Bug writing guidelines

Contents

[Creating bugs 2](#_Toc466640232)

[Default questions 2](#_Toc466640233)

[Additional questions 3](#_Toc466640234)

[Modifying bugs 3](#_Toc466640235)

[Filling in the fields of a TFS bug 4](#_Toc466640236)

[Title 4](#_Toc466640237)

[Area path 5](#_Toc466640238)

[Iteration path 5](#_Toc466640239)

[Assigned to 5](#_Toc466640240)

[Priority 5](#_Toc466640241)

[Severity 6](#_Toc466640242)

[RightNow Task ID 6](#_Toc466640243)

[Description 6](#_Toc466640244)

[Steps to reproduce 7](#_Toc466640245)

[System info tab 7](#_Toc466640246)

[Found in build 7](#_Toc466640247)

[System info 7](#_Toc466640248)

[Attachments 8](#_Toc466640249)

[Logfiles 8](#_Toc466640250)

[Screenshot and video 8](#_Toc466640251)

[Configuration file 8](#_Toc466640252)

[Sample files 9](#_Toc466640253)

[Field templates 9](#_Toc466640254)

# Creating bugs

Let’s say you are facing a problem that you would like to report to development, so the issue can be fixed. You observed the symptoms, and are ready to write down what you know. This guide’s purpose it to help and standardize the description provided in TFS/Jira bugs.

You can find templates for the **System info** and **Description** fields in the [Field templates](#_Field_templates) chapter which are to provide a crutch to fill in these fields properly.

## Default questions

Before and during creating a bug, ask yourself the following questions always.

|  |  |
| --- | --- |
| **Question** | **Considerations** |
| 1. Is the bug already reported? | Write down one or two keywords about your problem.  Search in TFS/Jira for these keywords.  If you find the same problem, check the guidelines for modifying bugs in the *Modifying bugs* section.  If you find a similar problem (that may have the same root cause as your problem), make sure you link this similar problem to your bug (using the Related link type). |
| 1. Do you think the issue has multiple root causes or multiple modules where development needs to fix? | Make sure you break down the issue to the lowest level possible. If you think an issue has to be fixed in 3 different modules, submit 3 different bugs describing the parts of the symptoms.  Vice-versa, if there is only one single root cause, then write up only 1 bug. |
| 1. Is the problem client-dependent? | Physical device:  Check with another physical device, different family from the same vendor. If the issue is reproducible on the other device, then let’s move on to check on a 3rd device but from a different device family.  If the issue is still reproducible on the 3rd device, then it’s very likely not a device dependent issue.  If your conclusion is that the issue is NOT device specific, then the Title should NOT contain the model.  Software client:  If you suspect that the issue affects other client’s behavior, then check it where you think it makes sense. For example, a form issue that appears in Module A, may be present in Module B too (A, B has familiar common attribute such as same type input, same design at some path) |
| 1. Would a screenshot help? | If so, create and attach one.  If screenshots are not enough, consider attaching a video. When doing so, please keep the size to a possible minimum. |
| 1. Is there a reason why NOT to attach logs? | Always attach a trace log, except when...   * you have an evident problem (easily reproducible in any environments), * you have a UI problem (screenshot needed instead of logs), * a developer confirmed that no logs are needed, * the issue is documentation related, * there is good reason why not to include logs – make sure to include your reason in the **Description**.   Bottom line: if there are no logs attached, you should include a reason for its absence in the **Description** field. |
| 1. Is there a reason why NOT to attach configuration file? | Always attach configuration file, except when…   * the issue describes a UI problem which can be reproduced easily without a device or backend configured, * a developer confirmed that no config file is needed, * the issue is documentation related, * the issue is installation related, * there is good reason why not to include config file – make sure to include your reason in the **Description**.   Bottom line: if there is no config file attached, you should include a reason for its absence in the **Description** field. |
| 1. Are all details included in the description? | Put yourself out of context and imagine the situation where you don’t know the details of the issue and you have to get those from the bug. Do you get the minimal set of information needed to understand the bug? |
| 1. Is the bug free of typos, and does it make sense? | Re-read what you wrote to ensure that content is free of typing errors (typos), and that it can be understood and reproduced by anyone else. |

