

0 - Introduction	2
1 - Preparations	7
2 - Setup GSE Actions	10
3.1 - Usage - Create Issue	17
3.2 - Usage - Add Comment	23
3.3 - Usage - Check Status	29
3.4 - Usage - Trouble Shooting	38
A.1 - Example - Create Issue	41
A.2 - Example - Add Comment	44
A.3 - Example - Check Status	47
A.4 - Example - Check Status and Add Comment	50
A.5 - Example - Check and Announce Status	54
B.1 - Access modify the code behind	59
B.2 - Version History	68

Jira Service Integration

INTRODUCTION

0 - Introduction

PREPARATIONS

1 - Preparations

SETUP

2 - Setup GSE Actions

USAGE

3.1 - Create Issue

3.2 - Add Comment

3.3 - Check Status

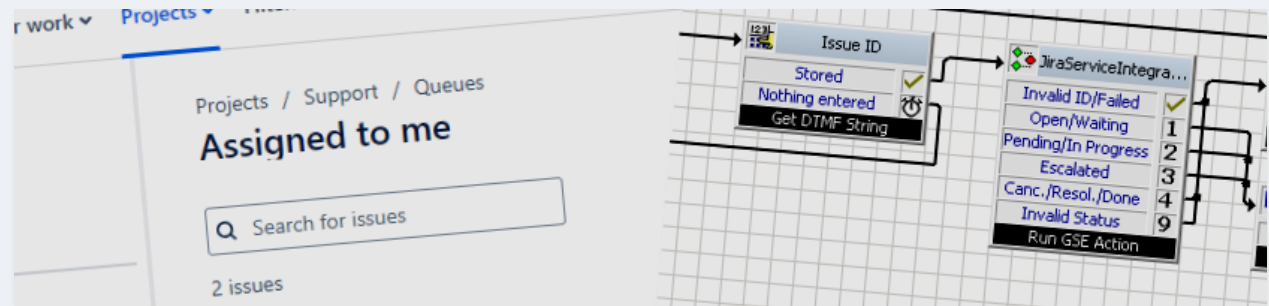
3.4 - Trouble Shooting

APPENDIX A

A.1 - Example: Create Issue

A.2 - Example: Add Comment

A.3 - Example: Check Status



Jira Service Integration - 0 - Introduction

Followers

0

VBScript

Lua

This extension provides an integration of the **Jira Service Management** into the SwyxWare call routing. It

- integrates into the Jira Service Management
- comes for **VBscript based** and **Lua based** call routing
- requires **SwyxWare 12.40** (or newer) for **VBScript based** call routing
- requires **SwyxWare 13.10** (or newer) for **Lua based** call routing

This integration provides the following functionality as GSE Actions within the SwyxWare call routing:

A.4 - Example: Check Status and Add Comment

A.5 - Example: Check and Announce Status

APPENDIX B

B.1 - Access/modify the code behind

B.2 - Version History

- **Create Issue** (create a new issue into an existing service project) - for example to register a callback request
- **Add Comment** (create a new comment (internal or public) to an existing service issue)
- **Check Status** (check the current status of an existing service issue)

The integration makes use of the [Jira REST API](#).



Please refer to the [Forums](#) to discuss the Jira Service Integration extension or for support requests.



Please find the download for this project [here](#).



For the complete documentation explaining the setup, usage and all included examples just read the following chapters from the menu on the left.

As with all other Swyx Forum Open Source Projects, Support is **EXCLUSIVELY** provided in the Project Froum (see link above).

Lua based call routing has been introduced to SwyxWare in version **13.10** on. As of the time of the release of the Jira Service Management Integration extension the Lua based call routing is still in **BETA state** and should not be used in productive environments. However, this project already comes also for **Lua based** call routing as an example of its ease of use.

License

Jira Service Integration

v1.1.0

This is a Swyx Forum Open Source Project.

<https://www.swyxforum.com/projects/>

<https://www.swyxforum.com/jira-service-integration/introduction/0-introduction-r1/>

The MIT License (MIT)

Copyright (c) 2024 by Swyx Forum

Copyright (c) 2024 by Tom Wellige

All Rights Reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice must be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

For the VBScript based integration this project also includes the following Open Source project:

JSON object class 3.5.4 - May, 29th - 2016

Licence:

The MIT License (MIT)

Copyright (c) 2016 RCDMK - rcdmk[at]hotmail[dot]com

<https://github.com/rcdmk/aspJSON>

<https://github.com/rcdmk/aspJSON/blob/master/README.md>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Modifications needed for SwyxWare Call Routing

Tom Wellige, 03.05.2017

For the Lua based integration this project also includes the following Open Source project:

Simple JSON encoding and decoding in pure Lua.
Copyright 2010-2016 Jeffrey Friedl

<http://regex.info/blog/>

Latest version: <http://regex.info/blog/lua/json>

This code is released under a Creative Commons CC-BY "Attribution" License:

http://creativecommons.org/licenses/by/3.0/deed.en_US

It can be used for any purpose so long as:

- 1) the copyright notice above is maintained
- 2) the web-page links above are maintained
- 3) the 'AUTHOR_NOTE' string below is maintained



By Tom Wellige
September 3, 2024

 Share

Theme ▼

Copyright (c) 2007-2025 by Tom Wellige
Powered by Invision Community

Jira Service Integration

INTRODUCTION

0 - Introduction

PREPARATIONS

1 - Preparations

SETUP

2 - Setup GSE Actions

USAGE

3.1 - Create Issue

3.2 - Add Comment

3.3 - Check Status

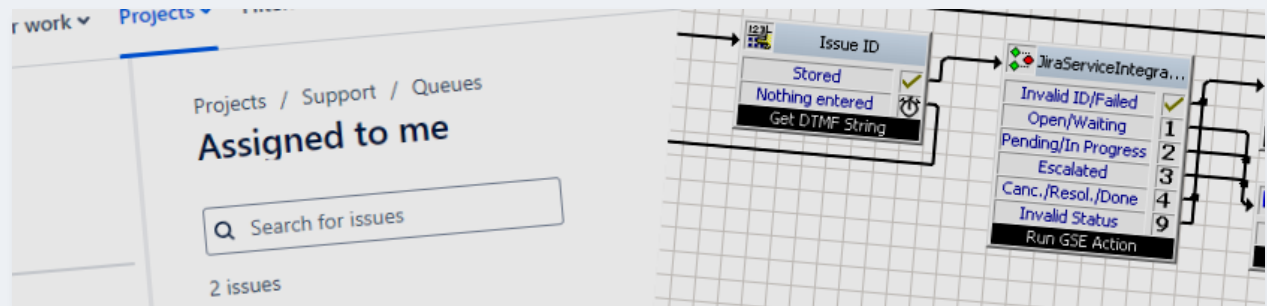
3.4 - Trouble Shooting

APPENDIX A

A.1 - Example: Create Issue

A.2 - Example: Add Comment

A.3 - Example: Check Status



Jira Service Integration - 1 - Preparations

Followers 0

VBScript

Lua

[Download](#) the latest version of this project.

To get access to the **Jira Service Management** portal via its **REST API** some means of authentication are required. Jira offers **OAuth 2.0** as also **API Token** authentication.

This extension makes use of the **API Token** authentication.

You need to create such an **API Token** from with the **ATLASSIAN Account** portal. The following Jira help center article explains in detail how to obtain the needed token:

A.4 - Example: Check Status and Add Comment

A.5 - Example: Check and Announce Status

APPENDIX B

B.1 - Access/modify the code behind

B.2 - Version History

- [Manage API tokens for your Atlassian account](#)

This link brings you directly into the **API Token** management of the ATlassian Account portal:

- [API Tokens](#)

Note: you need to create an API Token for the Jira user you want to use for the integration. New service issues or new comments into an existing issue are being created under this user (name). For later usage within this integration you need the **email address** of this user (as configured within the ATlassian Account portal) and the above created **API Token**.

Update 26.11.2024: Atlassian is going to make some changes in their security policy. Therefore:

- From December 15, 2024, new API tokens created by users will have a configurable duration of up to one year. This does not apply to keys being created before that date.

So you need to make sure to update your API Token in in time (yearly) when creating one the first time **after** the above given date.



By Tom Wellige
September 3, 2024

 Share

Theme ▼

Copyright (c) 2007-2025 by Tom Wellige

Powered by Invision Community

Jira Service Integration

INTRODUCTION

0 - Introduction

PREPARATIONS

1 - Preparations

SETUP

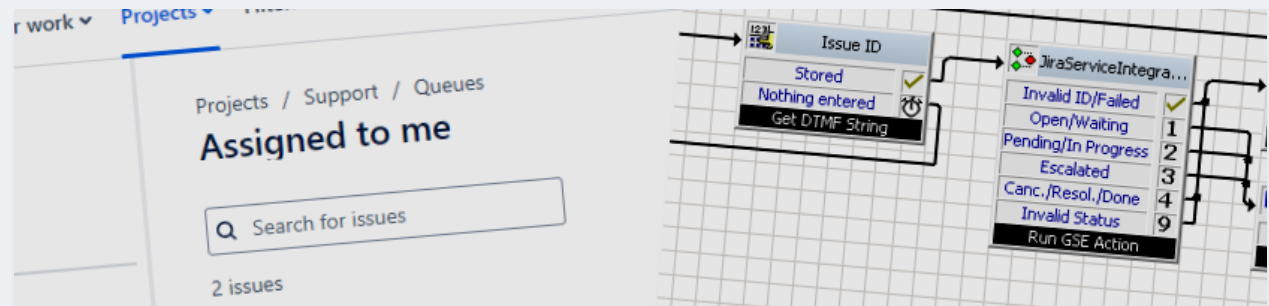
2 - Setup GSE Actions

USAGE

- 3.1 - Create Issue
- 3.2 - Add Comment
- 3.3 - Check Status
- 3.4 - Trouble Shooting

APPENDIX A

- A.1 - Example: Create Issue
- A.2 - Example: Add Comment
- A.3 - Example: Check Status



Jira Service Integration - 2 - Setup GSE Actions

Followers 0

VBScript

Lua

The **Jira Service Integration** extension is designed as a set of GSE actions. To install the GSE actions you need to use the **SwyxWare Administration**.

Step 1

Open the SwyxWare Administration and open the properties of your Swyx Server.

Step 2

Switch to the **Files** page and click on **Edit....**

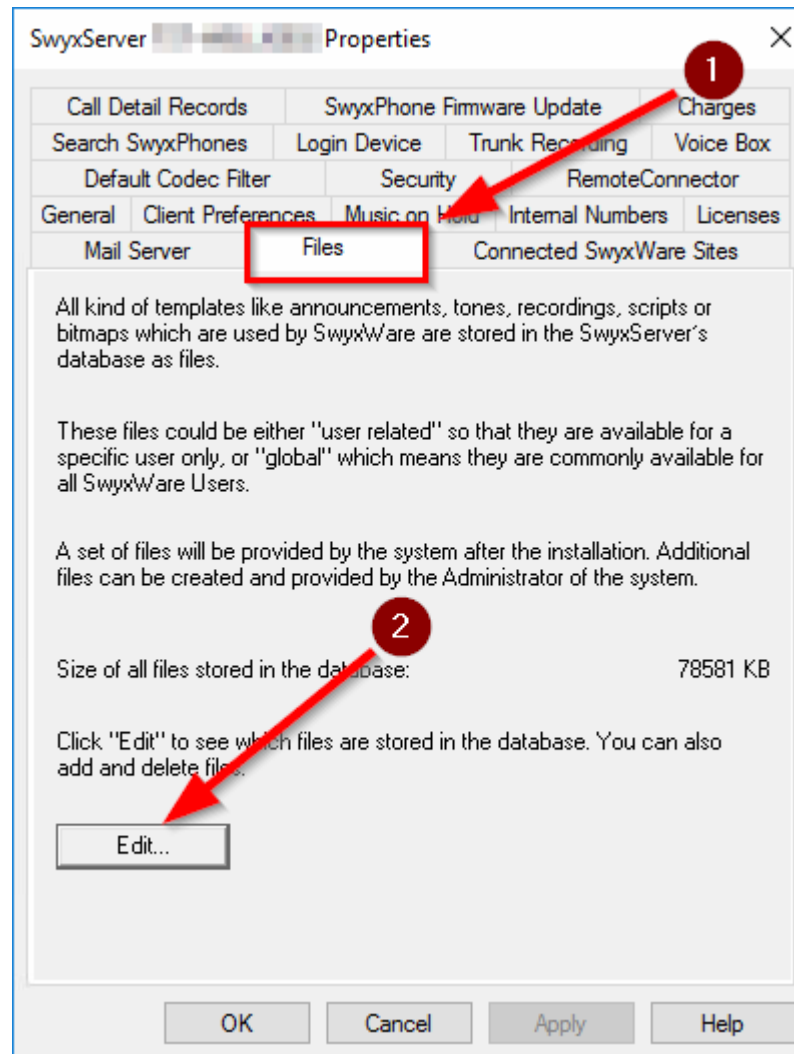
A.4 - Example: Check Status and Add Comment

