Roadmap

Xray's roadmap is constantly reviewed and redefined. We update it often depending on the feedback we receive from our clients and internal stakeholders.

Our release plan is available in our Jira issue tracker, although account registration is required. Feel free to vote on the issues that you would like to see implemented.

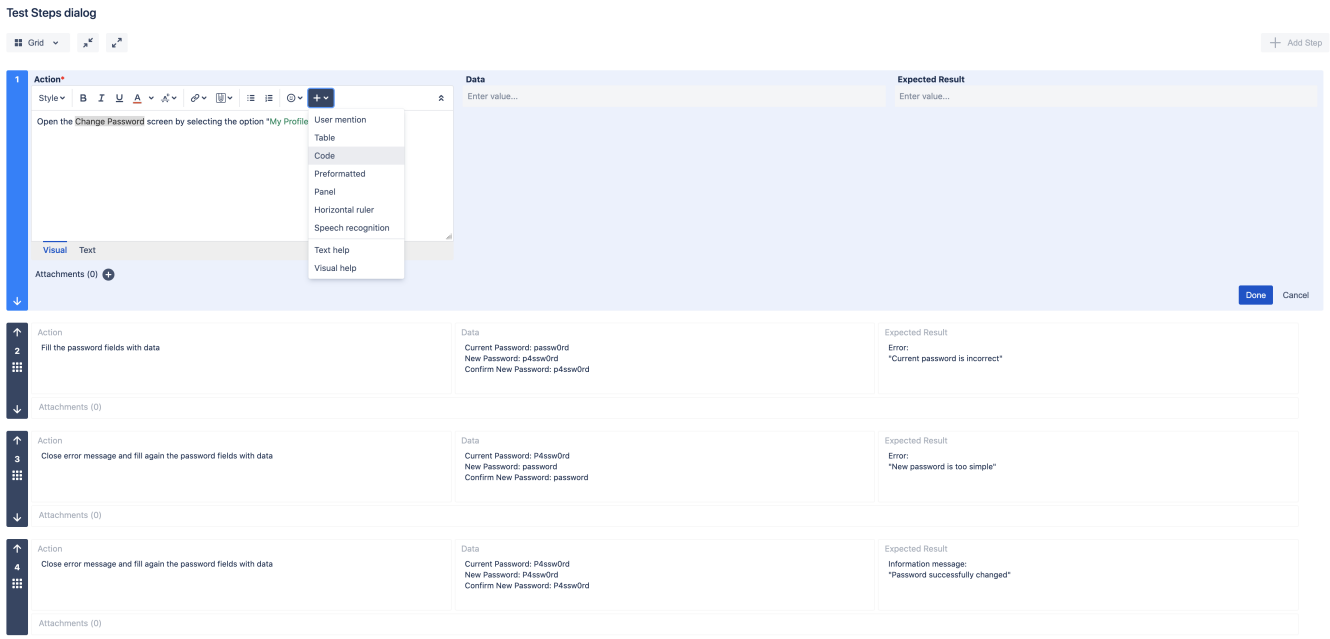
Here you can find a list of features that define our main goals for future releases. This doesn't mean that other, potentially smaller features, won't be implemented as well.

