MFG127988: Isn’t It Great When We Integrate?

Orrin Bourne

CAD/PLM Administrator
Greenpoint Technologies

Ravi Dharmalingam

Solution Architect
Autodesk

Join the conversation #AU2017

Use JavaScript to send a JSON packet to a Jitterbit Harmony API
Use Jitterbit Harmony to receive and transform data
Insert data into an on-premise SQL database via stored procedure
Pass arguments to a console application (C# Executable) and perform an action

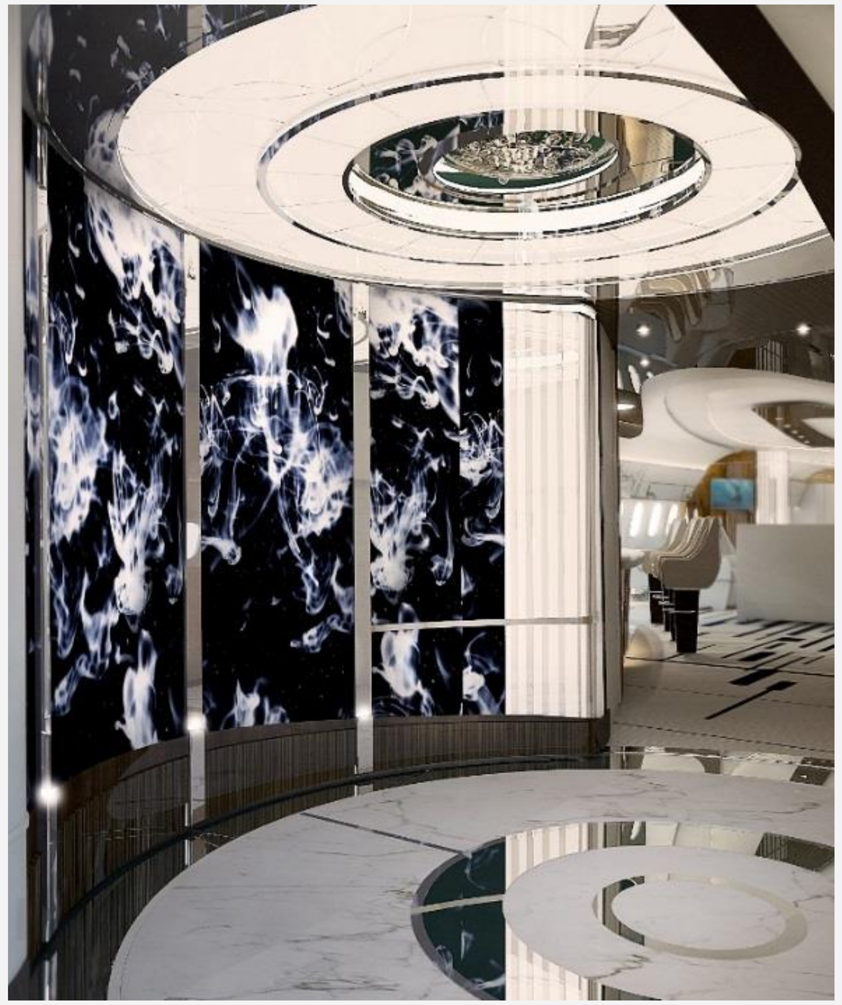
DISCLAIMER

Greenpoint Technologies, a Zodiac Aerospace company, is providing this information for educational purposes only. The presentation of the following information is to facilitate the free exchange of ideas amongst industry peers. This class and its contents should not be viewed as an endorsement of any product or company.

