ReDirect Version 1

Description

ReDirect facilitates implementing a manual intervention point for redirecting a job in a flow from an external web application. Our use case for this is the development of a *Problem Job Dashboard* to allow our production personnel, who have no access to Switch, to resolve issues with a job that could not be handled within the flow programmatically.

There are two webhooks, one to inquire about held jobs and the other to redirect each job to one of the available outgoing connections. The inquiry response can pass additional job information back to the inquiring web application. The redirect request can pass back one or more items to be added to the job's private data.

Compatibility

Switch 2021 Spring and higher.

Connections

At least one incoming connection required. Multiple output connections are allowed, and this list is passed back to the inquiring web application.

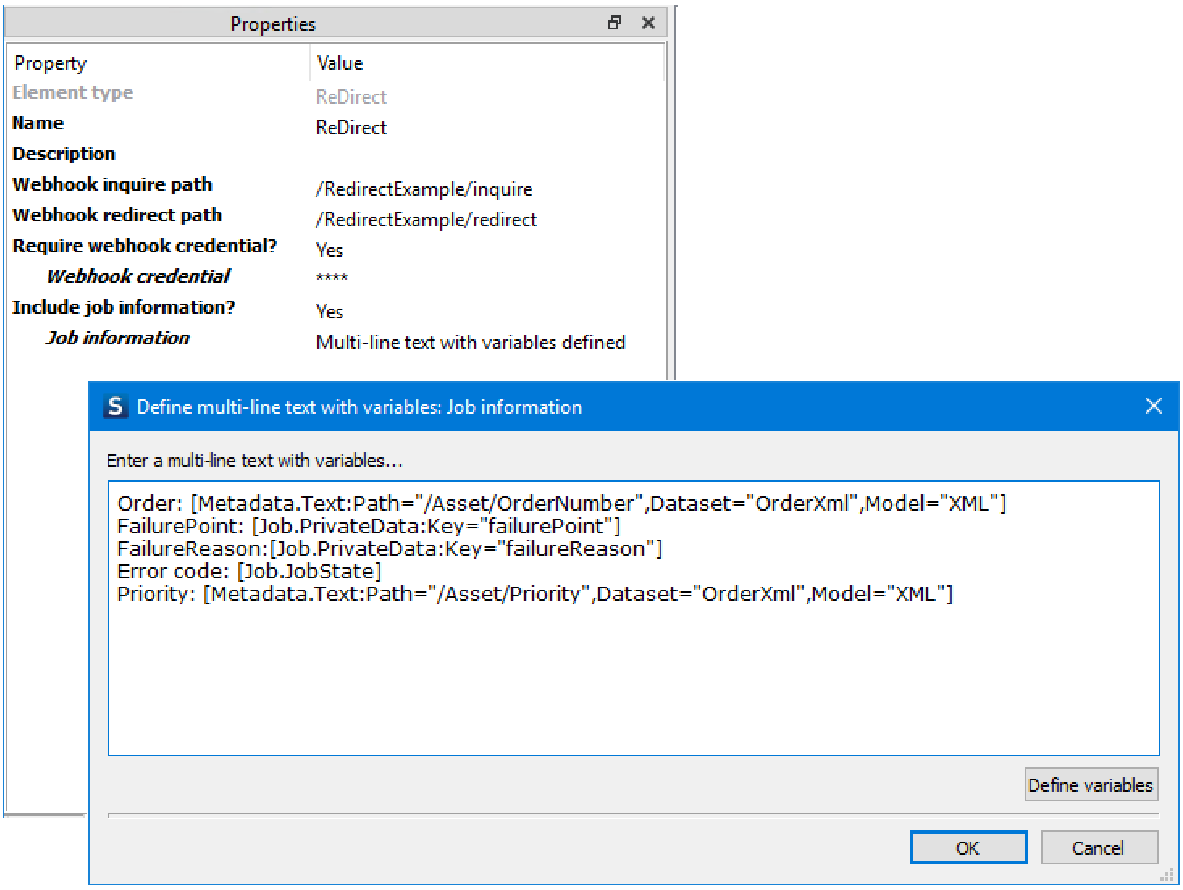
Properties detailed info

An *inquire* webhook provides the ability to query the app for a list of queued jobs and available outgoing connections returned as a JSON response. The *redirect* webhook expects a JSON request containing an array of redirects objects that include the job id, connection id and, optionally, an array of *tag:value* pairs to be added to the job's private data.