A.5 - Example: Check and Announce Status

APPENDIX B

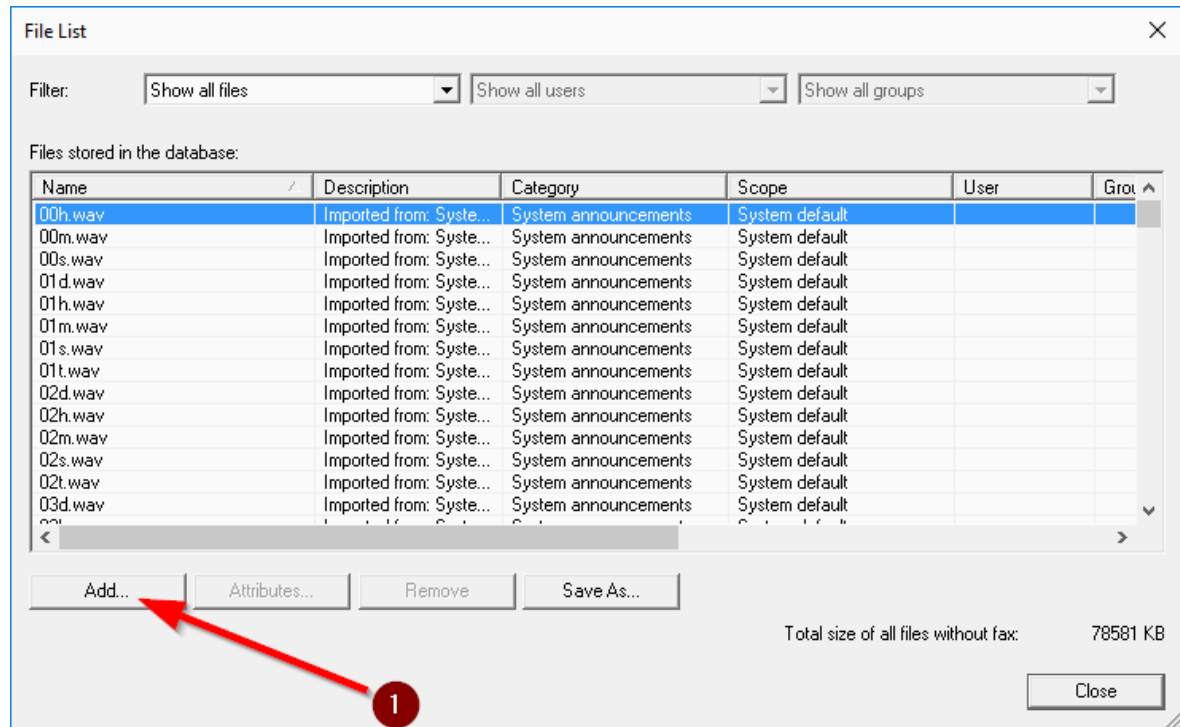
B.1 - Access/modify the code behind

B.2 - Version History



Step 3

Click on **Add...**



Step 4.1 - VBScript usage

1. Select all files from the **VBScript based\ase** folder from the download package.
2. Select **Global** as **Scope**. (you can also select User to load the files into the user scope a your script user)
3. Select **Call Routing VBS scripts** as **Category**.
4. It is recommended to enter **Jira Service Integration** into the **Description** field, but not necessary.

The screenshot shows a dialog box titled "Add File to Database" with a close button (X) in the top right corner. The dialog contains the following fields and options:

- File to add:** A text field containing "C:\Projects\JiraServiceIntegration\VBScript bas" followed by a red circle with the number "1" and a browse button "...".
- Name:** A text field containing "actionJiraServiceIntegrationAddCom".
- Scope:** A dropdown menu with "Global" selected, preceded by a red circle with the number "2".
- Category:** A dropdown menu with "Call Routing VBS scripts" selected, preceded by a red circle with the number "3".
- User:** A dropdown menu with "Botsquad" selected.
- Group:** A dropdown menu with "Everyone" selected.
- File Properties:** A section with three checkboxes: "Private", "Hidden", and "System", all of which are unchecked.
- Description:** A text field containing "Jira Service Integration" preceded by a red circle with the number "4".
- Buttons:** "OK" and "Cancel" buttons at the bottom right.

Step 4.2 - Lua usage

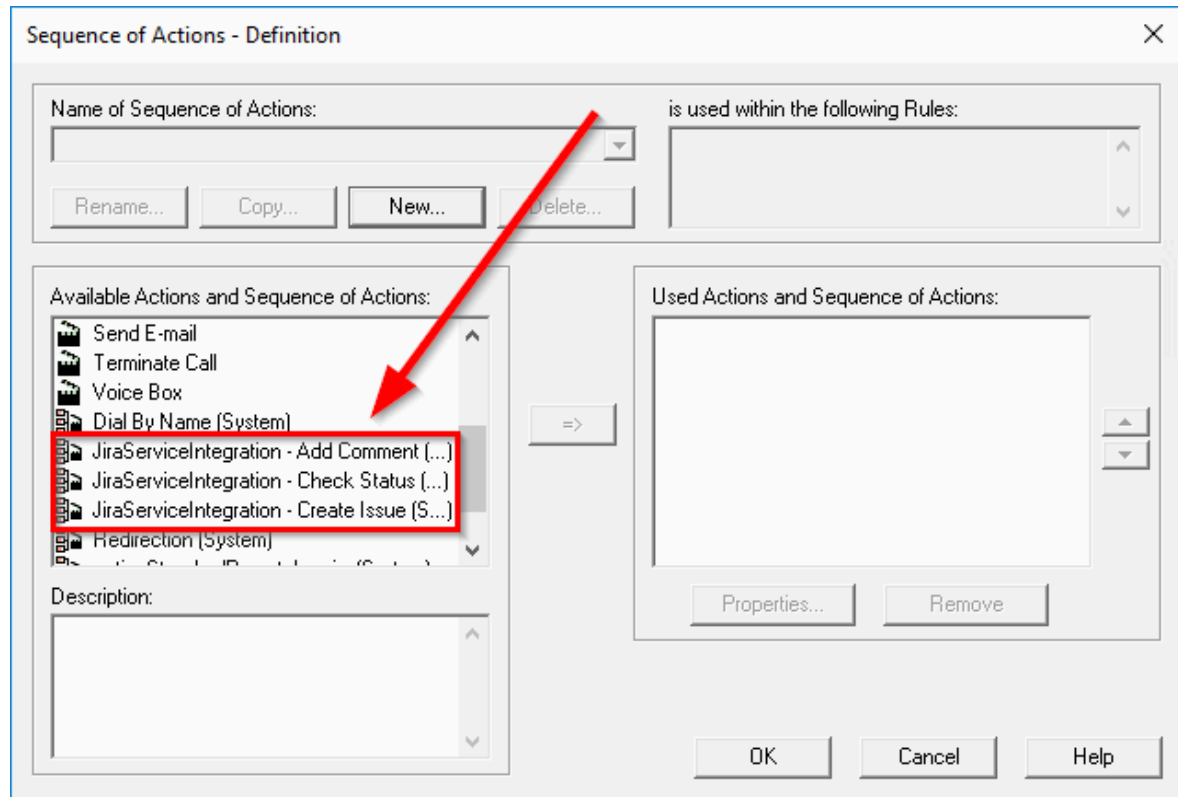
1. Select all files from the **Lua based\ase** folder from the download package.
2. Select **Global** as **Scope**. (you can also select User to load the files into the user scope a your script user)
3. Select **Call Routing Lua scripts** as **Category**.
4. It is recommended to enter **Jira Service Integration** into the **Description** field, but not necessary.

The screenshot shows a 'Add File to Database' dialog box with the following fields and callouts:

- File to add:** A text field containing the path 'C:\Projects\JiraServiceIntegration\Lua based\as...' with a red circle '1' over the file explorer button.
- Name:** A text field containing 'actionJiraServiceIntegrationAddCom'.
- Scope:** A dropdown menu set to 'Global' with a red circle '2' over it.
- Category:** A dropdown menu set to 'Call Routing Lua scripts' with a red circle '3' over it.
- User:** A dropdown menu set to 'Botsquad'.
- Group:** A dropdown menu set to 'Everyone'.
- File Properties:** A section with three unchecked checkboxes: 'Private', 'Hidden', and 'System'.
- Description:** A text field containing 'Jira Service Integration' with a red circle '4' over it.
- Buttons:** 'OK' and 'Cancel' buttons at the bottom right.

Step 5

To check if the GSE actions are available within call routing scripts open the **Call Routing Manager** of any user and click the button **Sequence of Actions**. Scroll the list on the left side down until you reach the **JiraServiceIntegration...** actions. The **(System)** behind the GSE action name shows that this is a global action being available for every SwyxWare user.



The Jira Service Integration actions are now installed and can be fully used within GSE call routing scripts.



By Tom Wellige
September 3, 2024

 Share

Theme ▼

Copyright (c) 2007-2025 by Tom Wellige

Powered by Invision Community

Jira Service Integration

INTRODUCTION

0 - Introduction

PREPARATIONS

1 - Preparations

SETUP

2 - Setup GSE Actions

USAGE

3.1 - Create Issue

3.2 - Add Comment

3.3 - Check Status

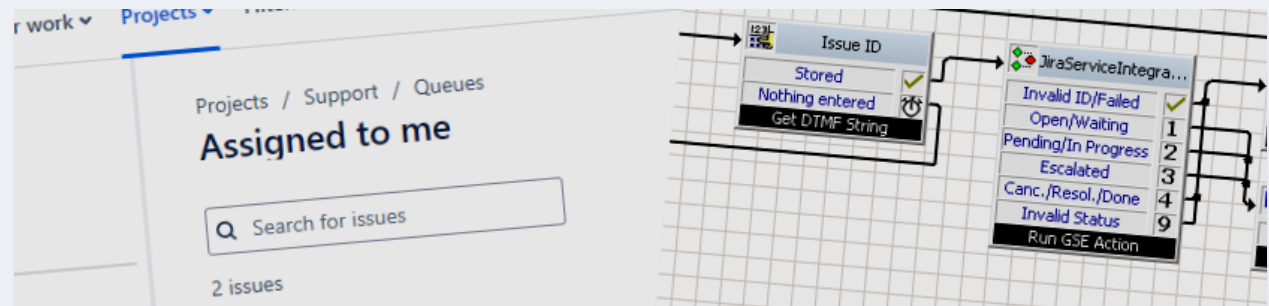
3.4 - Trouble Shooting

APPENDIX A

A.1 - Example: Create Issue

A.2 - Example: Add Comment

A.3 - Example: Check Status



Jira Service Integration - 3.1 - Create Issue

Followers 0

VBScript

Lua

This GSE action creates a new service issue. A common usage of this is to register a callback request.

An example call routing script can be found in [A.1 - Example: Create Issue](#).

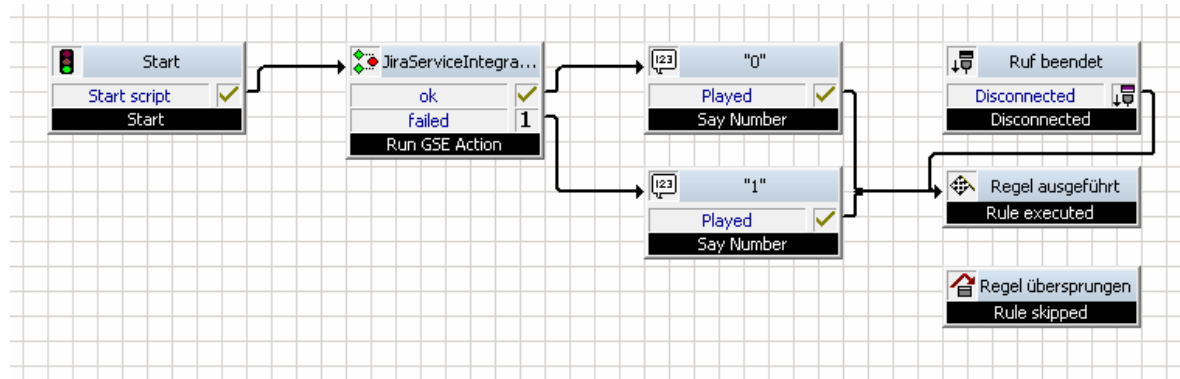
A.4 - Example: Check Status and Add Comment

A.5 - Example: Check and Announce Status

APPENDIX B

B.1 - Access/modify the code behind

B.2 - Version History



Configure action parameters

Run GSE Action Properties

General Parameters Links

Select GSE action: JiraServiceIntegration - Create Issue

Set action parameters:

Name	Value	Default Value
BaseURL	= ""	= ""
LoginUser	= ""	= ""
LoginToken	= ""	= ""
ProjectKey	= ""	= ""
Summary	= ""	= ""
Description	= ""	= ""

Edit Parameter...

Description:

Jira Service Integration v1.0.0
Creates a new issue.

Return Values:

0 - ok
1 - failed

OK Cancel Help

By double clicking a parameter in the list, you can edit it.

BaseURL Required

This is the ase URL of your Jira Service page, for example: <https://my-service.atlassian.net>

LoginUser Required

This is the email address of the Jira user you want to authenticate with. Please refer to [1- Preparations](#) for more details.

LoginToken Required

This is the API Token of the Jira user you want to authenticate with. Please refer to [1 - Preparations](#) for more details.

ProjectKey Required

When you created your Jira service project you defined a 3 character key which is used as prefix in all issue ids, for example "SUP" for your "Support" project.

Summary Required

This is the summary or title of the new issue.

Description Required

This is the description of the new issue.

IssueType Required

This is the type the new issue should be created with. When creating a new Jira service project you can select from many template how the project should look like, what types and states it will. You are also free to add any number of own types and states to your project. This example assumes you used the simple "Support" template and made no changes. This means, that the project has the following possible issue types: New Feature, Support, Developer escalation or Bug.

