Exporting a Test Execution

Xporter for Jira allows you to get the following data from the Xray Test Execution:

> ⊘   If a Test Execution contains a lot of information, it can decrease Jira performance

Overall Execution Status

Export the Overall Execution Status with name and percentage for each Test Execution Status

```
${Overall Execution Status}
```

You can print the status of the Test Execution by using the following notation:

% per Status	Number of Tests per Status
${Overall Execution Status.NameOfStatus}	${Overall Execution Status.NameOfStatus.Count}

See the real example:

Expand to see the examples on sample code of a Test Execution details

```
Todo: ${Overall Execution Status.TODO}% (${Overall Execution Status.TODO.Count})
Fail: ${Overall Execution Status.FAIL}% (${Overall Execution Status.FAIL.Count})
Pass: ${Overall Execution Status.PASS}% (${Overall Execution Status.PASS.Count})
Executing: ${Overall Execution Status.EXECUTING}% (${Overall Execution Status.EXECUTING.Count})
Aborted: ${Overall Execution Status.ABORTED}% (${Overall Execution Status.ABORTED.Count})
```

You can also use the translated name of Overall Execution Status to get the same information.

Language	Translated Name
ES	Estado de ejecución general
DE	Allgemeiner Ausführungsstatus
FR	État d'Exécution Global

See the example:

Expand to see the examples on sample code

```
Todo: ${Estado de ejecución general.TODO}% (${Estado de ejecución general.TODO.Count})
Fail: ${Estado de ejecución general.FAIL}% (${Estado de ejecución general.FAIL.Count})
Pass: ${Estado de ejecución general.PASS}% (${Estado de ejecución general.PASS.Count})
Executing: ${Estado de ejecución general.EXECUTING}% (${Estado de ejecución general.EXECUTING.Count})
Aborted: ${Estado de ejecución general.ABORTED}% (${Estado de ejecución general.ABORTED.Count})
```

Iterating Test Runs associated with a Test Execution

Xporter for Jira allows rendering of all the Test Tuns associated with a Test Execution.

Definition:

$ {TestRuns[n].Field}

n is the index of the Test Run, starting from 0. The field **TestRunsCount** or **TestrunsCount** was created in order to give the total number of Test Runs.

Since a Test Run isn't a Jira Issue, you can render only the following mappings.

Example:

Expand to see the example on sample code

```
#Iterating over Test Runs
#{for testruns}
        Execution Status: ${TestRuns[n].Execution Status}
        AssigneeID: ${AssigneeId}
        Rank: ${TestRuns[n].Rank}
        Executed By: ${TestRuns[n].Executed By}
        Started On: ${TestRuns[n].Started On}
        Finished On: ${TestRuns[n].Finished On}
        Comment: ${wiki:TestRuns[n].Comment}
        Execution Defects Count: ${TestRuns[n].ExecutionDefectsCount}
        TestSteps Defects Count: ${TestRuns[n].TestStepsDefectsCount}
        Evidences Count: ${TestRuns[n].ExecutionEvidencesCount}
#{end}
```

Exporting Test Runs custom fields

To export Test Runs Custom Fields you just have to defined the placeholder with the name of you custom field.

Example: Image that you have a custom field called "Run CF". To get the value printed on you document you just have to use the following placeholder:

```
#{for testruns}
The Run CF value is: ${TestRuns[n].Run CF}
#{end}
```

ⓘ If your custom field type is a Number, Data or Date Time you can use formatting functions.

Iterating Pre-Conditions associated with a Test Run

Xporter for Jira allows rendering of the Pre-Conditions associated with a Test from a Test Run.

