ALM Octane

Git Integration Tool

User Guide

Document Release Date: October 2019

Introduction

This tool can be used to:

- Fetch pull requests from Git repositories into Octane.
- Get branch information from Git repositories into Octane.
- Create branches from Octane on a Git repository.

> **Note:** Only Bitbucket is supported at the moment.

Limitations

- This solution can be used for one instance of Octane with multiple shared spaces and workspaces. In case you desire to use this utility on more than one Octane instance, you will need to reinstall the utility for each instance.

- In the current version we support only one Bitbucket instance connected to an Octane instance. Multiple repositories and projects from that instance of Bitbucket can be connected to Octane.

- In order to fetch the pull requests and branch information correctly, the commits data must be present in Octane, not only in the repository.

- If the configuration file (**\conf\configuration.properties**) is modified, the server must be restarted.

- After creating a new branch, the branch will be available in Octane only if these three conditions are met:
    - the commits data for that specific branch must be present in Octane
    - the newly created branch stems from a branch that would have already existed in the branch information field
    - branch information is refreshed by pressing the “Get Branch Information” button

  If the branch stems from any other branch, it means that it will not contain any of the commits from that branch that exist in Octane and are not already merged in the default branch and thus it will not appear in the branch information.

Prerequisites

One instance of each application below is required for the integration:

- Octane – Version 12.60.47 and higher
- Bitbucket – Version 5.16 and higher

One of the following web servers:

- Tomcat – Version 7.0 and higher
- Jetty – Version 9.4.20 and higher

Deployment

Fill the attributes in the configuration file

In order to install the tool, you will need the **git-integration-for-octane.war** file and a java web application server such as Tomcat/Jetty, where the artifact can be deployed to.

Please follow the steps below after adding the war to your web server (Tomcat/Jetty):

Fill in the configuration file, **configuration.properties**, which can be found in the **\conf** folder.

Key	Value
Octane Fields	
octane.server	The **URL** to the Octane server. Example: `http://octane.company.com:8080`
octane.sharedSpace	The shared spaces of Octane where the utility will be used. The shared spaces must be separated by a comma. Example: `1001,1002`

octane.user	A username (or API key) which has **Space Admin** and **Workspace Admin** permissions. If the utility is used for more than one shared space, you should add a username (or API key) for each shared space separated by a comma. Please pay attention because the order matters. Example: user1,user2
octane.password	The password (or Secret) for the user. If there are multiple users listed, all the passwords must be provided in the same order. Example: password1,password2
octane.pullRequestsInformationUDFName	In order to display the pull requests, the utility will create a memo field. The name of the field must have "_udf" at the end. Example: pull_requests_udf
octane.pullRequestsInformationUDFLabel	The name of the field. This is what users will see in the entities Edit form. Example: Pull Requests
octane.branchInformationUDFName	In order to display the branches information, the utility will create a memo field. The name of the field must have "_udf" at the end. Example: branch_information_udf
octane.branchInformationUDFLabel	The name of the field. This is what users will see in the entities Edit form. Example: Branch Information
Bitbucket fields	
repo.host	The only supported value for this field at the moment is **bitbucketserver**.
bitbucketserver.url	The Bitbucket URL.

	Example: `http://bitbucket.com:7990`
bitbucketserver.access	A Bitbucket Personal access token. Here you can find out how to create an access token for Bitbucket.
Proxy fields (Optional)	
proxy.host	The proxy host. Example: `webcache.example.com`
proxy.port	The proxy port. Example: 8080

Note: Please pay attention to enter the shared spaces and credentials in the correct order!

Example: user1 with password1 is Space Admin in the Shared Space 1001 and user2 with password2 is Space Admin in the Shared Space 1002. The correct values can be:

octane.sharedSpace=1001,1002
octane.user=user1,user2
octane.password=password1,password2

OR

octane.sharedSpace=1002,1001
octane.user=user2,user1
octane.password=password2,password1

Create a button using the External action editor

Please complete the following steps in order to add a button in Octane.

1. Login to Octane with a Site Admin user.
2. Click on the **Settings** icon.
3. Go to **External action editor**. You can find help about this feature here. All the used attributes are explained in the Help Center.
4. Add the JSON in the editor. (There are some JSON examples below)
5. After adding this JSON (with the updated URL) in the External action editor, you need to press **UPDATE** so that Octane will add the button to the entities actions menu.
6. Refresh Octane.

Add the Get Pull Requests button to Octane

Please read these steps first.