Shipped

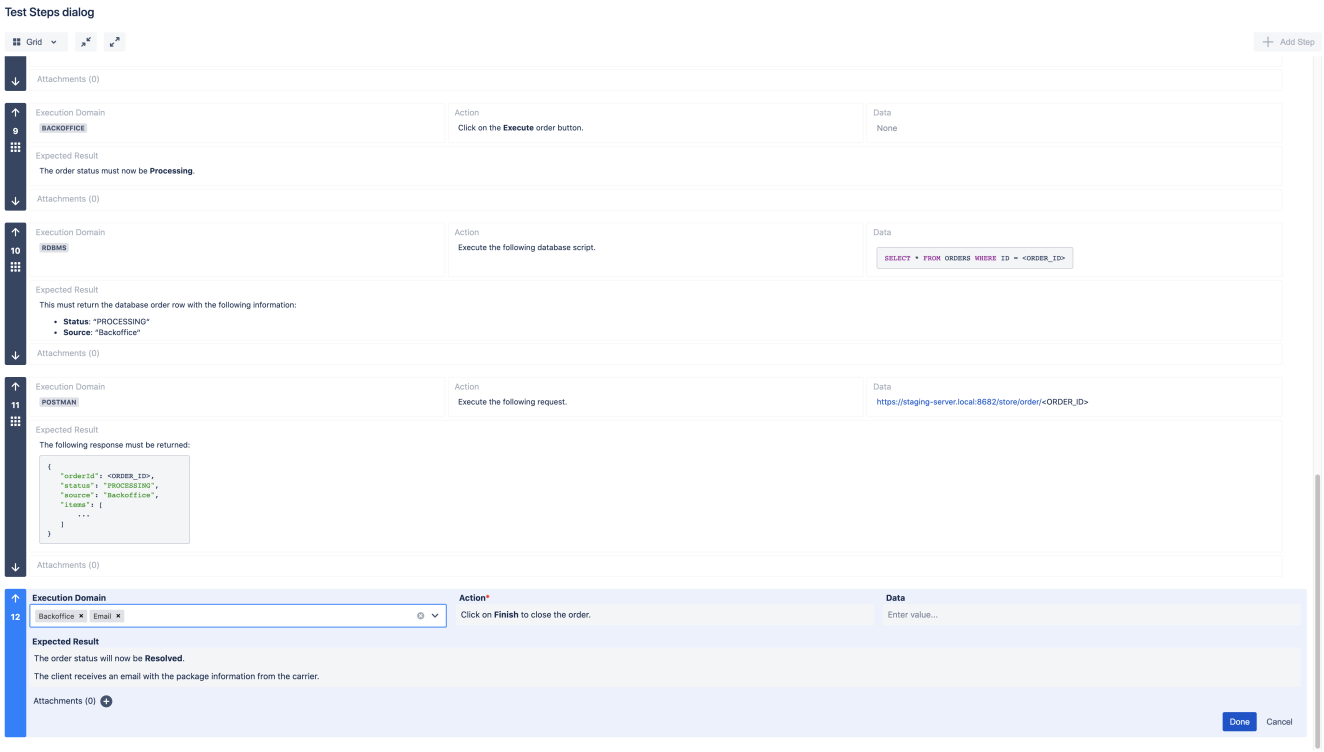
V4.0

The Test Steps component was the target of a major facelift with the goal of providing a wider, more usable UI that will have the ability to edit all step-related fields at once, in a grid or column-based layout.

Editing, reading, and navigating through Test steps are now much easier with this new UI.



Along with the Test steps UI revamp, Xray also provides the ability to configure and specify new custom step fields for manual test cases that can complement the standard ones (Action/Step, Data, Expected Result). The standard custom fields can also be hidden if desired. All of this can be configured in the project settings.



Starting from v4.0, It is possible to define additional Test Run custom fields. These fields can be useful to add extra information to Test Runs, usually only available during or after executing Tests.

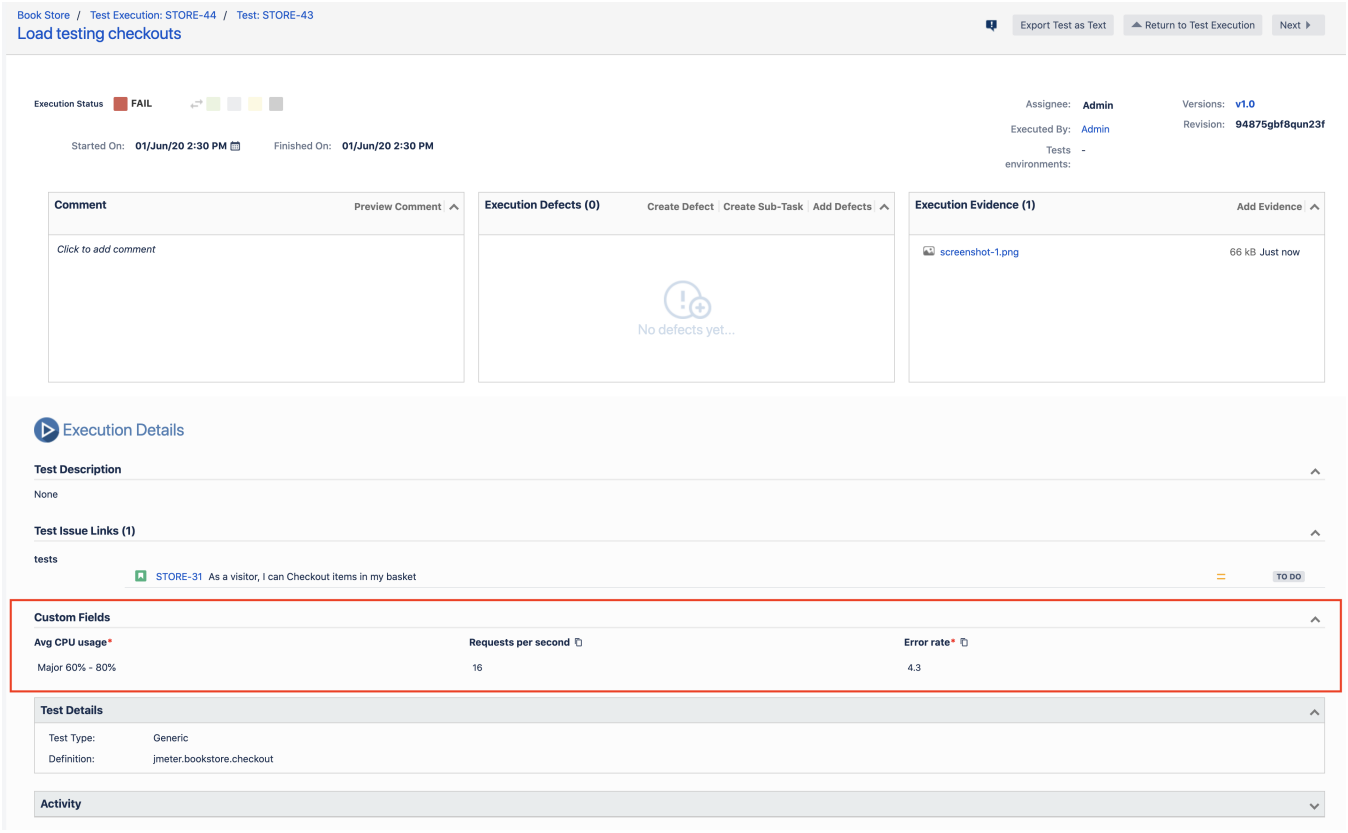
Test Run custom fields can be configured by project and by Test Type. Therefore, these settings will not affect other projects within your Jira instance. For example, it is possible to have custom fields just for Manual Tests within a project.

