Xporter Template: Defect Summary Report

Purpose

This report lists some details of the selected Defects in Xray, enabling them to be extracted in an Excel format. With this ability to extract the report you can use it for analysis of trends and current testing status, or process this information to generate some metrics, for example, or even share it with someone else who still needs access to Jira.

Possible usage scenarios:

- see all the Defect statuses & count for a given version
- see the linked Requirements, Tests, and Defects
- track the Defects that are taking the most time to get resolved

Output Example(s)

The template contains 4 tabs, the following tables show an example of the columns/rows you should expect.

Impacted Requirements (Story)				
Story Key	**Summary**	**Priority**	**Component**	**Total linked defects**
FIN-87	LinkedTestForStory	Medium	TestC2	3
FIN-14	Application Security and Access Control	Medium		1

Defect Snapshot				
Total defects reported	**Total defects resolved**	**Total defects unresolved**	**Total found outside of testing**	**Total found in testing**
25	3	22	4	21

Key	Summary	Workflow Status	Affected Version	Fix version	Priority	Component	Environment	Reporter	Assignee	Linked	Linked Test Ke	Linked Re	Linked	Source	Date Created	Time to fix
FIN-89	CreateIssue-Bug	In Progress	v1, v2	v1	Medium			Ivan Filippov	Ivan Filippov			FIN-87 - LinkedTestForStory FIN-14 - Application	FIN-57 - ET MoT Demo	found outside testing	21-12-2023 20:08:56	
FIN-84	Banner problem	To Do			Medium			Ivan Filippov		FIN-54 FIN-56	FIN-85 - FAILED			found in testing	13-12-2023 15:20:46	
FIN-81	Page 2 - Banner	To Do			Medium			Ivan Filippov		FIN-54	FIN-82 - FAILED			found in testing	10-12-2023 20:51:10	
FIN-74	"No" button is u	Done	v2	v1, v2	Medium	TestC2	chrome ff	Ivan Filippov	Ivan Filippov	FIN-54	FIN-82 - FAILED FIN-83 - FAILED FIN-85 - FAILED FIN-86 - PASSED			found in testing	20-11-2023 21:16:48	21:09:52

Key	Issue Type	Summary	Priority	Component(s)	Status	Assignee
FIN-89	Bug	CreateIssue-Bug-fromTimelineItem	Medium		In Progress	Ivan Filippov
FIN-84	Bug	Banner problem	Medium		To Do	
FIN-81	Bug	Page 2 - Banner - "Yes" is not responding	Medium		To Do	
FIN-74	Bug	"No" button is unresponsive and should be red	Medium	TestC2	Done	Ivan Filippov
FIN-73	Bug	Logo not visible at 175% zoom	Medium	TestC, TestC2	To Do	
FIN-72	Bug	Page 2 demo defect 1	Medium		To Do	

How to use

This report can be generated from different places/contexts, including:

- Issue view screen
- Issue search page (main search page or as a bulk operation)

> **Learn more**
>
> General information about all the existing places available to export from and how to perform it is available on the Exporting page.

Source data

In the store, you can find 2 subtypes for this template:

- "Bugsormix" - 1 or more Bug issue as well as a mix of Bug and Test Execution issues
- "Testexecs" - 1 or more Test Execution issues

Output format

The standard output format is .XLSX so you can open it in Microsoft Excel, Google Sheets, and other tools compatible with this format. From those tools, you can generate a .CSV file.

Report assumptions

The template has a set of assumptions that you have to make sure your Jira/Xray environment complies with:

1. Issue types having the name: "Test Execution", "Bug"
2. Bug final status being "Done" or "Resolved"
3. Finished Bugs having the "Resolved date" populated

If any of these assumptions is not met, you need to update the template or the environment accordingly.

Usage examples

Export Test Executions of your project

1. from the Issue Navigator/Search, search by the issueType (i.e., "Test Execution") from your project (e.g., "BOOK") and then use bulk export or Export->Xporter

example of JQL expression to use

```
project = "BOOK" and issuetype="Test Execution" order BY created DESC
```

Export Bugs associated with a given fix version

1. from the Issue Navigator/Search, search by the release (i.e., "fixVersion") of your project (e.g., "EWB") and then use bulk export or Export->Xporter

example of JQL expression to use

```
project = "BOOK" and issuetype= "Bug" and fixVersion=1.2 order BY created DESC
```

Export Test Executions and Bugs obtained in a given Environment

1. from the Issue Navigator/Search, search by Test Executions or Bugs assigned to that Test Environment (e.g., "chrome") and then use bulk export or Export->Xporter

example of JQL expression to use
`project = "BOOK" and issuetype in ("Test Execution", Bug) and environment IS chrome order BY created DESC`

Export Bug or Test Execution from its detailed view

1. open the Bug/Test Execution issue and export it using this template

Understanding the report

The report shows information about the Defects in a list form as well as provides summary tables.

