Java API: Groovy script examples

The public API is delivered in v1.18.0 or higher of the add-on.

Read Grid Data in Edit Mode, **Grid Row Count in Edit Mode** and **Synchronize Custom Field Values** API calls are only available since v1.19.0.

Below please find a set of groovy scripts which use Table Grid Editor Java API to read and manipulate grid data and metadata

Make scripts work

You can easily test the scripts in a Script Runner console. All of them are using some JIRA environment data like issue keys or grid names. So, there are some pre-requisites to make things work:

- You did setup a Table Grid Editor custom field named "TGE_TEST" with a default configuration.
- You have an issue "TEST-1" in a project "TEST" and there is some data in "TGE_TEST" grid in that issue (the presence of data is not mandatory, but it will make example scripts more clear for you).
- You installed the Script Runner

Now you can go to the Script Console of the Script Runner, copy any of the scripts there and check how they work. The scripts will provide some output in the "Result:" section on top of the page. If you want to use another grid or another issue for testing - you will have to change the grid name and/or issue key in the script to the names you need.

The java documentation can be found here.

Check out the Groovy examples of API usage
Add Rows Clear Duplicates Clear Grid Clear Not Modified Delete Rows Edit Grid Grid Datasource Info Grid List Grid Row Count in Edit Mode Grid Rows Count Read Grid Data Read Grid data for All Issues Read Grid Data in Create/Edit Mode Reload Grid Synchronize Custom Field Values

---

Table Grid Reader API

To use Table Grid Reader Java API, you need to use an object of **com.idalko.jira.plugins.igrid.api.data.TGRGridTableDataManager** class. You can get it using the following code in your Groovy script:

```
Class dataManagerClass = pluginAccessor.getClassLoader().findClass("com.idalko.jira.plugins.igrid.api.data.
TGEGridTableDataManager");
def tgeGridDataManager = ComponentAccessor.getOSGiComponentInstanceOfType(dataManagerClass);
```

TGRGridTableDataManager has the following methods:

```
@param gridId the Table Grid Reader customfield's ID
@param issueId the ID of the relevant issue
@param startAt the index of the row to start reading of the data from
@param maxResult the number of maximum number of rows that will be shown
@return a wrapper object that contains the values retrieved from the Table Grid Reader of the specified issue
@see TGEPaginatedData
TGEPaginatedData<Map<String, Object>> readGridData(Long gridId, Long issueId, Integer startAt,Integer
maxResult);



@param gridId the Table Grid Reader customfield's ID
@param issueId the ID of the relevant issue
@return the number of total row count for the specifield Table Grid Reader customfield and issue
int countRows(Long gridId, Long issueId);


@param gridId the Table Grid Reader customfield's ID
@param user the user which has the access rights to perform the operation
@return a wrapper object that contains the name of the configuration table, database table, and the connection
URL
@see TGEDataSourceInfo
```

TGEPaginatedData:

```
@return the total number of rows
Integer getTotal()


@return a list of rows represented as a map where the key is the name of the column and the value is its actual
value
List<Map<String, Object>> getValues()
```

TGEDataSourceInfo:

```
@return the name of the configuration table
String getConfigurationTableName()


@return the name of the database table
String getDatabaseTableName()


@return the URL of the established connection
String getConnectionUrl()
```