Reporting

The Test Runs List report provides an additional column that displays the Test Run Custom Field values for each Test Run.

It is also possible to include these fields on the Tests data-table within the Test Execution issue.

Learn more about this feature here.



V5.0

Test parameterization is a powerful practice that allows the same test to be executed multiple times with different parameters. Parameters are similar to input values (variables) that can change with each execution.

Parameterized tests in Xray are defined just like any other test with the addition of some parameter names within the specification using the following notation: **${PARAMETER_NAME}.** This notation is used to reference parameters within the **test steps**.

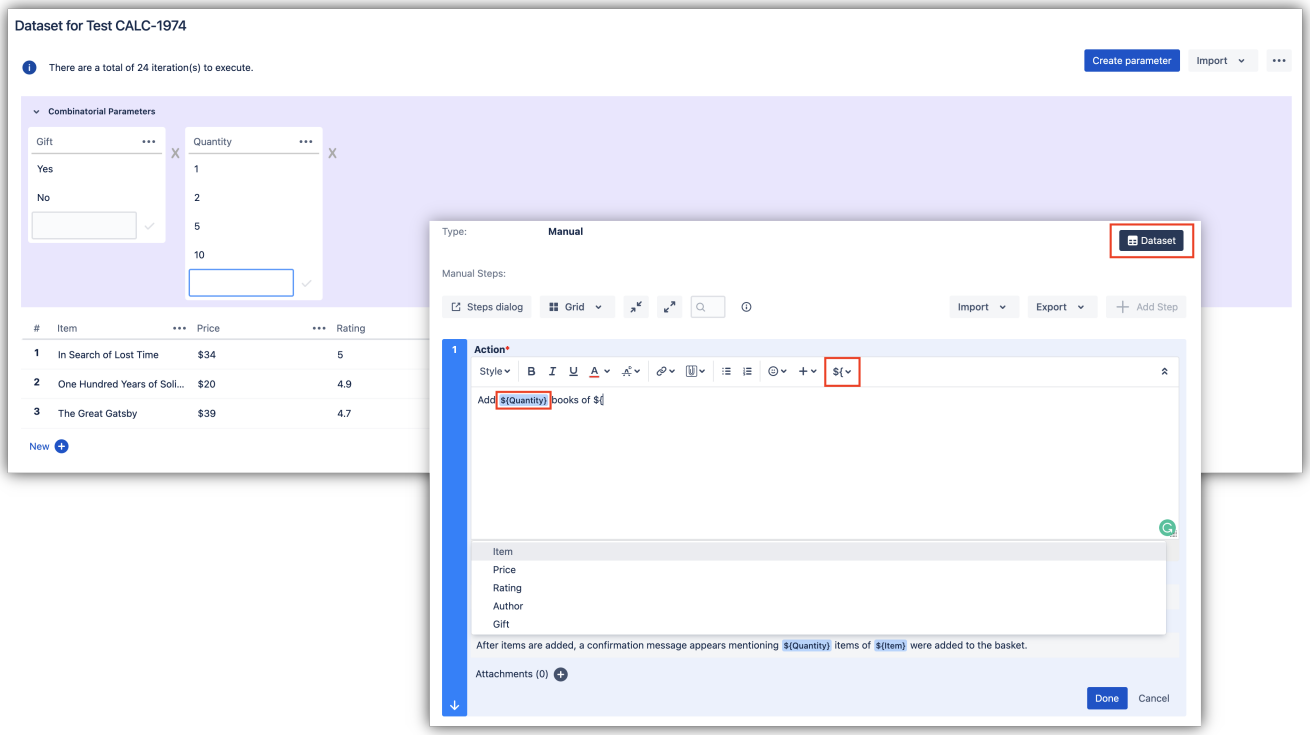
Precondition issues can also be parameterized by including parameter names in the precondition specification.