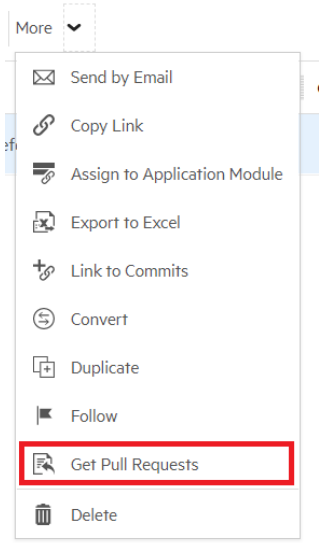
This example JSON can be used to create the button. The only thing which must be replaced is the highlighted first part of the Tomcat/Jetty URL (for example **http://localhost:9090/git-integration-for-octane**).

Example:

```
[ {
    "name": "pull_requests_button",
    "title": "Get Pull Requests",
    "entity_type": ["work_item"],
    "views": ["list", "details"],
    "icon": "import",
    "url": "http://localhost:9090/git-integration-for-octane/pull-requests?ids={entity_ids}&entityType={entity_type}&sharedSpace={shared_space}&workSpace={workspace}&dialogId={dialog_id}&server={octane_url}",
    "single_entity": false,
    "events": true,
    "dialog": "small"
}]
```

Some of these attributes, i.e. name or title, can be modified. Please follow the instructions from the Help Center before modifying any of the attributes' values.

The button should be available in the entities action menu if you select at least one entity, on the detailed view, Backlog or Team Backlog views, as below:



Add the Get Branch Information button to Octane

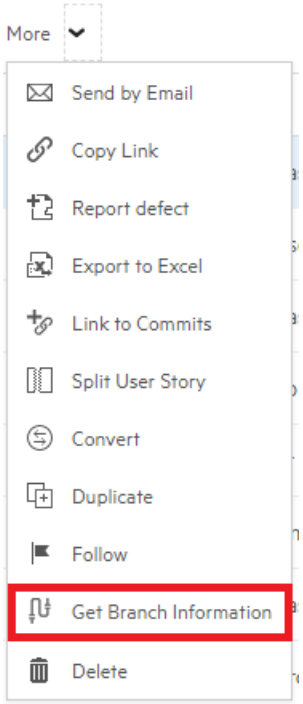
Please read these steps first.

The following JSON example can be copied in the External action editor (Do not forget to replace the highlighted first part of the Tomcat/Jetty URL for example **http://localhost:9090/git-integration-for-octane**):

Example:
```
[ {
    "name": "branch_information_button",
    "title": "Get Branch Information",
    "entity_type": ["work_item"],
    "views": [ "list", "details"],
    "icon": "type-pipeline",
    "url": "http://localhost:9090/git_integration_for_octane/branch-information?ids={entity_ids}&entityType={entity_type}&sharedSpace={shared_space}&workSpace={workspace}&dialogId={dialog_id}&server={octane_url}",
    "single_entity": false,
    "events": true,
    "dialog": "small"
} ]
```

Some of these attributes, i.e. name or title, can be modified. Please follow the instructions from the Help Center before modifying any of the attributes' values.

The button should be available in the entities action menu if you select at least one entity, on the detailed view, Backlog or Team Backlog views, as below:



Add the Create Branch button to Octane

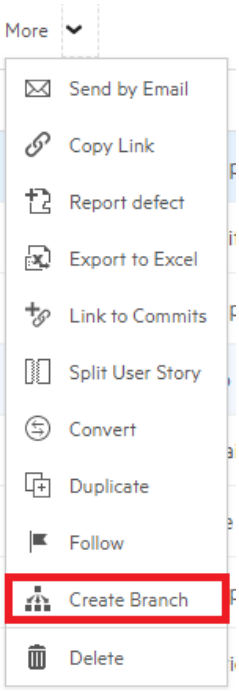
Please read these steps first.

The following JSON example can be copied in the External action editor (Do not forget to replace the highlighted first part of the Tomcat/Jetty URL for example **http://localhost:9090/git-integration-for-octane**):

Example:
```
[ {
    "name": "create_branch_button",
    "title": "Create Branch",
    "entity_type": ["work_item"],
    "views": ["list", "details"],
    "icon": "children",
    "url": "http://localhost:9090/git_integration_for_octane/create-branch-page?ids={entity_ids}&entityType={entity_type}&sharedSpace={shared_space}&workSpace={workspace}&server={octane_url}",
    "single_entity": true,
    "events": true
}]
```

Some of these attributes, i.e. name or title, can be modified. Please follow the instructions from the Help Center before modifying any of the attributes' values.