You can protect access to these webhooks by enabling the *Require webhook credential* and providing a value in *Webhook credential*. This credential then must be passed in a header named "password" by the external web site.

Enabling *Include job information* allows you to configure additional information that will be passed back in the JSON response of the *inquire* webhook. This is a *multi-line text with variables* editor. Place each ***tag:value*** pair on a separate line. Each pair will be transformed into a ***"key":"value"*** element in the JSON response within a *jobinformation* element.

Example multi-line text entries:



Property	Value
Element type	ReDirect
Name	ReDirect
Description	
Webhook inquire path	/RedirectExample/inquire
Webhook redirect path	/RedirectExample/redirect
Require webhook credential?	Yes
Webhook credential	****
Include job information?	Yes
Job information	Multi-line text with variables defined

Enter a multi-line text with variables...

```
Order: [Metadata.Text:Path="/Asset/OrderNumber",Dataset="OrderXml",Model="XML"]
FailurePoint: [Job.PrivateData:Key="failurePoint"]
FailureReason:[Job.PrivateData:Key="failureReason"]
Error code: [Job.JobState]
Priority: [Metadata.Text:Path="/Asset/Priority",Dataset="OrderXml",Model="XML"]
```

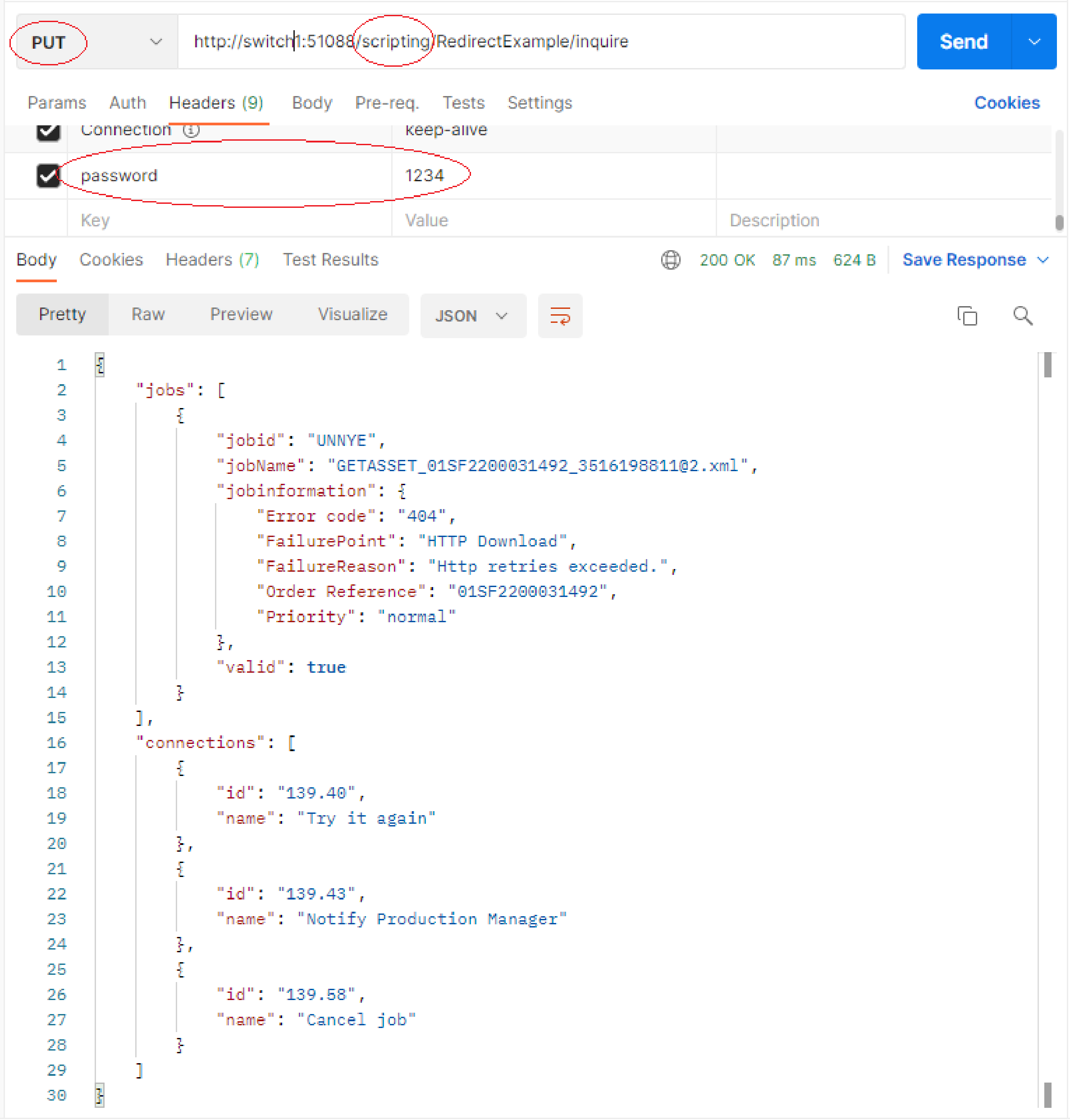
Resulting JSON response:

```
{
    "jobs": [
        {
            "jobid": "UNNYD",
            "jobName": "GETASSET_01SF2200031492_3516198811@1.xml",
            "jobinformation": {
                "Error code": "404",
                "FailurePoint": "HTTP Download",
                "FailureReason": "Http retries exceeded.",
                "Order": "01SF2200031492",
                "Priority": "normal"
            }
        }
    ],
    "connections": [
        { "id": "139.40", "name": "Try it again" },
        { "id": "139.43", "name": "Notify Production Manager" },
    ]
}
```