## Additional questions

The following items are optional as take significantly more time to answer, but definitely help creating better quality bugs. More important the answers to these questions can influence bug severity.

|  |  |
| --- | --- |
| 1. Is there a workaround? | If so, please state this clearly in the **Description** filed, and describe the steps needed to perform the workaround.  Not all bugs have workarounds, but for those that do, please make sure the workaround is included. |
| 1. Is it a regression issue? | If so, please state this clearly in the **Description** filed. A bug is a regression issue if it was not present in a previously tested and released build, but it is present in the current one. |
| 1. Is the bug language-dependent? | If you found the bug on an OS or client with a language different from English, check it with an English OS and client. If the bug is not present there, check it with other languages to see whether it is reproducible. |

# Modifying bugs

Any fields of a bug can be modified later if needed (Title, Description, etc.).

* For any non-trivial changes you do to a bug, please include a comment in the history on why you made the change.
* Do not include comments in bugs that are not closely related to the given bug. If your comment is not closely related, consider whether you need to add a new bug.
* Please never reactivate a bug that was fixed and closed in a past project. Instead, create a copy of the recurring bug, and include a comment in the symptom regarding the recurring nature of the issue. When you use copy function of TFS/Jira then the 2 bugs will be automatically linked together. In case you manually record the recurring bug, make sure it’s linked to the closed one as *Related*.

# Filling in the fields of a TFS/Jira bug

The following part gives a guidance on how to fill in all the fields where the one recording the TFS/Jira bug has to modify fields according to the QA procedure.

The following list gives a summary of the required fields which are to be modified while recording a new bug:

* [Title](#_Title)
* [Area path](#_Description)
* [Iteration path](#_Iteration_path_1)
* [Assigned to](#_Assigned_to)
* [Priority](#_Priority)
* [Severity](#_Severity)
* [RightNow Task ID](#_RightNow_Task_ID)
* [Description](#_Description_2)
  + [Steps to reproduce](#_Steps_to_reproduce)
* System info tab
  + [Found in Build](#_Found_in_build)
  + [System info](#_System_info)
* [Attachments](#_Attachments)
  + [Logfile](#_Logfile)s
  + [Screenshots and video](#_Screenshot_and_video)
  + [Configuration file](#_Configuration_file)
  + [Sample files](#_Sample_files)

## Title

*Title* is for describing the symptoms in the most specific, but the shortest way possible. Make sure that by reading the Title anyone can have a general understanding of what the bug is about.

Tips for creating the **Title**:

1. Make a list of all the components that played a part in the problem.
2. Include everything important, but nothing that is not needed. For each component, ask whether omitting that would make the title less precise - if not, drop it.
3. Ask yourself: if I met this problem without having any additional info or details, what are the words I would search for? Make a list of the words that you would try to look for when seeing the symptom.
4. Create a final word list from the above.
5. Phrase a sentence that includes all these words, and answers the following question: What’s the problem? This is going to be your final title.

The template of the **Title** field is: <”Escalation: “><”Area name – “><Bug title>

* Escalation: this is optional and needed only when recording a bug to track a customer escalation
* Area name: this is an optional field and needed only when QA engineer wants to emphasize the area/component affected by the bug. It’s a good practice to emphasize the affected area/component in the **Title** field, as it eases reading when bugs are listed and not viewed one by one.
* Bug title: mandatory part, which describes the issue

For example

* Component A help – Not supported Field value is show as an example
* Escalation: Component B – Number field boundaries are not taken into consideration

## Area path

This value reflects the area that the issue affects. QA should select an area according to the assumed place for the bug’s root cause. Please browse the tree and select the most appropriate option.

For example:

* Any bugs related to the components should go to the appropriate sub node.
* Process Designer related bugs should go.
* All the general documentation related bugs should go if there’s a bug about a component’s description in help then the area path should be set according to the component and not to the general documentation path.
* Installer related issues should go.