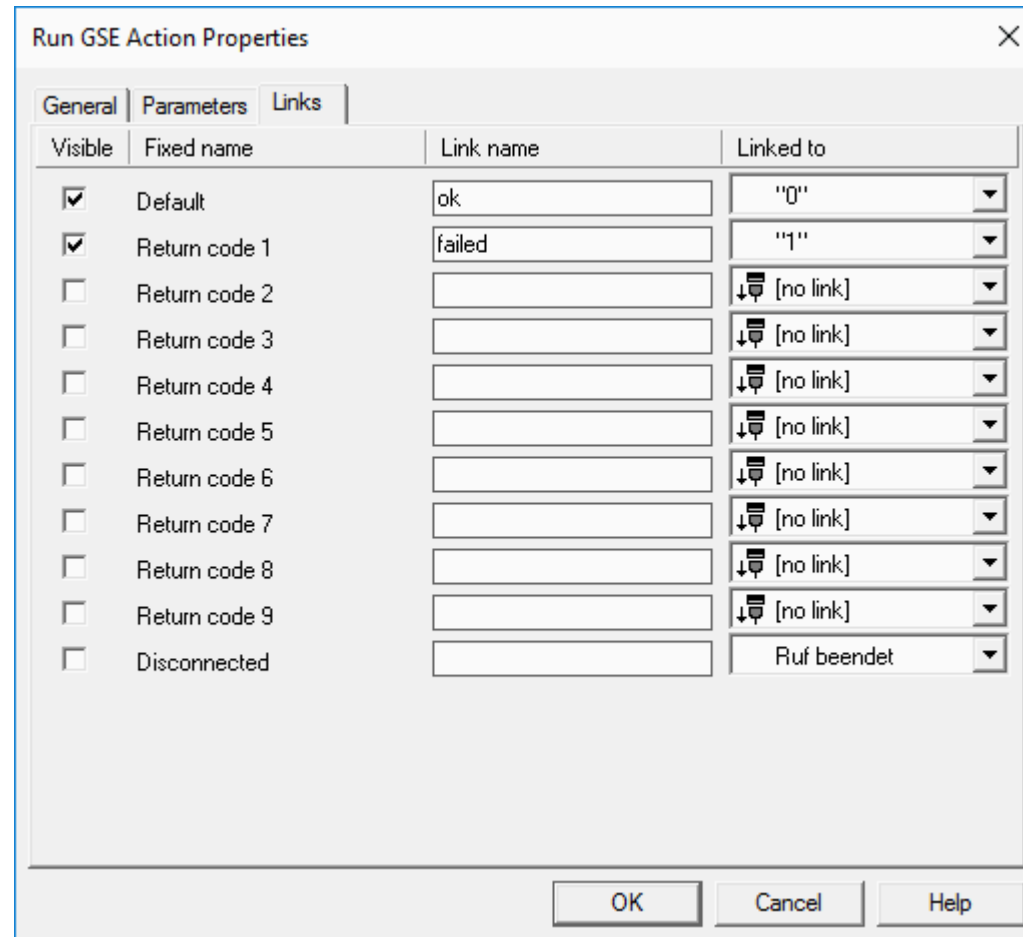
This GSE action checks the issue type you enter here against the list of available/valid types, which can be modified with the next parameter, if needed.

ValidIssueTypes Required

When creating a new Jira service project you can select from many template how the project should look like, what types and states it will. You are also free to add any number of own types and states to your project. This example assumes you used the simple "Support" template and made no changes. This means, that the project has the following possible issue types: New Feature, Support, Developer escalation or Bug.

If you modified your project you need to set the complete list of valid types here, as the GSE action checks the issue type you enter above against the types in this list.

Configure action exits



The image shows a dialog box titled "Run GSE Action Properties" with a close button (X) in the top right corner. It has three tabs: "General", "Parameters", and "Links". The "Links" tab is selected. Inside the "Links" tab, there is a table with four columns: "Visible", "Fixed name", "Link name", and "Linked to". The table contains ten rows. The first two rows are checked in the "Visible" column. The "Link name" column has text input fields, and the "Linked to" column has dropdown menus. At the bottom of the dialog are three buttons: "OK", "Cancel", and "Help".

Visible	Fixed name	Link name	Linked to
☒	Default	ok	"0"
☒	Return code 1	failed	"1"
☐	Return code 2		[no link]
☐	Return code 3		[no link]
☐	Return code 4		[no link]
☐	Return code 5		[no link]
☐	Return code 6		[no link]
☐	Return code 7		[no link]
☐	Return code 8		[no link]
☐	Return code 9		[no link]
☐	Disconnected		Ruf beendet

Exit 0 (Default)

This exit will be reached when everything worked fine and the service issue was created. It is recommended to name this exit **"ok"** or **"created"**.

Exit 1

This exit will be reached when there was any kind of problem and no topic has been created. It is recommended to name this exit "**failed**".

If you reach this exit you can refer to [3.4 - Trouble Shooting](#) to figure what went wrong.

Additional return value (as global variable)

g_sLatestJiraIssueID (string)

This global variable holds ID (in PROJECTKEY-XXX format) of the newly created service issue after the **ok** (0) exit has been reached.



By Tom Wellige
September 3, 2024

 Share

Next Page 
[3.2 - Add Comment](#)

Theme ▼

Copyright (c) 2007-2025 by Tom Wellige
Powered by Invision Community

Jira Service Integration

INTRODUCTION

0 - Introduction

PREPARATIONS

1 - Preparations

SETUP

2 - Setup GSE Actions

USAGE

3.1 - Create Issue

3.2 - Add Comment

3.3 - Check Status

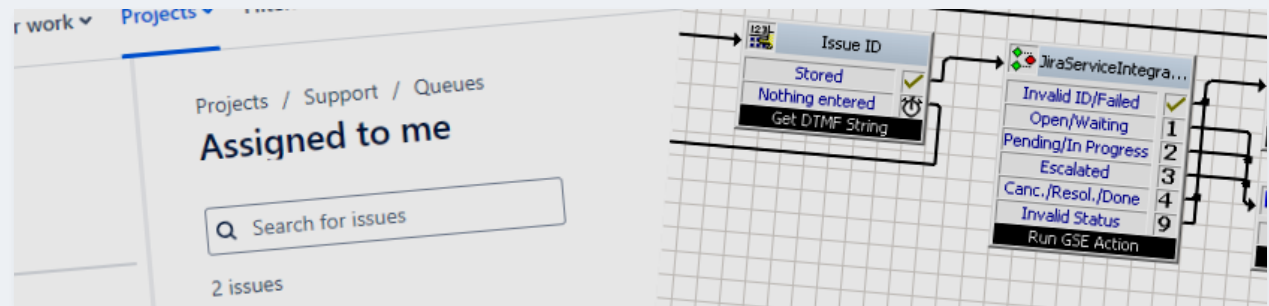
3.4 - Trouble Shooting

APPENDIX A

A.1 - Example: Create Issue

A.2 - Example: Add Comment

A.3 - Example: Check Status



Jira Service Integration - 3.2 - Add Comment

Followers 0

VBScript

Lua

This GSE action add a new comment to an existing service issue.

Example call routing scripts can be found here [A.2 - Example: Add Comment](#) and here [A.4 - Example: Check Status and Add Comment](#).

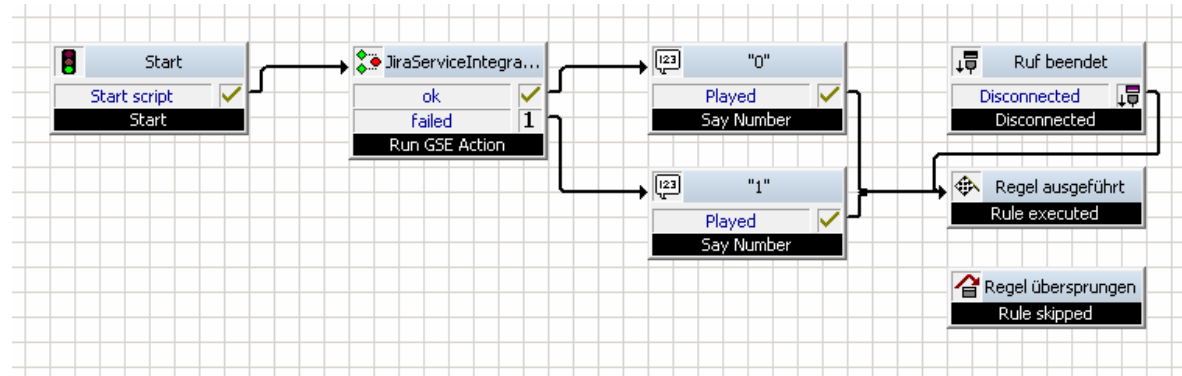
A.4 - Example: Check Status and Add Comment

A.5 - Example: Check and Announce Status

APPENDIX B

B.1 - Access/modify the code behind

B.2 - Version History



Configure action parameters

Run GSE Action Properties

General Parameters Links

Select GSE action: JiraServiceIntegration - Add Comment

Set action parameters:

Name	Value	Default Value
BaseURL	= ""	= ""
LoginUser	= ""	= ""
LoginToken	= ""	= ""
IssueID	= ""	= ""
Body	= ""	= ""
Internal	1	1

Edit Parameter...

Description:

Jira Service Integration v1.0.0
Adds new comment to existing issue.

Return Values:

0 - ok
1 - failed

OK Cancel Help

By double clicking a parameter in the list, you can edit it.

BaseURL Required

This is the ase URL of your Jira Service page, for example: <https://my-service.atlassian.net>

LoginUser Required

This is the email address of the Jira user you want to authenticate with. Please refer to [1- Preparations](#) for more details.

LoginToken Required

This is the API Token of the Jira user you want to authenticate with. Please refer to [1-Preparations](#) for more details.

IssueID Required

This is the id of the service issue you want to add a comment to. The format of the id is typically PROJECTKEY-XXX, e.g. SUP-42

Body Required

This is the text of the comment.

Internal

You can define, if the new comment will only be visible internally (1) or is also visible to your customer (0).

Valid values or 0 and 1. Default: 1

Configure action exits

Run GSE Action Properties

General Parameters Links

Visible	Fixed name	Link name	Linked to
☒	Default	ok	"0"
☒	Return code 1	failed	"1"
☐	Return code 2		↓ [no link]
☐	Return code 3		↓ [no link]
☐	Return code 4		↓ [no link]
☐	Return code 5		↓ [no link]
☐	Return code 6		↓ [no link]
☐	Return code 7		↓ [no link]
☐	Return code 8		↓ [no link]
☐	Return code 9		↓ [no link]
☐	Disconnected		Ruf beendet

OK Cancel Help

Exit 0 (Default)

This exit will be reached when everything worked fine and the comment was created. It is recommended to name this exit "**ok**" or "**created**".

Exit 1

This exit will be reached when there was any kind of problem and no comment has been created. It is recommended to name this exit "**failed**".

If you reach this exit you can refer to [3.4 - Trouble Shooting](#) to figure what went wrong.

Additional return value (as global variable)

g_sLatestJiraIssueID (string)

This global variable holds ID (in PROJECTKEY-XXX format) of the issue the comment was added to. after the **ok** (0) exit has been reached.



By Tom Wellige

September 3, 2024

 Share

< Previous Page
3.1 - Create Issue

Next Page >
3.3 - Check Status

Theme ▼

Copyright (c) 2007-2025 by Tom Wellige
Powered by Invision Community

Jira Service Integration

INTRODUCTION

0 - Introduction

PREPARATIONS

1 - Preparations

SETUP

2 - Setup GSE Actions

USAGE

3.1 - Create Issue

3.2 - Add Comment

3.3 - Check Status

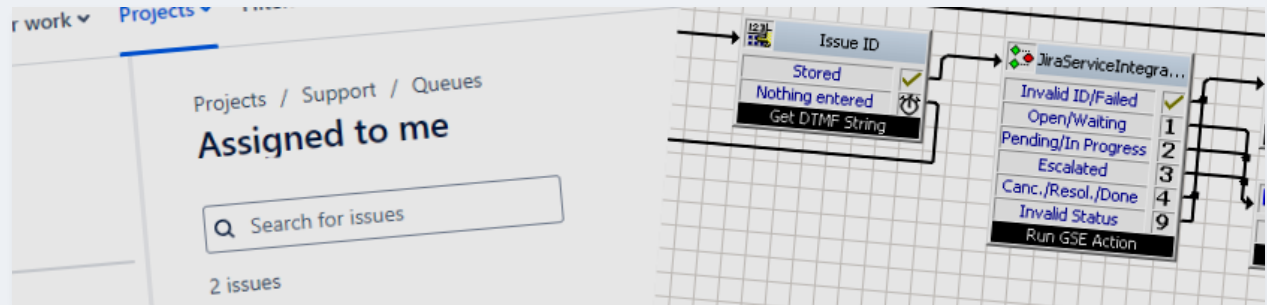
3.4 - Trouble Shooting

APPENDIX A

A.1 - Example: Create Issue

A.2 - Example: Add Comment

A.3 - Example: Check Status



Jira Service Integration - 3.3 - Check Status

Followers

0

VBScript

Lua

This GSE action checks the status of an existing service issue.

Example call routing scripts can be found here [A.3 - Example: Check Status](#) and here [A.4 - Example Check Status and Add Comment](#).

