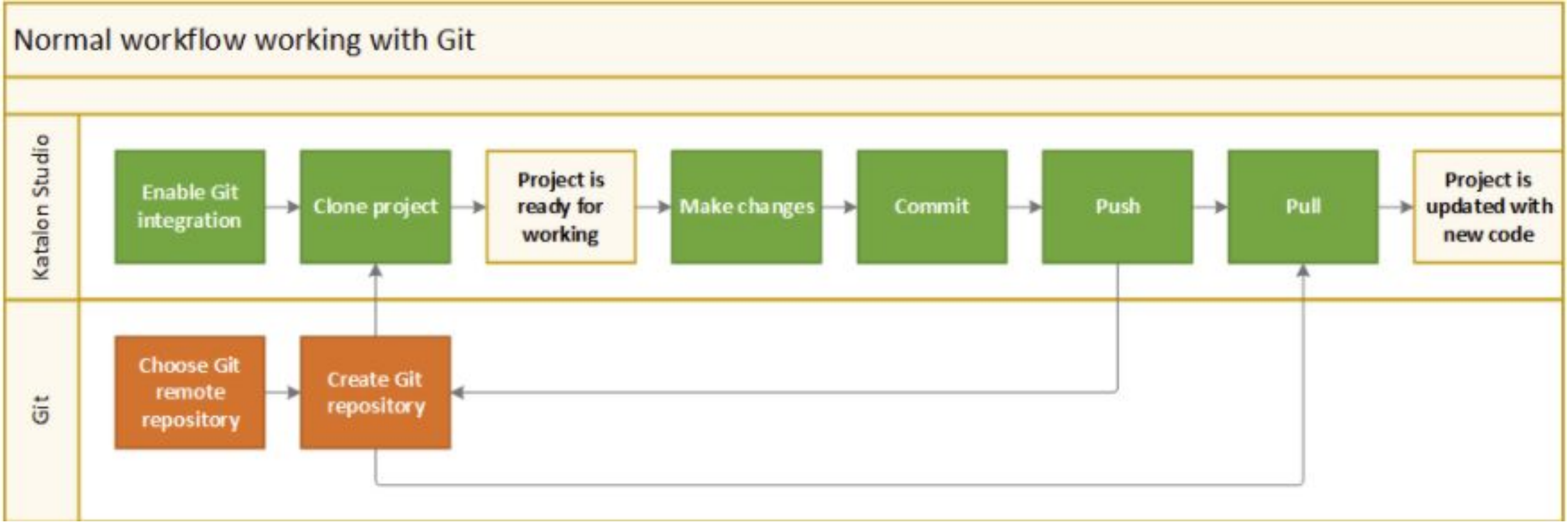
Katalon Studio For Automated Testing

Sesi 8

Integrations With GIT & JIRA

GIT X KATALON WORKFLOW



Normal workflow working with Git

Katalon Studio

Enable Git integration

Clone project

Project is ready for working

Make changes

Commit

Push

Pull

Project is updated with new code

Git

Choose Git remote repository

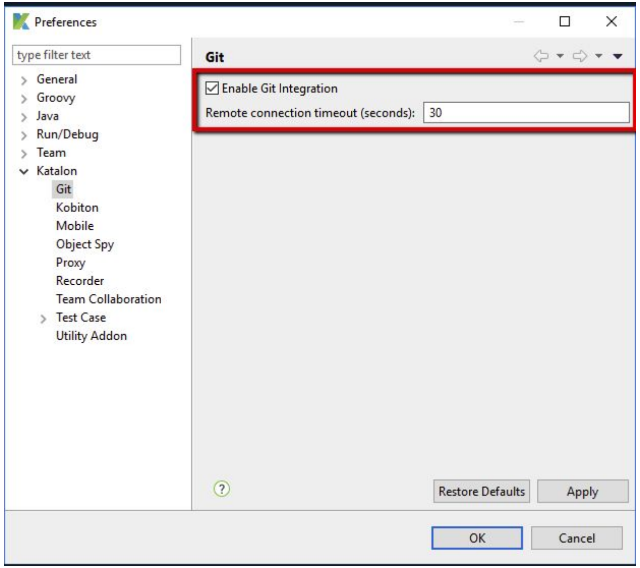
Create Git repository

INTEGRATION WITH GIT

Steps to enable Git integration:

1. Enable Git Integration. In order to access all Git features, you need to enable Git Integration first by going to Window > Katalon Studio Preferences > Katalon > Git.

Once this option is enabled, you can use Git from the Katalon Studio main toolbar.

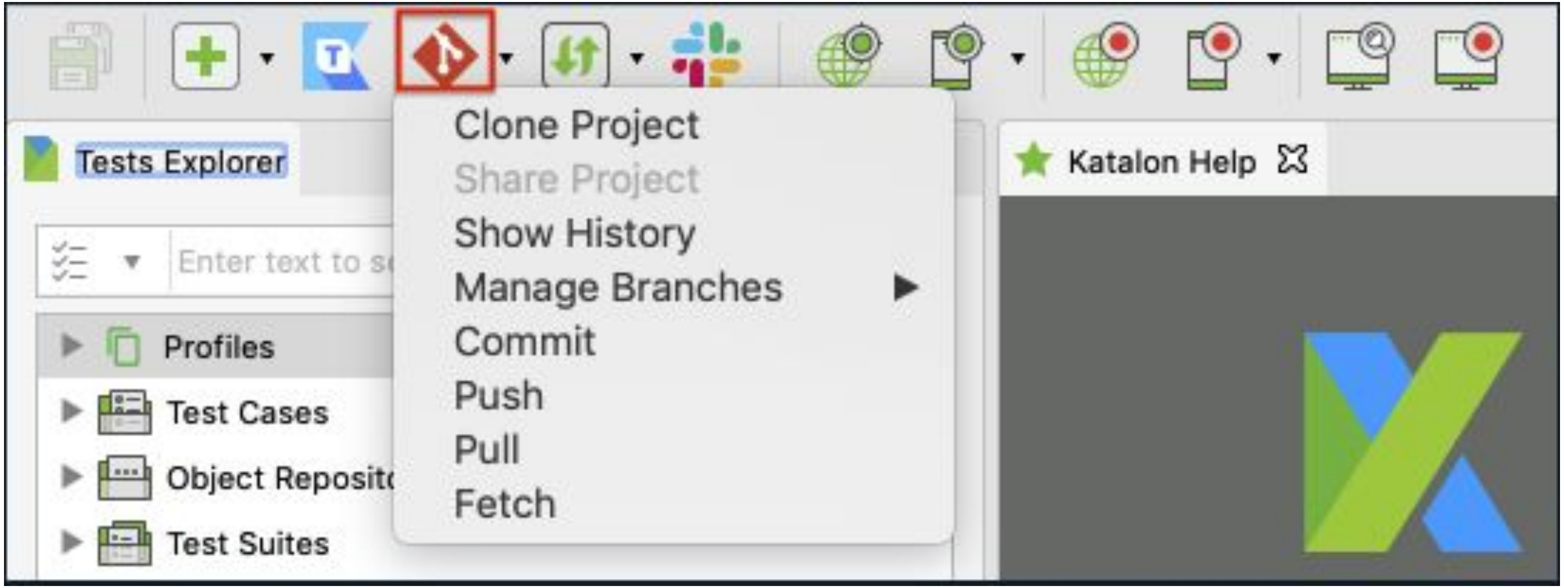


☑ Enable Git Integration

Remote connection timeout (seconds): 30

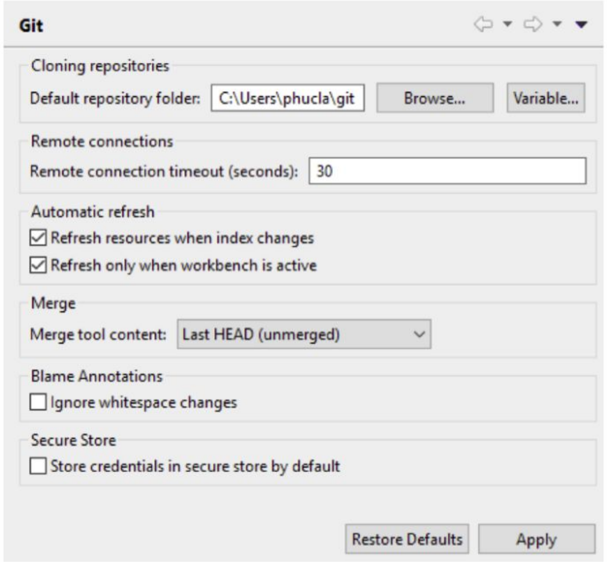
INTEGRATION WITH GIT

2. Now the Git integration feature should be enabled. We are ready to use Git from Katalon Studio.



INTEGRATION WITH GIT

3. Advanced configurations are available from Window > Preferences > Team > Git in case you want more detailed setups.



Git

Cloning repositories

Default repository folder: C:\Users\phucla\git

Remote connections

Remote connection timeout (seconds): 30

Automatic refresh

☑ Refresh resources when index changes

☑ Refresh only when workbench is active

Merge

Merge tool content: Last HEAD (unmerged)

Blame Annotations

□ Ignore whitespace changes

Secure Store

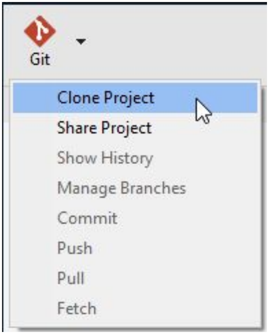
□ Store credentials in secure store by default

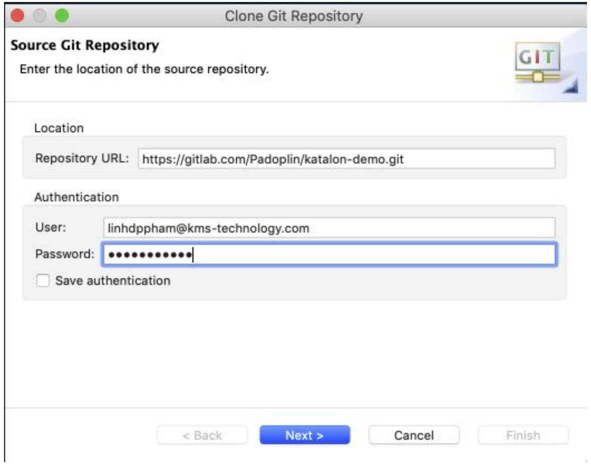
After enabling Git integration, you can perform Git commands such as sharing and cloning projects, committing changes, and managing branches.

CLONE PROJECT

After enabling Git Integration, you can clone an existing Git repository into a newly-created directory on the local machine.

1. From the main toolbar, select the Git icon > Clone Pr





Clone Git Repository

Source Git Repository

Enter the location of the source repository.

Location

Repository URL: https://gitlab.com/Padoplin/katalon-demo.git

Authentication

User: linhdppham@kms-technology.com

Password: ●●●●●●●●●●●

☐ Save authentication

2. The Source Git Repository dialog is displayed.

CLONE PROJECT

Connecting to Git with HTTPS

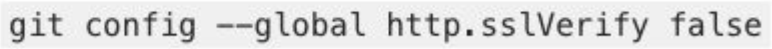
Enter all required information and click Next to let Katalon Studio get details about your repository.

Where:

- Repository URL: the remote URL to your Git repository in HTTPS protocol.
- Username: the username to access the Git repository.
- Password: the password to access the Git repository.

ISSUE WITH SSL VERIFICATION

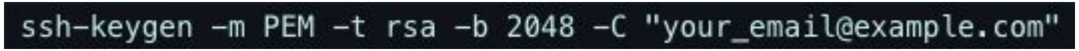
If your network cannot access the repository, there may be a chance that it is not allowed by SSL verification from your connecting network. You can use the following command in your Git bash to bypass SSL verification:



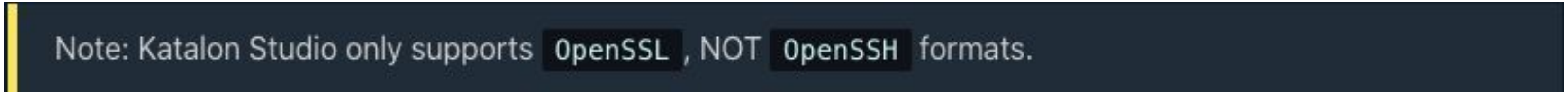
```
git config --global http.sslVerify false
```

CONNECTING TO GIT WITH SSH KEYS

1. Generating a public/private rsa key pair with this command:



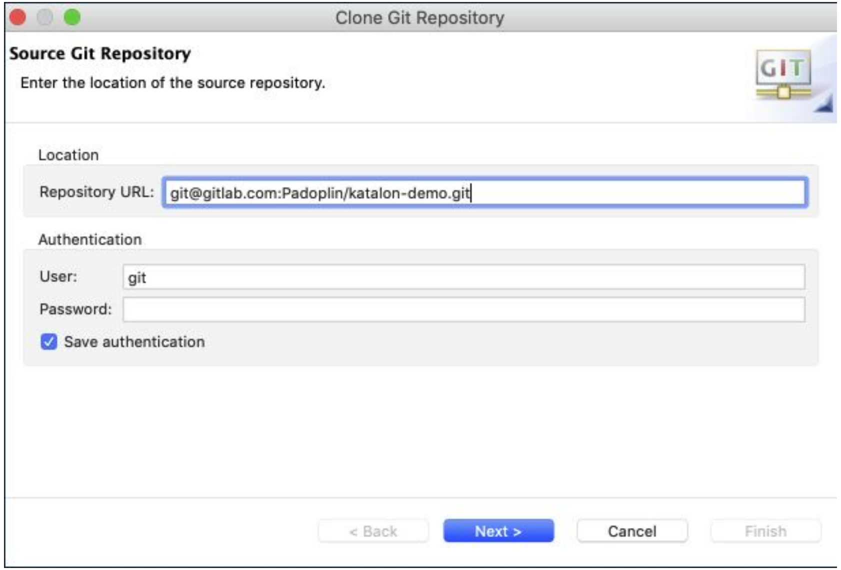
```
ssh-keygen -m PEM -t rsa -b 2048 -C "your_email@example.com"
```



Note: Katalon Studio only supports `OpenSSL`, NOT `OpenSSH` formats.

2. Enter file in which to save the key.
3. Enter the SSH key passphrase and confirm it.
4. Add your SSH key to the ssh-agent.
5. Add a new SSH key to your Git account.

In the Source Git Repository dialog,
enter a repository URL with SSH Protocol and click
Next.



Clone Git Repository

Source Git Repository

Enter the location of the source repository.