## Iteration path

Value of **Iteration path** reflects the iteration where the bug was found. This reflects the release cycle (e.g. AS7 SP2 or AS6 SP1) and NOT the Sprint (build) where the bug was found. (The **Found in Build** field tracks this information anyway)

QA should always set it to the current iteration path that is communicated throughout the team.

For example, for *AS7 SP3* all new bugs should be recoded under *Work Management\Releases\AS 7.0\AS 7.0 SP3*.

Note: when a bug is deferred, the Iteration path can change to a later development cycle but this is managed by the team leads.

## Assigned to

While you’re working on the bug, keep it assigned to yourself. The **Assigned to** field has to be set to the QA lead when you think you are done with recording the bug.

This way it’s easy to distinguish between bugs that have to be reviewed and ones which are still being worked on.

## Priority

This field is for development to manage the order in which they fix the bugs. QA should always leave the default value.

## Severity

Determining Severity is never an easy thing to do. There are no absolute rules, you have to assess this for every bug by how big the impact for the bug would be for the client using our product. You have to consider how much trouble the bug would cause if it were found in the field by one of our clients. If needed, please include a description for the chosen Severity value in the **Description** field.

To give guidelines on determining a bug’s severity follow the below guidelines:

* **S1 - Critical**: The defect affects critical functionality or critical data. It does not have a workaround. Example: Unsuccessful installation, complete failure of a feature.
* **S2 - High**: The defect affects major functionality or major data. It has a workaround but is not obvious and is difficult. Example: A feature is not functional from one module but the task is doable if 10 complicated indirect steps are followed in another module.
* **S3 - Medium**: The defect affects minor functionality or non-critical data. It has an easy workaround (even including a system crash which can be avoided with a few easy steps). Example: A minor feature that is not functional in one module but the same task is easily doable from another module.
* **S4 - Low**: The defect does not affect functionality or data. It does not even need a workaround. It does not impact productivity or efficiency. It is merely an inconvenience. Example: Petty layout discrepancies, spelling/grammatical errors.

## RightNow Task ID

The **RightNow Task ID** field is used only in case when the bug being recorded is to track a customer escalation coming for the *RightNow* system. This is a numeric ID which should be identical to the task ID in the *RightNow* system.

## Description

The aim of the **Description** field is to describe the symptoms in details.

* Include every little piece of detail that might be important.
* The *Title* field's length and simplicity constraints do not apply here, so be verbose.
* If your title is very good, it might be enough to copy the title to the **Description** field.
* Make sure you describe not only the current situation, but the *desired situation* as well (in other words, please always provide description for ***Actual Results*** and ***Expected Results***).
* You can include the reasoning behind your chosen Severity value.
* If you have any other suspicions about the bug, but do not have time to perform all needed tests, please share your thoughts. You may give development good ideas where to start.
* If you think the bug can affect other areas of the software (e.g. a textbox issue could affect all textboxes), either prove your ideas by testing, and include your results, or include a comment that there may be related areas, but you have not tested them.
* If you work with different backend versions, include the version used in the test.
* If you can or cannot identify workaround process, write it here. Avoid using the “Workaround is missing” phrase, use “Workaround not identified” instead.

### Steps to reproduce

Steps to reproduce part of the **Description** is for providing precise guidance for reproducing the problem. Include any steps that are not evident, but do not include redundant instructions.

* Do not include OS installation, or build installation, if there is a general consensus on how to do these, and the versions are stored elsewhere in the bug, BUT
* include special steps, for example, if different service packs had to be installed on each other to reproduce the issue (e.g. SP2 had to be installed on SP1 instead of RTM).
* It is a very good practice to attach the configuration file to the bug.
* Please do not include any comment on a bug being device-specific in the Steps to reproduce field.