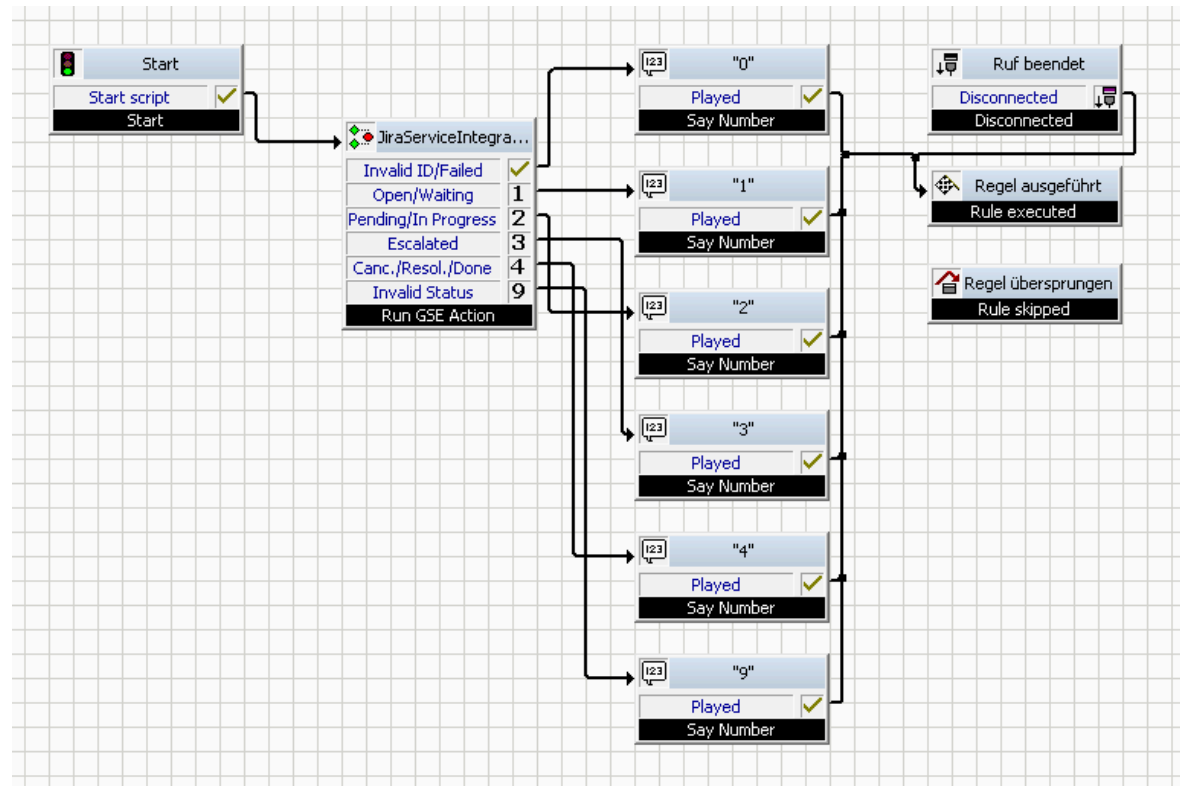
A.4 - Example: Check Status and Add Comment

A.5 - Example: Check and Announce Status

APPENDIX B

B.1 - Access/modify the code behind

B.2 - Version History



Configure action parameters

Run GSE Action Properties

General Parameters Links

Select GSE action: JiraServiceIntegration - Check Status

Set action parameters:

Name	Value	Default Value
BaseURL	= ""	= ""
LoginUser	= ""	= ""
LoginToken	= ""	= ""
IssueID	= ""	= ""

Edit Parameter...

Description:

Jira Service Integration v1.0.0
Checks current status of existing issue.

Return Values:

0 - ok
1 - failed

OK Cancel Help

By double clicking a parameter in the list, you can edit it.

BaseURL Required

This is the base URL of your Jira Service page, for example: <https://my-service.atlassian.net>

LoginUser Required

This is the email address of the Jira user you want to authenticate with. Please refer to [1- Preparations](#) for more details.

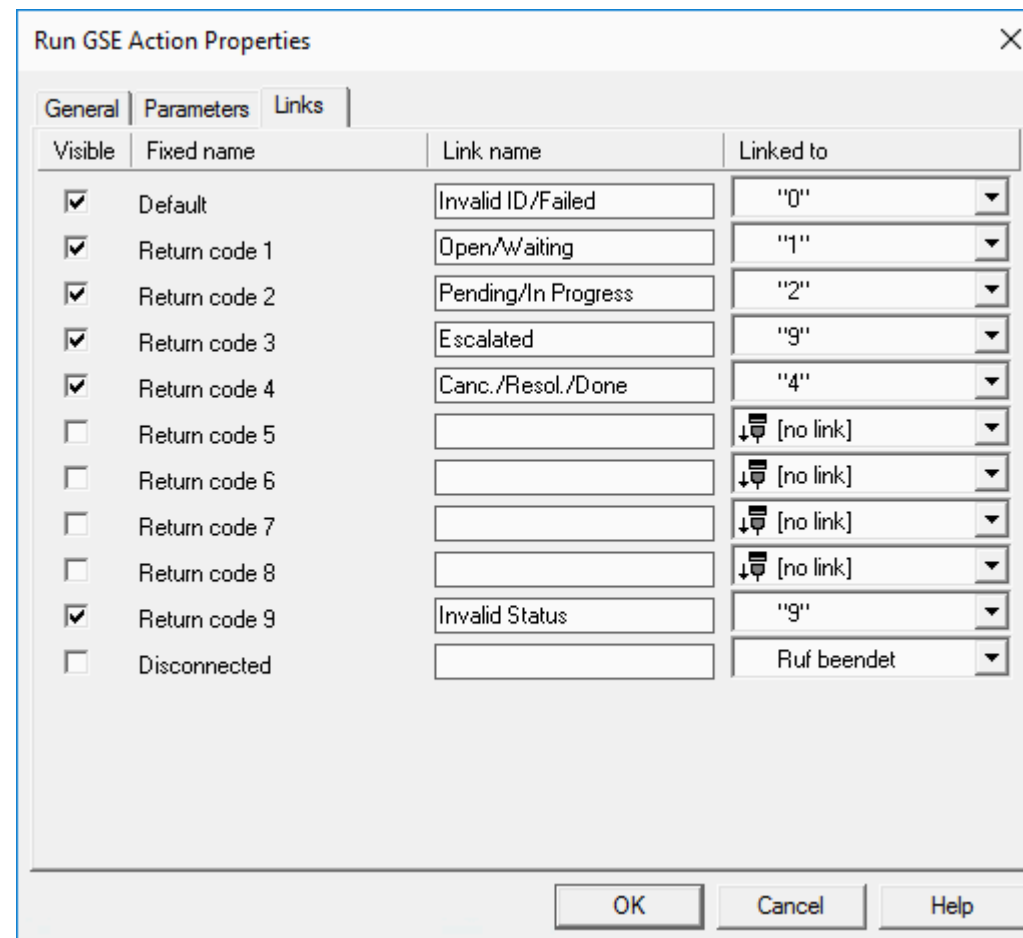
LoginToken Required

This is the API Token of the Jira user you want to authenticate with. Please refer to [1- Preparations](#) for more details.

IssueID Required

This is the id of the service issue you to check the current status for. The format of the id is typically PROJECTKEY-XXX, e.g. SUP-42

Configure action exits



The image shows a dialog box titled "Run GSE Action Properties" with a close button (X) in the top right corner. It has three tabs: "General", "Parameters", and "Links". The "Links" tab is selected. The dialog contains a table with four columns: "Visible", "Fixed name", "Link name", and "Linked to".

Visible	Fixed name	Link name	Linked to
☒	Default	Invalid ID/Failed	"0"
☒	Return code 1	Open/Waiting	"1"
☒	Return code 2	Pending/In Progress	"2"
☒	Return code 3	Escalated	"9"
☒	Return code 4	Canc./Resol./Done	"4"
☐	Return code 5		↓ [no link]
☐	Return code 6		↓ [no link]
☐	Return code 7		↓ [no link]
☐	Return code 8		↓ [no link]
☒	Return code 9	Invalid Status	"9"
☐	Disconnected		Ruf beendet

At the bottom of the dialog are three buttons: "OK", "Cancel", and "Help".

Exit 0 (Default)

This exit will be reached if an invalid issue id was given or any kind of problem occurred. It is recommended to name this exit "**Invalid ID/Failed**".

If you reach this exit you can refer to [3.4 - Trouble Shooting](#) to figure what went wrong.

Exit 9

This exit will be reached if an unknown status was returned by the Jira REST API. It is recommended to name this exit "**Unknown Status**".

If you reach this exit you can refer to [3.4 - Trouble Shooting](#) to figure the returned status and modify your status mapping.

Dynamic Status Mapping

Jira service projects are highly flexible and depending on the selected template when you created your project and the made modifications, your project can have any number of different workflows and states.

This GSE action respects this flexibility and therefore doesn't have a fixed binding of states to block exits. All block exits **1 to 8** are of your free choosing.

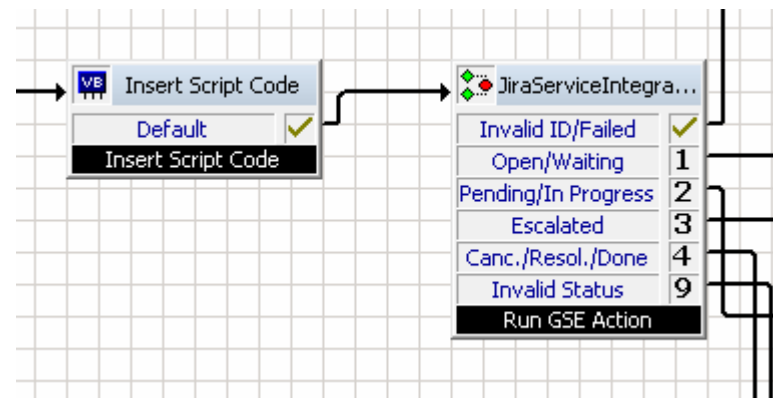
To have something to start with, this GSE action assumes you used the **Support** template when you created your service project and haven't made any modifications in the workflows. This gives you a certain set of possible states, which by default are bound like this to the exits of the block:

Exit States

- | | |
|---|---|
| 1 | Open, Waiting for support, Waiting for customer |
| 2 | Pending, In progress, Work in progress |
| 3 | Escalated |
| 4 | Canceled, Resolved, Done |

As already mentioned, you can modify this mapping to your precise needs. Just keep in mind that only the exits **1 to 8** are available to you.

All you need to do is to modify a global variable before you use the **Check Status** GSE action. This can best be done in a **Insert Script Code** block.



The content of the Insert Script Code block is then the definition of the mapping list, which is either a **VBScript Array** or a **Lua Table**. In both cases you are free to map any number of states to any exit (1..8).

Lets assume you want to map your own states **State A** and **State B** to exit **5** and **State C** to exit **6**: This is the code you need to place into the **Insert Script Code** block:

For **VBScript** based call routing:

```
UseExit = 0

g_aJiraServiceStatusMapping = Array( _
    "1;Waiting for support", _
    "1;Waiting for customer", _
    "1;Open", _
```

```
"2;Pending", _  
"2;In progress", _  
"2;Work in progress", _  
"3;Escalated", _  
"4;Canceled", _  
"4;Resolved", _  
"4;Done", _  
"5;State A", _  
"5;State B", _  
"6;State C")
```

For **Lua** based call routing:

```
UseExit = 0  
  
g_aJiraServiceStatusMapping = {  
    "1;Waiting for support",  
    "1;Waiting for customer",  
    "1;Open",  
    "2;Pending",  
    "2;In progress",  
    "2;Work in progress",  
    "3;Escalated",  
    "4;Canceled",  
    "4;Resolved",  
    "4;Done",  
    "5;State A",  
    "5;State B",  
    "6;State C"  
}
```

As you can see, you are absolutely free in terms of the mapping of the different possible **states** of your service issue to an **exit** of the **Run GSE Action** block.

All you need to do is to do your mapping in the **Insert Script Code** block and afterwards open and name the **exits** of the **Run GSE Action block** accordingly.

An example of some own status mapping can be found in [A.5 - Example: Check and Announce Status](#).

Additional return values (as global variables)

g_sLatestJiralssueID (string)

This global variable holds the ID of the latest checked issue after the **ok** (0) exit has been reached.

g_sLatestJiralssueStatus (string) 1.1.0

This global variable holds the status of the latest checked issue after the **ok** (0) exit has been reached.

The [A.5 - Example: Check and Announce Status](#) makes use of this variable to announce the status via [AzureTTS](#) (text-to-speech).

g_sLatestJiralssueModified (string) 1.1.0

This global variable holds the latest modification date of the latest checked issue after the **ok** (0) exit has been reached.

The date is as the Jira API provides it, e.g. "2024-09-03T14:48:17.138+0200".

g_sLatestJiralssueModifiedReadable (string) 1.1.0

This global variable holds the latest modification date of the latest checked issue after the **ok** (0) exit has been reached.

The date is in a better readable format, e.g. "03.09.2024 14:48:17".

The [A.5 - Example: Check and Announce Status](#) makes use of this variable to announce the latest modification date via [AzureTTS \(text-to-speech\)](#).



By Tom Wellige

September 3, 2024

 Share



Previous Page

[3.2 - Add Comment](#)

Next Page

[3.4 - Trouble Shooting](#)



Theme ▼

Copyright (c) 2007-2025 by Tom Wellige

Powered by Invision Community

Jira Service Integration

INTRODUCTION

0 - Introduction

PREPARATIONS

1 - Preparations

SETUP

2 - Setup GSE Actions

USAGE

3.1 - Create Issue

3.2 - Add Comment

3.3 - Check Status

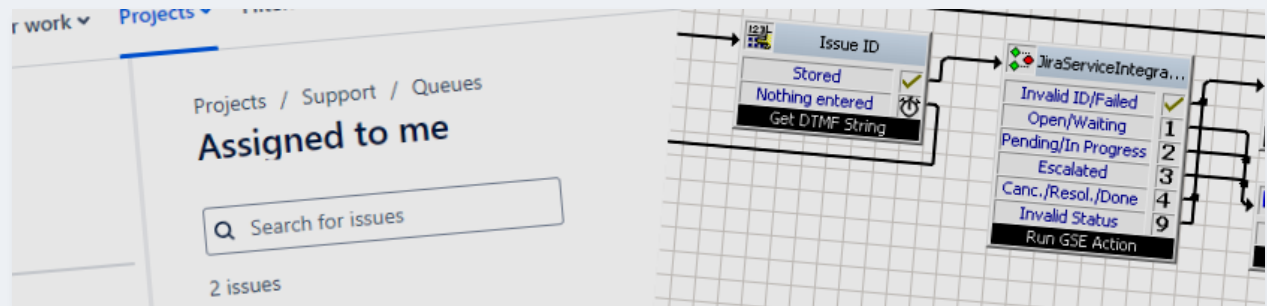
3.4 - Trouble Shooting

APPENDIX A

A.1 - Example: Create Issue

A.2 - Example: Add Comment

A.3 - Example: Check Status



Jira Service Integration - 3.4 - Trouble Shooting

Followers

0

VBScript

Lua

The Jira Service Integration extension writes meaningful trace information into the server trace file, according to the trace recommendation from this blog post:

- [Don't be shy, be chatty!](#)

So, to figure the exact reason for the problem you need to take a look into that file. The following link explains where to find it and how to filter its content down to the call in question and call routing output only:

- [How to filter SwyxWare traces for call routing output of single call](#)

A.4 - Example: Check Status and Add Comment

A.5 - Example: Check and Announce Status

APPENDIX B

B.1 - Access/modify the code behind

B.2 - Version History

Once you have opened and filtered the trace file you need to search for the first trace lines written by the GSE action. Just look for these lines:

For **VBScript** based call routing:

```
-----> JiraService.Class_Initialize
```

For **Lua** based call routing:

```
-----> JiraServiceIntegrationClass:Create
```

The following lines will contain information of all things happened, like setting the parameters and calling the Jira REST API.