GREENPOINT

Artistry Engineered

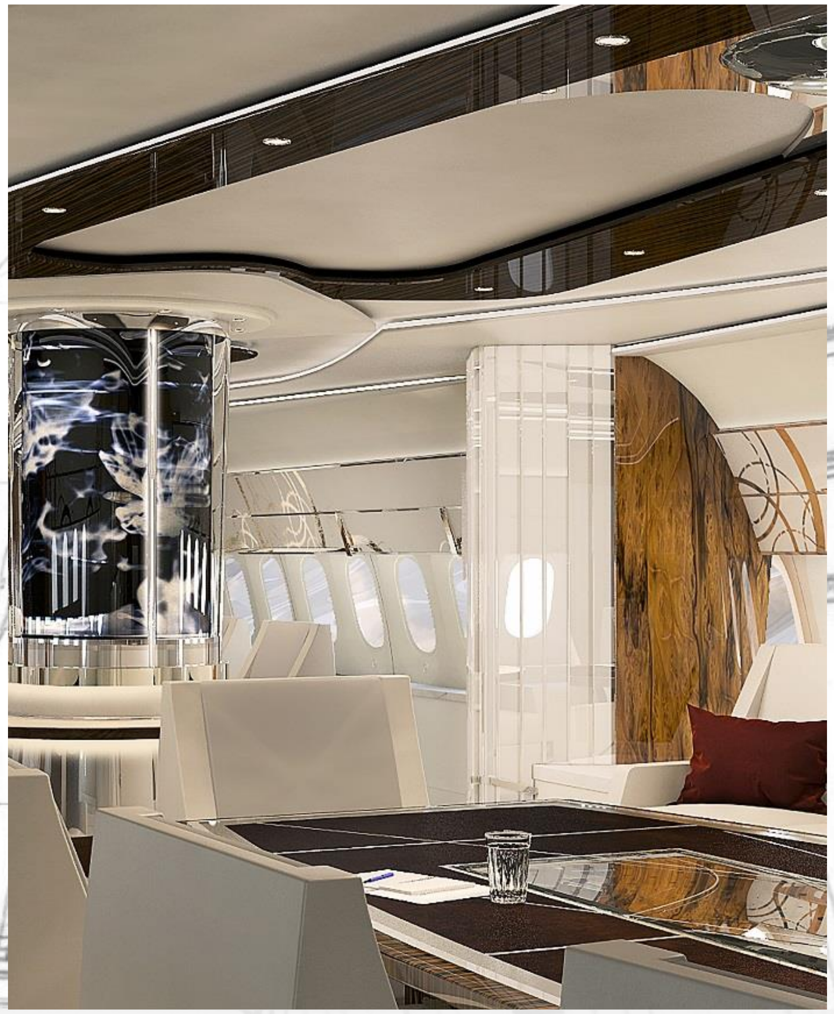
VIP 787 – *Azure* Interior



VIP 787 – *Azure* Interior

SBID Design Award 2015

IIDA InConcept Award 2014



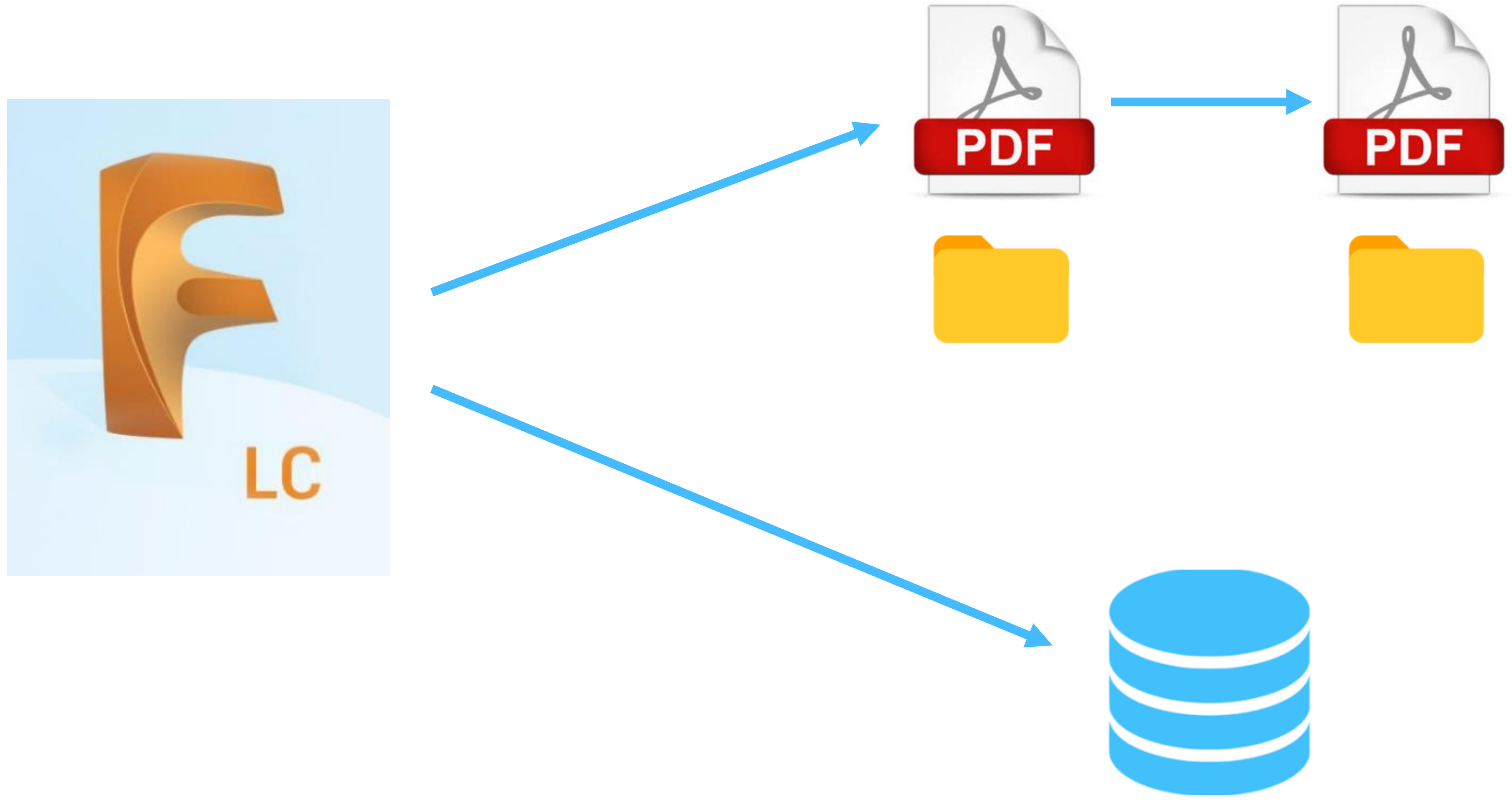
Crystal AirCruises 777-200LR

Delivered early August 2017

Crystal AirCruises 777-200LR

Sleeps up to 88 guests comfortably

Real World Business Application

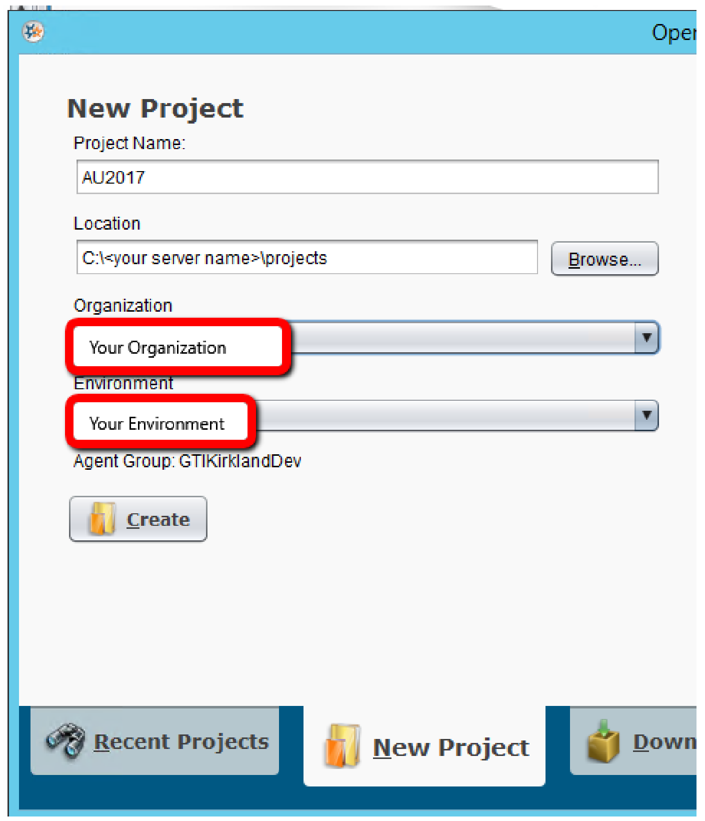


How Do We Achieve This With Fusion Lifecycle

- A triggering event that will initiate the scripting
- The script will aggregate the FLC data and send to a Jitterbit API
- Receive API request and transform the data
- Use the data to run external systems

Start In Jitterbit Studio

- Create a project
- Create an operation
- Choose a transformation type
- Give it a name