Location

Repository URL: git@gitlab.com:Padoplin/katalon-demo.git

Authentication

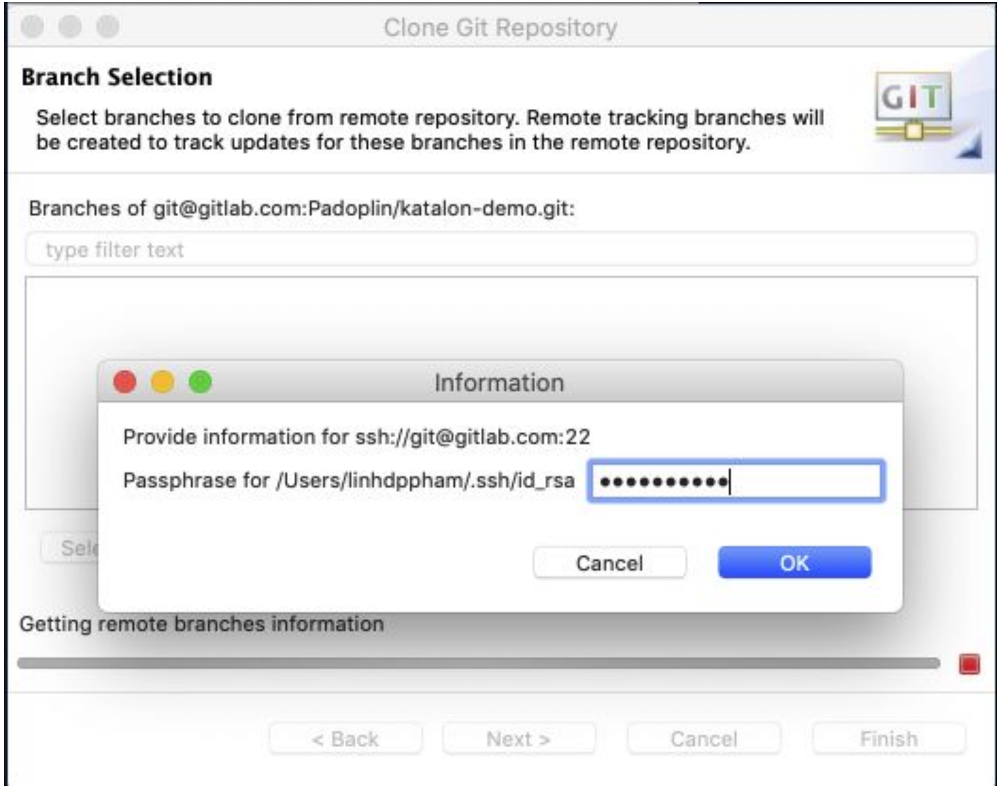
User: git

Password:

☑ Save authentication

CONNECTING TO GIT WITH SSH KEYS

Enter the passphrase for the key generated above.



Clone Git Repository

Branch Selection

Select branches to clone from remote repository. Remote tracking branches will be created to track updates for these branches in the remote repository.

Branches of git@gitlab.com:Padoplin/katalon-demo.git:

Information

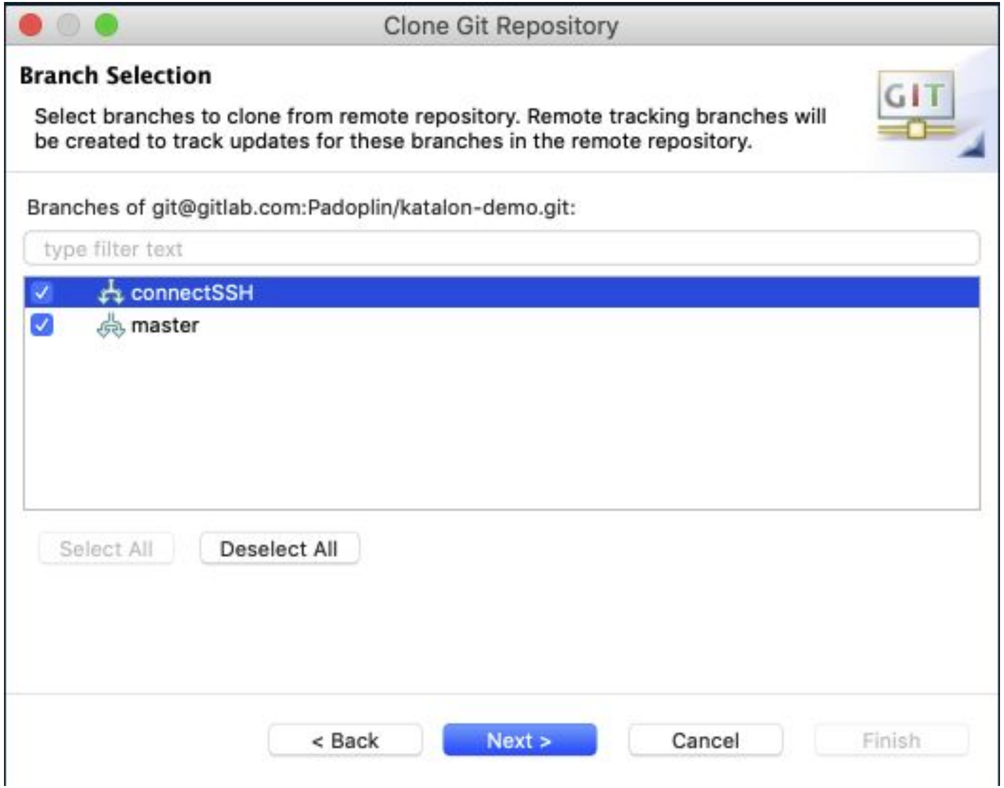
Provide information for ssh://git@gitlab.com:22

Passphrase for /Users/linhdppham/.ssh/id_rsa ●●●●●●●●●●●

Getting remote branches information

CONNECTING TO GIT WITH SSH KEYS

At Branch Selection screen, you can choose which branches to be checked out as local branches. Click Next to continue.



Branch Selection

Select branches to clone from remote repository. Remote tracking branches will be created to track updates for these branches in the remote repository.

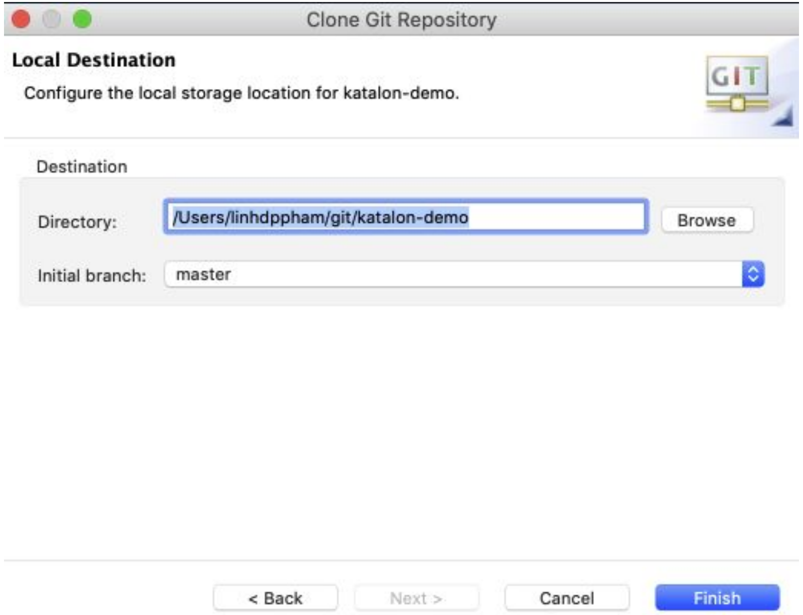
Branches of git@gitlab.com:Padoplin/katalon-demo.git:

☑ connectSSH

☑ master

CONNECTING TO GIT WITH SSH KEYS

At Local Destination dialog, specify the local location for cloning as well as the initial branch.



Clone Git Repository

Local Destination

Configure the local storage location for katalon-demo.

Destination

Directory: /Users/linhdppham/git/katalon-demo

Initial branch: master

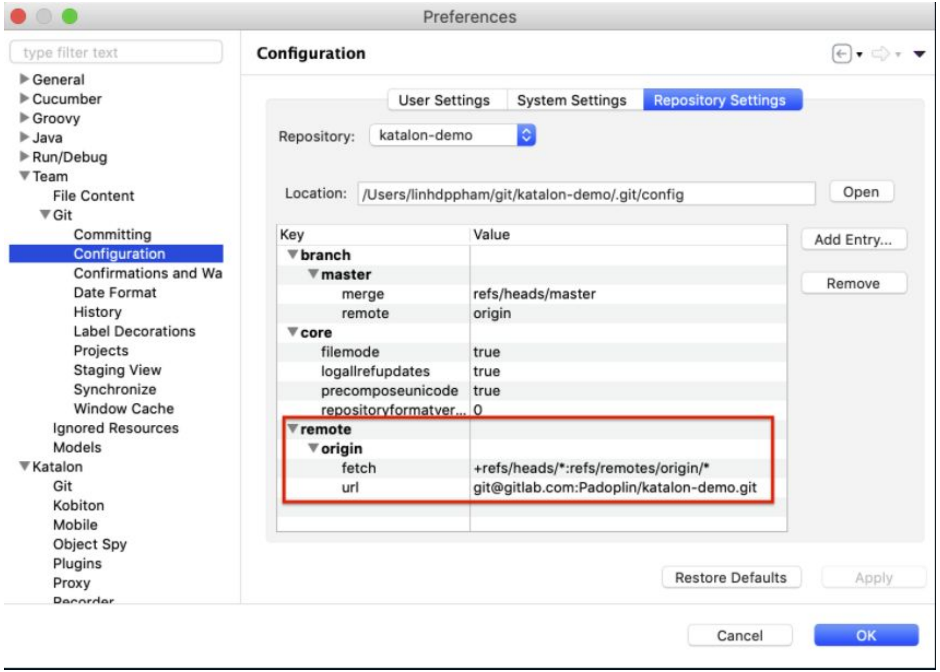
Where:

- Directory: the local storage location you want to store your Git's repository.
- Initial branch: all selected branches from the previous step are displayed here. Select the branch to be used initially from this list.

Click Finish when you are done. Katalon Studio automatically opens your cloned project.

CONNECTING TO GIT WITH SSH KEYS

To verify settings, go to Katalon Studio> Preferences > Team > Git > Configurations > Repository Settings. Ensure that the repository is selected correctly with the URL specified.



Repository: katalon-demo

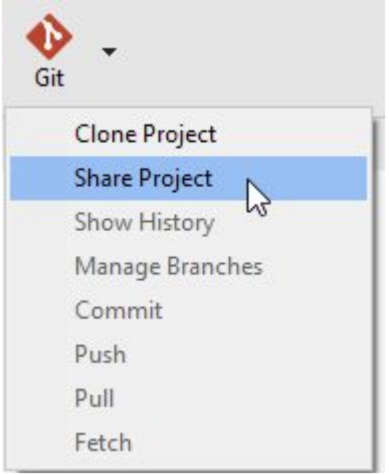
Location: /Users/linhdppham/git/katalon-demo/.git/config

Key	Value
branch	
master	
merge	refs/heads/master
remote	origin
core	
filemode	true
logallrefupdates	true
precomposeunicode	true
repositoryformatver...	0
remote	
origin	
fetch	+refs/heads/*:refs/remotes/origin/*
url	git@gitlab.com:Padoplin/katalon-demo.git

PUBLISH non-GIT PROJECT 1st TIME

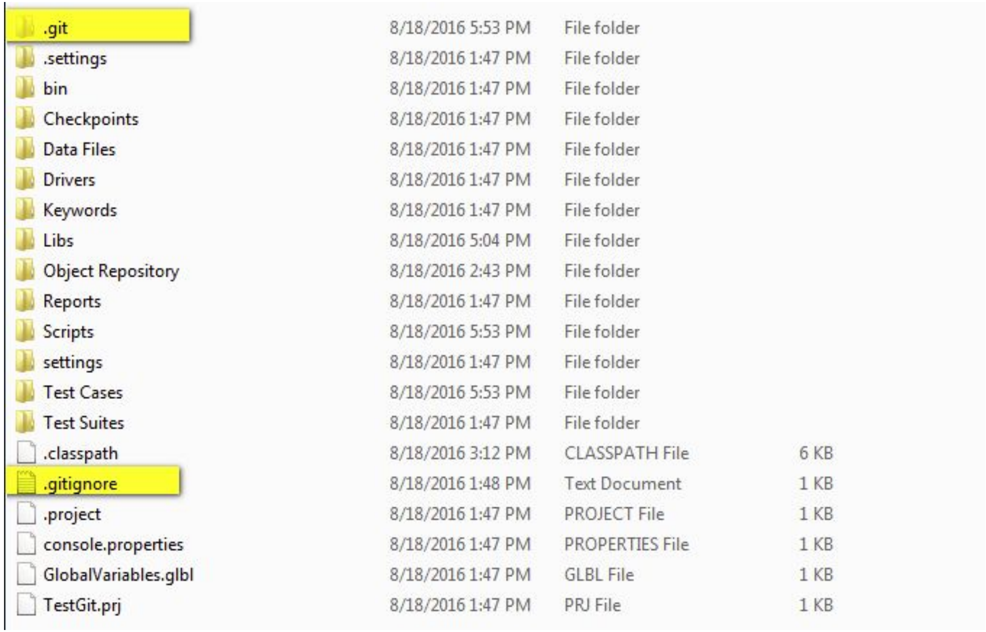
Share Project is a step to enable Git configuration for your new Katalon Studio project.

1. From the main toolbar, select the Git icon > Share Project.



PUBLISH non-GIT PROJECT 1st TIME

2. Folder .git and file .gitignore are created within the Katalon project.



.git	8/18/2016 5:53 PM	File folder	
.settings	8/18/2016 1:47 PM	File folder	
bin	8/18/2016 1:47 PM	File folder	
Checkpoints	8/18/2016 1:47 PM	File folder	
Data Files	8/18/2016 1:47 PM	File folder	
Drivers	8/18/2016 1:47 PM	File folder	
Keywords	8/18/2016 1:47 PM	File folder	
Libs	8/18/2016 5:04 PM	File folder	
Object Repository	8/18/2016 2:43 PM	File folder	
Reports	8/18/2016 1:47 PM	File folder	
Scripts	8/18/2016 5:53 PM	File folder	
settings	8/18/2016 1:47 PM	File folder	
Test Cases	8/18/2016 5:53 PM	File folder	
Test Suites	8/18/2016 1:47 PM	File folder	
.classpath	8/18/2016 3:12 PM	CLASSPATH File	6 KB
.gitignore	8/18/2016 1:48 PM	Text Document	1 KB
.project	8/18/2016 1:47 PM	PROJECT File	1 KB
console.properties	8/18/2016 1:47 PM	PROPERTIES File	1 KB
GlobalVariables.glbl	8/18/2016 1:47 PM	GLBL File	1 KB
TestGit.prj	8/18/2016 1:47 PM	PRJ File	1 KB

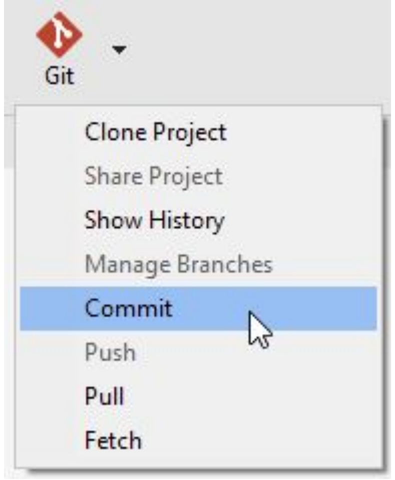
.gitignore tells Git which files (or patterns) it should ignore. By default, .gitignore contains these files and patterns:

