bin/java -jar \
  /usr/local/devel/VIRIFX/users/pedworth/bin/atlassian-jira-cli-2.1.0/lib/jira-cli-2.1.0.jar \
  -a runFromCSV \
  --server http://jira/ \
  --user sampletracking \
  --password a2c4e6g8 \ embedding the password isn't good
  --file /tmp/ST-jira.csv \ the input is from the sqsh script
  --common "--action createIssue" \
  --continue \ don't stop if one of the samples can't be created
  > /tmp/ST-jira.created this hides the feedback from the user, change to tee?
#convert the insertion messages into sql < /tmp/ST-jira.created > /tmp/ST-jira-update.sql
cat /tmp/ST-jira.created | \ start with the output of jiracli
sed '/^$/d'| \ remove blank lines
sed '$!N;s/\n/ /' | \ merge every other line 1\n2\n3\n4\n -> 12\n23\n34
grep createIssue | \ filter out the summary messages
Find the values of customfield_10120, customfield_10224 and the Issue created. Create an insert statement from them
See 'Parsing the output of the create script for details'
sed 's/^.*customfield_10120:\([a-z]*\)*.*customfield_10224:\([0-9]*\)[^0-9].*Issue \(ST-[0-9]*\) .*$/<originally on a single line>
       insert \1..ExtentAttribute (Extent_id, ExtentAttributeType_id, value) <originally on a single line>
              select \2 as Extent_id, ExtentAttributeType_id, "\3" as value from \1..ExtentAttributeType where type = "jira_id";/' \
> /tmp/ST-jira-update.sql
sqsh -S SYBPROD -i /tmp/ST-jira-update.sql run the sql generated from the insertion log
```

Parsing the output of the create script

Breaking the command down:

cat /tmp/ST-jira.created

Passes the output into the chain

sed '/^$/d'

Removes blank lines, particularly important as we are trying to merge every other non blank line

The 'command' is the pattern for the empty string with d for delete appended.

sed '$!N;s/\n/ /'

Merge alternate lines (1\n2\n3\n4\n -> 12\n23\n34\n)

grep createIssue

The final two lines aren't from samples being added and should be deleted. This only allows lines that contain createIssue to carry on.

sed 's/<search pattern>/<replace pattern>/' /tmp/ST-jira-update.sql

The search and replace command, as the pattern matches from the start of the string '^' to the end '$' it acts like the output is generated from the input, not like the output is actually the input that has been manipulated.

The search pattern is:

```
^.*customfield_10120:\([a-z]*\)*.*customfield_10224:\([0-9]*\)[^0-9].*Issue \(ST-[0-9]*\) .*$
```

^	from the start of the string
.*	match anything
customfield_10120:	until this pattern is found (Database field)
\(	remember everything that matches until the close bracket (as \1)
[	the start of a set of characters to match
a-z	alphabetic characters
]	the end of the set
*	match as many characters, that match the type of the set, as possible
\)	finish storing the matched section
*	possibly a mistake, although it still works
.*	match anything
customfield_10224:	until this pattern is found (Extent ID field)
\(	remember everything that matches until the close bracket (as \2)
[0-9]	The set of number characters
*	match as many characters, that match the type of the set, as possible
\)	finish storing the matched section
[^0-9]	until a character this is not a number
.*	match anything
Issue	until this pattern is found
\(	remember everything that matches until the close bracket (as \3)
ST-	match these characters exactly
[0-9]*	As many number characters as possible
\)	finish storing the matched section
<space>	until a space is encountered
.*	match anything
$	until the end of the string

Example outputs

/tmp/ST-jira.csv