New Project

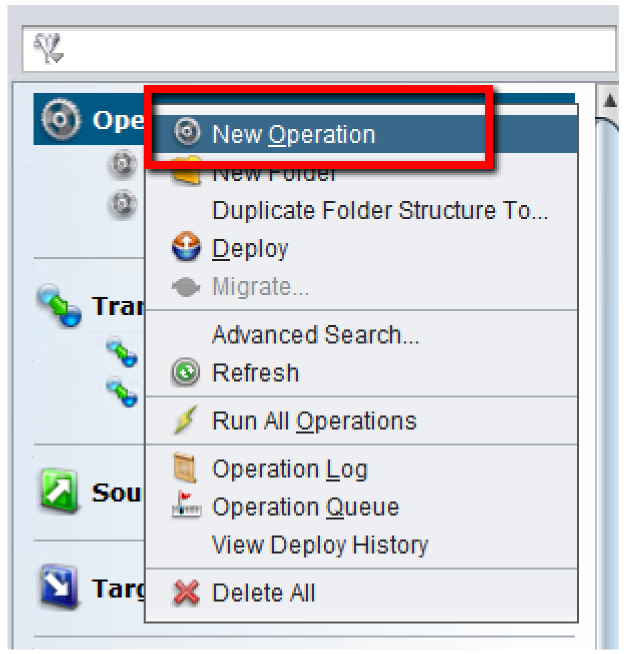
Project Name: AU2017

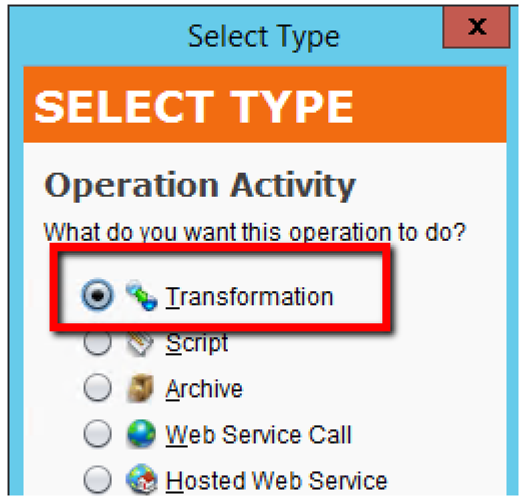
Location: C:\<your server name>\projects

Organization: Your Organization

Environment: Your Environment

Agent Group: GTIKirklandDev





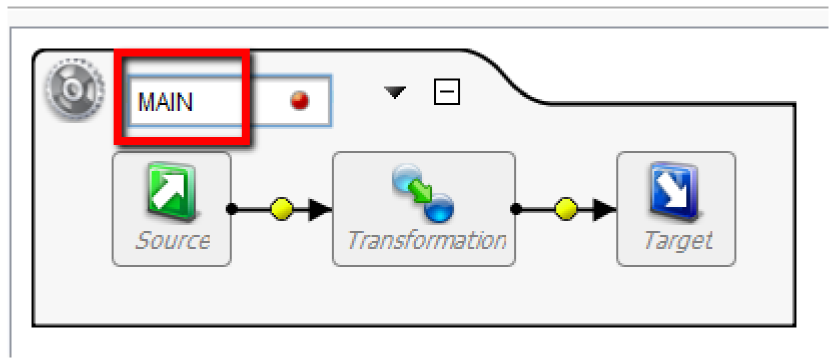
Select Type

SELECT TYPE

Operation Activity

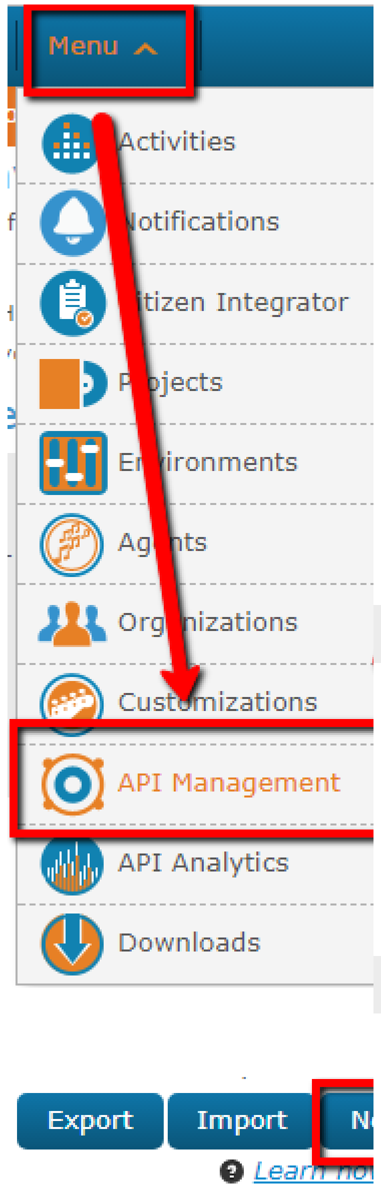
What do you want this operation to do?

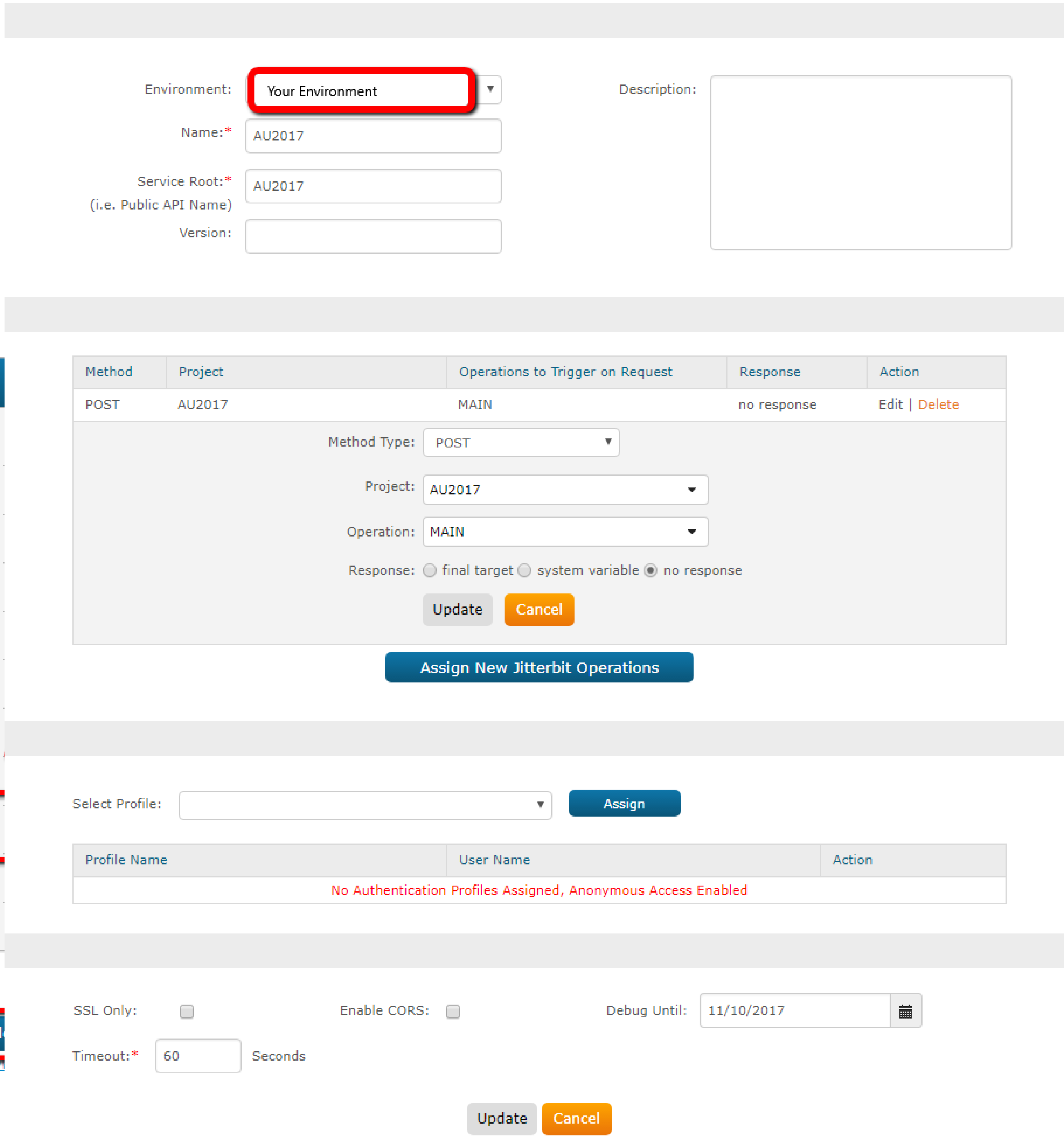
- [✓ Transformation]
- Script
- Archive
- Web Service Call
- Hosted Web Service