Implementing the webhooks

Switch webhook requests must be sent to the host name or IP address of your Switch server and use the port identified as the ***Port for the switch Web Service*** in the *Web Services* section of *Preferences*. Also, *and this is important*, you must add **/scripting** to the beginning of the paths that are configured for the two webhooks in the request URL. Note that you should use the protocol (http or https) based on the ***Protocol*** setting in the *Web Services* preference.

Below is an example request and response from Postman for the ***inquiry*** webhook:



PUT http://switch1:51088/scripting/RedirectExample/inquire

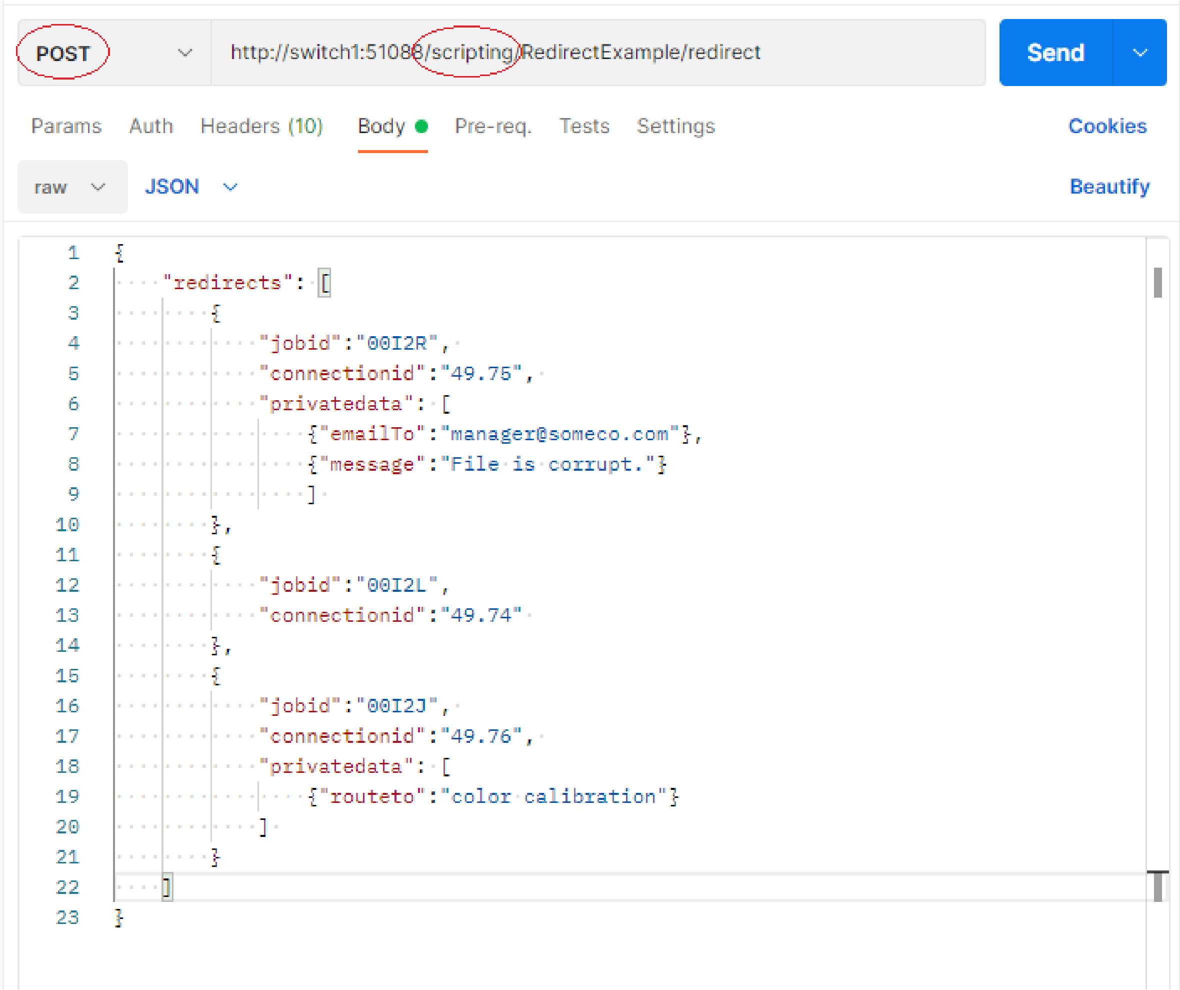
☑ Connection: keep-alive

☑ password: 1234

200 OK 87 ms 624 B

```
{
    "jobs": [
        {
            "jobid": "UNNYE",
            "jobName": "GETASSET_01SF2200031492_351619881102.xml",
            "jobinformation": {
                "Error code": "404",
                "FailurePoint": "HTTP Download",
                "FailureReason": "Http retries exceeded.",
                "Order Reference": "01SF2200031492",