Definition:

${TestRuns[n].PreCondition.Field} - **DEPRECATED**
${TestRuns[n].PreConditions[p].Field}

n is the index of Test Runs, starting from 0. The fields available for Pre-Conditions are:

- Key
- Summary
- Conditions
- Pre-Condition Type

Expand to see the example on sample code

```
#Iterating over Test Runs
#{for testruns}
    #Getting data from Pre-Conditions
    ${TestRuns[n].PreConditionsCount}
    #{for p=TestRuns[n].PreConditionsCount}
      Pre-Condition Key: ${TestRuns[n].PreConditions[p].Key}
      Pre-Condition Summary: ${TestRuns[n].PreConditions[p].Summary}
      Condition:  ${TestRuns[n].PreConditions[p].Conditions}
      Type: ${TestRuns[n].PreConditions[p].Pre-Condition Type}
    #{end}
#{end}
```

DEPRECATED - Expand to see the example on sample code

```
#Iterating over Test Runs
#{for testruns}
        #Getting data from Pre-Condition
        Pre-Condition Key: ${TestRuns[n].PreCondition.Key}
        Pre-Condition Summary: ${TestRuns[n].PreCondition.Summary}
        Condition:        ${TestRuns[n].PreCondition.Conditions}
        Type: ${TestRuns[n].PreCondition.Pre-Condition Type}
#{end}
```

Iterating Attachments associated with a Test Run

Xporter for Jira allows rendering of all the Attachments associated with a Test Run.

Definition:

$ {TestRuns[n].AttachmentsCount[sa]}

sa is the index of the Attachments, starting from 0. The field **AttachmentsCount** was created in order to give the total number of Attachments of a Test Run.

Since a Test Run isn't a Jira Issue, you can render only the following mappings.

Example:

Expand to see the example on sample code

```
#Iterating over Test Runs
#{for testruns}
        #Iterating Test Attachments
        #{for sa=TestRuns[n].AttachmentsCount}
                Name: ${TestRuns[n].Attachments[sa].Name}
                Author: ${TestRuns[n].Attachments[sa].Author}
                ID: ${TestRuns[n].Attachments[sa].ID}
                Size: ${TestRuns[n].Attachments[sa].Size}
        #{end}
#{end}
```

> ⊕ If a Test Execution contains a lot of information, it can decrease Jira performance

Iterating Evidences associated with a Test Run

Xporter for Jira allows rendering of all the Evidences associated with a Test Run.

Definition:

$ {TestRuns[n].ExecutionEvidences[d]}

d is the index of the Evidences, starting from 0. The field **ExecutionEvidencesCount** was created in order to give the total number of Evidences of a Test Run.

Since a Test Run isn't a Jira Issue, you can render only the following mappings.

Example:

Expand to see the example on sample code

```
#Iterating over Test Runs
#{for testruns}
        #Iterating over Evidences
        #{for d=TestRuns[n].ExecutionEvidencesCount}
                Id: ${TestRuns[n].ExecutionEvidences[d].Id}
                Name: ${TestRuns[n].ExecutionEvidences[d].Name}
                Author: ${TestRuns[n].ExecutionEvidences[d].Author}
                Link: @{title=${TestRuns[n].ExecutionEvidences[d].FileURL}|href=${TestRuns[n].ExecutionEvidences
[d].FileURL}}
                Size: ${TestRuns[n].ExecutionEvidences[d].Size}
                Created: ${TestRuns[n].ExecutionEvidences[d].Created}
                HumanReadableSize: ${TestRuns[n].ExecutionEvidences[d].HumanReadableSize}
                MimeType: ${TestRuns[n].ExecutionEvidences[d].MimeType}
                Evidence:${TestRuns[n].ExecutionEvidences[d].Evidence}
        #{end}
#{end}
```

Iterating Defects associated with a Test Run

Xporter for Jira allows rendering of all the defects associated with a Test Run.

Definition:

$ {TestRuns[n].ExecutionDefects[e]}

e is the index of the defects, starting from 0. The field **ExecutionDefectsCount** was created in order to give the total number of Defects of a Test Run.

Since a Test Run isn't a Jira Issue, you can render only the following mappings.