/bin /Libs .settings .classpath /.svn

COMMIT

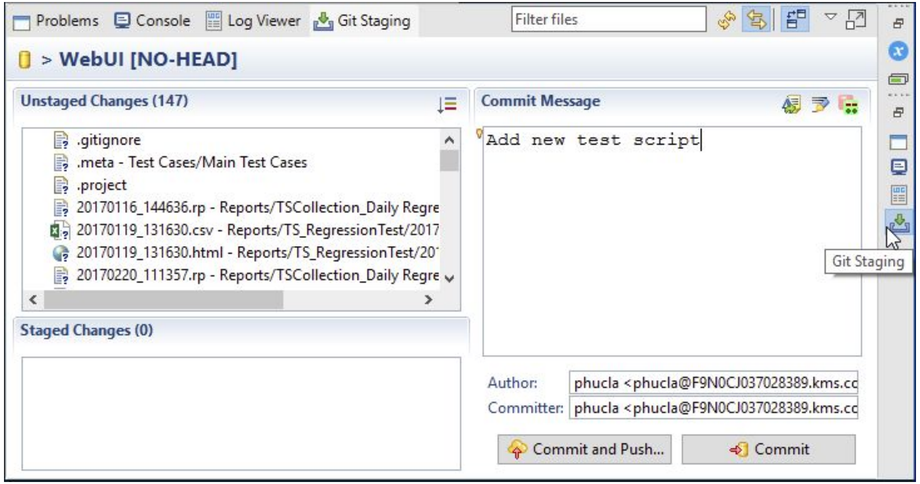
The Commit option allows users to view all current changes and decide which changes to be stored in the local branch

1. From the main toolbar, select the Git icon > Commit.



COMMIT

2. The Git Staging tab is displayed for configuration.



> WebUI [NO-HEAD]

Unstaged Changes (147)

- .gitignore
- .meta - Test Cases/Main Test Cases
- .project
- 20170116_144636.rp - Reports/TSCollection_Daily Regre
- 20170119_131630.csv - Reports/TS_RegressionTest/2017
- 20170119_131630.html - Reports/TS_RegressionTest/20
- 20170220_111357.rp - Reports/TSCollection_Daily Regre

Staged Changes (0)

Commit Message

```
Add new test script
```

Author: phucla <phucla@F9N0CJ037028389.kms.cc

Committer: phucla <phucla@F9N0CJ037028389.kms.cc

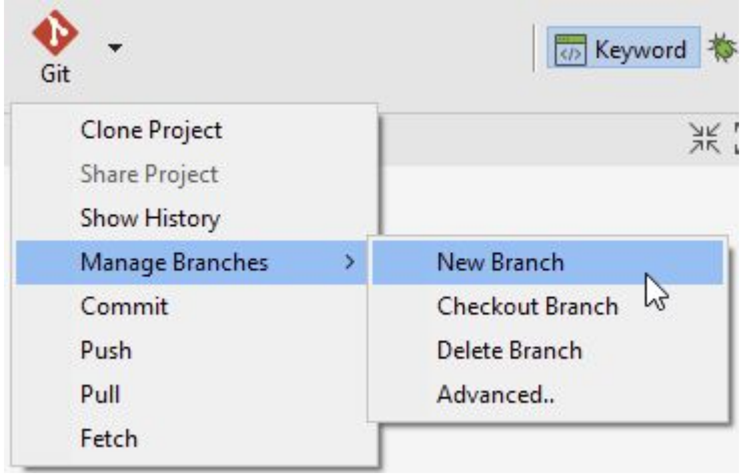
3. From the Unstaged Changes list, select the changes to be committed, then right-click on them and select Add To Index. Selected changes are added to the Staged Changes list.

4. Enter your comments into the Commit Message then click on Commit to store your staged changes into the local branch.

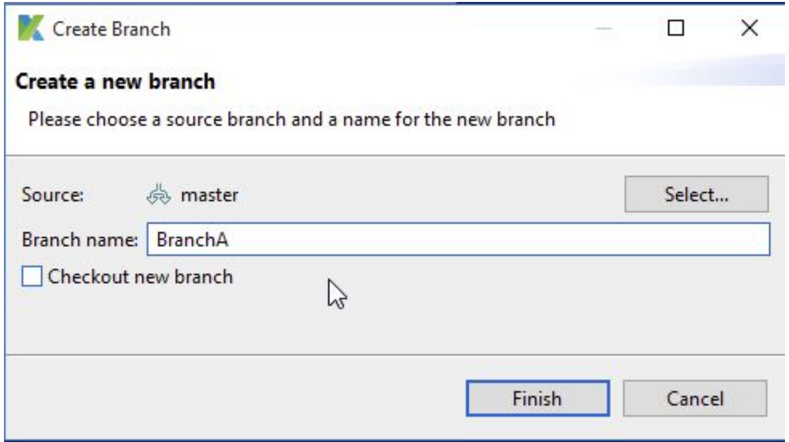
Unstaged Changes = > Changes which have been made.
Staged Changes = > Selected changes from Unstaged Changes. These changes are committed.

MANAGE BRANCHES

1. From the main toolbar, select the Git icon > Manage Branches > New Branch.



2. The Create Branch dialog is displayed.



Create a new branch

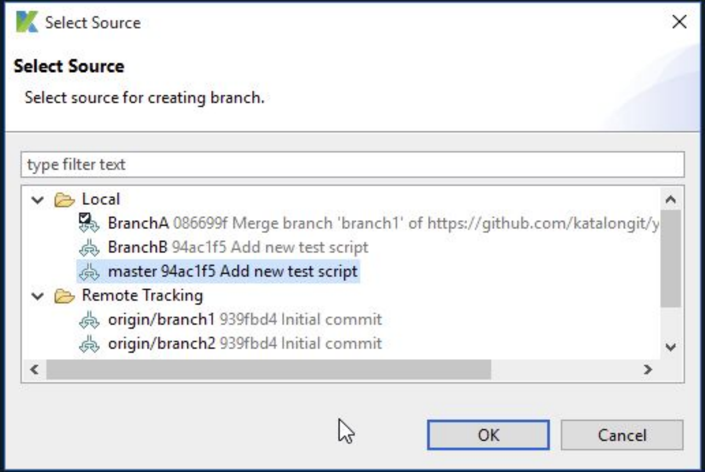
Please choose a source branch and a name for the new branch

Source: master

Branch name: BranchA

□ Checkout new branch

MANAGE BRANCHES

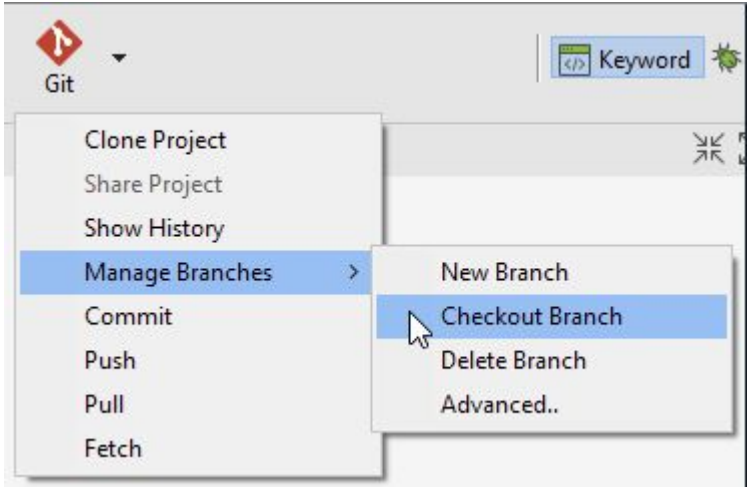
Field	Description
Source	Select either remote or local branch, which is your source branch.  Select Source **Select Source** Select source for creating branch. type filter text Local BranchA 086699f Merge branch 'branch1' of https://github.com/katalongit/y BranchB 94ac1f5 Add new test script master 94ac1f5 Add new test script Remote Tracking origin/branch1 939fbd4 Initial commit origin/branch2 939fbd4 Initial commit OK Cancel
Branch name	The name to be used for the new branch.
Checkout new branch	Option to let Katalon Studio checkout that branch after created.

Click Finish to create a new branch.

CHECKOUT BRANCHES

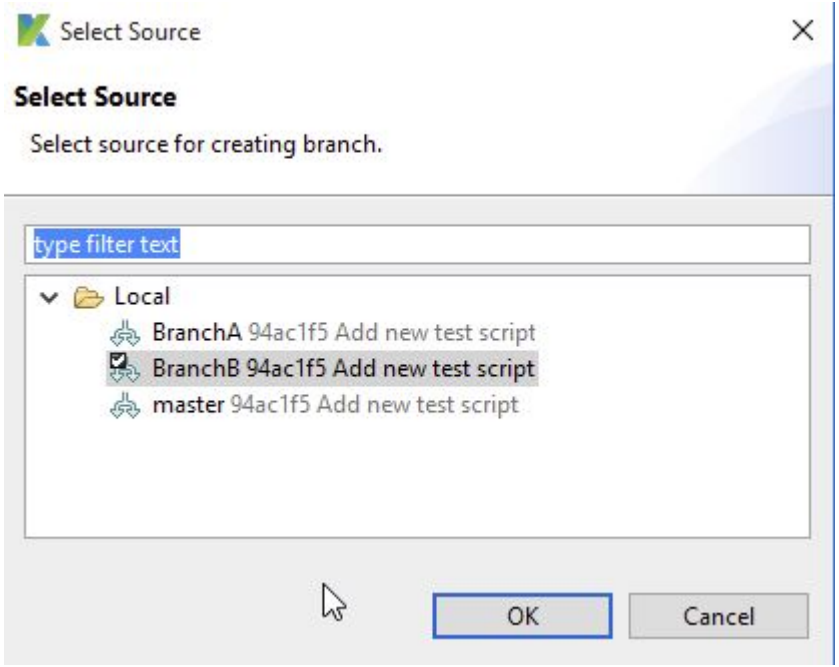
The Checkout Branch option allows you to switch from one branch to another.

1. From the main toolbar, select the Git icon > Manage Branches > Checkout Branch.



CHECKOUT BRANCHES

2. The Select Source dialog is displayed. Select the local branch you want to check out to be the current branch. The branch with an √ icon is your current local branch.



Select Source

Select Source

Select source for creating branch.

type filter text

Local

BranchA 94ac1f5 Add new test script

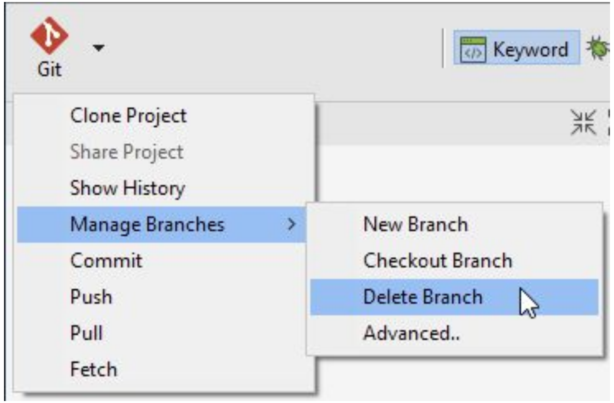
☑ BranchB 94ac1f5 Add new test script

master 94ac1f5 Add new test script

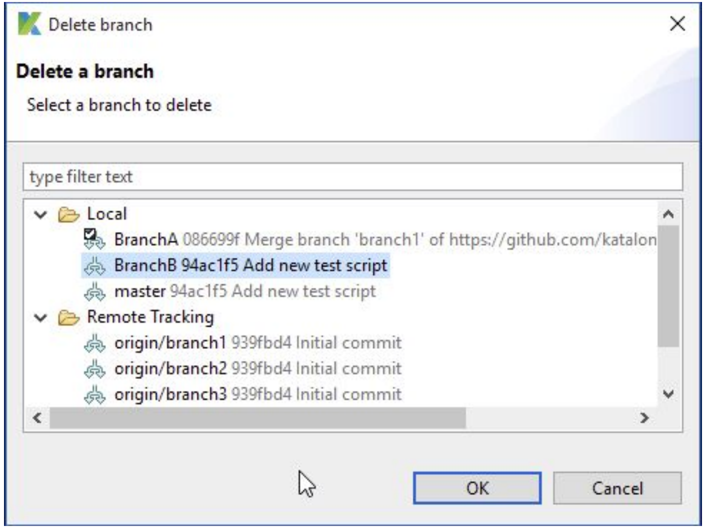
3. Click OK to finish checking out to the new local branch.

DELETE BRANCH

1. From the main toolbar, select the Git icon > Manage Branches > Delete Branch.



2. In this dialog, both local and remote branches are displayed. Select a branch to be deleted then click OK.



Delete a branch

Select a branch to delete

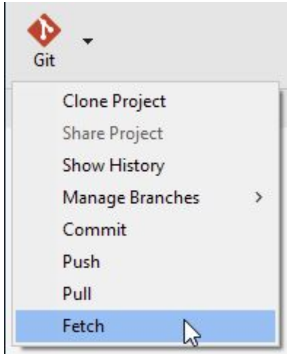
type filter text

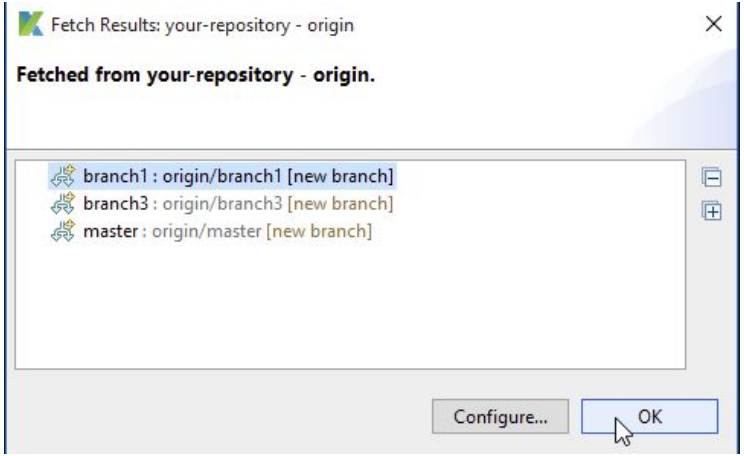
- Local
  - BranchA 086699f Merge branch 'branch1' of https://github.com/katalon
  - BranchB 94ac1f5 Add new test script
  - master 94ac1f5 Add new test script
- Remote Tracking
  - origin/branch1 939fbd4 Initial commit
  - origin/branch2 939fbd4 Initial commit
  - origin/branch3 939fbd4 Initial commit

FETCH

Retrieve all information about changes that have occurred in remote branches.

1. From the main toolbar, select the Git icon > Fetch.




Fetch Results: your-repository - origin

Fetched from your-repository - origin.