                "Priority": "normal"
            },
            "valid": true
        }
    ],
    "connections": [
        {
            "id": "139.40",
            "name": "Try it again"
        },
        {
            "id": "139.43",
            "name": "Notify Production Manager"
        },
        {
            "id": "139.58",
            "name": "Cancel job"
        }
    ]
}
```

Here is a Postman example request for the ***redirect*** webhook:



http://switch1:51088/scripting/RedirectExample/redirect

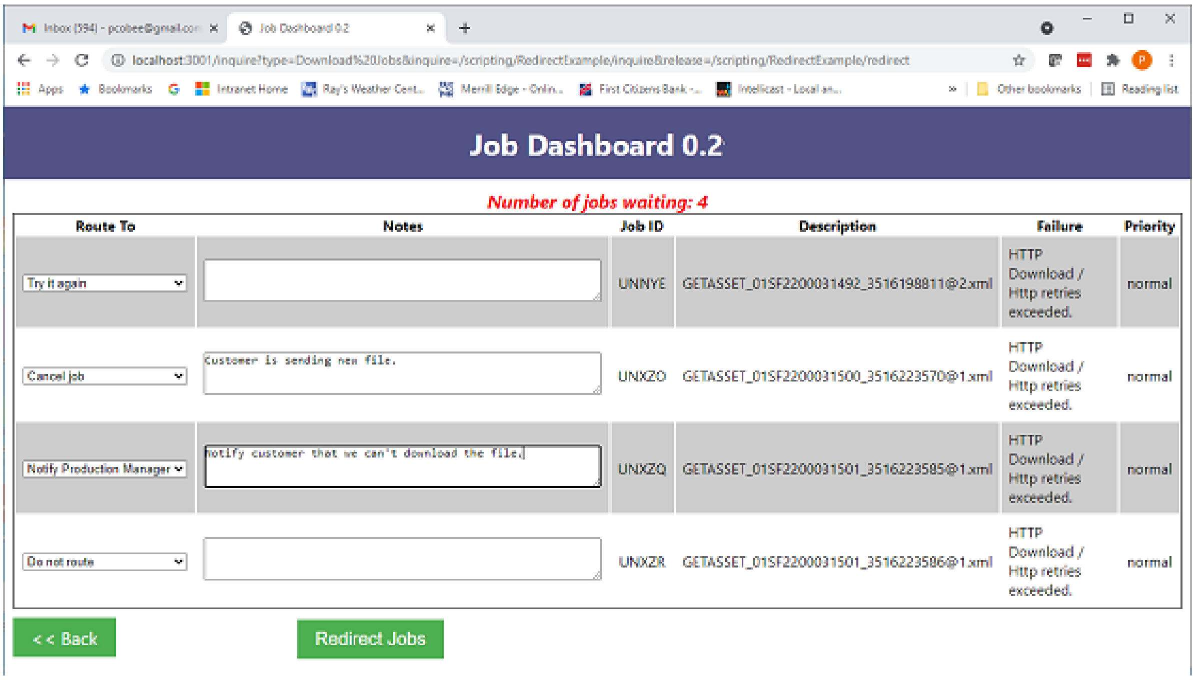
```
{
    "redirects": [
        {
            "jobid":"00I2R",
            "connectionid":"49.75",
            "privatedata": [
                {"emailTo":"manager@someco.com"},
                {"message":"File is corrupt."}
                ]
        },
        {
            "jobid":"00I2L",
            "connectionid":"49.74"
        },
        {
            "jobid":"00I2J",
            "connectionid":"49.76",
            "privatedata": [
                {"routeto":"color calibration"}
            ]
        }
    ]
}
```