When you finished creating the steps, check them one-by-one, and ask the following questions:

|  |  |
| --- | --- |
| **Question** | **Consideration** |
| 1. After this step, can I perform the next one without doing anything else? | If not, you missed something between these steps. Include another step so your workflow is clear. |
| 2. Is this step needed? | Try to connect the step before this one with the step after this one. If you left this step out, could you perform the workflow with 100% confidence? If so, drop this step. |

## System info tab

### Found in build

This field is to show, in which build the bug was found in. The goal is to make it clearly identifiable which build is the bug reported for. This is most of the time the latest build being tested by QA.

Good examples:

* 20161102\_7.0.3.0\_SP3\_Sprint1
* AS7 SP3 Sprint 1

Bad examples:

* AS7 SP2
* 7.0.2.0

### System info

Environment description in **System info** field is to provide information on the environmental needs of the issue. This section gives information on all standard and non-standard parameters of the environment where the issue is present.

* **Device vendor(s) model(s)** and the device SDK version/firmware where the issue is reproducible. Mention always all tested devices together with the information whether the issue is reproducible on the given device or not. This can help isolating the bug’s location in the code.
* **Backend product name and version** together with the backend computer domain name. For example *iManage WorkSite Server 9.3 (QA1WS93R2)*. List all the backend servers you tried to reproduce the issue with. It’s again to give guidance on the issue’s location in the code.
* **Additional non-build installed DLLs, EXEs, etc.** This is the part where you list all the binaries used which were not coming from the standard build procedure. The list has to contain the binaries’ file names and their corresponding version information. If you used these kinds of binaries to reproduce the issue, then put a remark in the **Description** filed too to notify the developer that non-standard build was used.

## Attachments

There are additional possibilities to extend the information included in the bug with attachments. Sometimes a good screenshot or even a video recording can spare a huge amount of text.

When attaching files, consider that there’s a file size limit set on the TFS/Jira server for attachments. That is 4MB per attachment, so you can attach multiple files to the same bug which altogether may exceed the 4MB limit.

The item described in the following chapter should be attached to the bug.

### Logfiles

By attaching logfile to the bug, you can spare some time to the developer as the bug might not need to be reproduced but can be fixed based on the information uncovered from the logfile.

When attaching logs to the bug always make sure those contain only the logs for the steps related to bug reproduction. E.g. do not include all logs from the past 2 days because the issue was reproduced there. Since you have to verify the Steps to Reproduce procedure anyway it’s a good practice to clean up the log folder before and save the logs after the steps in Steps to Reproduce have been executed.

### Screenshot and video

As a general rule – as for the logs too –, record only what’s relevant to the bug. There are a bunch of good image and video manipulation tools which can help you editing the image attachments. (for example Paint .NET, IrfanView for images and Movie Maker for videos).

There are 2 major factors when it comes to images and videos, quality and size. You have to find the balance between the 2 since good quality images cause big size, but small size images may cause low quality on the output. It’s a good practice to create the original content with a high resolution and low compression and then editing the image and only play around with the compression at the end. Always try to get to the smallest possible file size where the content is still OK for everyone else too (e.g. do not chose high compression levels where the image content is no longer visible/readable)

Always check the output file if all is conveyed to the audiences what you wanted to show with it!

### Configuration file

It’s required to add the configuration file to the bug when a specific configuration is needed to reproduce the issue.

Always consider the config file as a required attachment and look for an exception why not to attach it.

### Sample files

It’s a good practice to attach sample files in case when the bug is related to the document/image file used for testing and reproducing the bug. The size consideration described above in the *Screenshot and Video* chapter applies to these items too.

When attaching modified files to the bug always make sure they are usable to reproduce the bug.

# Field templates

Template for **Description** field:

<Bug’s detailed description>

**Steps to reproduce**

1. <step 1>
2. <step 2>
3. …

**Actual result:** <description of actual situation caused by the above steps>

**Expected outcome:** <description of expected situation caused by the above steps>

**Workaround:** <description of the workaround identified for the issue described above>

Template for **System info** field (remove the ones that are not applicable):

**OS: <**OS name, update level, type and language>

**Device:** <Device vendor(s) model(s) and the device SDK version/firmware>

**Backend:** <Backend product name, version and backend computer name>

**Non-build binaries:** <Additional non-build installed binary files>