If at some point an error occurs, you will see an error message in the trace.

If the REST API call fails, for example because you entered an invalid API Token, you will see the **response body** of the Jira REST API in the trace which contains usually proper information on why the request failed.

You will also find the original **response code** of the web request. If everything went fine, the response code is **201**, otherwise its any other number (most likely ≥ 400), e.g.

```
nResponseCode = 401
```

By taking the detailed error information from the server trace file you should be able to fix your problem.

In case you can't get your problem fixed or have specific questions, please open your own topic in the [project forum](#).



By Tom Wellige

September 3, 2024

 Share



Previous Page

3.3 - Check Status

Theme ▼

Copyright (c) 2007-2025 by Tom Wellige

Powered by Invision Community

Jira Service Integration

INTRODUCTION

0 - Introduction

PREPARATIONS

1 - Preparations

SETUP

2 - Setup GSE Actions

USAGE

3.1 - Create Issue

3.2 - Add Comment

3.3 - Check Status

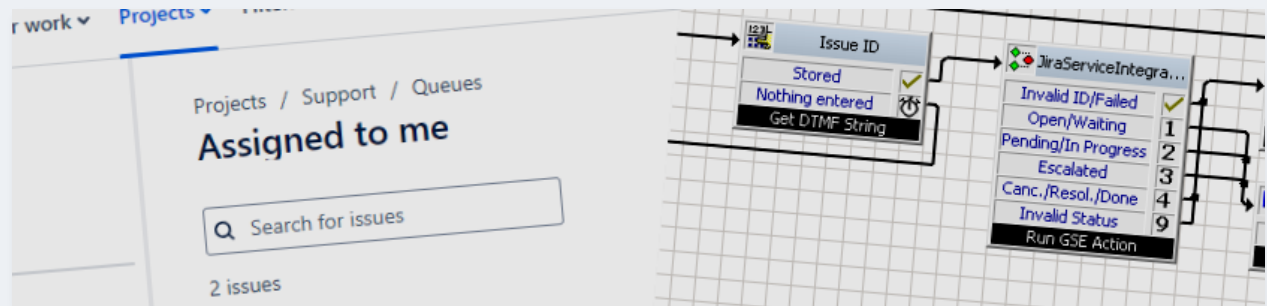
3.4 - Trouble Shooting

APPENDIX A

A.1 - Example: Create Issue

A.2 - Example: Add Comment

A.3 - Example: Check Status



Jira Service Integration - A.1 - Example: Create Issue

Followers 0

VBScript

Lua

This example demonstrates the usage of the **JiraServiceIntegration - Create Issue** GSE action.

It creates a new service topic in your Jira service project and announces afterwards "0" on success or "1" on failure.

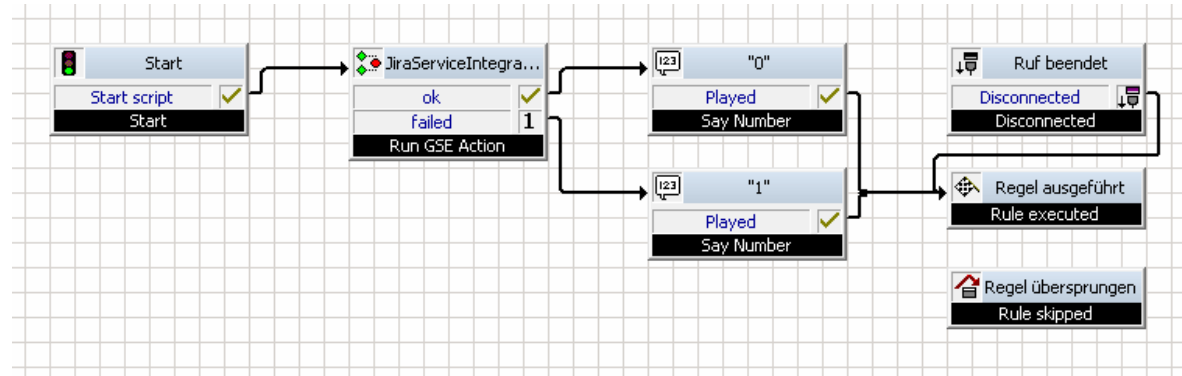
A.4 - Example: Check Status and Add Comment

A.5 - Example: Check and Announce Status

APPENDIX B

B.1 - Access/modify the code behind

B.2 - Version History



To install it:

1. Open the **Call Routing Manager** of your desired SwyxWare user.
2. Click the "New Rule..." button.
3. Select "**Graphical Script Editor**" and click **Ok**.
4. Within the GSE open the **File | Import...** menu and click **No**.
5. From the download package select the following file:

For **VBScript** usage:

```
VBScript based\rse\Create Issue.rse
```

For **Lua** usage:

```
Lua based\rse\Create Issue.rse
```

6. You need to make some modifications in the **JiraServiceIntegration - Create Issue** (Run GSE Action) block . They are explained in detail in chapter [3.1 - Create Issue](#).



By Tom Wellige
September 3, 2024

 Share

Next Page >
[A.2 - Example: Add Comment](#)

Theme ▼

Copyright (c) 2007-2025 by Tom Wellige
Powered by Invision Community

Jira Service Integration

INTRODUCTION

0 - Introduction

PREPARATIONS

1 - Preparations

SETUP

2 - Setup GSE Actions

USAGE

3.1 - Create Issue

3.2 - Add Comment

3.3 - Check Status

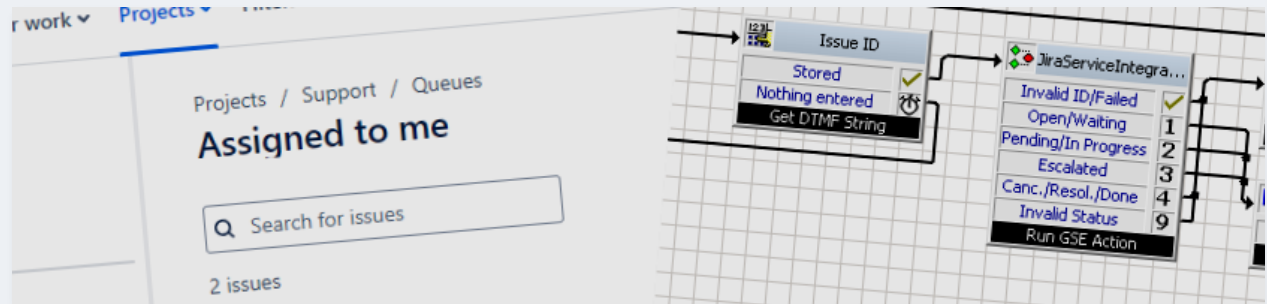
3.4 - Trouble Shooting

APPENDIX A

A.1 - Example: Create Issue

A.2 - Example: Add Comment

A.3 - Example: Check Status



Jira Service Integration - A.2 - Example: Add Comment

Followers 0

VBScript

Lua

This example demonstrates the usage of the **JiraServiceIntegration - Add Comment** GSE action.

It adds a new comment to an existing service issue and announces afterwards "0" on success or "1" on failure.

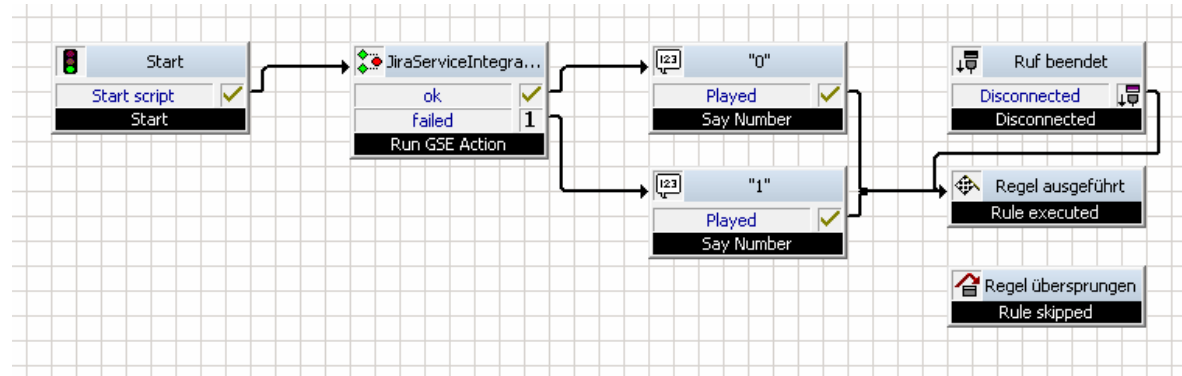
A.4 - Example: Check Status and Add Comment

A.5 - Example: Check and Announce Status

APPENDIX B

B.1 - Access/modify the code behind

B.2 - Version History



To install it:

1. Open the **Call Routing Manager** of your desired SwyxWare user.
2. Click the "New Rule..." button.
3. Select "**Graphical Script Editor**" and click **Ok**.
4. Within the GSE open the **File | Import...** menu and click **No**.
5. From the download package select the following file:

For **VBScript** usage:

```
VBScript based\rse\Add Comment.rse
```

For **Lua** usage:

```
Lua based\rse\Add Comment.rse
```

6. You need to make some modifications in the **JiraServiceIntegration - Add Comment** (Run GSE Action) block . They are explained in detail in chapter [3.2 - Add Comment](#).



By Tom Wellige
September 3, 2024

 Share

< Previous Page
A.1 - Example: Create Issue

Next Page >
A.3 - Example: Check Status

Theme ▼

Copyright (c) 2007-2025 by Tom Wellige
Powered by Invision Community

Jira Service Integration

INTRODUCTION

0 - Introduction

PREPARATIONS

1 - Preparations

SETUP

2 - Setup GSE Actions

USAGE

3.1 - Create Issue

3.2 - Add Comment

3.3 - Check Status

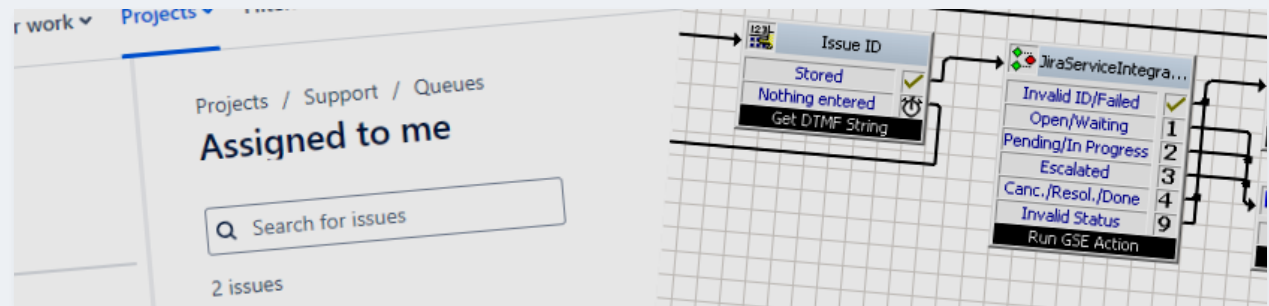
3.4 - Trouble Shooting

APPENDIX A

A.1 - Example: Create Issue

A.2 - Example: Add Comment

A.3 - Example: Check Status



Jira Service Integration - A.3 - Example: Check Status

Followers

0

VBScript

Lua

This example demonstrates the usage of the **JiraServiceIntegration - Check Status** GSE action.

It checks the current status of an existing service issue and announces the result.