Example:

Expand to see the example on sample code

```
#Iterating over Test Runs
#{for testruns}
        #Iterating over defects from each test run
        #{for e=TestRuns[n].ExecutionDefectsCount}
                Link: @{title=${TestRuns[n].ExecutionDefects[e].Key}|href=${BaseURL}/browse/${TestRuns[n].
ExecutionDefects[e].Key}}
                Summary: ${TestRuns[n].ExecutionDefects[e].Summary}
        #{end}
#{end}
```

Iterating Automated Test details associated with a Test Run

Xporter for Jira allows rendering of the Details from Automated Tests associated with a Test Run.

Definition:

Cucumber Scenario: $ {TestRuns[n].Cucumber Scenario}

Test Definition: $ {TestRuns[n].Generic Test Definition}

n is the index of the Test Runs, starting from 0. The fields **Cucumber Scenario/Generic Test Definition** were created in order to give the step details of Automated Tests of a Test Run.

Since a Test Run isn't a Jira Issue, you can render only the following mappings.

Example:

Expand to see the example on sample code

```
#Iterating over Test Runs
#{for testruns}
        #Test Run Step Details
        Cucumber Scenario: ${TestRuns[n].Cucumber Scenario}
        Test Definition: ${TestRuns[n].Generic Test Definition}
#{end}
```

ⓘ   If a Test Execution contains a lot of information, it can decrease Jira performance

Iterating Manual Test Step details associated with a Test Run

Xporter for Jira allows rendering of the Details from Manual Tests associated with a Test Run.

Definition:

$ {TestRuns[n].TestSteps[r]}

r is the index of the Test Steps, starting from 0. The field **TestStepsCount** or **TeststepsCount** was created in order to give the step details of Manual Tests of a Test Run.

Since a Test Run isn't a Jira Issue, you can render only the following mappings.

Example:

Expand to see the example on sample code

```
#Iterating over Test Runs
#{for testruns}
        #Iterating over Manual Test Steps from Test Run
        #{for r=TestRuns[n].TestStepsCount}
                StepNumber:        ${TestRuns[n].TestSteps[r].StepNumber}
                Action: ${wiki:TestRuns[n].TestSteps[r].Action}
                Data: ${wiki:TestRuns[n].TestSteps[r].Data}
                Expected Result: ${wiki:TestRuns[n].TestSteps[r].ExpectedResult}
                Comment: ${wiki:TestRuns[n].TestSteps[r].Comment}
                Status:          ${TestRuns[n].TestSteps[r].Status}
                Actual Result: ${wiki:TestRuns[n].TestSteps[r].Actual Result}
        #{end}
#{end}
```

Iterating Manual Test Step Attachments associated with a Test Run

Xporter for Jira allows rendering of the Attachments from Manual Tests Steps associated with a Test Run.

Definition:

$ {TestRuns[n].TestSteps[r].Attachments[sa]}

sa is the index of the Test Step Attachments, starting from 0. The field **AttachmentsCount** was created in order to give the step attachments of Manual Tests of a Test Run.

Since a Test Run isn't a Jira Issue, you can render only the following mappings.

Example:

Expand to see the example on sample code

```
#Iterating over Test Runs
#{for testruns}
        #Iterating over Manual Test Steps from Test Run
        #{for r=TestRuns[n].TestStepsCount}
                #Iterating over Test Step Attachments
                #{for sa=TestRuns[n].TestSteps[r].AttachmentsCount}
                        Name: {TestRuns[n].TestSteps[r].Attachments[sa].Name}
                        Author: {TestRuns[n].TestSteps[r].Attachments[sa].Author}
                        Link: {title={TestRuns[n].TestSteps[r].Attachments[sa].FileURL}|href={TestRuns[n].
TestSteps[r].Attachments[sa].FileURL}}
                        Size: {TestRuns[n].TestSteps[r].Attachments[sa].Size}
                #{end}
        #{end}
#{end}
```

> ⊕   If a Test Execution contains a lot of information, it can decrease Jira performance

Iterating Manual Test Step Defects associated with a Test Run

Xporter for Jira allows rendering of the Defects from Manual Tests Steps associated with a Test Run.