- job *00I2R* is being routed to the connection identified as *49.75* and includes two private data items – an email address and a message. This would most likely be used when redirecting to a *Mail send* flow element to ask for assistance with the file.

- job *00I2L* is just routing the file to the *49.74* connection with no private data necessary. Maybe we just want to cancel this job and it's routed to a *Recyle bin*.

- job *00I23* is being routed to the *49.76* connection and includes a ***routeto*** privatedata item. ***routeto*** could be used downstream for further routing of the job.

You can download a sample Node.js server app from my web site. It is based on *Express* and *Pug*…



Inbox (594) - pcobee@gmail.com
Job Dashboard 0.2
localhost:3001/inquire?type=Download%20Jobs&inquire=/scripting/RedirectExample/inquire&release=/scripting/RedirectExample/redirect
Apps
Bookmarks
Intranet Home
Ray's Weather Cent...
Merrill Edge - Onlin...
First Citizens Bank -...
Intellicast - Local an...
Other bookmarks
Reading list

Job Dashboard 0.2

Number of jobs waiting: 4

Route To	Notes	Job ID	Description	Failure	Priority
Try it again		UNNYE	GETASSET_01SF2200031492_3516198811@2.xml	HTTP Download / Http retries exceeded.	normal
Cancel job	Customer is sending new file.	UNXZO	GETASSET_01SF2200031500_3516223570@1.xml	HTTP Download / Http retries exceeded.	normal
Notify Production Manager	Notify customer that we can't download the file.	UNXZQ	GETASSET_01SF2200031501_3516223585@1.xml	HTTP Download / Http retries exceeded.	normal
Do not route		UNXZR	GETASSET_01SF2200031501_3516223586@1.xml	HTTP Download / Http retries exceeded.	normal

Expand the contents of the zip file into a directory on your system. Install Node.js if you haven't already done so (https://nodejs.org/en/download/).

Only a few minor changes should be needed:

- edit the ***app.js*** file and update *switchPassword* to match the password credential for your webhooks and change the *port* if it is in conflict with another service
- edit ***views\waitingJobs.pug*** to edit the table columns to match your job information. The JSON response is passed to *pug* so all you need to do is reference the specific items you want to display as ***td*** items. *(Note: be careful to maintain the indents as this is important to pug.)* Remember that your custom information is included in the *job.jobinformation* element so if you configured an item called *description* you would reference it as *job.jobinformation.description*.
- edit appconfig.json to set **switchHost** to match your host name or ip address. Under **jobTypes**, create an entry for each instance of *ReDirect* to be monitored. The value for *name* will appear under *Job Type* and the values for *inquire* and *release* should match the configured webhook paths preceded by ***/scripting***.

To run on Windows, open a command prompt instance, set the working directory to the ***jobDashboard*** directory (where app.js can be found) and enter ***node app.js.*** Ctrl-C will stop the server. *(I'm not sure how you do this on a Mac.)*

From a browser, type in the host name or ip address and port to access the server, examples: http://192.168.1.20:3001 or http://localhost:3001

Clicking on the link under *Inquiry* will return a list of the currently held jobs. Specify the route/connection for one or more jobs and optionally enter information into the *Notes* field. Click *Redirect Jobs* to redirect the jobs to the corresponding connection. A private data field called ***notes*** will be added to the redirected jobs.