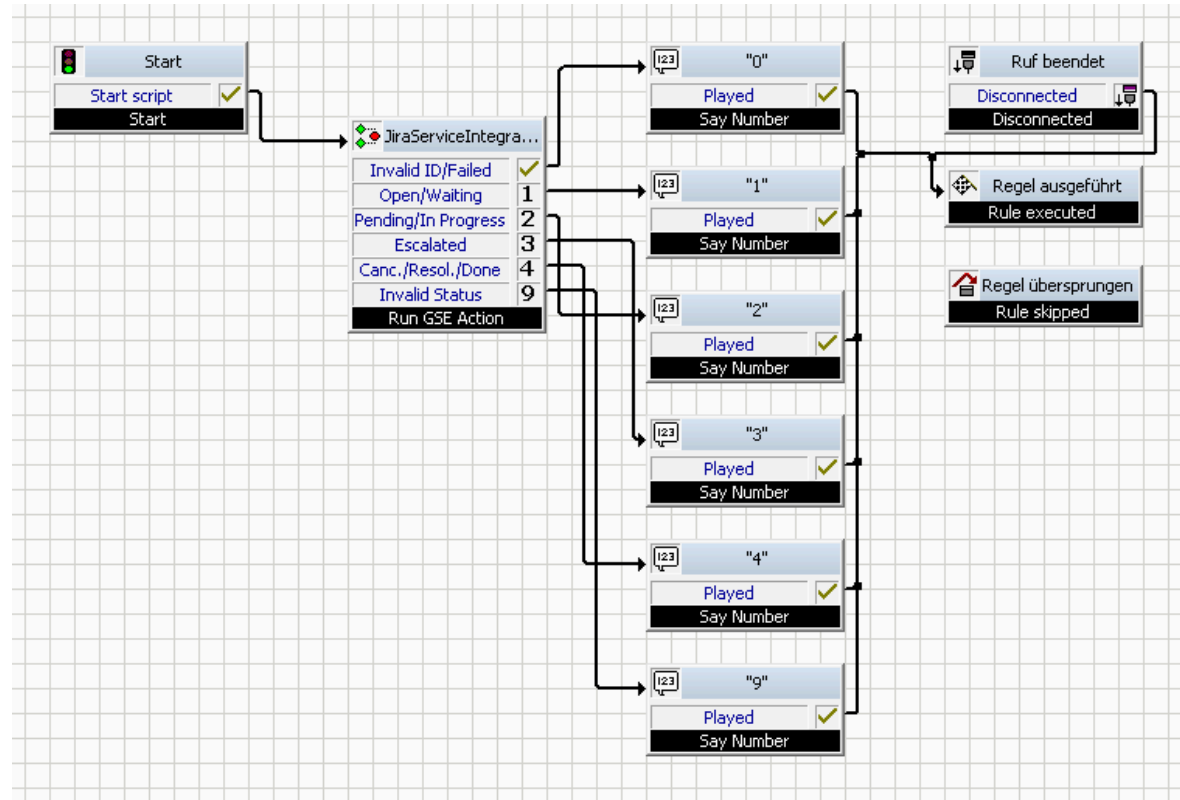
A.4 - Example: Check Status and Add Comment

A.5 - Example: Check and Announce Status

APPENDIX B

B.1 - Access/modify the code behind

B.2 - Version History



To install it:

1. Open the **Call Routing Manager** of your desired SwyxWare user.
2. Click the "New Rule..." button.
3. Select "**Graphical Script Editor**" and click **Ok**.
4. Within the GSE open the **File | Import...** menu and click **No**.
5. From the download package select the following file:

For **VBScript** usage:

```
VBScript based\rse\Check Status.rse
```

For **Lua** usage:

```
Lua based\rse\Check Status.rse
```

6. You need to make some modifications in the **JiraServiceIntegration - Check Status** (Run GSE Action) block . They are explained in detail in chapter [3.3 - Check Status](#).



By Tom Wellige
September 3, 2024

Share

< Previous Page
A.2 - Example: Add Comment

Next Page >
A.4 - Example: Check Status and Add Comment

Theme ▼

Copyright (c) 2007-2025 by Tom Wellige
Powered by Invision Community

Jira Service Integration

INTRODUCTION

0 - Introduction

PREPARATIONS

1 - Preparations

SETUP

2 - Setup GSE Actions

USAGE

3.1 - Create Issue

3.2 - Add Comment

3.3 - Check Status

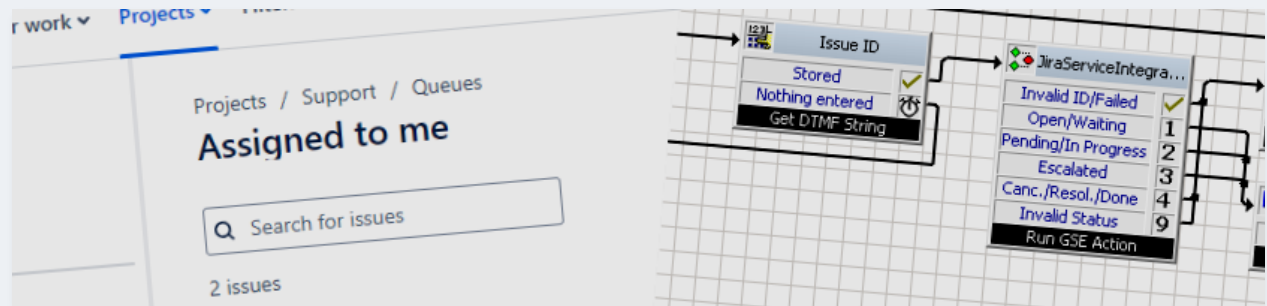
3.4 - Trouble Shooting

APPENDIX A

A.1 - Example: Create Issue

A.2 - Example: Add Comment

A.3 - Example: Check Status



Jira Service Integration - A.4 - Example: Check Status and Add Comment

Followers 0

VBScript

Lua

This example demonstrates the usage of the **JiraServiceIntegration - Check Status** and **JiraServiceIntegration - Add Comment** GSE actions.

It is assumed to be used on a support help desk, where only callers with a valid issue id are getting connected. Additionally all call details will be added as comment to the issue at the end of the call.

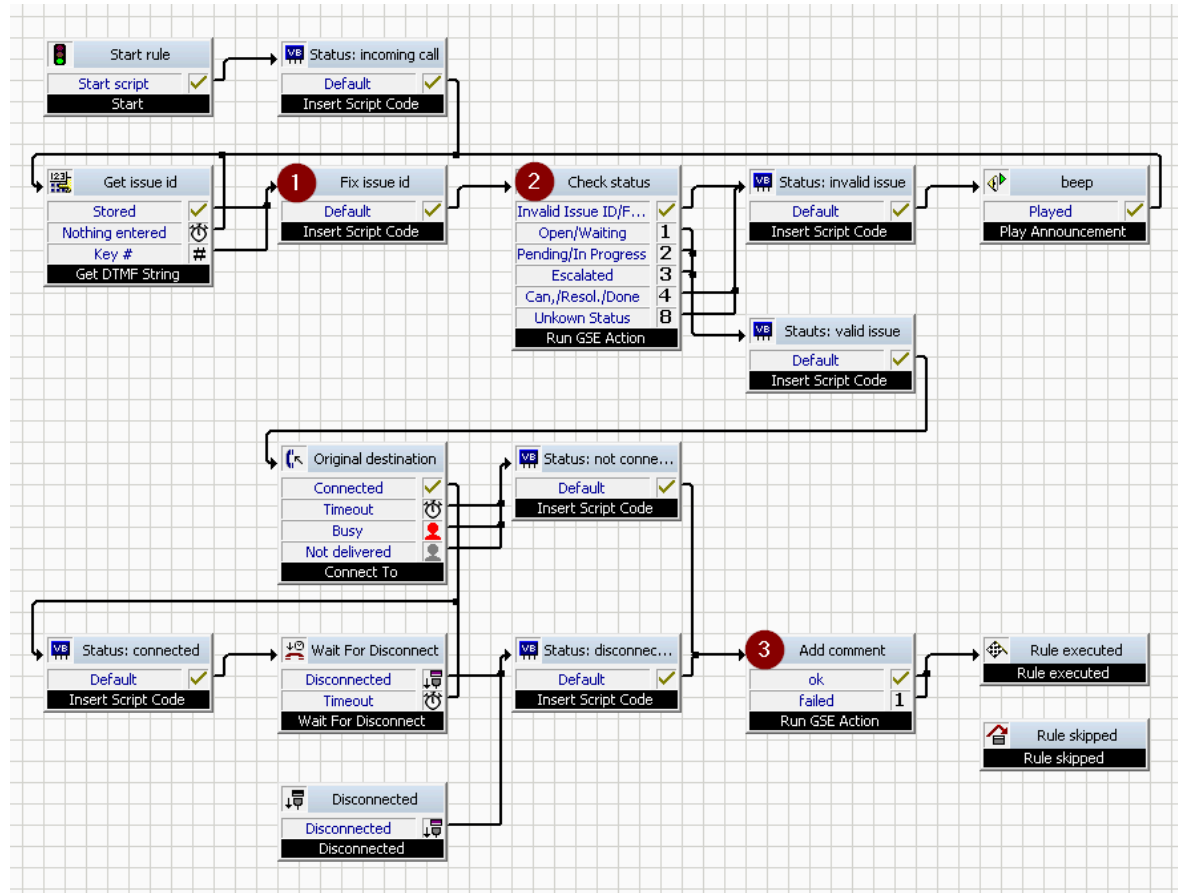
A.4 - Example: Check Status and Add Comment

A.5 - Example: Check and Announce Status

APPENDIX B

B.1 - Access/modify the code behind

B.2 - Version History



To install it:

1. Open the **Call Routing Manager** of your desired SwyxWare user.
2. Click the "New Rule..." button.
3. Select "**Graphical Script Editor**" and click **Ok**.
4. Within the GSE open the **File | Import...** menu and click **No**.

5. From the download package select the following file:

For **VBScript** usage:

```
VBScript based\rse\Check Status and Add Comment.rse
```

For **Lua** usage:

```
Lua based\rse\Check Status and Add Comment.rse
```

6. You need to make some modifications to make it work:

(1) - Configure your own **ProjectKey**

(2) - Make all needed configurations according to [3.3 - Check Status](#)

(3) - Make all needed configurations according to [3.2 - Add Comment](#)

After a call has been completely handled by the call routing script and also afterwards by an agent (and finally disconnected), you will find the following call details added as a new comment into the issue:

```
04.09.2024 21:17:34 : Incoming call from 'Test User', 101
04.09.2024 21:17:55 : Issue validated.
04.09.2024 21:18:04 : Call connected to Agent 1
04.09.2024 21:19:41 : Call connected to Agent 2
04.09.2024 21:22:17 : Call disconnected.
```



By Tom Wellige
September 4, 2024

 Share

< Previous Page
A.3 - Example: Check Status

Next Page >
A.5 - Example: Check and Announce Status

Theme ▼

Copyright (c) 2007-2025 by Tom Wellige
Powered by Invision Community

Jira Service Integration

INTRODUCTION

0 - Introduction

PREPARATIONS

1 - Preparations

SETUP

2 - Setup GSE Actions

USAGE

3.1 - Create Issue

3.2 - Add Comment

3.3 - Check Status

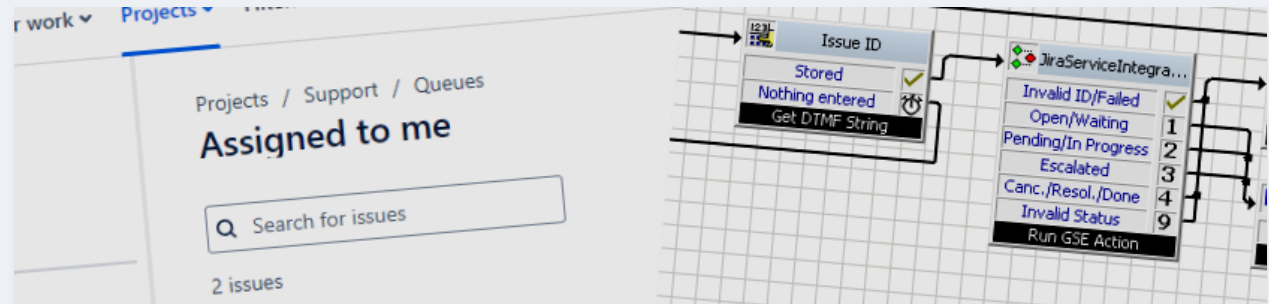
3.4 - Trouble Shooting

APPENDIX A

A.1 - Example: Create Issue

A.2 - Example: Add Comment

A.3 - Example: Check Status



Jira Service Integration - A.5 - Example: Check and Announce Status

Followers

0

VBScript

Lua

1.1.0

This example demonstrates the usage of the **JiraServiceIntegration - Create Issue** and **AzureTTS (text-to-speech)** GSE actions.

You need to have the **AzureTTS (text-to-speech)** Open ECR Extension installed beside the Jira Service Integration.

It asks for an issue id (close id with #), checks the status and announces its details.