Definition:

$ {TestRuns[n].TestSteps[r].Defects[dc]}

dc is the index of the Test Step Defects, starting from 0. The field **DefectsCount** was created in order to give the step defects of Manual Tests of a Test Run.

Since a Test Run isn't a Jira Issue, you can render only the following mappings.

Example:

Expand to see the example on sample code

```
#Iterating over Test Runs
#{for testruns}
        #Iterating over Manual Test Steps from Test Run
        #{for r=TestRuns[n].TestStepsCount}
                #Iterating over Test Step Defects
                #{for dc=TestRuns[n].TestSteps[r].DefectsCount}
                        Link: {title={TestRuns[n].TestSteps[r].Defects[dc].Key}|href={BaseURL}/browse/{TestRuns
[n].TestSteps[r].Defects[dc].Key}}
                        Summary: {wiki:TestRuns[n].TestSteps[r].Defects[dc].Summary}
                #{end}

        #{end}
#{end}
```

Iterating Manual Test Step Evidences associated with a Test Run

Xporter for Jira allows rendering of the Evidences from Manual Tests Steps associated with a Test Run.

Definition:

$ {TestRuns[n].TestSteps[r].Evidences[e]}

e is the index of the Test Step Evidences, starting from 0. The field **EvidencesCount** was created in order to give the step evidences of Manual Tests of a Test Run.

Since a Test Run isn't a Jira Issue, you can render only the following mappings.

Example:

Expand to see the example on sample code

```
#Iterating over Test Runs
#{for testruns}
        #Iterating over Manual Test Steps from Test Run
        #{for r=TestRuns[n].TestStepsCount}
                #Iterating over Test Step Evidences
                #{for e=TestRuns[n].TestSteps[r].EvidencesCount}
                        Name: ${TestRuns[n].TestSteps[r].Evidences[e].Name}
                        Author: ${TestRuns[n].TestSteps[r].Evidences[e].Author}
                        Link: @{title=${TestRuns[n].TestSteps[r].Evidences[e].FileURL}|href=${TestRuns[n].
TestSteps[r].Evidences[e].FileURL}}
                        Size: ${TestRuns[n].TestSteps[r].Evidences[e].Size}
                Created: ${TestRuns[n].TestSteps[r].Evidences[e].Created}
                HumanReadableSize: ${TestRuns[n].TestSteps[r].Evidences[e].HumanReadableSize}
                MimeType: ${TestRuns[n].TestSteps[r].Evidences[e].MimeType}
                Evidence:${TestRuns[n].TestSteps[r].Evidences[e].Evidence}
                        #{end}

        #{end}
#{end}
```

ⓘ   If you want to export the images, for example ${TestRuns[n].ExecutionEvidences[d].FileURL} you can check here for instructions on how to do it.

Manual Test Step Custom Fields associated with a Test Run

To export Test Steps Custom Fields within a Test Run you just have to define the placeholder with the name of your custom field.

Example: Image that you have a custom field called "Run CF". To get the value printed on your document you just have to use the following placeholder:

```
#Iterating over Test Runs
#{for testruns}
        #Iterating over Manual Test Steps from Test Run
        #{for r=TestRuns[n].TestStepsCount}
                #Add Custom Field name, for example Run CF
                Run CF: ${TestRuns[n].TestSteps[r].Run CF}
        #{end}
#{end}
```

ⓘ If your custom field type is a Number, Data, or Date Time you can use formatting functions.

Exporting Test Run Activity

Xporter for Jira allows export all the activity of a Test Run.

Definition:

$ {TestRuns[n].ActivityEntries[ac]}

ac is the index of the Activity entry, starting from 0. The field **ActivityEntriesCount** was created in order to give the Activity entry of a Test Run.