Create an API in Jitterbit Harmony

- Navigate to Custom API
- Select New Custom API Service
- Fill out the form
  - Method: POST
  - Operation: Your Operation





Environment:	Your Environment
Name:*	AU2017
Service Root:* (i.e. Public API Name)	AU2017
Version:	
Description:	

Method	Project	Operations to Trigger on Request	Response	Action
POST	AU2017	MAIN	no response	Edit | Delete

Method Type:	POST
Project:	AU2017
Operation:	MAIN
Response:	final target / system variable / [✓ no response]

Update Cancel

Assign New Jitterbit Operations

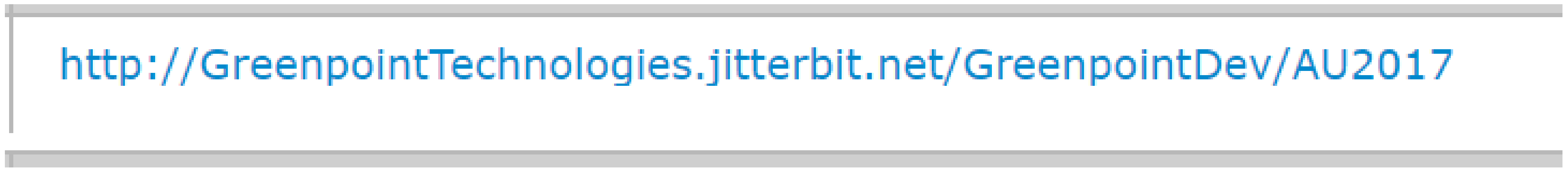
Select Profile: Assign

Profile Name	User Name	Action
No Authentication Profiles Assigned, Anonymous Access Enabled		

SSL Only:	☐	Enable CORS:	☐	Debug Until:	11/10/2017
Timeout:*	60 Seconds				

Update Cancel

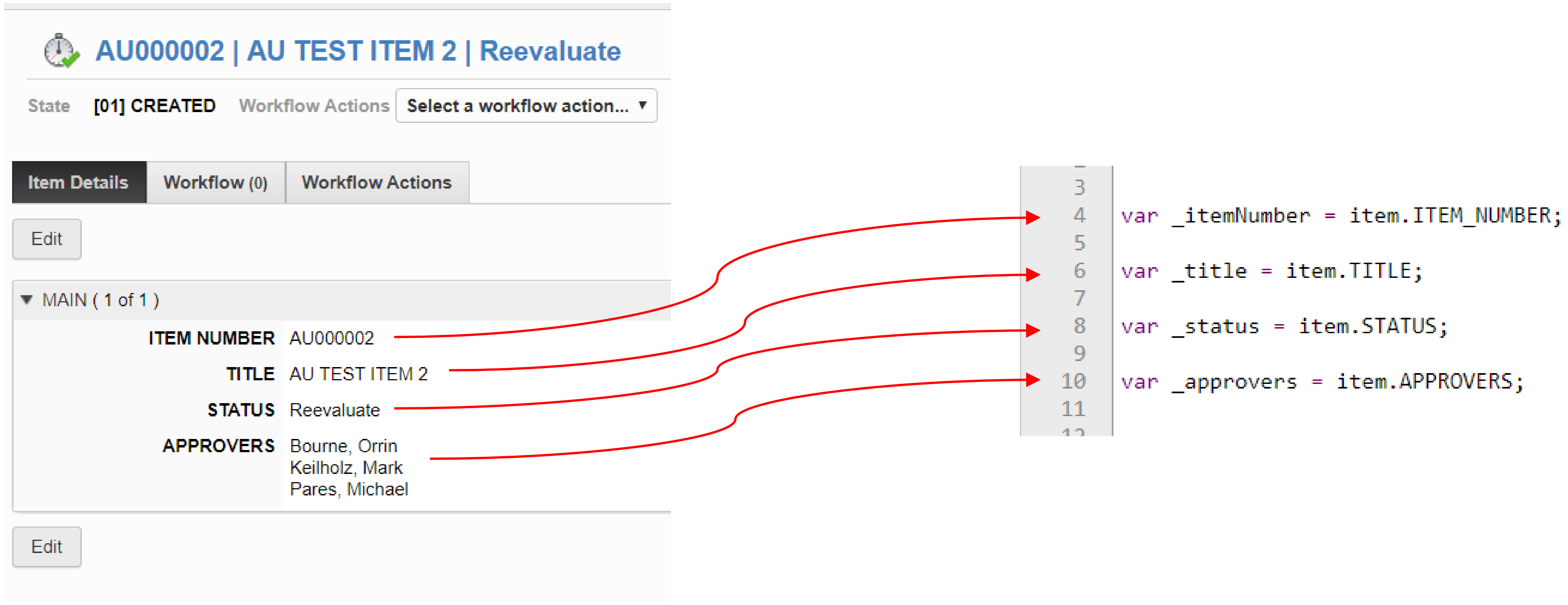
Create an API in Jitterbit Harmony (cont.)



http://GreenpointTechnologies.jitterbit.net/GreenpointDev/AU2017

Write Some Code in Fusion Lifecycle

- Collect pertinent meta data through scripting
- Consider how to handle special coding needs (i.e to arrays, linking pick lists etc.)