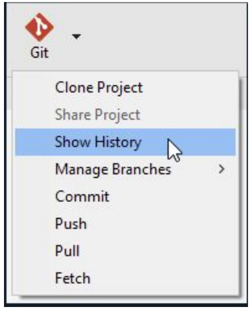
branch1 : origin/branch1 [new branch]
branch3 : origin/branch3 [new branch]
master : origin/master [new branch]

2. It automatically fetches remote branches, tags, and remote changes.

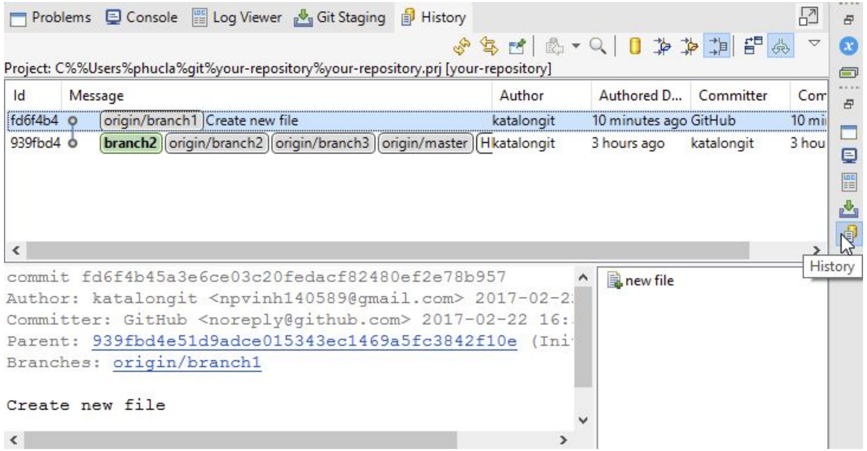
FETCH

Retrieve all information about changes that have occurred in remote branches.

3. Select History from the main toolbar.



4. Details regarding all the branches and tags you've just fetched are displayed.



Project: C%%Users%phucla%git%your-repository%your-repository.prj [your-repository]

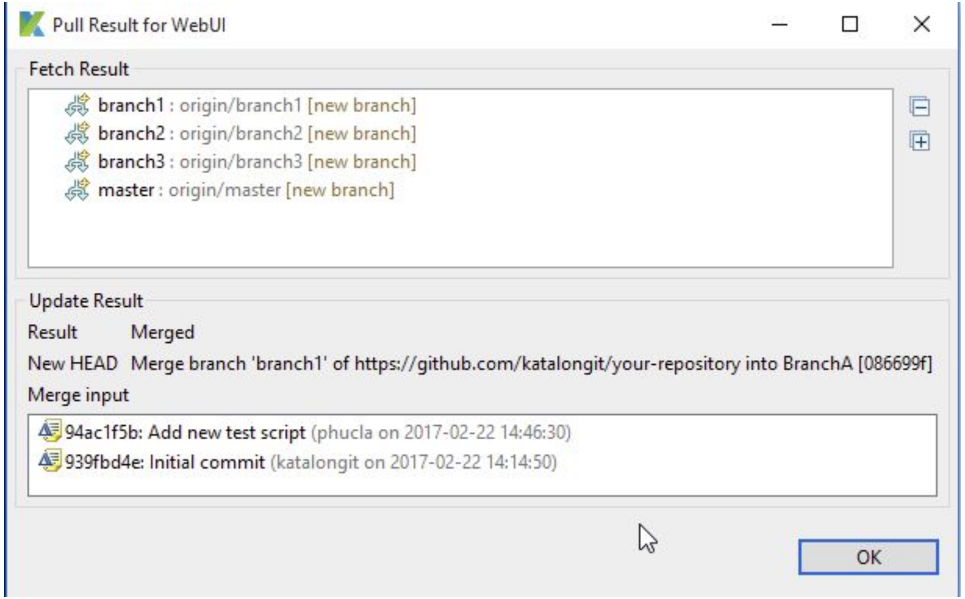
Id	Message	Author	Authored D...	Committer	Com
fd6f4b4	origin/branch1 Create new file	katalongit	10 minutes ago	GitHub	10 mi
939fbd4	branch2 origin/branch2 origin/branch3 origin/master H	katalongit	3 hours ago	katalongit	3 hou

```
commit fd6f4b45a3e6ce03c20fedacf82480ef2e78b957
Author: katalongit <npvinh140589@gmail.com> 2017-02-2
Committer: GitHub <noreply@github.com> 2017-02-22 16:
Parent: 939fbd4e51d9adce015343ec1469a5fc3842f10e (Ini
Branches: origin/branch1

Create new file
```

PULL

Incorporate changes from a remote repository into the current branch.

3. The Pull Result dialog is displayed with all data about pulling requests on the selected branch.



Fetch Result

branch1 : origin/branch1 [new branch]
branch2 : origin/branch2 [new branch]
branch3 : origin/branch3 [new branch]
master : origin/master [new branch]

Update Result

Result Merged
New HEAD Merge branch 'branch1' of https://github.com/katalongit/your-repository into BranchA [086699f]
Merge input

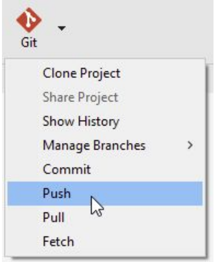
94ac1f5b: Add new test script (phucla on 2017-02-22 14:46:30)
939fbd4e: Initial commit (katalongit on 2017-02-22 14:14:50)

PUSH

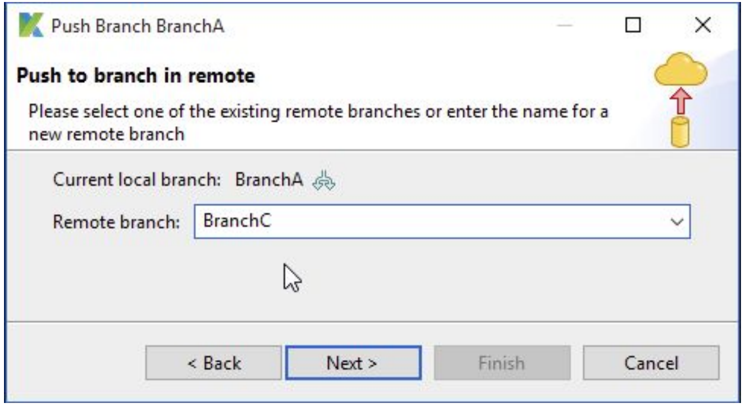
Update the remote branch using the local branch.

Before doing any push, you have to commit your changes first.

1. **From the main toolbar, select the Git icon > Push.**



2. The Push to Branch dialog is displayed. Choose from the Remote branch list which branch to be updated (All remote branches in your Git repository are listed here).


Push Branch BranchA

Push to branch in remote

Please select one of the existing remote branches or enter the name for a new remote branch

Current local branch: BranchA

Remote branch: BranchC

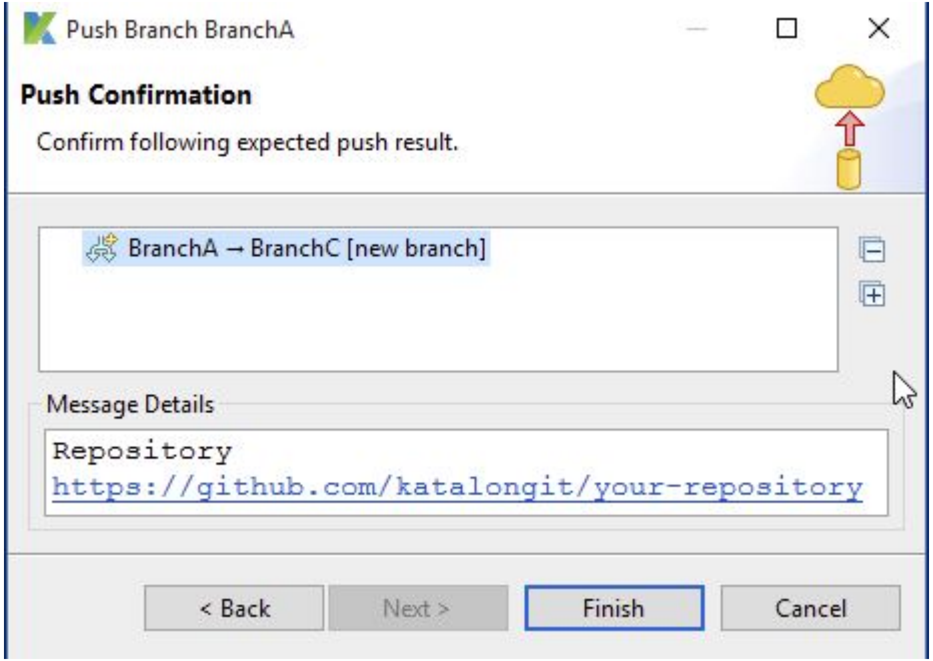
Click Next after finished selecting your remote branch.

If you enter a different name besides the listed branches, a new remote branch with that name is created accordingly.

PUSH

Update the remote branch using the local branch.

3. The Push Confirmation Dialog is displayed with details regarding your commit.



Push Branch BranchA

Push Confirmation

Confirm following expected push result.

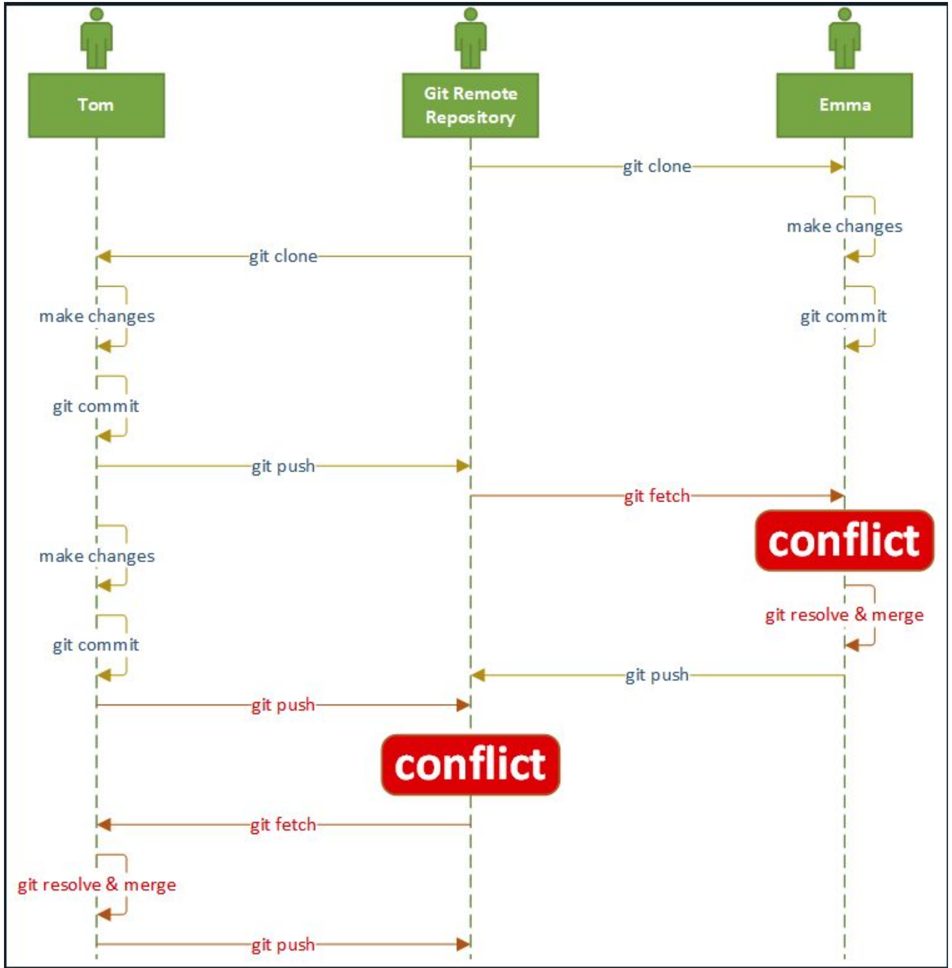
BranchA → BranchC [new branch]

Message Details

```
Repository
https://github.com/katalongit/your-repository
```

Click on Finish to push your commits to the remote repository.

Resolve Git conflicts



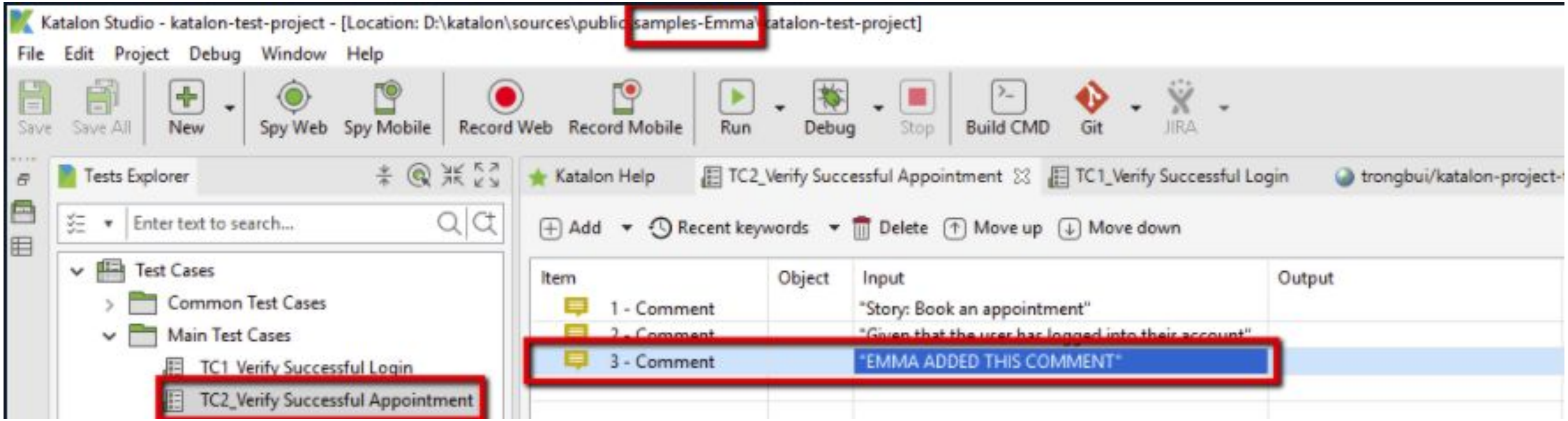
Resolve Git conflicts

In a source control system like Git, conflicts may occur when two or more people make changes to the same file concurrently. The conflicts may appear at a member's local repository or Git remote repository. In order to avoid conflicts, the team must collaborate following several Git practices. For example, before pushing new source code to the Git remote repository, one must remember to fetch the latest version from Git remote repository, resolve any conflicts and merge the code with local version.

The chart below demonstrates how conflicts may occur when Tom and Emma are working in the same project. The conflicts occur when Tom and Emma try to push new code to the Git remote repository without updating the changes from each other.

RESOLVE GIT

Let's consider the following situation: Tom and Emma are working on the same test case in a test project. Emma added a new comment ("EMMA ADDED THIS COMMENT"), then committed and pushed the change to the Git remote repository.