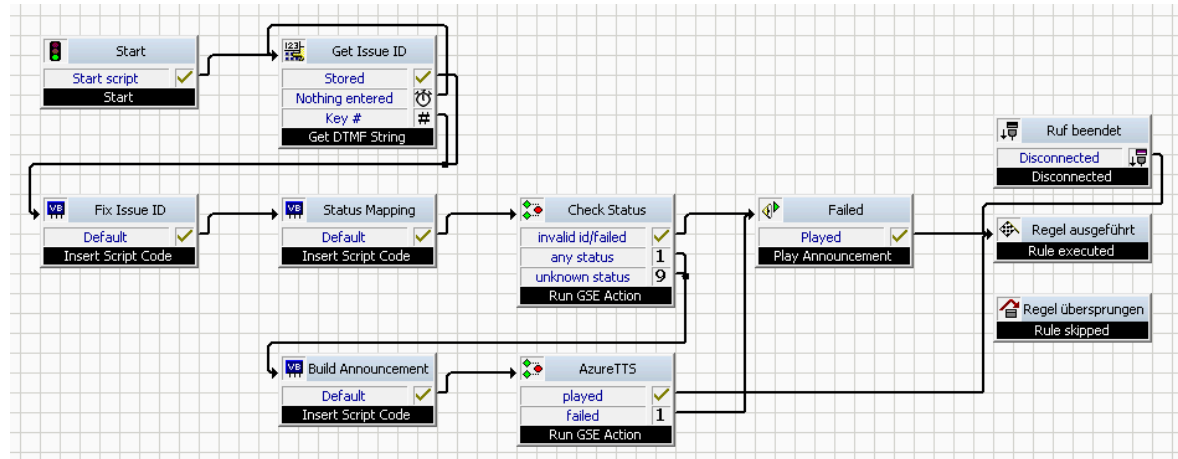
A.4 - Example: Check Status and Add Comment

A.5 - Example: Check and Announce Status

APPENDIX B

B.1 - Access/modify the code behind

B.2 - Version History



To install it:

1. Open the **Call Routing Manager** of your desired SwyxWare user.
2. Click the "New Rule..." button.
3. Select "**Graphical Script Editor**" and click **Ok**.
4. Within the GSE open the **File | Import...** menu and click **No**.
5. From the download package select the following file:

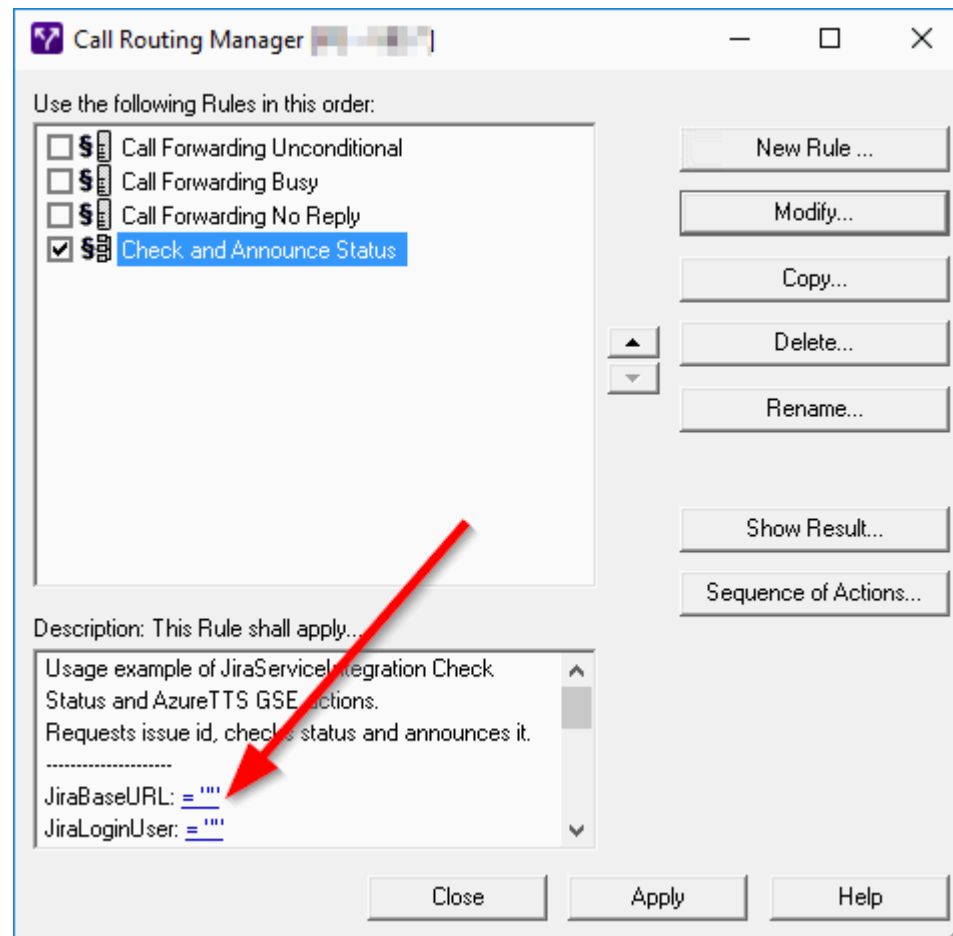
For **VBScript** usage:

```
VBScript based\rse\Check and Announce Status.rse
```

For **Lua** usage:

```
Lua based\rse\Check and Announce Status.rse
```

6. All needed configuration is done via **GSE Rule Parameters** directly in the **Call Routing Manager**.



For all **Jira...** parameters please refer to [Jira 3.1 - Create Issue](#) for instructions.
For all **Azure...** parameter please refer to [AzureTTS 3.1 - Usage](#) for instructions.

EnterIssueID

Name of the announcement wav file to be played when you have to enter the Jira issue id.

It should announce something like "Please enter your Jira Issue ID (the trailing number only) and finish with the # (hash) key."

Default: *beep.wav*

StatusAnnouncement

The text of the status announcement. You can make use of placeholders, to insert current values:

#ID# - the complete issue id (ProjectKey-ID)

#STATUS# - the current status of the issue

#UPDATE# - the date/time of the latest modification of the issue

Default: *The current status of issue #ID# is #STATUS# and it was modified latest at #UPDATE#*

FailedAnnouncement

Name of the announcement wav file to be played if either the Jira or the Azure request fails.

Please refer to [Jira 3.4 - Trouble Shooting](#), or [AzureTTS 3.2 - Trouble Shooting](#), for information on how to figure what went wrong in detail.

Default: *beep.wav*

Hint:

It makes a lot of sense to use the **AzureTTS (text-to-speech)** extension once to create the announcement files for **EnterIssueID** and **FailedAnnouncement**. This ensures that the entire call routing uses the same voice.

To do so, just use the [A.1 - Example: Announce Text](#) example and use an **Insert Script Code** block right behind the **Played** exit to copy the generated temporary wav file to a permanent location.

This only works with the **VBScript** version of the extension.

```
Dim fso
Set fso = CreateObject("Scripting.FileSystemObject")
fso.CopyFile g_sAzureTTS_LatestWavFile, "C:\MyFiles\"
Set fso = Nothing
UseExit = 0
```



By Tom Wellige
September 6, 2024

 Share

[< Previous Page](#)
[A.4 - Example: Check Status and Add Comment](#)

Theme ▼

Copyright (c) 2007-2025 by Tom Wellige
Powered by Invision Community

Jira Service Integration

INTRODUCTION

0 - Introduction

PREPARATIONS

1 - Preparations

SETUP

2 - Setup GSE Actions

USAGE

3.1 - Create Issue

3.2 - Add Comment

3.3 - Check Status

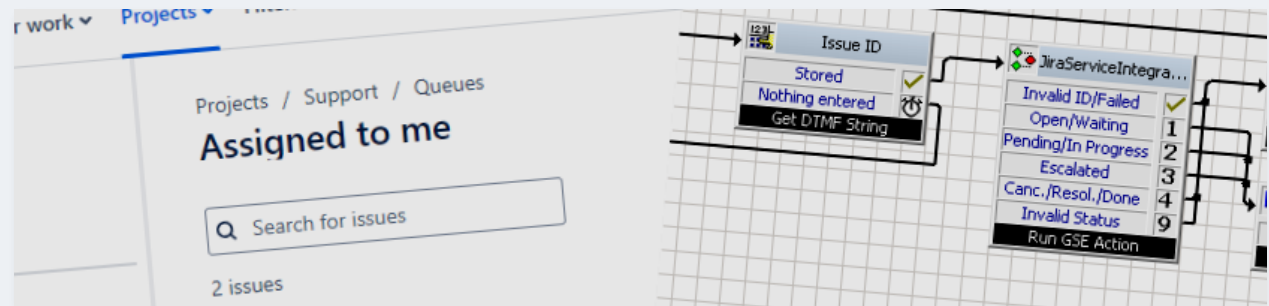
3.4 - Trouble Shooting

APPENDIX A

A.1 - Example: Create Issue

A.2 - Example: Add Comment

A.3 - Example: Check Status



Jira Service Integration - B.1 - Access/modify the code behind

Followers

0

How to make your own modifications or simply just take a look onto the implementation to get an idea of how everything works?

The following explanations will refer to the **VBScript** version, the **Lua** version however is more or less identical in its structure.

The entire code of this integration is located in one single file, **JiraServiceIntegration.vbs**.

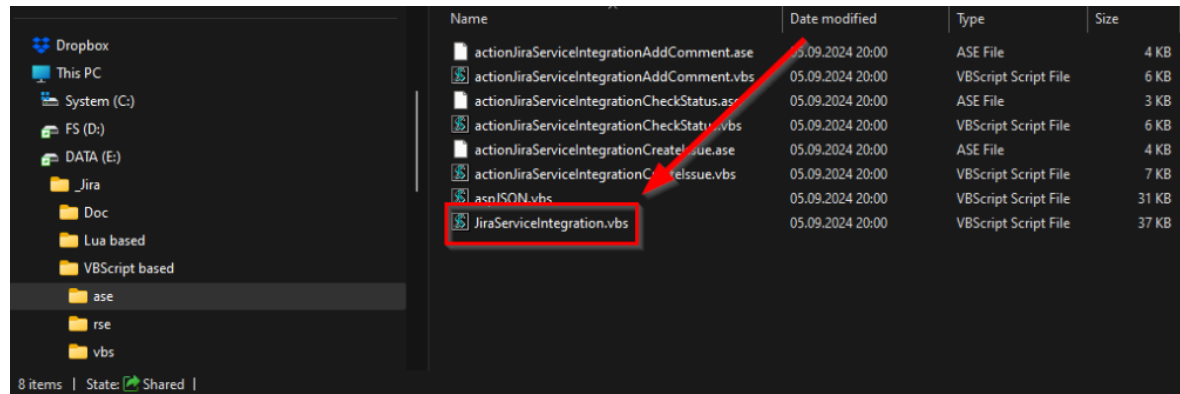
A.4 - Example: Check Status and Add Comment

A.5 - Example: Check and Announce Status

APPENDIX B

B.1 - Access/modify the code behind

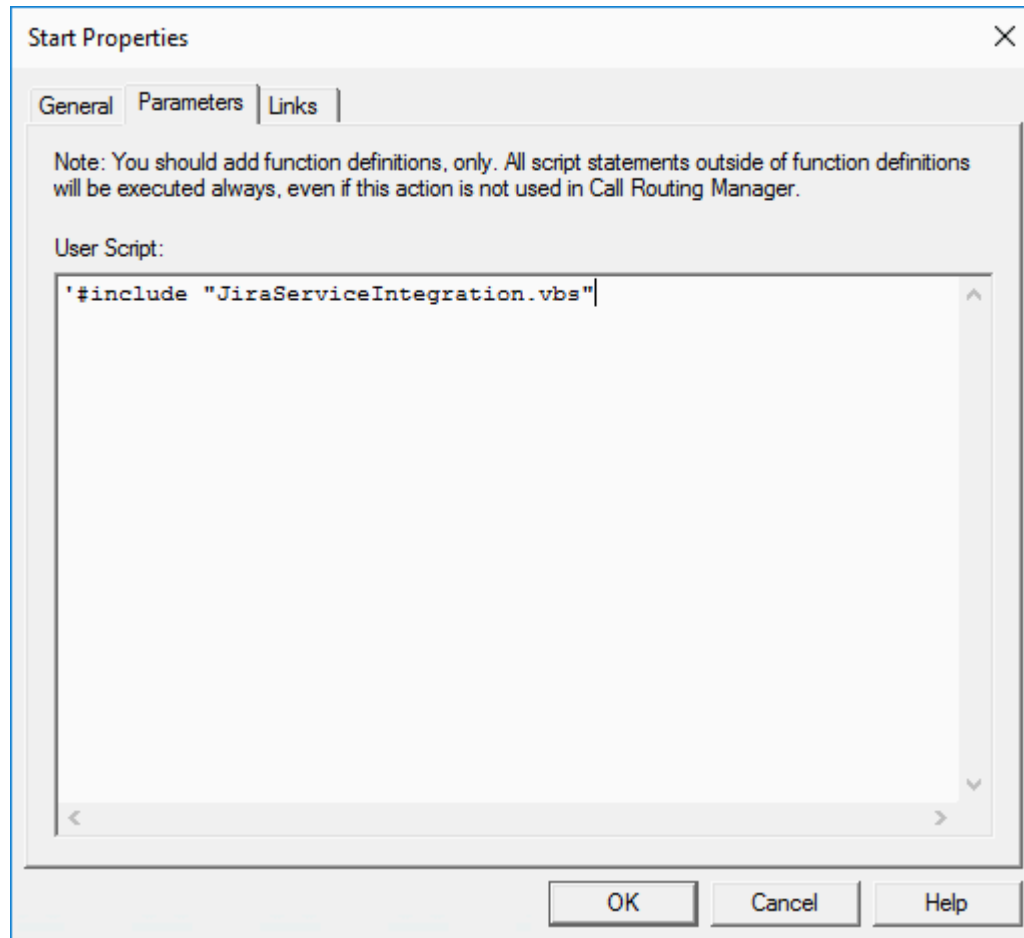
B.2 - Version History



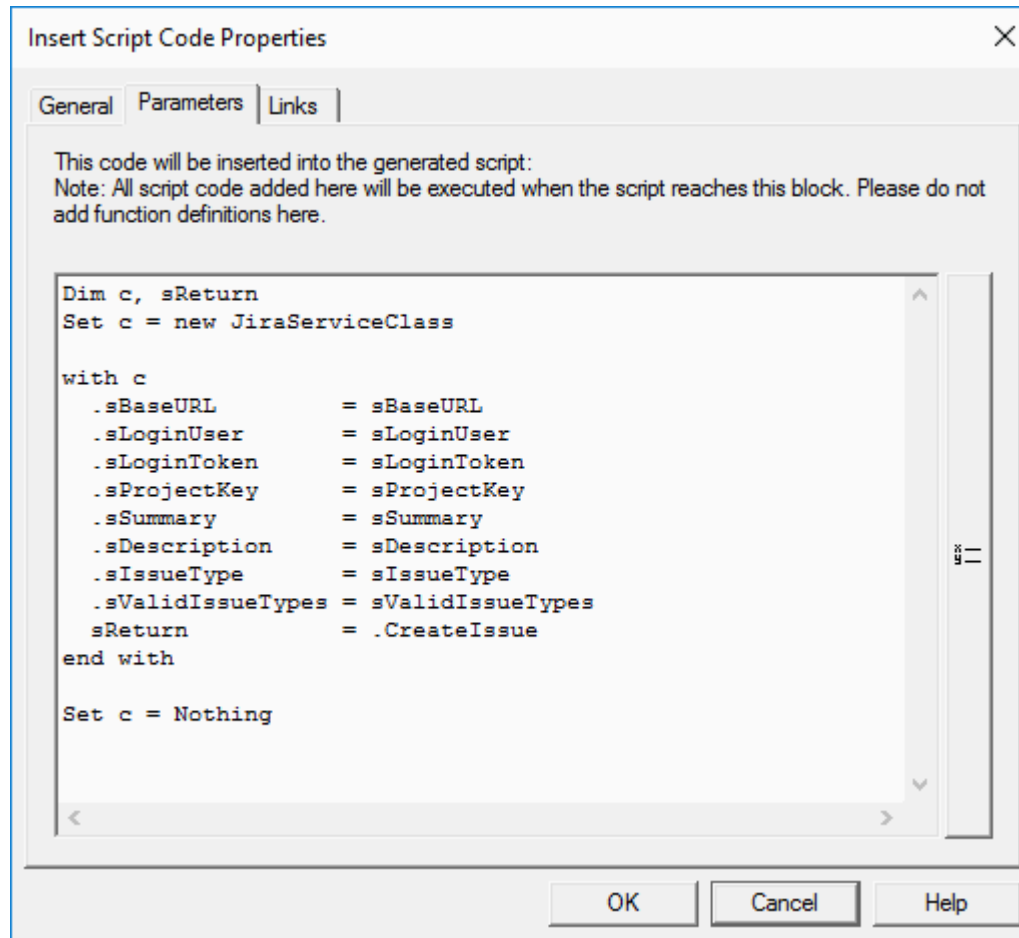
All functionality is encapsulated in a class (JiraServiceClass), which defines properties for all parameters (with some validation) as also private and public methods.