AU000002 | AU TEST ITEM 2 | Reevaluate

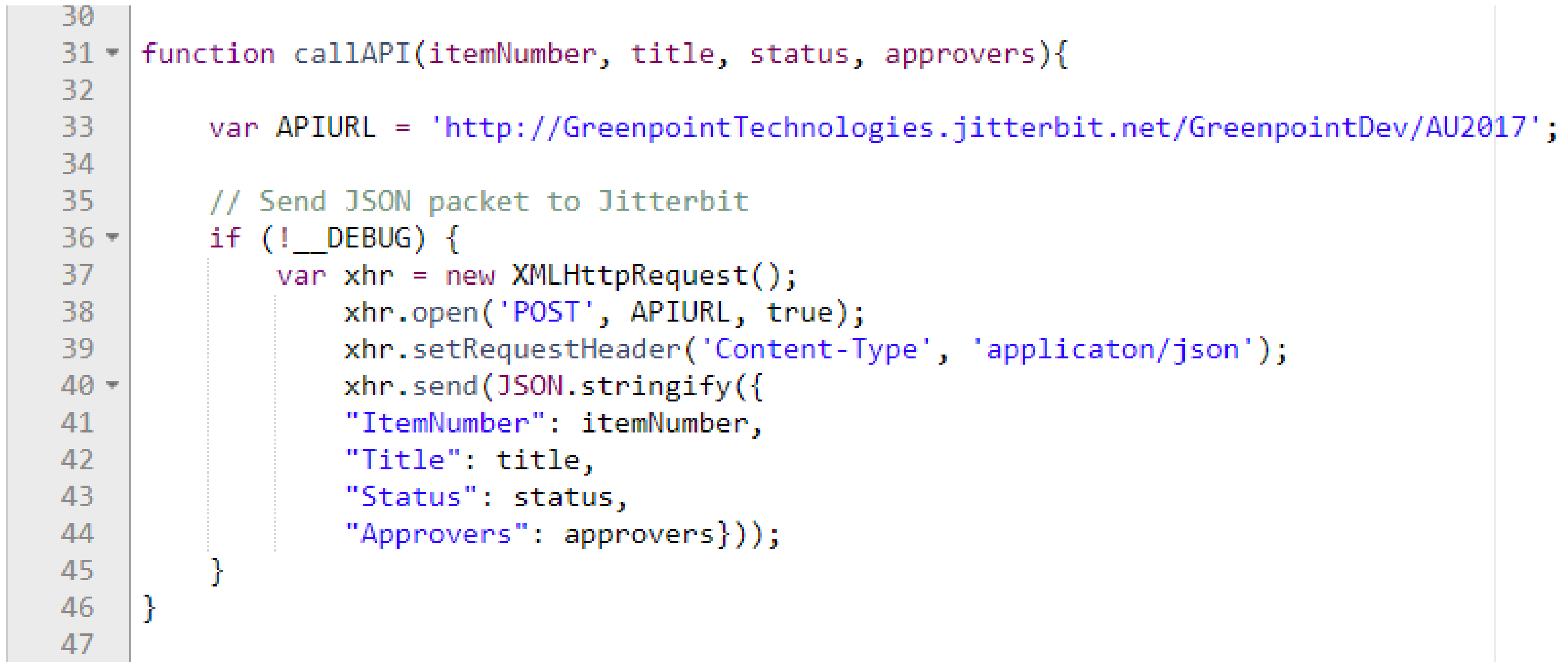
State [01] CREATED Workflow Actions Select a workflow action...

ITEM NUMBER	AU000002
TITLE	AU TEST ITEM 2
STATUS	Reevaluate
APPROVERS	Bourne, Orrin Keilholz, Mark Pares, Michael

```
var _itemNumber = item.ITEM_NUMBER;

var _title = item.TITLE;

var _status = item.STATUS;

var _approvers = item.APPROVERS;

```

Write Some Code in Fusion Lifecycle (cont.)

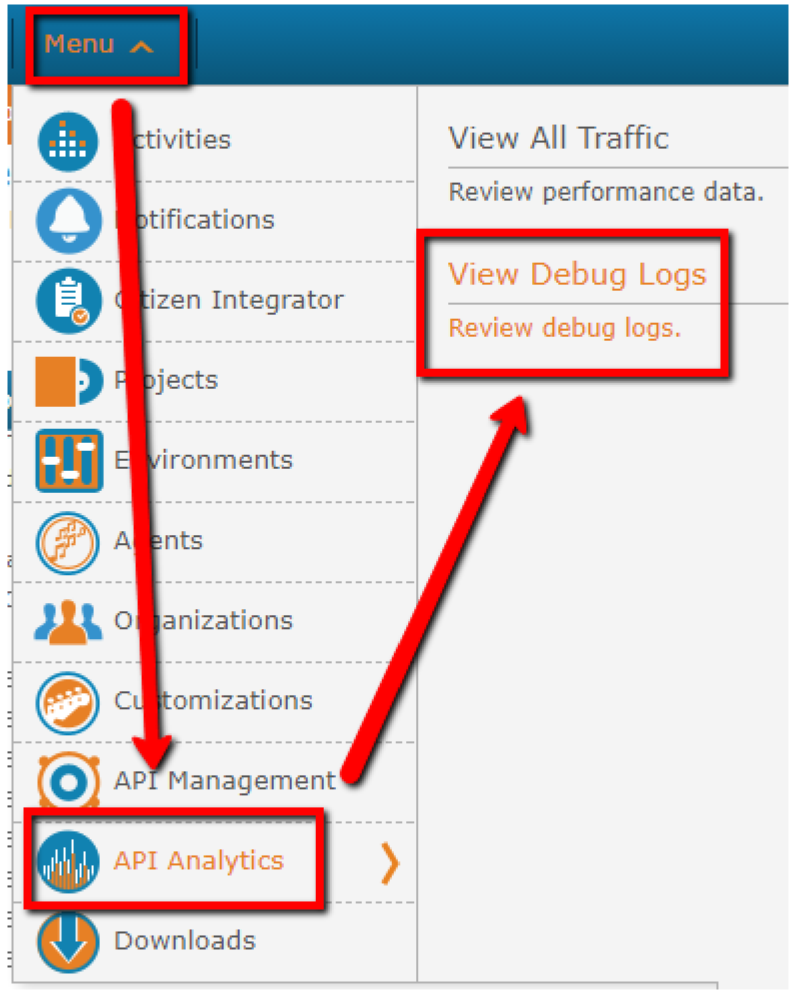
- Build the request (use simple names with an obvious meaning)
- Send JSON packet

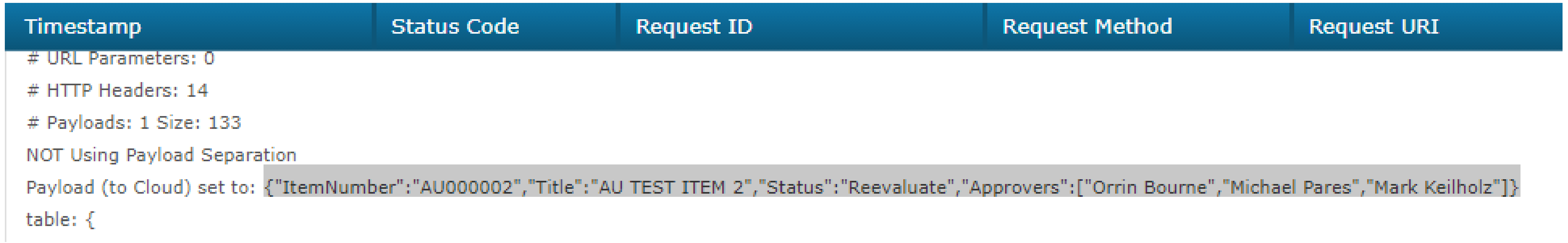


```
function callAPI(itemNumber, title, status, approvers){

    var APIURL = 'http://GreenpointTechnologies.jitterbit.net/GreenpointDev/AU2017';

    // Send JSON packet to Jitterbit
    if (!__DEBUG) {
        var xhr = new XMLHttpRequest();
            xhr.open('POST', APIURL, true);
            xhr.setRequestHeader('Content-Type', 'applicaton/json');
            xhr.send(JSON.stringify({
            "ItemNumber": itemNumber,
            "Title": title,
            "Status": status,
            "Approvers": approvers}));
    }
}
```