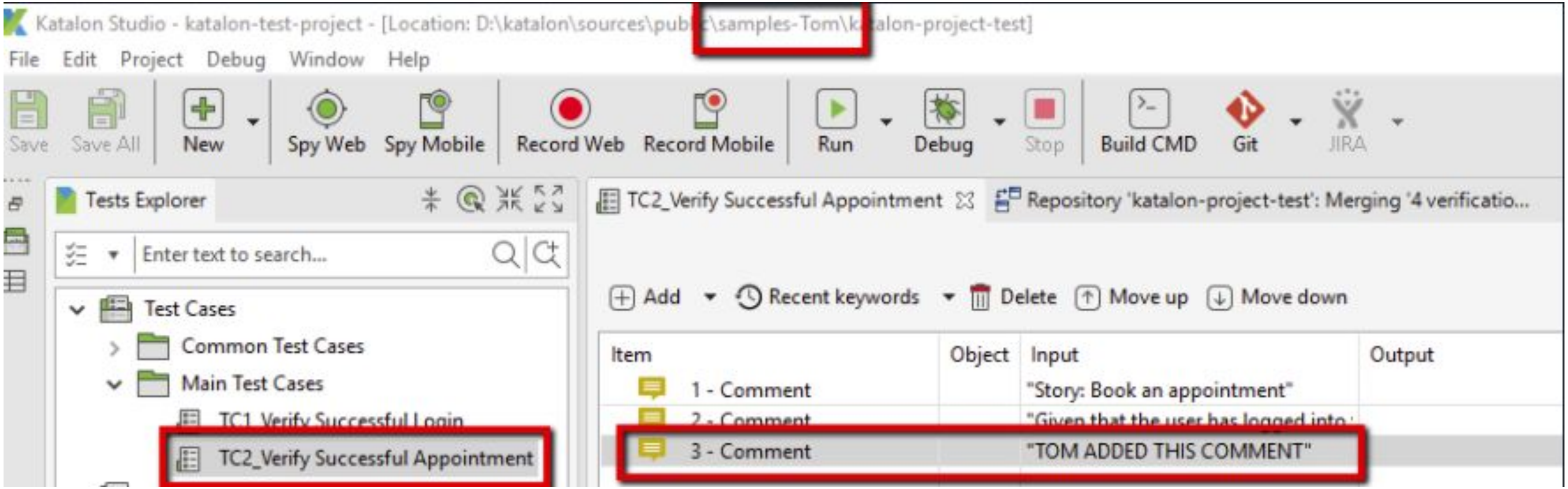


Katalon Studio - katalon-test-project - [Location: D:\katalon\sources\public\samples-Emma\katalon-test-project]

Item	Object	Input	Output
1 - Comment		"Story: Book an appointment"	
2 - Comment		"Given that the user has logged into their account"	
3 - Comment		"EMMA ADDED THIS COMMENT"	

RESOLVE GIT

At almost the same time, Tom also added a new comment ("TOM ADDED THIS COMMENT"), then committed and tried to push to the Git remote repository.


Katalon Studio - katalon-test-project - [Location: D:\katalon\sources\pub \samples-Tom\k alon-project-test]
File Edit Project Debug Window Help
Save
Save All
New
Spy Web
Spy Mobile
Record Web
Record Mobile
Run
Debug
Stop
Build CMD
Git
JIRA
Tests Explorer
Enter text to search...
Test Cases
Common Test Cases
Main Test Cases
TC1_Verify Successful Login
TC2_Verify Successful Appointment
TC2_Verify Successful Appointment
Repository 'katalon-project-test': Merging '4 verificatio...
Add
Recent keywords
Delete
Move up
Move down

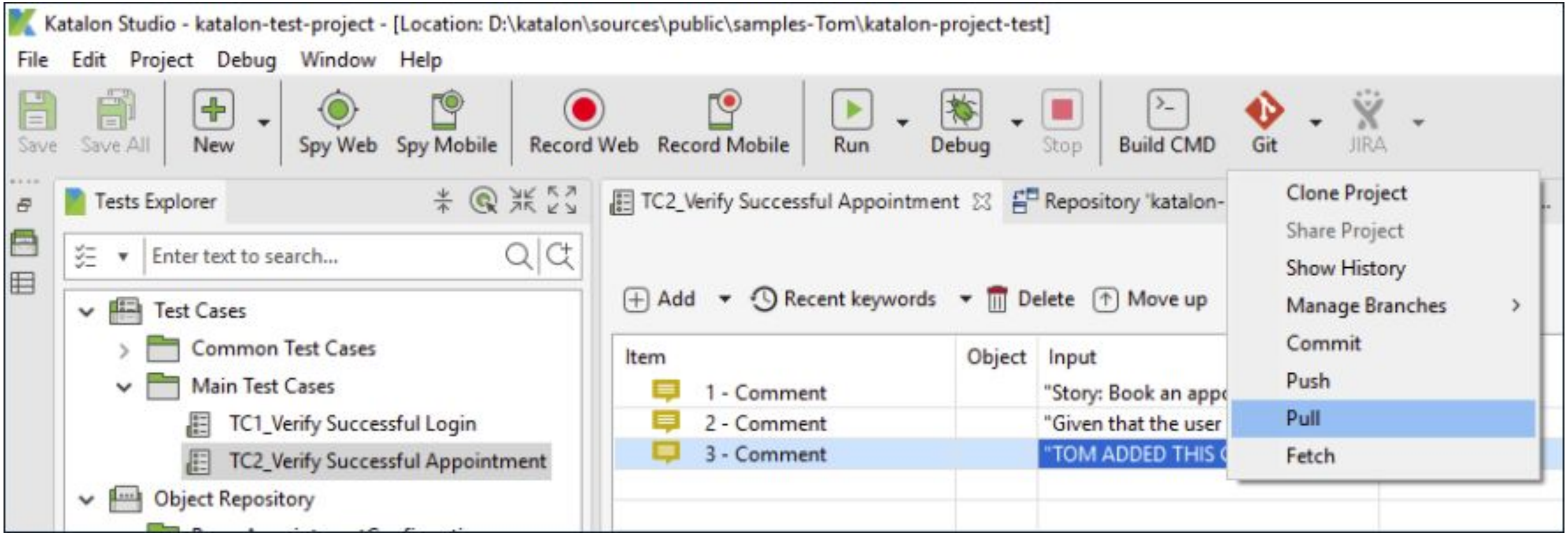
Item	Object	Input	Output
1 - Comment		"Story: Book an appointment"	
2 - Comment		"Given that the user has logged into	
3 - Comment		"TOM ADDED THIS COMMENT"	

What should Tom do to have push its change to the Git remote control?

Unfortunately, since Emma had pushed the code before Tom, so the version of code in Git was different from the version of code in Tom's local repository and therefore, Git rejected Tom's "push" action.

RESOLVE GIT

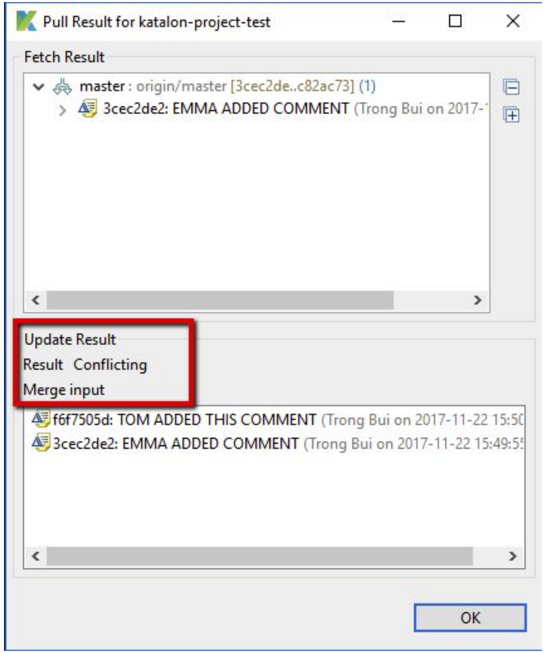
First, Tom has to "pull" the code from the Git remote repository to his local machine.



Item	Object	Input
1 - Comment		"Story: Book an appo
2 - Comment		"Given that the user
3 - Comment		"TOM ADDED THIS C

RESOLVE GIT

Obviously, Tom will see a message about the conflict:



Pull Result for katalon-project-test

Fetch Result

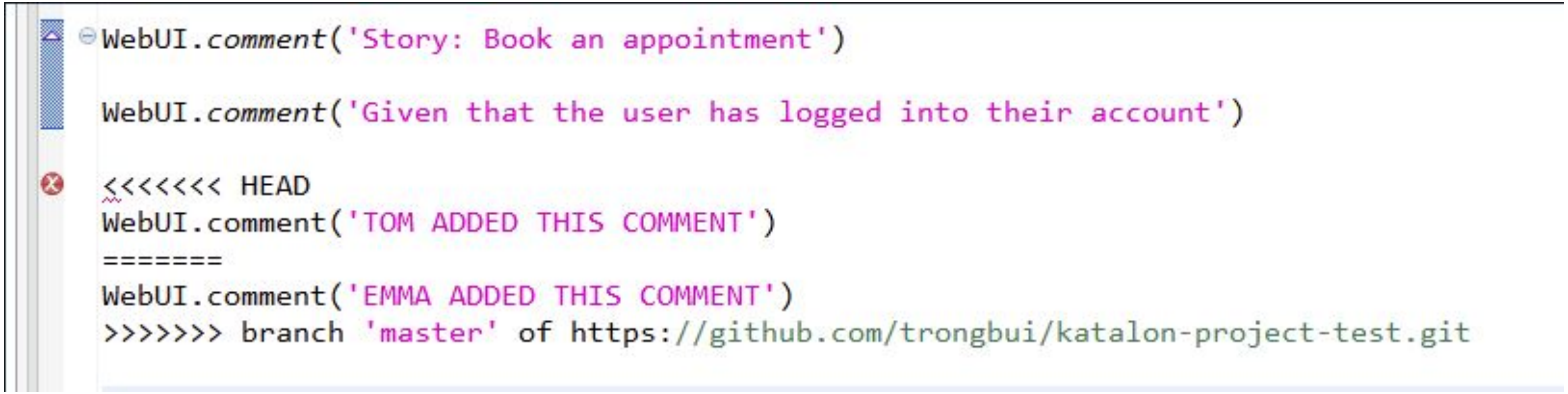
master : origin/master [3cec2de..c82ac73] (1)
3cec2de2: EMMA ADDED COMMENT (Trong Bui on 2017-

Update Result
Result Conflicting
Merge input

f6f7505d: TOM ADDED THIS COMMENT (Trong Bui on 2017-11-22 15:50
3cec2de2: EMMA ADDED COMMENT (Trong Bui on 2017-11-22 15:49:5

RESOLVE GIT

In the "Script" mode of the test case "TC2_Verify Successful Appointment" in Tom's Katalon Studio project, there are errors with indicators such as "<<<<<<<" (convention from Git). Let's look at the script more carefully:

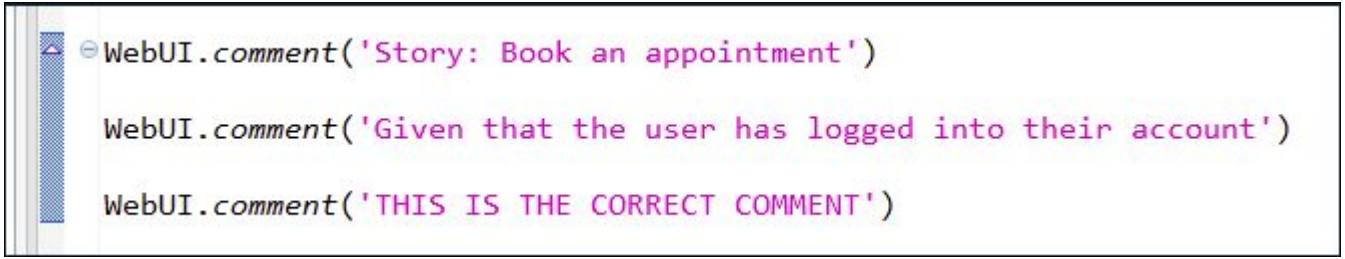


```
WebUI.comment('Story: Book an appointment')

WebUI.comment('Given that the user has logged into their account')

<<<<<<< HEAD
WebUI.comment('TOM ADDED THIS COMMENT')
=======
WebUI.comment('EMMA ADDED THIS COMMENT')
>>>>>>> branch 'master' of https://github.com/trongbui/katalon-project-test.git
```

Recall that the comments were added by Tom and Emma, and the "conflict" is now on Tom's Katalon Studio project. Everything within "<<<<<<< HEAD" and "=======" is the change from Tom. And, everything within "=======" and ">>>>>>> branch 'master'..." comes from Emma, which is currently in the Git remote repository.

RESOLVE GIT

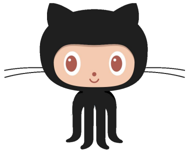
Now Tom has to decide which change is correct, or both are correct or wrong. Tom has to replace these lines of code by the correct one, e.g., "THIS IS THE CORRECT COMMENT":



```
WebUI.comment('Story: Book an appointment')

WebUI.comment('Given that the user has logged into their account')

WebUI.comment('THIS IS THE CORRECT COMMENT')
```

After resolving the conflict, Tom is now able to commit and push the change to the Git remote repository.

SUGGESTION



- Commit often: do not wait until a huge amount of scripts created to commit and push to the Git remote repository. The smaller set of the scripts is pushed, the easier you resolve the conflict.
- Pull changes from the Git remote repository before working on new scripts and before committing.
- Each member works on each feature at a time.

In order to minimize the conflict in a team having more than one members, you should define a process from the very beginning so that all team members are on the same page when using Git. Here are some suggestions for good practices:

Integrations With JIRA

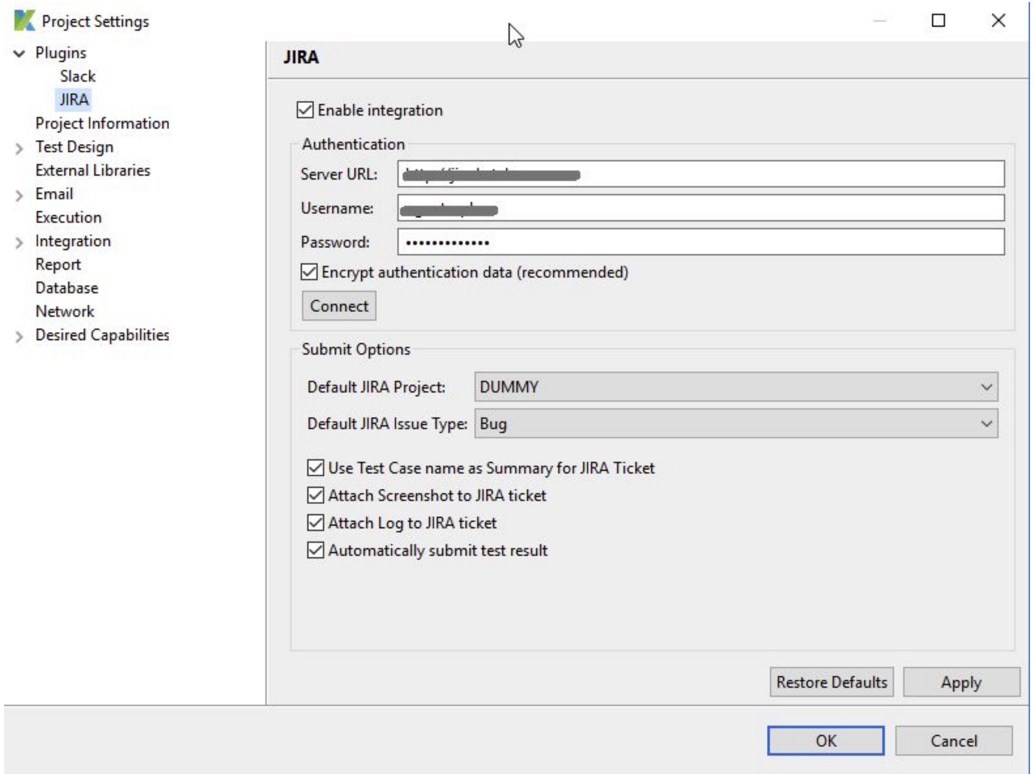
INTEGRATION WITH Jira

Prerequisites

- Install Jira Integration plugin for Katalon Studio from Katalon Store.
- Install Katalon BDD app for Jira from Atlassian Marketplace.