To make use of the JiraServiceClass class the **JiraServiceIntegration.vbs** file must be included first. This is done in the "**Start**" block of the GSE Action or your own GSE rule.

```
'#include "JiraServiceIntegration.vbs"
```



Afterwards the class needs to get instantiated, its properties set and the needed public method/function called. The return value of the function is either "0" (ok) or "1" (failed), so it can be used directly for the exits of a "**Insert Script Code**" or "**Run GSE Action**" block.



And this is exactly what the GSE actions of this project are doing.

In order to make modifications it is not simply possible to modify the JiraServiceIntegration.vbs file and re-upload it again into the SwyxWare database. This is because the file is digitally signed, which is necessary to get it included with the **#include** statement.

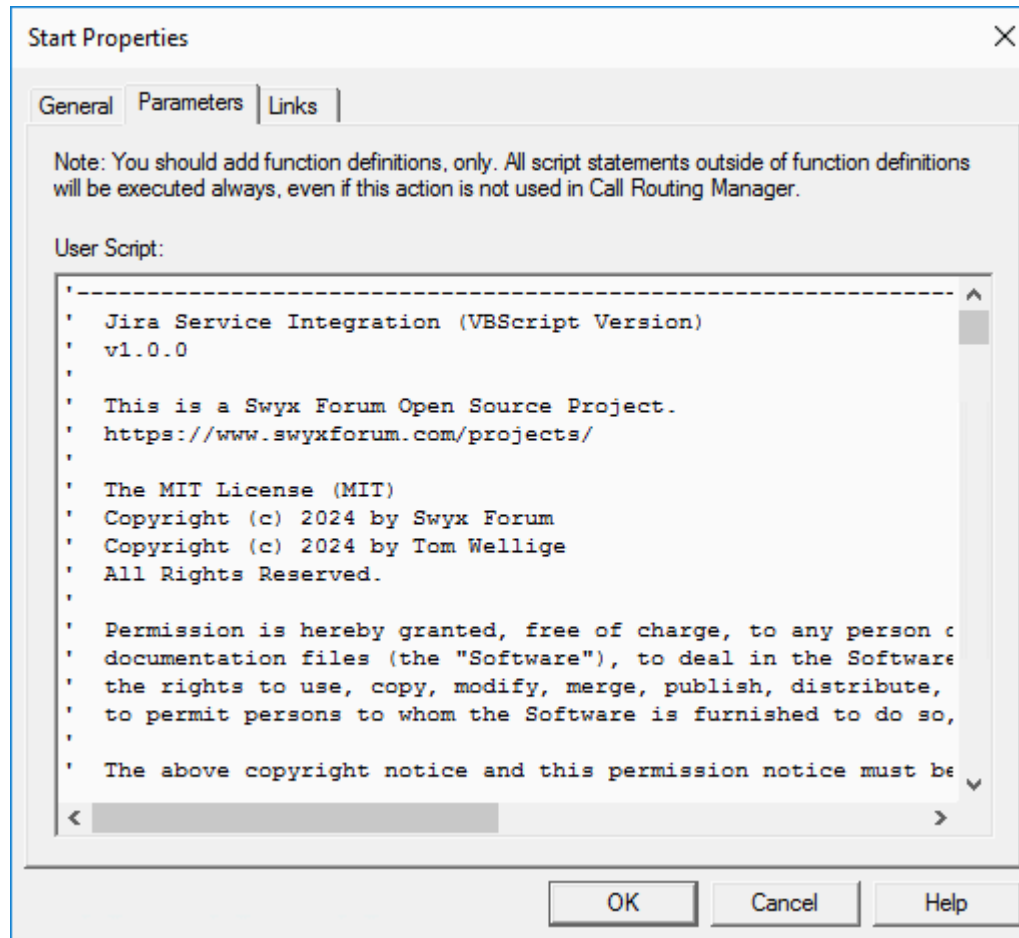
The easiest way to do your own modifications is to forget about the JiraServiceIntegration.vbs file all together and use just its content within a start block of a GSE rule (instead of the

#include statement), but without the digital signature.

You can strip the signature yourself from the code (first line and all lines from the bottom), or simply use the content of the start.vbs file from the "vbs" folder or the download package.

```
VBScript based\vbs\start.vbs
```

So just do all your modifications within the **start.vbs** file and afterwards just **copy & paste** the entire content of the file into the "**Start**" block



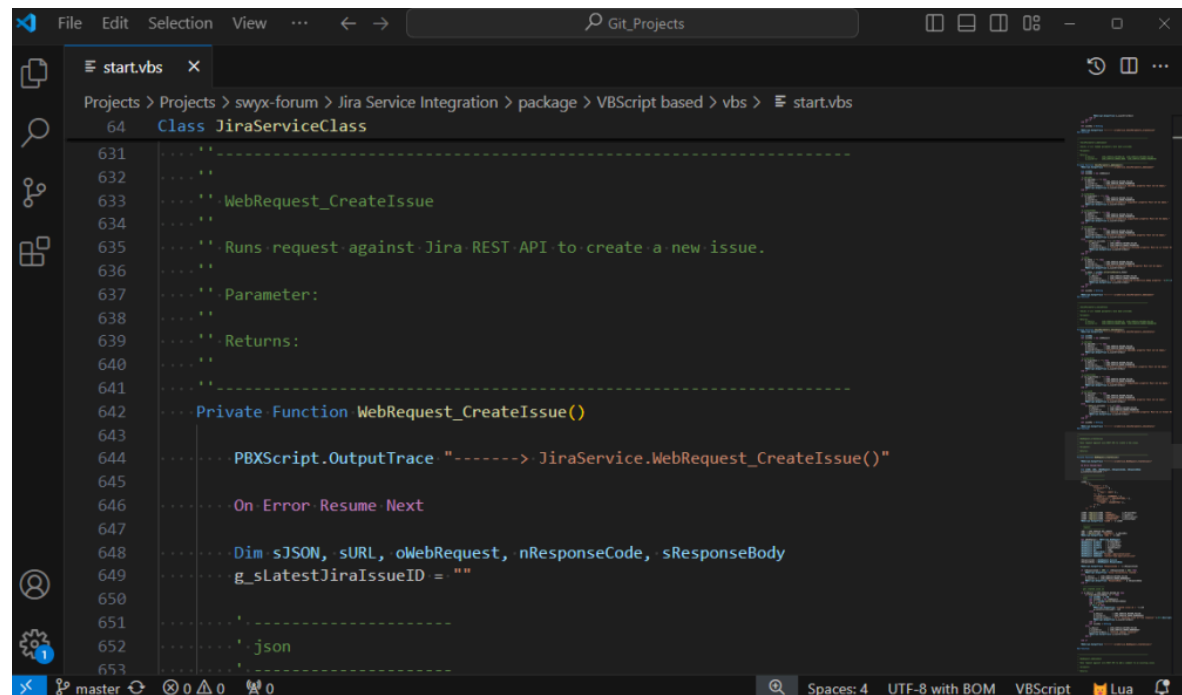
and then use the Insert "**Insert Script Code**" block to instantiate the IPS class and call its functionality

```
Dim c
Set c = new JiraServiceClass
with c
    .sBaseURL          = "https://my-service.atlassian.net"
    .sLoginUser        = "myname@gmail.com"
```



```
.sLoginToken      = "1234567890"  
.sProjectKey      = "SUP"  
.sSummary         = "My first issue via Jira REST API"  
.sDescription     = "Hello world"  
.sIssueType       = "Support"  
.sReturn          = .CreateIssue  
end with  
Set c = Nothing
```

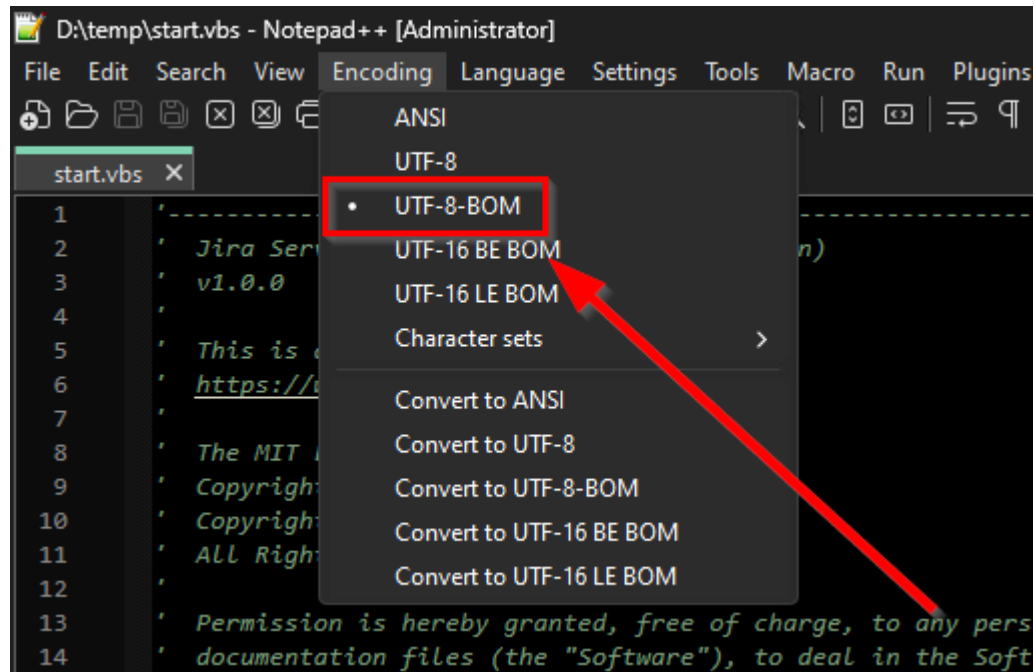
It is highly recommended to not do any code modifications directly in the **Start** block, but instead use a proper editor like [VS Code](#) or [Notepad++](#).



You are now free to to any modifications within the code and use it afterwards in your own GSE rules.

If you want to update the JiraServiceIntegration.vbs in the end, you need to update its signature as well. This can be done with the "**SignScript**" tool which can be found in the [Enreach Partner Net](#) (you need a partner login).

In order to get your own .vbs file signed, make sure that the text encoding is **UTF-8-BOM**. This can easily be checked/configured with Notepad++.



Please feel free to ask any questions regarding the code, implementation or distribution (signing, including, etc.) in the [Open ECR Extensions forum](#).

Most of the above mentioned is also true for the following extensions. So there is lots more to explore 😊

- [Invision Power Services \(IPS\) Integration](#) (REST API usage, API Key/Token authentication)
- [Longest Waiting](#) (Database usage)
- [Open Queue](#) (Database usage)
- [Zendesk Integration](#) (REST API usage, API Key/Token authentication)



By Tom Wellige

September 3, 2024

 Share

Next Page 
[B.2 - Version History](#)

Theme ▼

Copyright (c) 2007-2025 by Tom Wellige
Powered by Invision Community

Jira Service Integration

INTRODUCTION

0 - Introduction

PREPARATIONS

1 - Preparations

SETUP

2 - Setup GSE Actions

USAGE

3.1 - Create Issue

3.2 - Add Comment

3.3 - Check Status

3.4 - Trouble Shooting

APPENDIX A

A.1 - Example: Create Issue

A.2 - Example: Add Comment

A.3 - Example: Check Status

Jira Service Integration - B.2 - Version History

Followers

0

Version v1.0.0

Initial Release

Version v1.1.0

This version adds the following new features:

- New global variable that holds the latest status after "Check Status" call
- New global variable that holds the latest modification date after "Check Status" call
- New example that asks for an issue id, checks the status and uses the [Azure TTS Open ECR Extension](#) to announce everything

A.4 - Example: Check Status and Add
Comment

A.5 - Example: Check and Announce Status

APPENDIX B

B.1 - Access/modify the code behind

B.2 - Version History



By Tom Wellige

October 27, 2024

 Share



Previous Page

B.1 - Access/modify the code behind

Theme ▼

Copyright (c) 2007-2025 by Tom Wellige

Powered by Invision Community