Review Debug Logs in Jitterbit Harmony

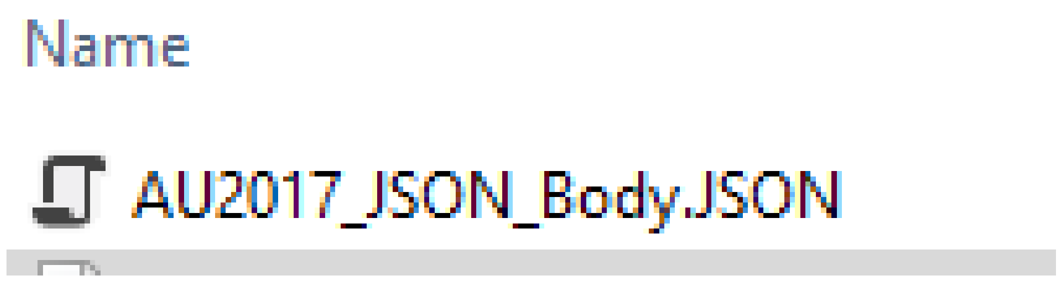
- Navigate to Debug logs
- Copy JSON body and save a text file in an accessible location
- Change extension to .JSON





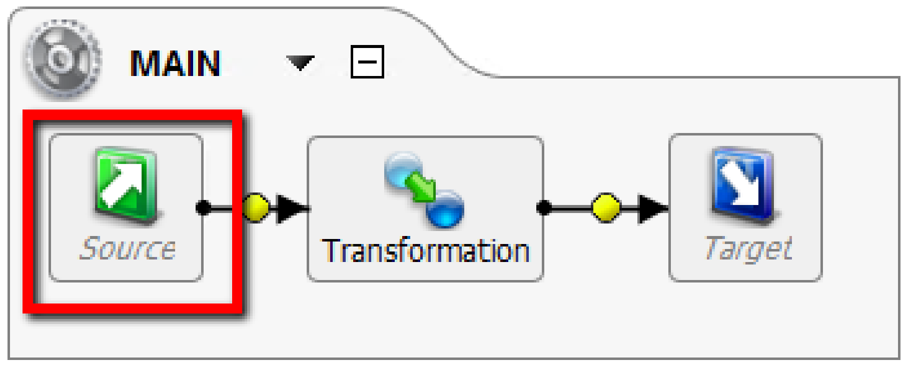
Timestamp	Status Code	Request ID	Request Method	Request URI

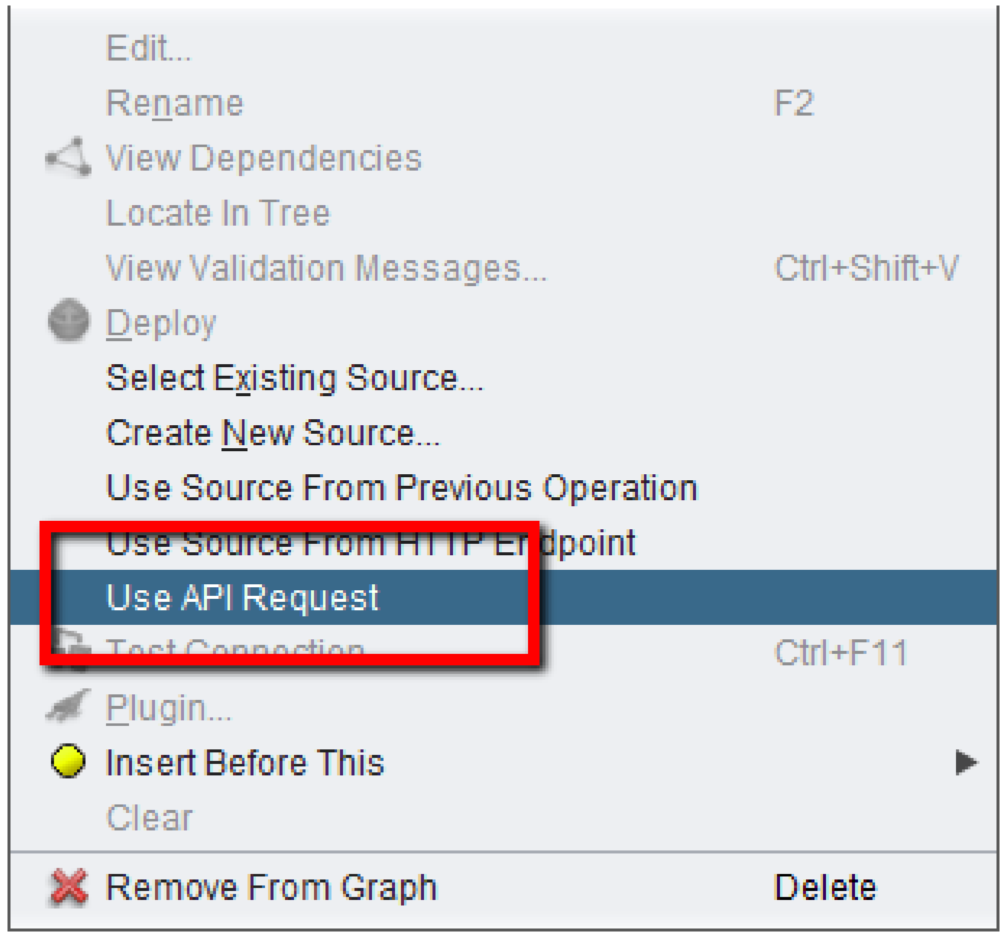
```
# URL Parameters: 0
# HTTP Headers: 14
# Payloads: 1 Size: 133
NOT Using Payload Separation
Payload (to Cloud) set to: {"ItemNumber":"AU000002","Title":"AU TEST ITEM 2","Status":"Reevaluate","Approvers":["Orrin Bourne","Michael Pares","Mark Keilholz"]}
table: {
```

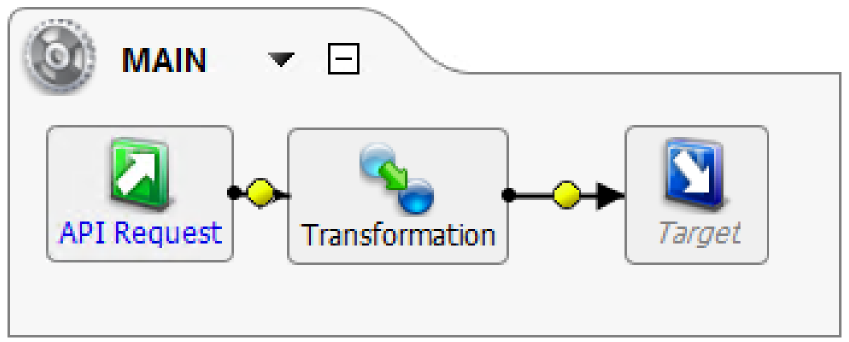


Create the Source

- RMC on the source
- Select “Use API Request”