INTEGRATION WITH Jira

You need to enable JIRA Integration in order to submit issues to JIRA. This setting is available at Project > Settings > Integration > JIRA.



JIRA

☑ Enable integration

Authentication

Server URL: [redacted]

Username: [redacted]

Password: •••••••••••••

☑ Encrypt authentication data (recommended)

Submit Options

Default JIRA Project: DUMMY

Default JIRA Issue Type: Bug

☑ Use Test Case name as Summary for JIRA Ticket

☑ Attach Screenshot to JIRA ticket

☑ Attach Log to JIRA ticket

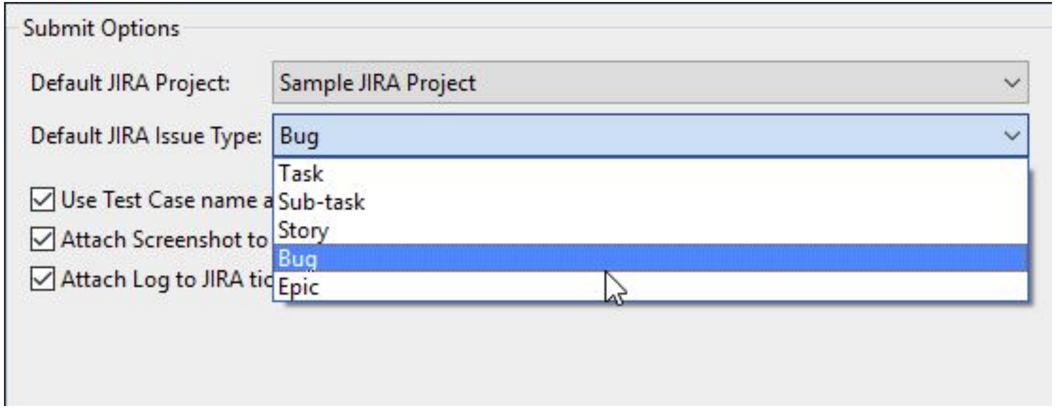
☑ Automatically submit test result

INTEGRATION WITH Jira

1. Select Enable integration option. The settings will be available for configuration.

- Jira Cloud
  - *Server URL* must be in the form *https://<site_name>.atlassian.net*.
  - Use a Atlassian Cloud's API token for *Password*. See instructions here.
- Jira Server
  - *Server URL* must be in the form *http(s)://domain* without any trailing parts e.g. */secure*.
  - Use username instead of email for *Username*.

2. Specify information regarding your JIRA Server and login credential then click Connect button.
3. After successfully authenticating with JIRA, all relevant JIRA Projects and Issue Types will be retrieved and displayed under Submit Options. You can specify the default project and issue type for submission here.



Submit Options

Default JIRA Project: Sample JIRA Project

Default JIRA Issue Type: Bug

☑ Use Test Case name a

☑ Attach Screenshot to

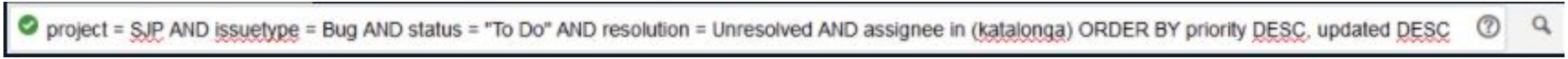
☑ Attach Log to JIRA tic

4. Click OK button to complete the JIRA Integration setup.

INTEGRATE TEST CASE

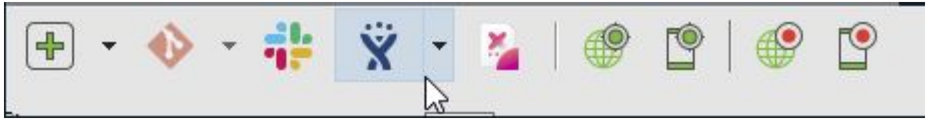
You need to enable Jira integration with Katalon TestOps to have an insightful look at your testing data and better test management.

1. Prepare Jira JQL Script

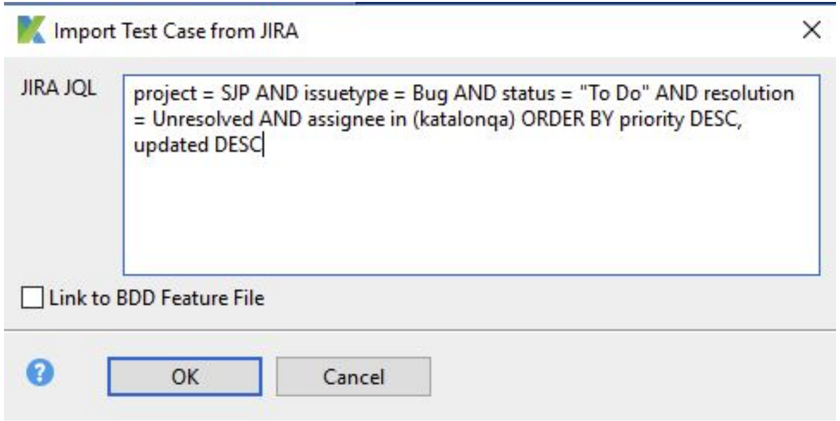


```
project = SJP AND issuetype = Bug AND status = "To Do" AND resolution = Unresolved AND assignee in (katalonqa) ORDER BY priority DESC, updated DESC
```

2. Open Jira Integration > click Import Test Case from JIRA JQL



3. Enter the Jira JQL. Click OK.



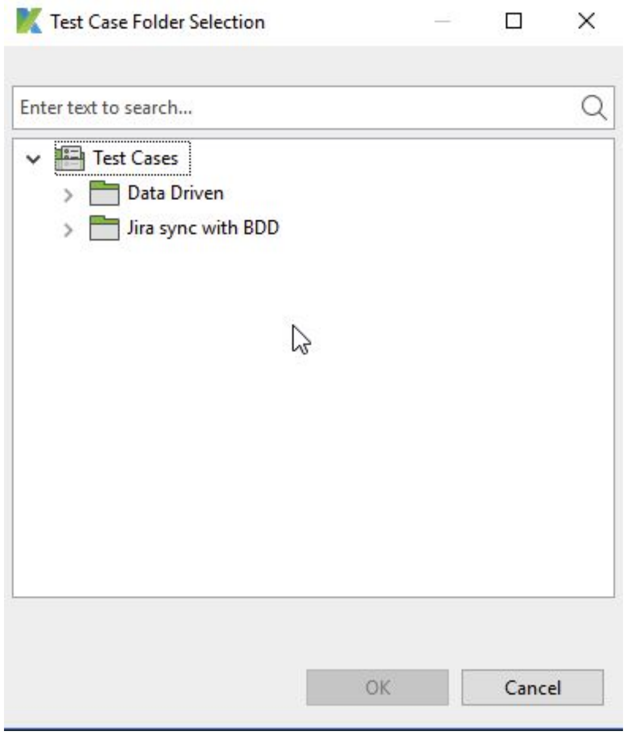
Import Test Case from JIRA

JIRA JQL: project = SJP AND issuetype = Bug AND status = "To Do" AND resolution = Unresolved AND assignee in (katalonqa) ORDER BY priority DESC, updated DESC

☐ Link to BDD Feature File

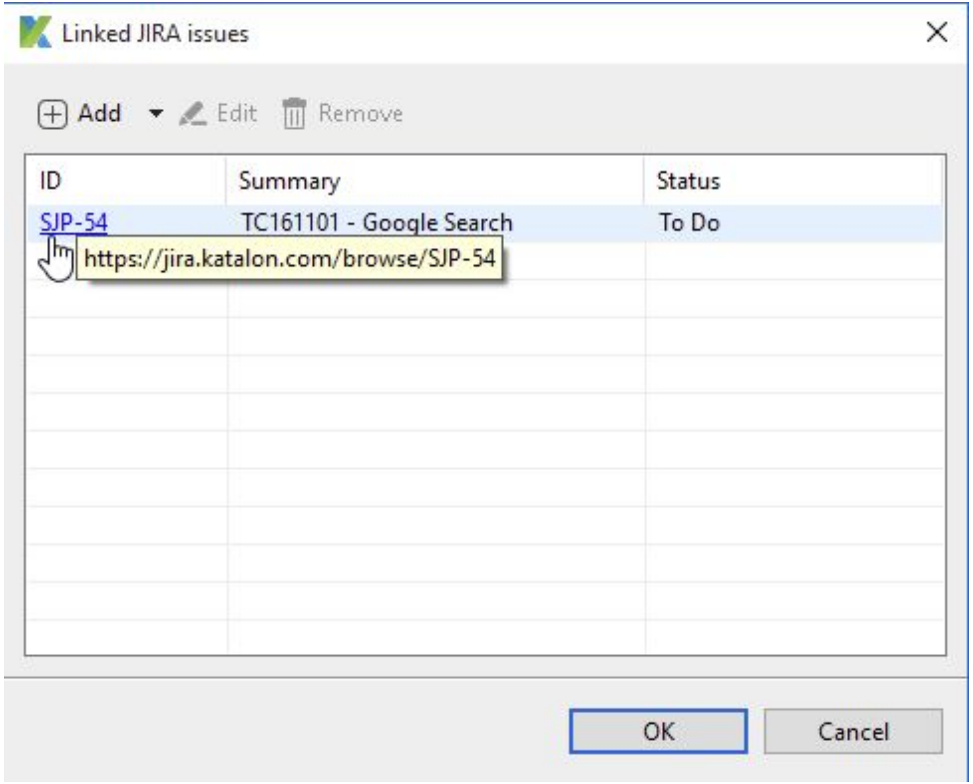
INTEGRATE TEST CASE

4. The Test Case Folder Selection window will appear. Choose the destination to store the issues. Click OK.



INTEGRATE TEST CASE

5. The Jira Issues window will appear. Click OK.



Linked JIRA issues

ID	Summary	Status
SJP-54	TC161101 - Google Search	To Do

https://jira.katalon.com/browse/SJP-54

“

If your test cases have already been linked to a JIRA ticket,
Katalon Studio will not sync them again.

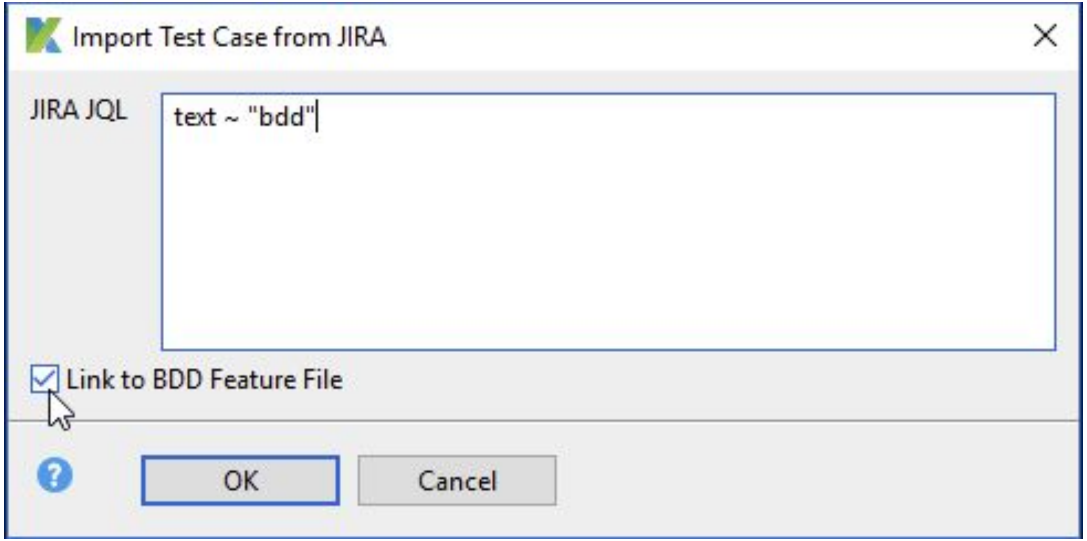
”

IMPORT BDD FEATURES FILES

Import BDD Feature Files

Jira Integration also allows you to import Jira BDD Feature Files to Katalon Studio.

When importing test cases from Jira, please check Link to BDD Feature File > OK > Choose the destination to store the issues.



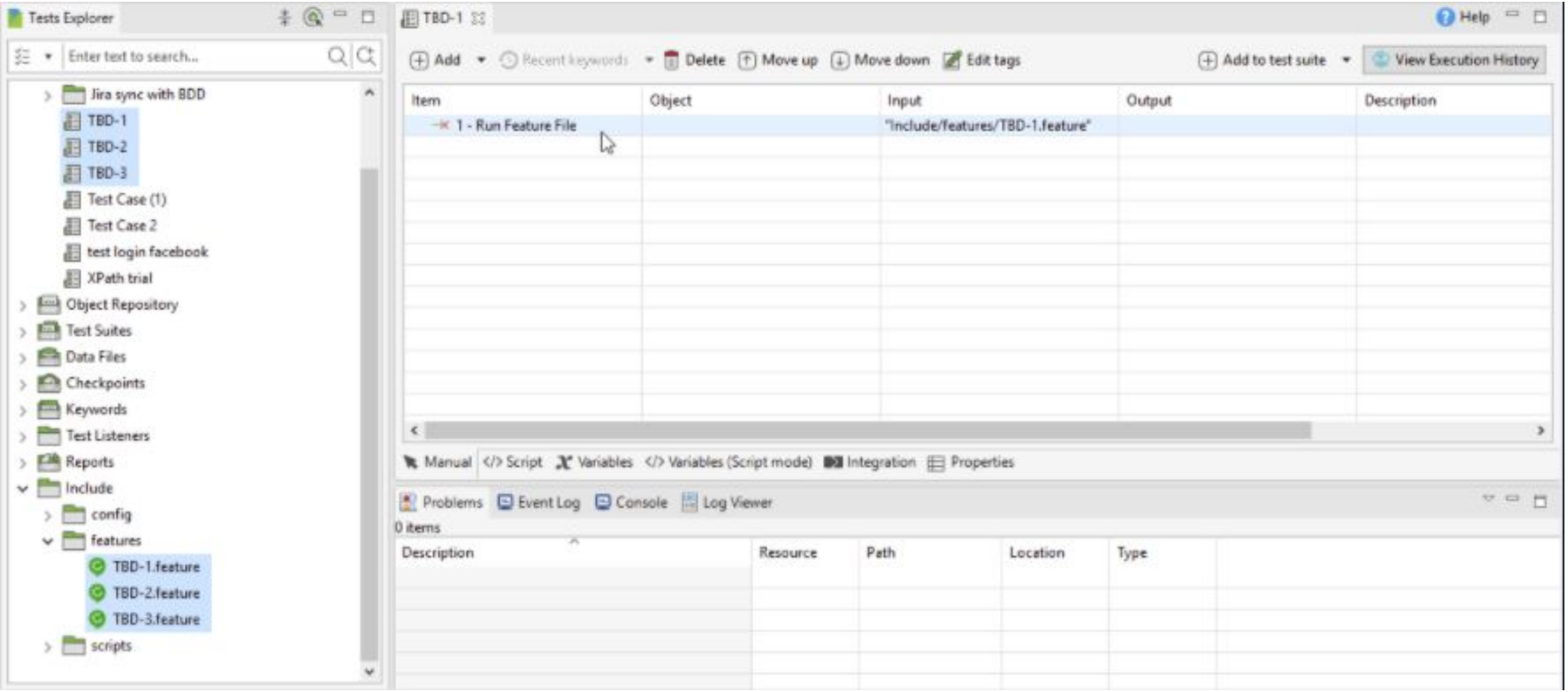
Import Test Case from JIRA

JIRA JQL: text ~ "bdd"

☑ Link to BDD Feature File

IMPORT BDD FEATURES FILES

A new Feature File (with the same name as the test case) will be created with the content from Jira BDD. Moreover, a RunFeature step will be created in the linked test case to Jira.