The parameters, along with their values, are defined within a **dataset**. A dataset is a collection of data represented with a tabular view where every column of the table represents a particular variable (or **parameter**), and each row corresponds to a given record (or **iteration**) of the dataset. The number of rows in the dataset determines the number of iterations to execute.

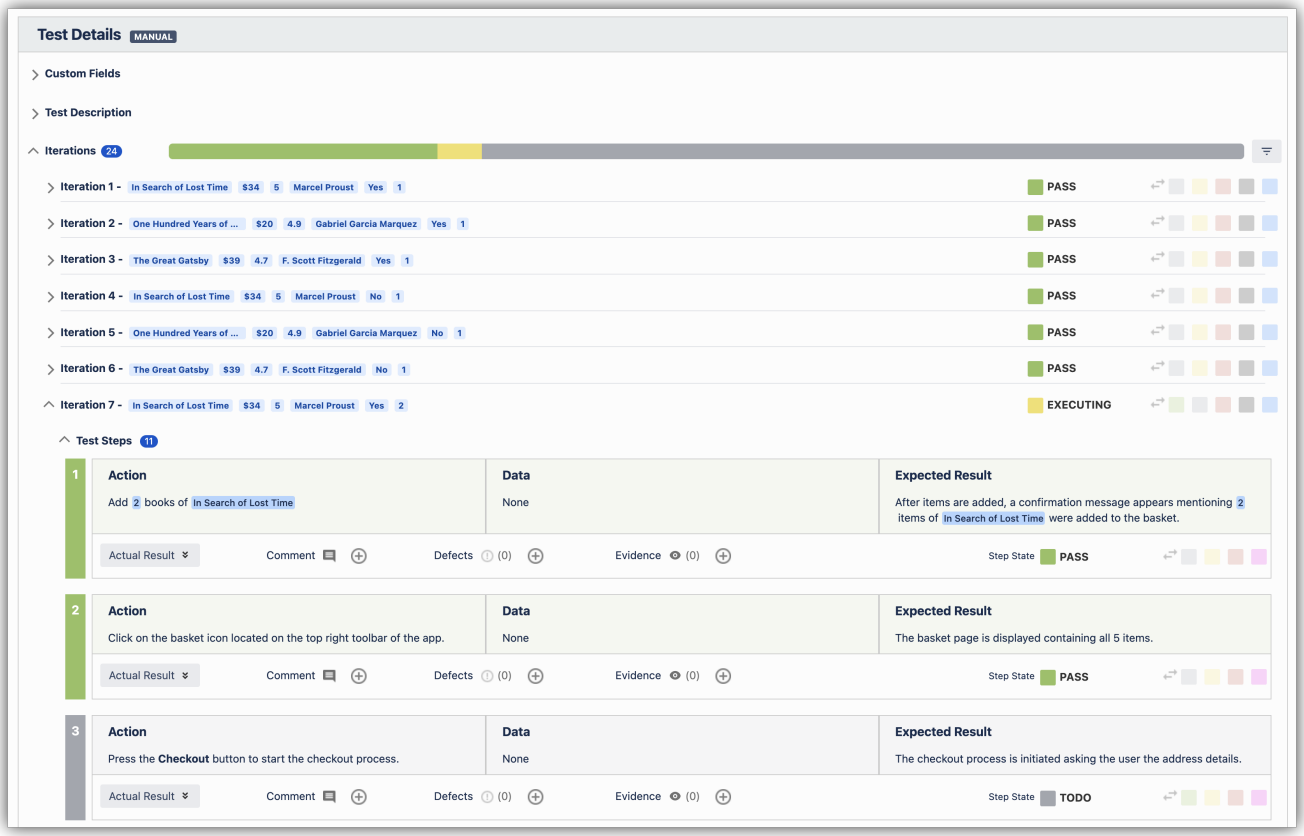
A dataset can be defined in the following entities/scopes:

1. Test (default dataset)
2. Test Plan - Test
3. Test Execution - Test (Test Run)

The closest dataset to the test run will be the one used to generate the iterations, effectively overriding any dataset defined in higher levels.