The button should be available in the entities action menu if you select at least one entity, on the detailed view, Backlog or Team Backlog views, as below:

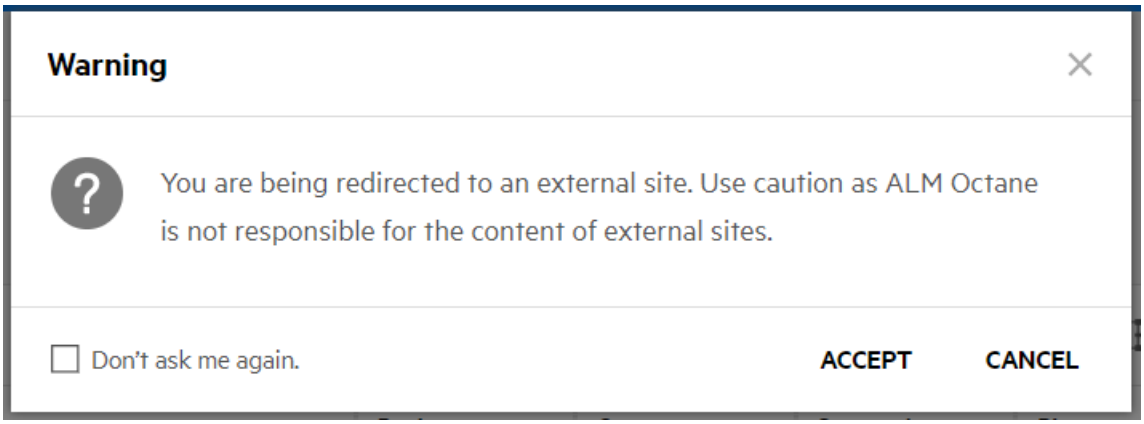


Note: This button will be available only for a single entity! If you select multiple items in the grid view, the button will not be displayed.

First time usage

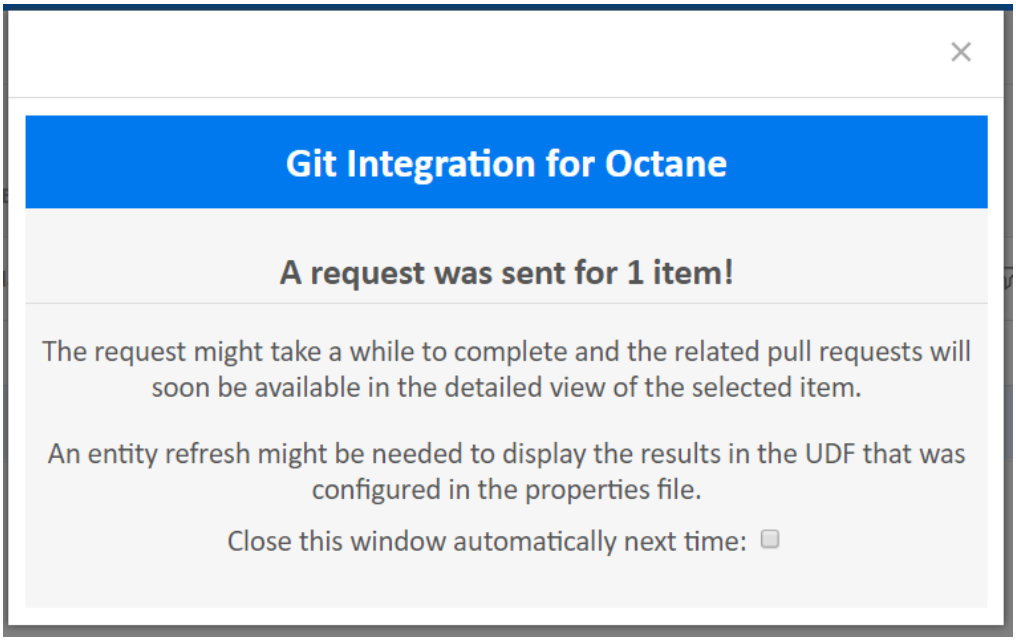
> **Note:** The process of getting branch information is similar to getting the pull requests which is detailed below.

Once you have your commits data from Bitbucket brought into Octane, you can start using the tool. You can select all the entities and press the "**Get Pull Requests**" button. For the first time, Octane will display this message:



This is because the Get Pull Requests button, which was just added to Octane, fires a request to the middleware. If you don't want to see this pop up every time you click the button, you can check the "**Don't ask me again**" checkbox.