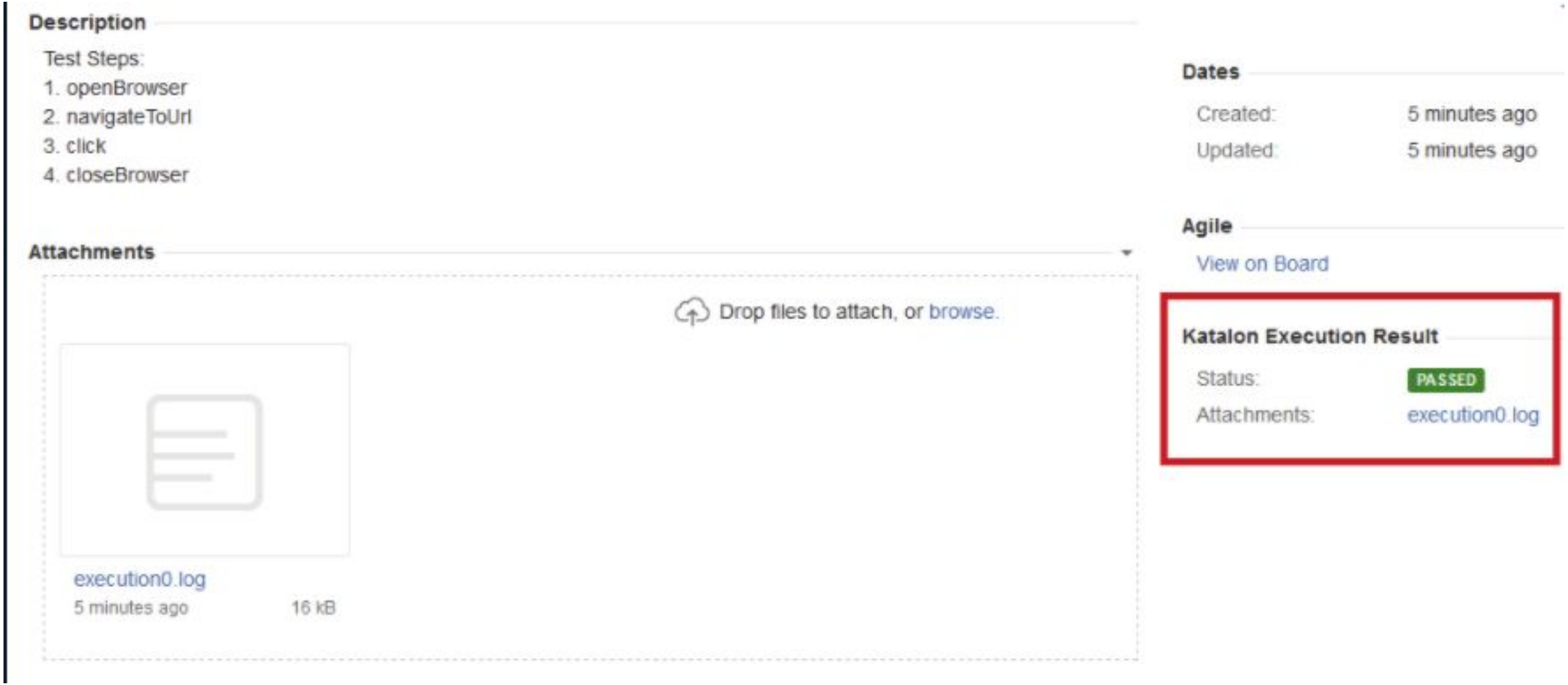
Item	Object	Input	Output	Description
1 - Run Feature File		"Include/features/TBD-1.feature"		

0 items

Description	Resource	Path	Location	Type

VIEW TEST RESULTS ON Jira

After you have successfully integrated test cases, test execution results will be automatically created in the associated Jira ticket. Review the status and attachments of Katalon Studio test cases right inside Jira.



Description

Test Steps:
1. openBrowser
2. navigateToUrl
3. click
4. closeBrowser

Attachments

Drop files to attach, or browse.

execution0.log
5 minutes ago 16 kB

Dates

Created:	5 minutes ago
Updated:	5 minutes ago

Agile

View on Board

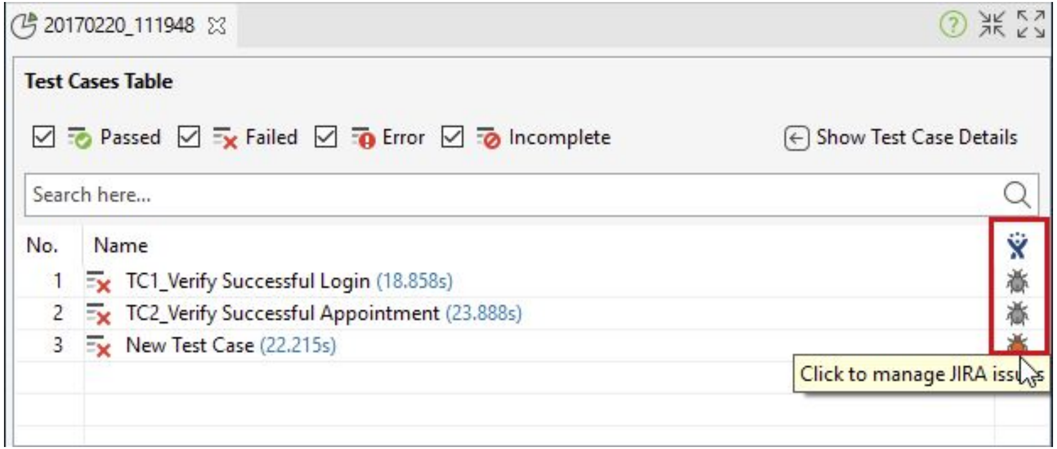
Katalon Execution Result

Status:	PASSED
Attachments:	execution0.log

SUBMIT AN ISSUE TO Jira

Bug submission options will be available in Test Reports after JIRA Integration setup is successfully configured.

1. Open a test execution in Reports that you want to review for issues. In Test Cases Table, a dedicated column for JIRA Integration will be enabled.



20170220_111948

Test Cases Table

☑ Passed ☑ Failed ☑ Error ☑ Incomplete

Show Test Case Details

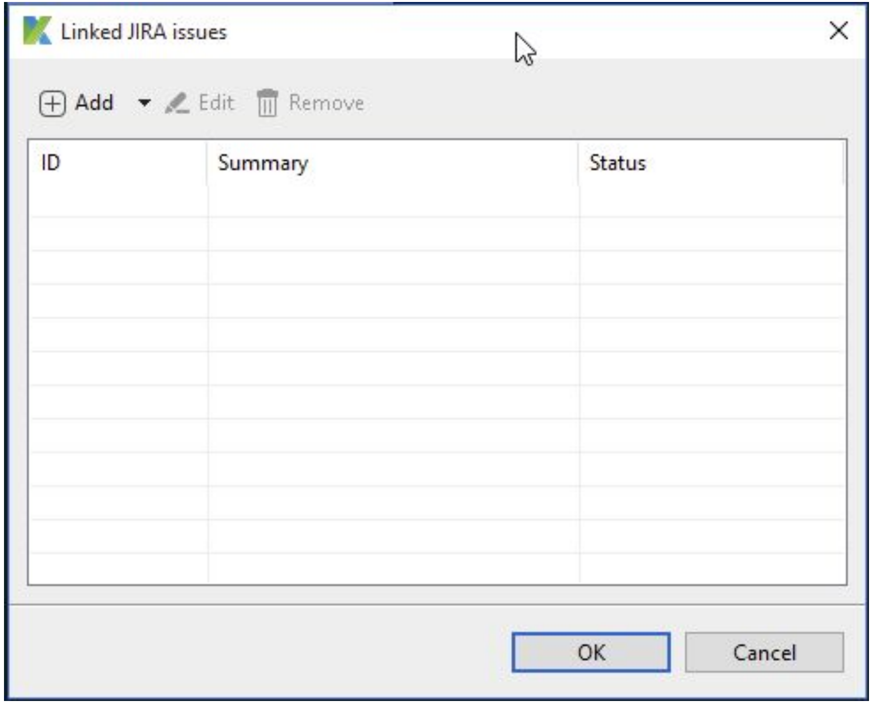
Search here...

No.	Name
1	TC1_Verify Successful Login (18.858s)
2	TC2_Verify Successful Appointment (23.888s)
3	New Test Case (22.215s)

Click to manage JIRA issues

SUBMIT AN ISSUE TO Jira

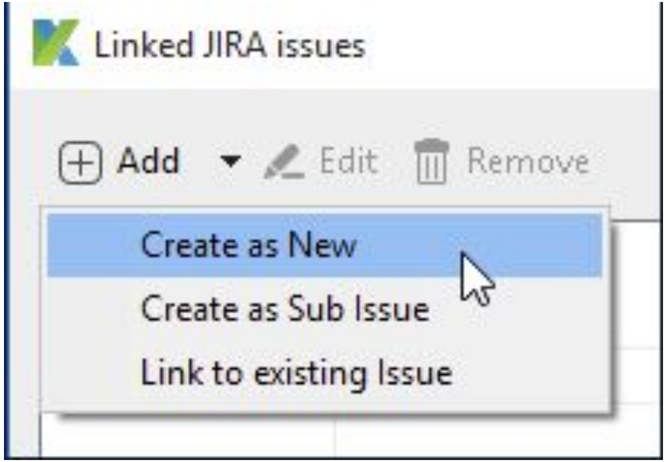
2. Click on the bug icon to display the list of related JIRA issues associated with the selected Test Case. The issues are shown in the following screen.



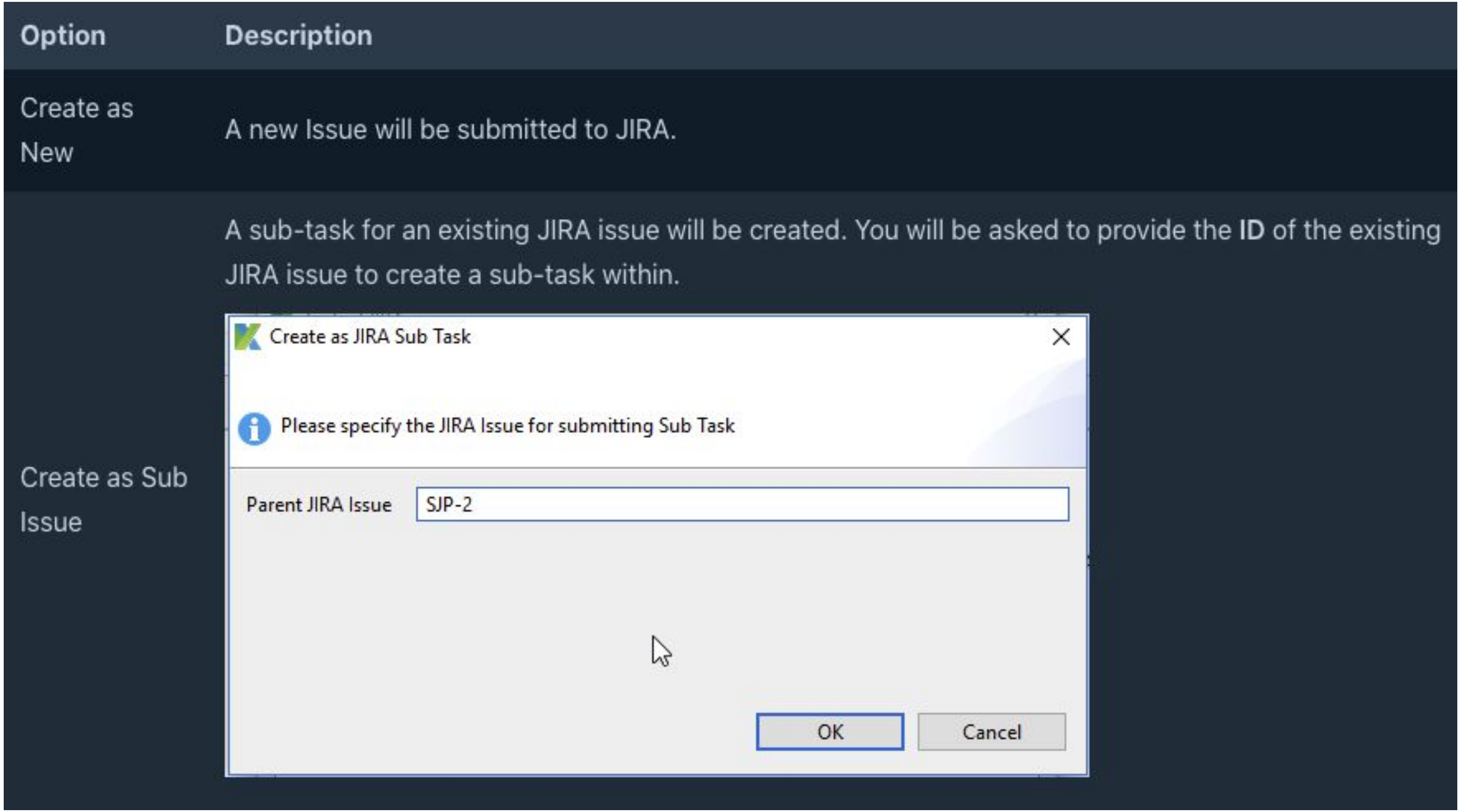
ID	Summary	Status

SUBMIT AN ISSUE TO Jira

3. Select submit option under the Add command.



SUBMIT AN ISSUE TO Jira



Option	Description
Create as New	A new Issue will be submitted to JIRA.
Create as Sub Issue	A sub-task for an existing JIRA issue will be created. You will be asked to provide the **ID** of the existing JIRA issue to create a sub-task within.