All iterations for a given test are executed within the context of the same test run. Each iteration can be expanded, and the steps executed individually. The step parameters will be replaced by the corresponding iteration values. The steps affect the iteration status that, in turn, affects the overall test run status.



Learn more about parameterized tests here.

V6.0

Modular test design is a way of promoting test case **reusability** and **composition** across a large test repository. To design modular tests, you can create a test where some of the steps *call* or include other test cases. This prevents testers from having to write the same steps over and over again in different high-level tests. Using a modular design approach, any test can become a building block of more extensive test scenarios. Still, they can also be executed individually if needed.

A called test can, in turn, also call other tests. You can compose a test scenario with up to five levels of depth.

Modular tests can also be parameterized. When calling a test, you can provide new parameter values according to the parent test data.

Upon executing, Xray will unfold all called test steps in the test run. This becomes transparent to testers as they only have to follow and execute the steps on the execution, even though test steps might come from different tests issues.

A common use case for modular tests is end-to-end testing. End-to-end tests often need to pass through the same area or component of the application before asserting the final result. With modular test design, you can **reuse** the tests for these common areas or components.

Learn more about modular tests here.

Jira Datacenter supports archiving of issues and projects since v8.1. By archiving issues, Jira will remove them from the indexes. The archived issues will still be accessible in read-only mode, however, they will not appear in Jira searches. This enhances Jira performance because Jira stores less data.

The goal of this feature is to support issue archiving so that Xray issues can be archived without some of the undesirable side effects (described here). A good use case for archiving Xray issues is to archive old Test Executions, especially if continuous integration processes are used. These can quickly create many Test Execution issues while importing new execution results into Xray on a daily basis. These issues are good candidates to be archived.

Most of Xray's native reports are tied to the project level. We plan to add multi-project support for these reports by allowing a saved filer with issues from multiple projects to be used as the report data source.

In the works

This feature will cover a common scenario where manual Test cases evolve to automated Tests. In this case, the user can either change/select the Test type to see different definitions (refer to the configuration option: Delete test definition when changing Test type) or create separate Test issues to cover all the natures for the Test.

With Test natures, we plan to make it more explicit that multiple Test definitions can coexist within the same Test case. Users will also be able to set the current definition so that Test Runs are always created using the "current" definition.

Currently, the Test Plan is composed of a static list of Test cases. This means you must explicitly add the Tests to the Test Plan. If the Tests are all known and well defined when you start your Test Plan, this is ok. However, if you are working in an agile context where a Test Plan is created for a specific sprint, Tests will only be specified during the sprint and later added to the Test Plan. This process is not ideal because users might forget to add the Tests to the Test Plan.

Dynamic Test Plans can be defined with a JQL query which will be the source for Test cases. Considering the use case described above, we can define a JQL query that will get all Tests covering requirements of a specific sprint.

Future Versions

This feature will allow test engineers to create or generate environments with variables such as Browser, Operating System, Database, etc. These environments (or configurations) can be managed and assigned to Test Plans and Test Plan folders. Test Executions can then be created automatically for each environment/configuration.

Xray will provide a report to track the progress of Test Runs by environment in the context of a specific Test Plan.

The Xray Connector app for Bamboo will be updated in order to support new functionality and APIs already available in Xray. The Xray Connector will also support Xray cloud APIs and connectivity.

Ability to provide management for test types and thus provide additional, user-defined test types besides the standard ones (i.e. Manual, Cucumber, Generic).

Currently, it is possible to create new Test types using the "Test Type" custom field. However, any custom test type (except for some pre-determined names) will default to the Generic kind in Xray.

We plan to create a custom view where users can define new Test types and also the kind for each Test type. For instance, one can create a new Test type named "SpecFlow" and choose the "Gherkin" field. This means SpecFlow Tests would display a field with Gherkin syntax highlight.

Right now, a Test Run contains the start and finish date which are set automatically, although users can also change the start date manually. However, users can not stop and start the execution as needed and the total time spent on the Test Run (finish - start) might not reflect the time spent by the user executing the Test.

We plan to provide a feature to allow users to pause and continue with the execution directly on the Test Run screen.

Most Xray settings are now global settings. This means only Jira administrators can change these settings. Also, having only global settings means that any change will affect all projects in Jira.

By moving some of the settings to the project level, each project/team can have a different configuration of Xray.