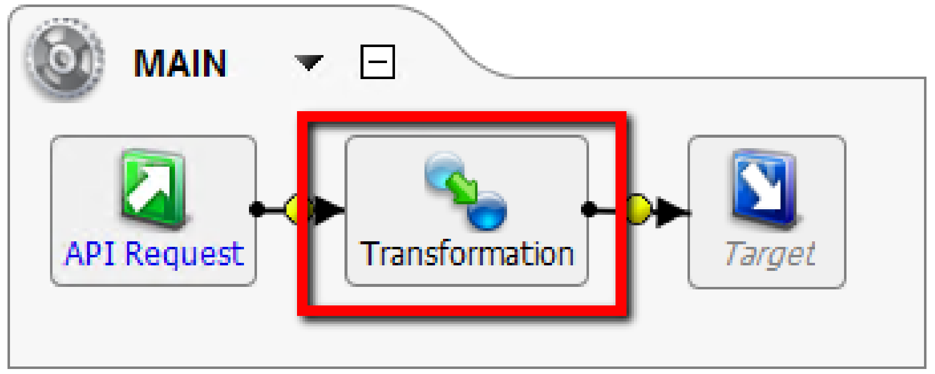


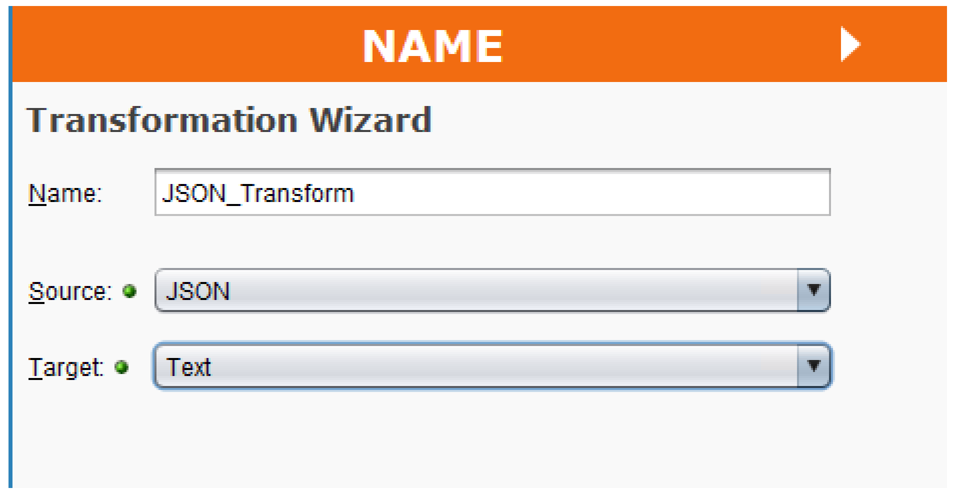




Create the Transformation

- Double click the Transformation
- Select Create New Transformation
- Set Source: JSON and Target:Text
- Select "Create a new JSON…"
- Click "NEXT"



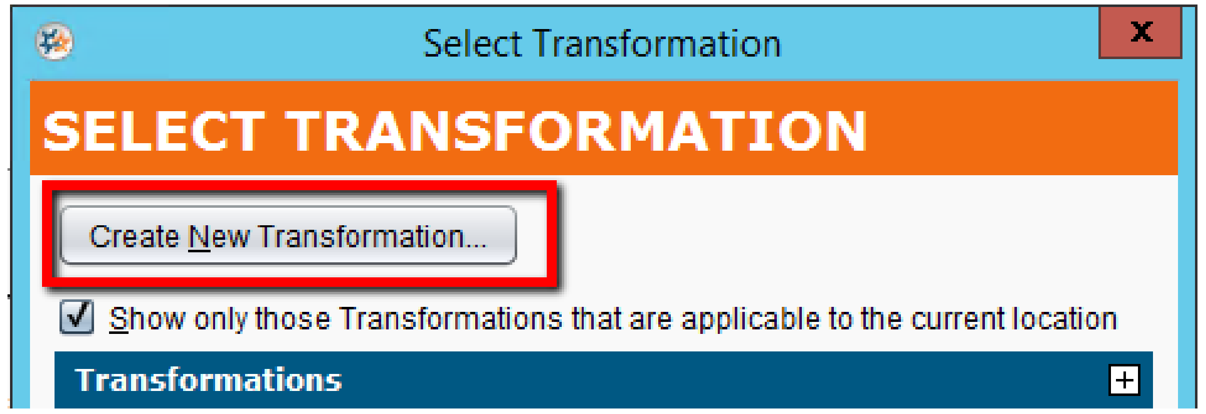

NAME

Transformation Wizard

Name: JSON_Transform

Source: JSON

Target: Text

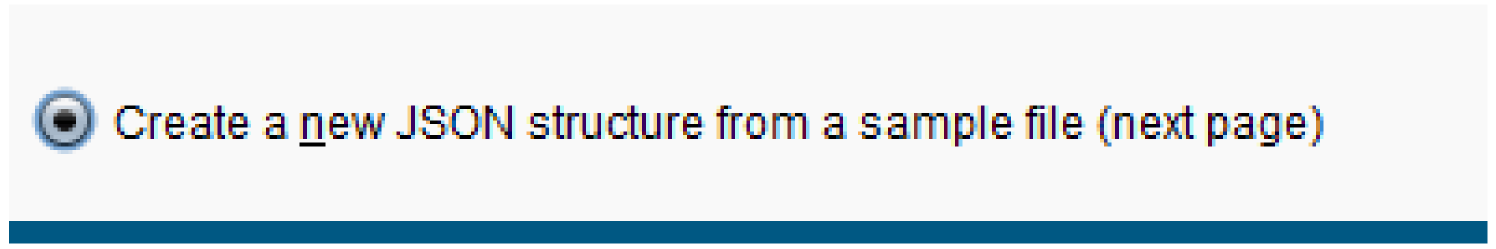

Select Transformation

SELECT TRANSFORMATION

Create New Transformation...