For every call (button press), the middleware will display the following content:



If the checkbox is selected, this window will automatically close after 4 seconds and the detailed view of the entities will be refreshed.

> **Note:** The content of the window will not change in case there are any errors or delays. The calls to fetch the pull requests are done while this message is displayed. In case of any latencies please check the logs (\git-integration-for-octane\logs) for errors or warnings.

The middleware will perform the following actions when the button is pressed for the first time:

1. For epics, features, defects, user stories and quality stories, the memo UDF (where the pull requests will be listed) will be created.

2. The UDF will be added into the Edit form for every entity listed above. This means that users will see the UDF when they edit an entity.

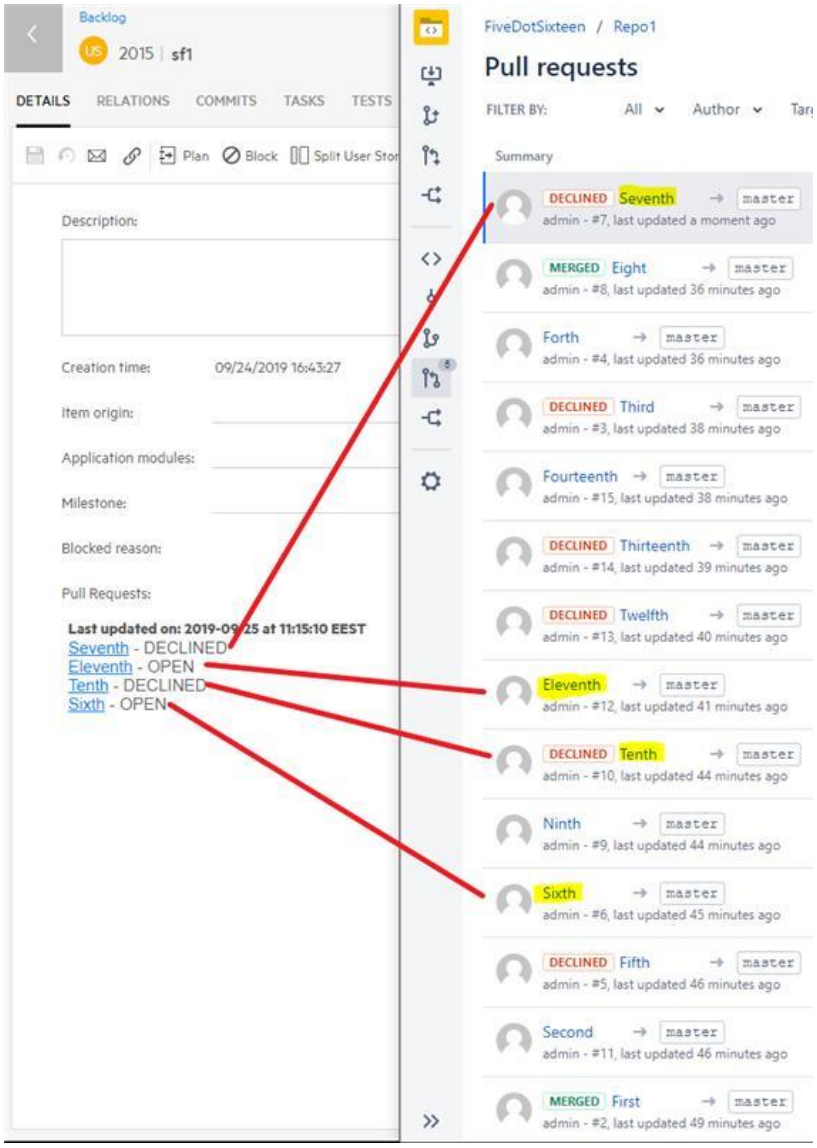
3. A new rule will be added, in order to make the UDF read only. This rule will be created for all the entities mentioned above.

   The rule makes the UDF read-only for all the users, except the ones with Space Admin permissions.