Example:

Expand to see the example on sample code

```
#{for testruns}
#{for d=TestRuns[n].ActivityEntriesCount}
Action: ${TestRuns[n].ActivityEntries[d].Action}
Author: ${TestRuns[n].ActivityEntries[d].Author}
Created at: ${dateformat("dd-MM-yyyy HH:mm:ss"):TestRuns[n].ActivityEntries[d].Created}
Changes:
#{for ch=TestRuns[n].ActivityEntries[d].ActivityItemsCount}
Field: ${TestRuns[n].ActivityEntries[d].ActivityItems[ch].Field}
OldValue: ${TestRuns[n].ActivityEntries[d].ChangedItems[ch].OldValue}
NewValue: ${TestRuns[n].ActivityEntries[d].ChangedItems[ch].NewValue}

#{end}
#{end}
#{end}
```

Exporting Parameterized Tests from a Test Run

To export all this data, we first need to execute the test run. If we change any data, we need to return to the Execution Page and merge it with the new data.

Fields	Description
IsDataDriven	Returns "true" if the current test run is data-driven; otherwise, it returns "false"
Iterations Overall Execution Status.STATUS.Percentage	The percentage of STATUS in the test run iterations
IterationsOverallExecutionStatus	List of all the statuses in the current test run iteration and their percentages
IterationsOverallExecutionStatus.STATUS.Count	The number of STATUS in the test run iterations
Parameters	List of the parameters from each test run iteration
ParametersCount	Total of Parameters declared in Dataset

ⓘ

> ⓘ **Data-driven**
>
> The field IsDataDriven is going to return TRUE only if the test run has more than one iteration.

Exporting Test Runs Parameters from a Test Execution

For each Test Run Parameter you can export the following fields:

- Key
- Value

Below you can find an example of how to iterate over the list of **Test Run Parameters** associated with a **Test Execution.**

Some mappings we can export from Test Runs Parameters

```
// Iterating each test run
#{for testruns}
    // Iterating over parameters for each test run
    Parameters Total: ${TestRuns[n].ParametersCount}
    #{for m=TestRuns[n].ParametersCount}
        Key: ${TestRuns[n].Parameters[m].Key}
        Value: ${TestRuns[n].Parameters[m].Value}
    #{end}
#{end}
```

Exporting Test Runs Iterations from a Test Execution

For each Test Run Iteration you can export the following fields:

- Overall Execution Status
- Data from Test Run Iterations
- Parameters from Test Run Iterations
- PreConditions from Test Run Iterations
- Test steps from Test Run Iterations

Below you can find an example of how to iterate over the list of **Test Run Iterations** associated with a **Test Execution**.