```
Project,type,summary,customfield_10120,customfield_10121,customfield_10122,customfield_10100,customfield_10126,customfield_10123,customfield_10224
Sample Tracking,Sample,giv_RFH3_38157,giv,RFH3,38157,giv_RFH3_38157,H3N2,NIGSP_CEIRS_CIP047_RFH3_00097,1131006253387
Sample Tracking,Sample,giv_RFH3_38125,giv,RFH3,38125,giv_RFH3_38125,H3N2,NIGSP_CEIRS_CIP047_RFH3_00061,1131006253355
Sample Tracking,Sample,giv_RFH3_38105,giv,RFH3,38105,giv_RFH3_38105,H3N2,NIGSP_CEIRS_CIP047_RFH3_00038,1131006253335
Sample Tracking,Sample,giv_RFH3_38339,giv,RFH3,38339,giv_RFH3_38339,H3N2,NIGSP_CEIRS_CIP047_RFH3_00283,1131006253569
Sample Tracking,Sample,giv_RFH3_38189,giv,RFH3,38189,giv_RFH3_38189,H3N2,NIGSP_CEIRS_CIP047_RFH3_00130,1131006253419
Sample Tracking,Sample,giv_RFH3_38151,giv,RFH3,38151,giv_RFH3_38151,H3N2,NIGSP_CEIRS_CIP047_RFH3_00091,1131006253381
Sample Tracking,Sample,giv_RFH3_38130,giv,RFH3,38130,giv_RFH3_38130,H3N2,NIGSP_CEIRS_CIP047_RFH3_00067,1131006253360
Sample Tracking,Sample,giv_RFH3_38299,giv,RFH3,38299,giv_RFH3_38299,H3N2,NIGSP_CEIRS_CIP047_RFH3_00241,1131006253529
Sample Tracking,Sample,giv_RFH3_38085,giv,RFH3,38085,giv_RFH3_38085,H3N2,NIGSP_CEIRS_CIP047_RFH3_00018,1131006253315
Sample Tracking,Sample,giv_RFH3_38248,giv,RFH3,38248,giv_RFH3_38248,H3N2,NIGSP_CEIRS_CIP047_RFH3_00189,1131006253478
Sample Tracking,Sample,giv_RFH3_38226,giv,RFH3,38226,giv_RFH3_38226,H3N2,NIGSP_CEIRS_CIP047_RFH3_00167,1131006253456
Sample Tracking,Sample,giv_RFH3_38204,giv,RFH3,38204,giv_RFH3_38204,H3N2,NIGSP_CEIRS_CIP047_RFH3_00145,1131006253434
Sample Tracking,Sample,giv_RFH3_38165,giv,RFH3,38165,giv_RFH3_38165,H3N2,NIGSP_CEIRS_CIP047_RFH3_00106,1131006253395
<continues for 200+ lines>
```

/tmp/ST-jira.created

```
<starts 200+ icreations earlier; the first line is blank>
<blank line>
Run: --action createIssue --summary "giv_RFH3_38150" --project "Sample Tracking" --type "Sample" --custom
"'customfield_10122:38150','customfield_10123:NIGSP_CEIRS_CIP047_RFH3_00090','customfield_10100:giv_RFH3_38150','customfield_10120:giv','customfield_10121:RFH3','custo
mfield_10224:1131006253380','customfield_10126:H3N2'"
Issue ST-1006 created.
<blank line>
Run: --action createIssue --summary "giv_RFH3_38263" --project "Sample Tracking" --type "Sample" --custom
"'customfield_10122:38263','customfield_10123:NIGSP_CEIRS_CIP047_RFH3_00204','customfield_10100:giv_RFH3_38263','customfield_10120:giv','customfield_10121:RFH3','custo
mfield_10224:1131006253493','customfield_10126:H3N2'"
Issue ST-1007 created.
<blank line>
Run: --action createIssue --summary "giv_RFH3_38236" --project "Sample Tracking" --type "Sample" --custom
"'customfield_10122:38236','customfield_10123:NIGSP_CEIRS_CIP047_RFH3_00177','customfield_10100:giv_RFH3_38236','customfield_10120:giv','customfield_10121:RFH3','custo
mfield_10224:1131006253466','customfield_10126:H3N2'"
Issue ST-1008 created.
<blank line>
Run completed successfully. 271 actions were successful from file: /tmp/ST-jira.csv
<blank line>
```

/tmp/ST-jira-updated.sql

```
insert giv..ExtentAttribute (Extent_id, ExtentAttributeType_id, value) select 1131006253355 as Extent_id, ExtentAttributeType_id, "ST-738" as value from giv..ExtentAttributeType where type = "jira_id";
insert giv..ExtentAttribute (Extent_id, ExtentAttributeType_id, value) select 1131006253335 as Extent_id, ExtentAttributeType_id, "ST-739" as value from giv..ExtentAttributeType where type = "jira_id";
insert giv..ExtentAttribute (Extent_id, ExtentAttributeType_id, value) select 1131006253419 as Extent_id, ExtentAttributeType_id, "ST-740" as value from giv..ExtentAttributeType where type = "jira_id";
insert giv..ExtentAttribute (Extent_id, ExtentAttributeType_id, value) select 1131006253381 as Extent_id, ExtentAttributeType_id, "ST-741" as value from giv..ExtentAttributeType where type = "jira_id";
insert giv..ExtentAttribute (Extent_id, ExtentAttributeType_id, value) select 1131006253360 as Extent_id, ExtentAttributeType_id, "ST-742" as value from giv..ExtentAttributeType where type = "jira_id";
```