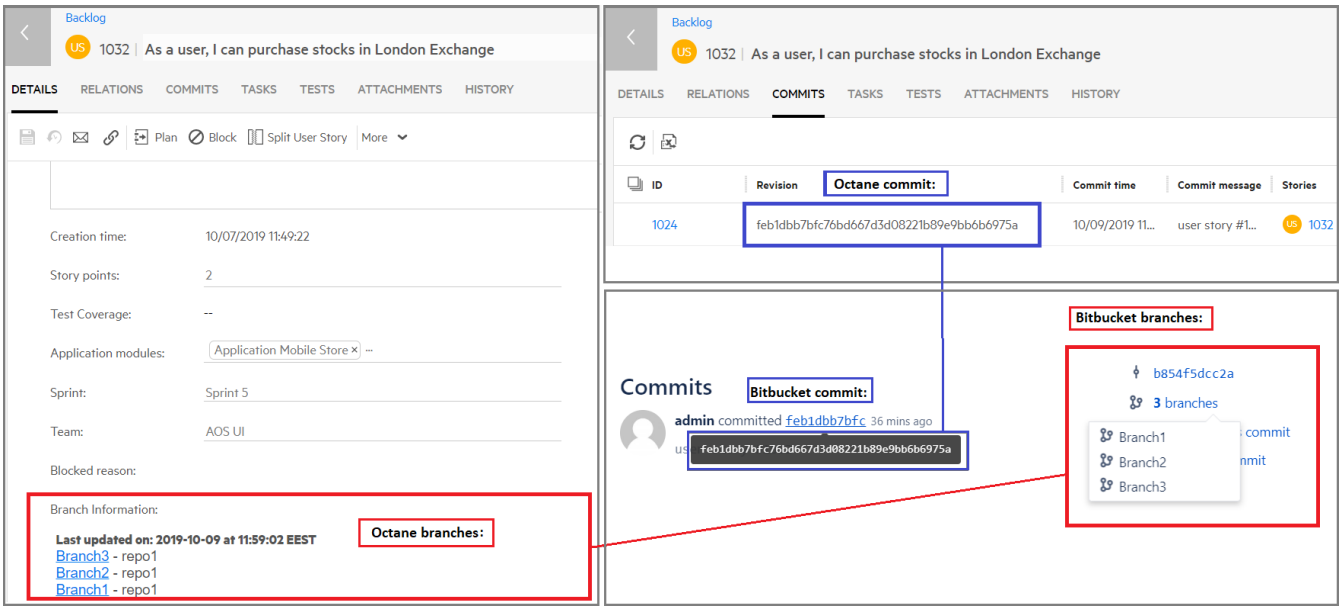
For Isolated Spaces, the items mentioned above will be created at workspace level. For Shared Spaces, the items will be created in the master workspace and inherited in the rest of the workspaces.

> **Note:** The UDFs will be created on the first request which is done on one or multiple entities (defect/user story/epic/feature/quality story), which are linked to at least one commit which is included in a pull request. This happens for the branch information UDFs as well.

After all the operations are completed the pull requests links and states will be entered into the memo UDF. The pull requests are ordered as they are in Bitbucket, last updated first. In the pull request memo UDF we can find the following information: the name of the pull request, a link to Bitbucket pull request and the state of the pull request.

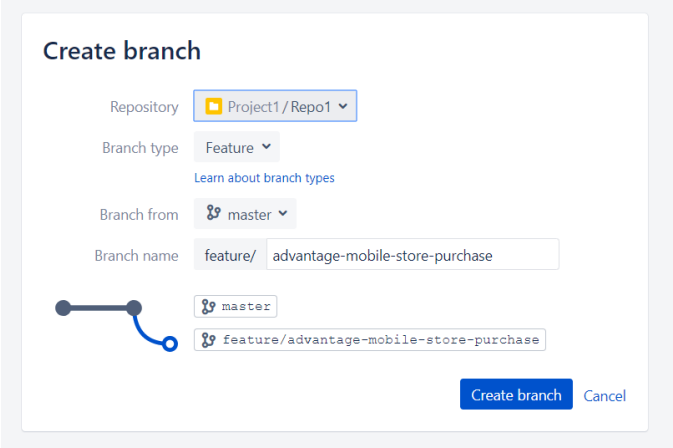


The branches are listed in Octane in a memo UDF, similar to the pull requests. The following information is found there: the name of the branch, a link to Bitbucket for source code on the branch and the repository where the branch has been created.



The commit linked to the user story is present in 3 branches in Bitbucket. The commit can be searched in Bitbucket, where a list with the branches which include the specific commit can be found.

The branch creation redirects to Bitbucket. The name of the branch and the type of the branch is selected based on the Octane work items.



Troubleshooting

By default, the middleware does not display any errors.

All the logs of the application are available in the **git-integration-for-octane\logs** folder. In case of any errors or latencies please check the logs for ERROR and WARNING messages.

Feedback

This is our GitHub page. Please feel free to share your feedback and suggestions with us there.