Some mappings we can export from Test Run Iterations

```
// Iterating each test run
#{for testruns}
    IsDataDriven: ${TestRuns[n].IsDataDriven}

    // Iterations Overall Execution Status (percentage + total of testes per status)
    List of Statuses: ${TestRuns[n].Iterations Overall Execution Status}
    TO DO: ${TestRuns[n].Iterations Overall Execution Status.TODO}% - ${TestRuns[n].Iterations Overall
Execution Status.TODO.Count}
    EXECUTING: ${TestRuns[n].Iterations Overall Execution Status.EXECUTING}% - ${TestRuns[n].Iterations Overall
Execution Status.EXECUTING.Count}
    PASSED: ${TestRuns[n].Iterations Overall Execution Status.PASSED}% - ${TestRuns[n].Iterations Overall
Execution Status.PASSED.Count}
    FAILED: ${TestRuns[n].Iterations Overall Execution Status.FAILED}% - ${TestRuns[n].Iterations Overall
Execution Status.FAILED.Count}
    ABORTED: ${TestRuns[n].Iterations Overall Execution Status.ABORTED}% - ${TestRuns[n].Iterations Overall
Execution Status.ABORTED.Count}

    // Iterating over test runs iterations
    Total of Iterations from a Test Run: ${TestRuns[n].IterationsCount}
    #{for m=TestRuns[n].IterationsCount}
        Name: ${TestRuns[n].Iterations[m].Name}
        Status: ${TestRuns[n].Iterations[m].Status}
        Parameters: ${TestRuns[n].Iterations[m].Parameters}

        // Iterating over parameters for each test run iteration
        Parameters Total: ${TestRuns[n].Iterations[m].ParametersCount}
        #{for l=TestRuns[n].Iterations[m].ParametersCount}
            Key: ${TestRuns[n].Iterations[m].Parameters[l].Key}
            Value: ${TestRuns[n].Iterations[m].Parameters[l].Value}
```

```
        #{end}

        // Iterating over preconditions for each test run iteration
        Preconditions Total: ${TestRuns[n].Iterations[m].PreConditionsCount}
        #{for l=TestRuns[n].Iterations[m].PreConditionsCount}
                    Key: ${TestRuns[n].Iterations[m].PreConditions[l].Key}
                    Summary: ${TestRuns[n].Iterations[m].PreConditions[l].Summary}
            Definition: ${TestRuns[n].Iterations[m].PreConditions[l].Conditions}
            Type: ${TestRuns[n].Iterations[m].PreConditions[l].Pre-Condition Type}
        #{end}

        //Iterating over test steps for each test run iteration
        #{for i=TestRuns[n].Iterations[m].TestStepsCount}
            Step Number: ${TestRuns[n].Iterations[m].TestSteps[i].StepNumber}
            Action: ${TestRuns[n].Iterations[m].TestSteps[i].Action}
            Data: ${TestRuns[n].Iterations[m].TestSteps[i].Data}
            Expected Result: ${TestRuns[n].Iterations[m].TestSteps[i].ExpectedResult}
            Status: ${TestRuns[n].Iterations[m].TestSteps[i].Status}
            Comment: ${TestRuns[n].Iterations[m].TestSteps[i].Comment}
            Actual Result: ${TestRuns[n].Iterations[m].TestSteps[i].ActualResult}
            // Replace the placeholder text to export any custom field associated with the test step.
            Step Custom Field:${TestRuns[n].Iterations[m].TestSteps[i].<Step Custom Field>}

            // Iteration Test Step Attachments
            #{for l=TestRuns[n].Iterations[m].TestSteps[i].AttachmentsCount}
                Id: ${TestRuns[n].Iterations[m].TestSteps[i].Attachments[l].Id}
                Name: ${TestRuns[n].Iterations[m].TestSteps[i].Attachments[l].Name}
                Image: ${TestRuns[n].Iterations[m].TestSteps[i].Attachments[l].Attachment}
                FileURL: ${TestRuns[n].Iterations[m].TestSteps[i].Attachments[l].FileURL}
            #{end}
            // Iteration Test Step Evidences
            #{for l=TestRuns[n].Iterations[m].TestSteps[i].EvidencesCount}
                Id: ${TestRuns[n].Iterations[m].TestSteps[i].Evidences[l].Id}
                Name: ${TestRuns[n].Iterations[m].TestSteps[i].Evidences[l].Name}
                Evidence: ${TestRuns[n].Iterations[m].TestSteps[i].Evidences[l].Evidence}
            #{end}
            // Iteration Test Step Defects
            #{for l=TestRuns[n].Iterations[m].TestSteps[i].DefectsCount}
                Description: ${TestRuns[n].Iterations[m].TestSteps[i].Defects[l].Description}
                Id: ${TestRuns[n].Iterations[m].TestSteps[i].Defects[l].Id}
                Key: ${TestRuns[n].Iterations[m].TestSteps[i].Defects[l].Key}
                Summary: ${TestRuns[n].Iterations[m].TestSteps[i].Defects[l].Summary}
            #{end}
        #{end}
    #{end}
#{end}
```