Layout

The report is composed of 4 sheets - "ReqsImpactSummary", "DefectSummary", "DetailedDefectList", and "Input". By default, all tabs are rendered. The layout is the same regardless of the template subtype used.

"ReqsImpactSummary" sheet

This sheet presents a table listing requirements ("Story" issue type) impacted by defects from your exported issues.

Column	Notes
Story Key	Issue key of the Story
Summary	Summary of the Story
Priority	Priority of the Story
Component	Component of the Story
Total linked defects	Count of unique defect issues linked to a given requirement story

"DefectSummary" sheet

Column	Notes
Total defects reported	Count of unique defects associated with all the items from the source JQL. Should match the number of content rows on the "DetailedDefectList" sheet.
Total defects resolved	Count of unique defects in "Done" or "Resolved" status
Total defects unresolved	Difference between the first 2 columns
Total found outside of testing	Count of defects from "DetailedDefectList" sheet that have "found outside of testing" value in the Source column (see the description of logic below).
Total found in testing	Difference between "Total defects reported" and "Total found outside of testing"

"DetailedDefectList" sheet

Column	Notes
Key	Issue key of the defect

Summary	Summary of the defect
Workflow Status	Workflow status of the defect
Affected Version	Affected Version(s) of the defect
Fix version	Fix Version(s) of the defect
Priority	Priority of the defect
Component	Component(s) of the defect
Environment	Environment of the defect
Reporter	Reporter of the defect
Assignee	Assignee of the defect
Linked Test Execution	Key for the Test Execution(s) linked to the defect (regardless of the link type)
Linked Test Key and Associated Latest Test Run Status	Key for the Test(s) linked to the defect (regardless of the link type) and the latest execution run status for the Test (s)
Linked Requirement Story Key and Summary	Key and Summary for the Story(ies) linked to the defect (regardless of the link type)
Linked Requirement Epic Key and Summary	Key and Summary for the Epic(s) linked to the defect (regardless of the link type)
Source	The value is "found in testing" if there is at least 1 linked Test, Test Execution, or Test Plan. The value is "found outside of testing" otherwise (e.g. if a bug is created directly from the story).
Date Created	"Created" date in the "dd-MM-yyyy HH:mm:ss" format
Time to fix	Difference between "Resolved" and "Created" dates

"Input" sheet

Column	Notes
Key	Issue key of the item from the source JQL
Issue Type	Issue Type of the item from the source JQL
Summary	Summary of the item from the source JQL
Priority	Priority of the item from the source JQL
Component	Component of the item from the source JQL
Status	Workflow Status of the item from the source JQL
Assignee	Assignee of the item from the source JQL

Customizing the report

The common customization actions are:

- adding/removing columns

- changing the level of detail in the columns

As this report is column-based, if some columns are not relevant to you, you should be able to delete them. Make sure that no temporary variables are created in the cells of those columns that are used in other subsequent columns.

Keep in mind that only 1 hyperlink will be active per Excel cell (the one associated with the first line item).

You can further finetune the content and formatting via JavaScript, you can find more useful snippets in this tutorial for Xporter and DocGen as well as Xporter documentation (Cloud, DC).

Exercise 1: extend the report to custom issue types

In the "bugsormix" subtype, find all instances of

```
#{if (%{'${IssueTypeName}'.equals('Bug')})}
```

and edit them to include more than 1 type with the OR (||) syntax like this

```
#{if (%{'${IssueTypeName}'.equals('Bug') || '${IssueTypeName}'.equals('Defect') || '${IssueTypeName}'.equals('Incident')})}
```

Similar changes would allow you to account for more statuses.

Performance

Performance can be impacted by the information that is rendered and by how that information is collected/processed (especially in the "bugsormix" subtype).

The number of Defects and associated items, depending on scenarios, can be considerably high, especially with CI/CD. As this report sums up quite a lot of information, please use it wisely.

Some tips

- limit the number of input issues; in Xporter there's a global setting for this purpose