Create as JIRA Sub Task

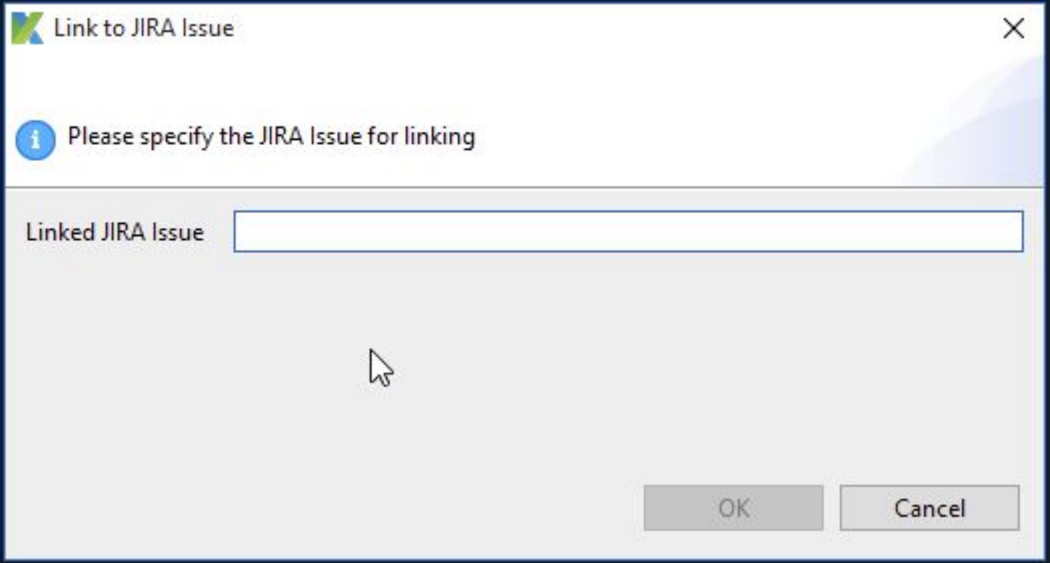
Please specify the JIRA Issue for submitting Sub Task

Parent JIRA Issue: SJP-2

SUBMIT AN ISSUE TO Jira

This option will append execution details to an existing JIRA issue. You will be asked to provide the ID of the existing JIRA issue for this.

Link to existing Issue



Link to JIRA Issue

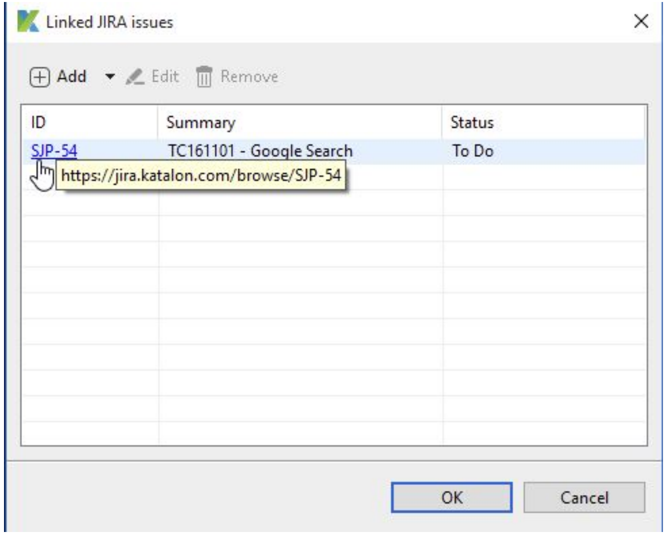
Please specify the JIRA Issue for linking

Linked JIRA Issue

SUBMIT AN ISSUE TO Jira

5. Based on your preferences in JIRA Integration settings, the Summary, Screenshots, Logs, Reporter, and Description of test cases will be populated and attached accordingly. Once done, click on the Create button at bottom of the form.

6. A created JIRA issue will have its ID recorded in the Linked JIRA issues list so that you can quickly navigate there from Katalon Studio. You can also edit linked JIRA issue or remove the linking of the created JIRA issue.

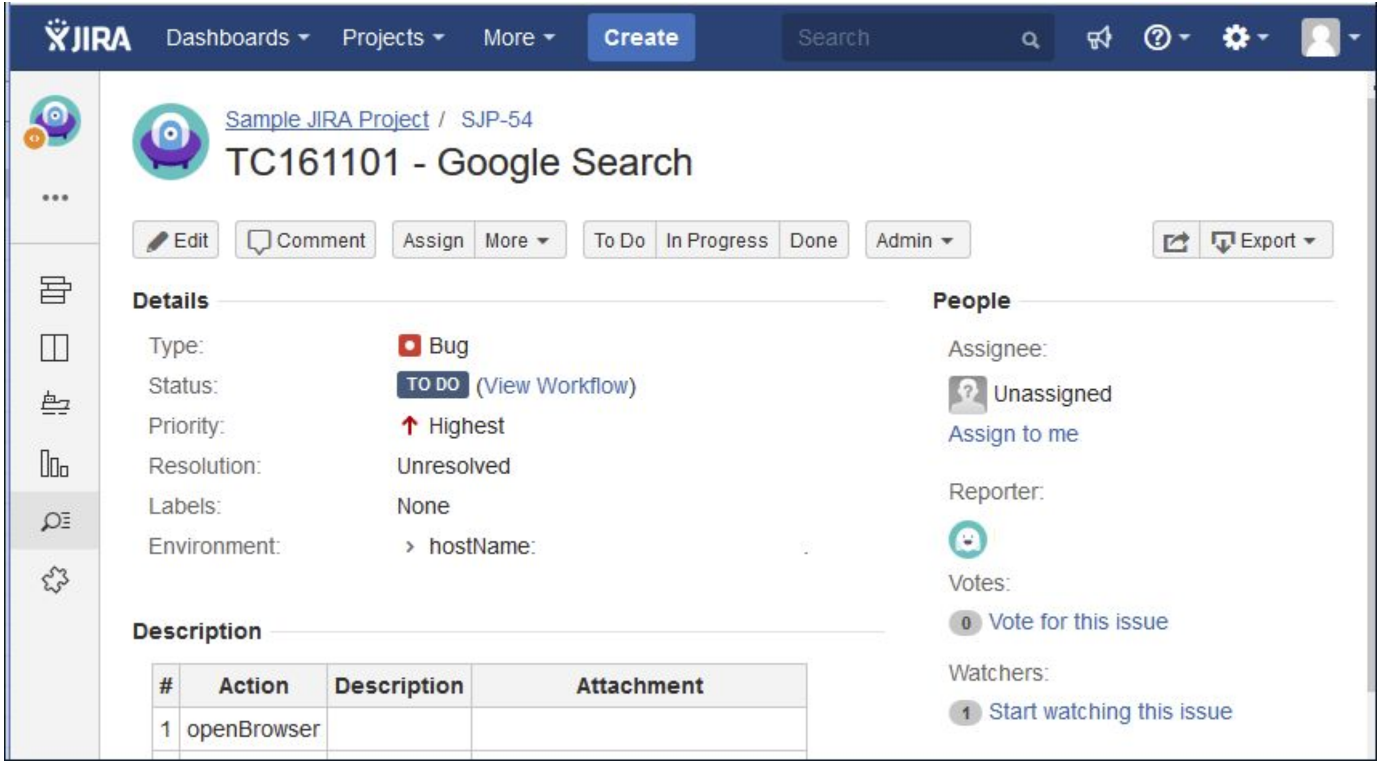


ID	Summary	Status
SJP-54	TC161101 - Google Search	To Do

https://jira.katalon.com/browse/SJP-54

SUBMIT AN ISSUE TO Jira

7. Once clicked on ID, you will be taken to JIRA issues page accordingly as shown below



Sample JIRA Project / SJP-54

TC161101 - Google Search

Details

Type:	Bug
Status:	TO DO (View Workflow)
Priority:	Highest
Resolution:	Unresolved
Labels:	None
Environment:	› hostName:

People

Assignee:

Unassigned

Assign to me

Reporter:

Votes:

0 Vote for this issue

Watchers:

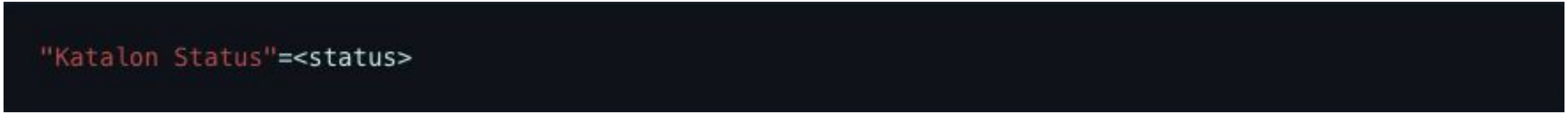
1 Start watching this issue

Description

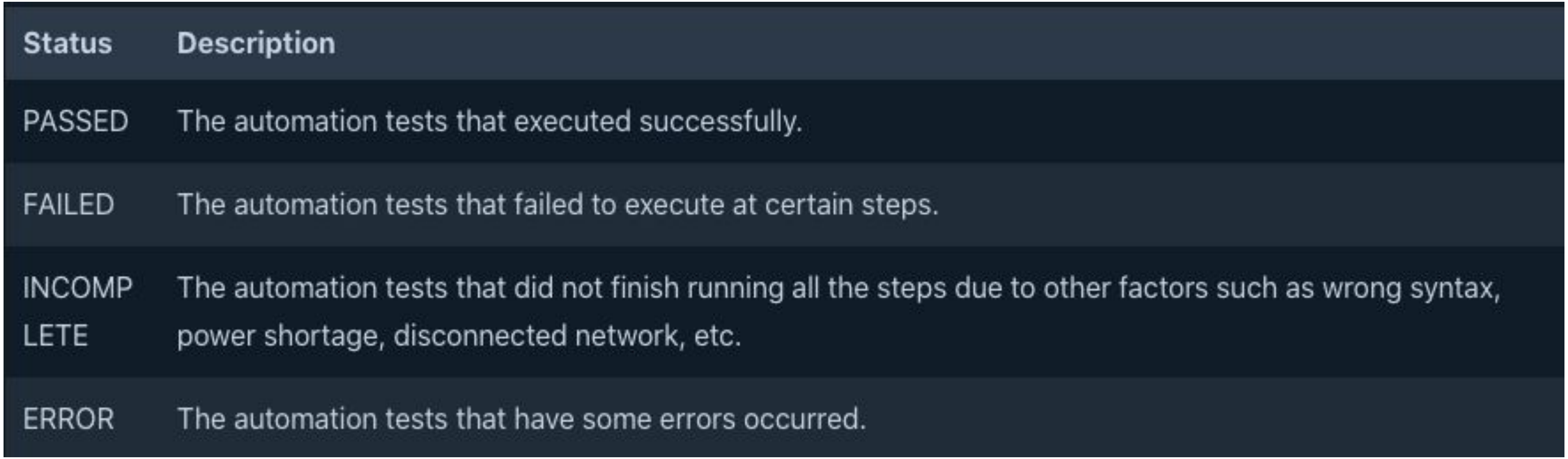
#	Action	Description	Attachment
1	openBrowser		

JQL SYNTAX

Katalon Studio test execution status can be queried via JQL. The syntax is as following:



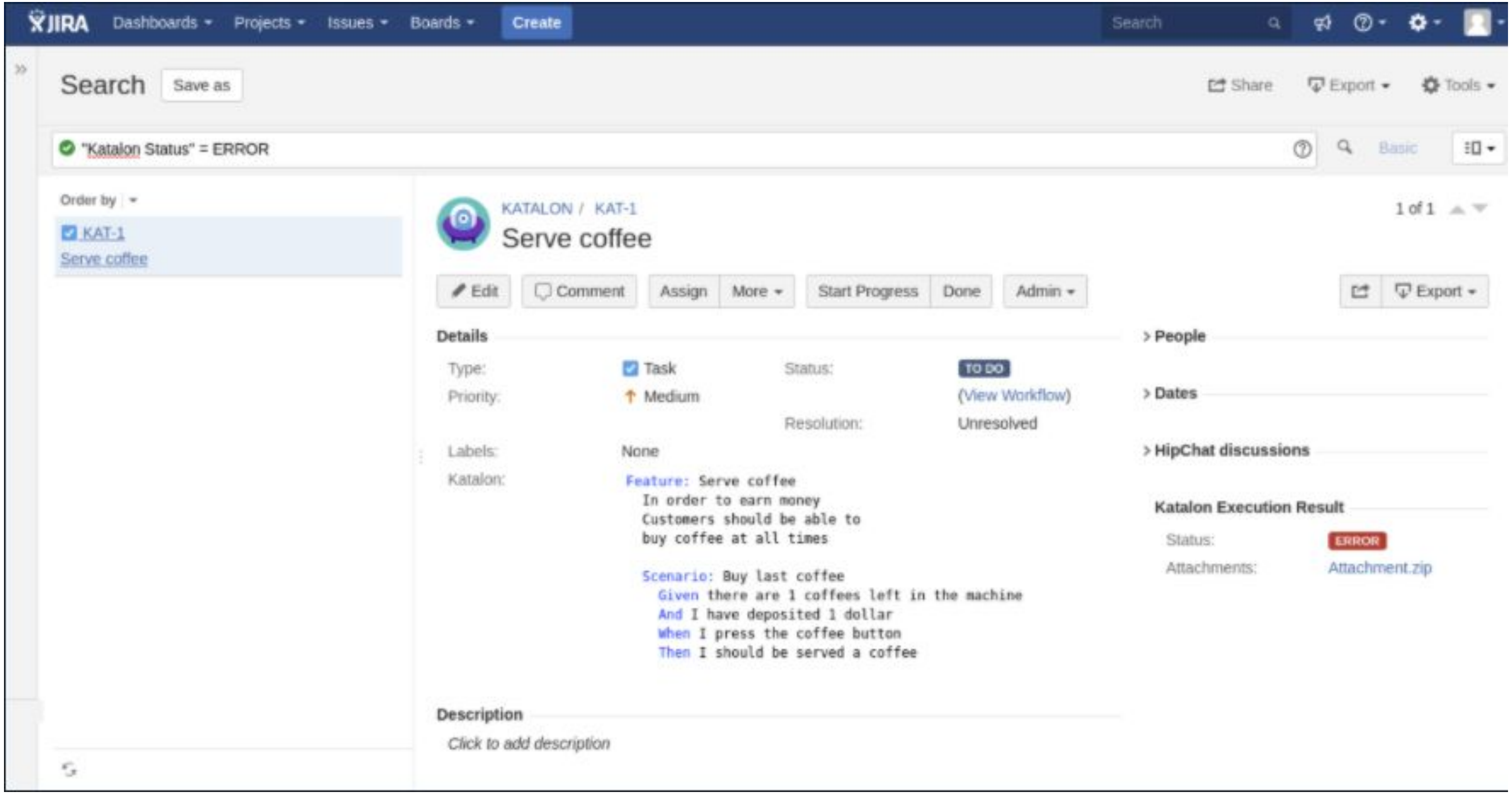
```
"Katalon Status"=<status>
```



Status	Description
PASSED	The automation tests that executed successfully.
FAILED	The automation tests that failed to execute at certain steps.
INCOMPLETE	The automation tests that did not finish running all the steps due to other factors such as wrong syntax, power shortage, disconnected network, etc.
ERROR	The automation tests that have some errors occurred.

JQL SYNTAX

For example, to search for all issues that have failed in Katalon Studio test execution_:_



"Katalon Status" = ERROR

KAT-1
Serve coffee

KATALON / KAT-1

Serve coffee

Details

Type:	☑ Task	Status:	TO DO
Priority:	↑ Medium		(View Workflow)
		Resolution:	Unresolved
Labels:	None		

Katalon:

```
Feature: Serve coffee
  In order to earn money
  Customers should be able to
  buy coffee at all times

  Scenario: Buy last coffee
    Given there are 1 coffees left in the machine
    And I have deposited 1 dollar
    When I press the coffee button
    Then I should be served a coffee
```

Description

Click to add description

Katalon Execution Result

Status:	ERROR
Attachments:	Attachment.zip