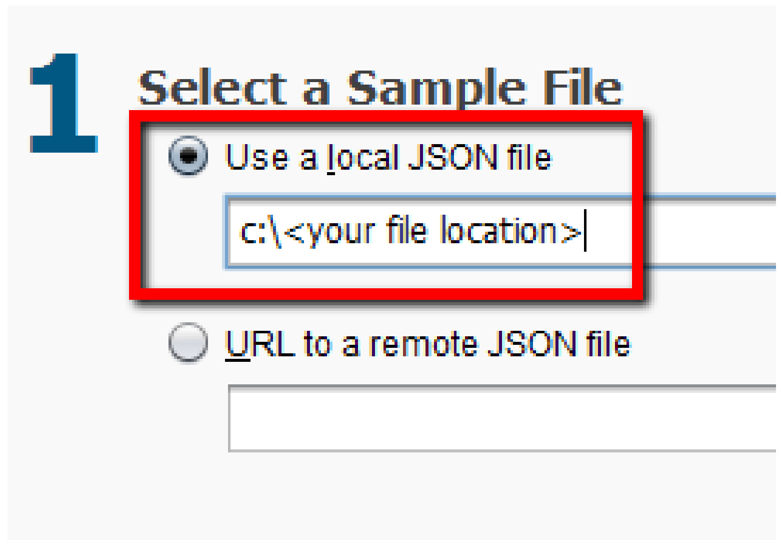
☑ Show only those Transformations that are applicable to the current location

Transformations


● Create a new JSON structure from a sample file (next page)

Define Source Data Structure

- Find the JSON file you saved earlier
- Generate the XSD file
- Review the results

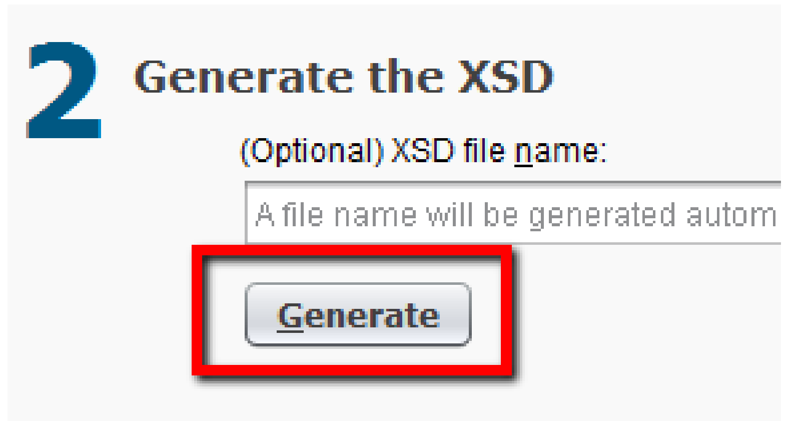


1 Select a Sample File

● Use a local JSON file

c:\<your file location>

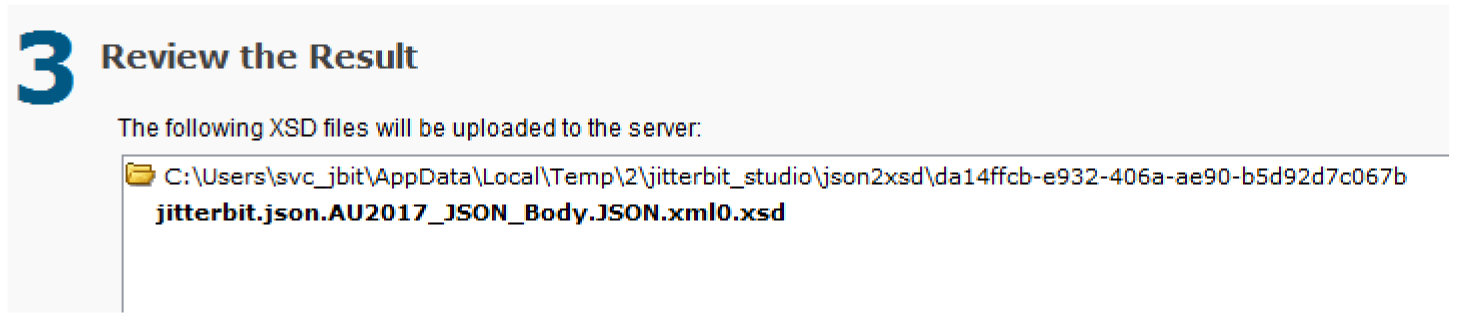
○ URL to a remote JSON file



2 Generate the XSD

(Optional) XSD file name:

A file name will be generated autom



3 Review the Result

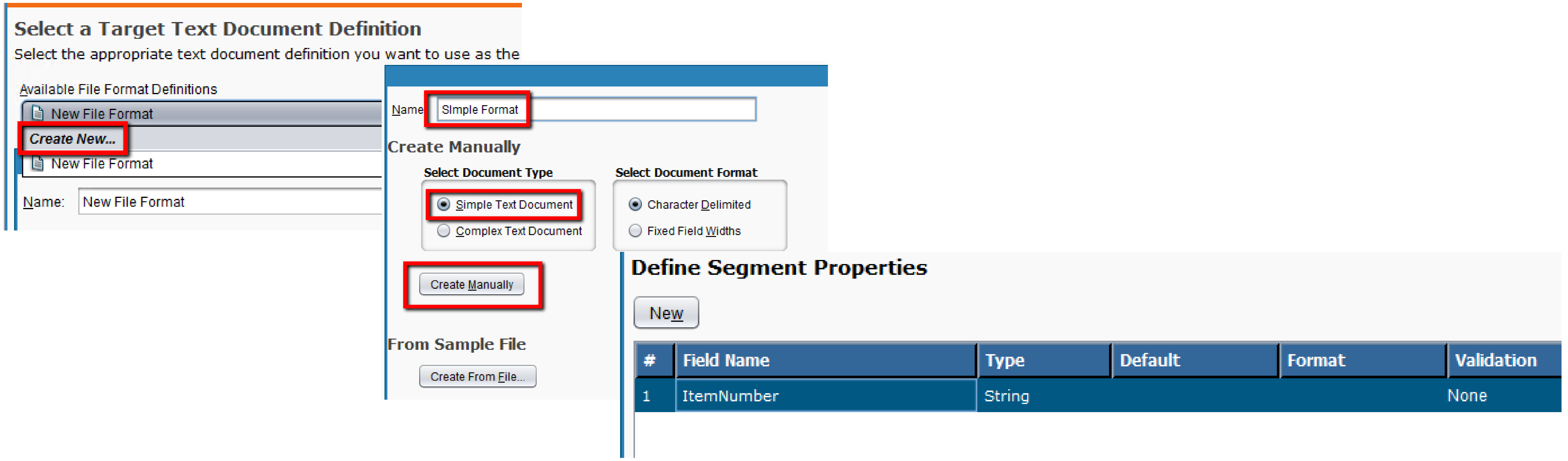
The following XSD files will be uploaded to the server:

C:\Users\svc_jbit\AppData\Local\Temp\2\jitterbit_studio\json2xsd\da14ffcb-e932-406a-ae90-b5d92d7c067b

jitterbit.json.AU2017_JSON_Body.JSON.xml0.xsd

Define a Target Data Structure: “Simple” Method

- Select “Create New” from drop down
- Name it. Select Simple Text Document and Create Manually
- Select “NEW”, add a field and repeat as needed. Select FINISH



Select a Target Text Document Definition

Select the appropriate text document definition you want to use as the

Available File Format Definitions

- New File Format
- *Create New...*
- New File Format

Name: New File Format

Name: SImple Format

Create Manually

Select Document Type: [✓ Simple Text Document] Complex Text Document

Select Document Format: [✓ Character Delimited] Fixed Field Widths

Create Manually

From Sample File

Create From File...

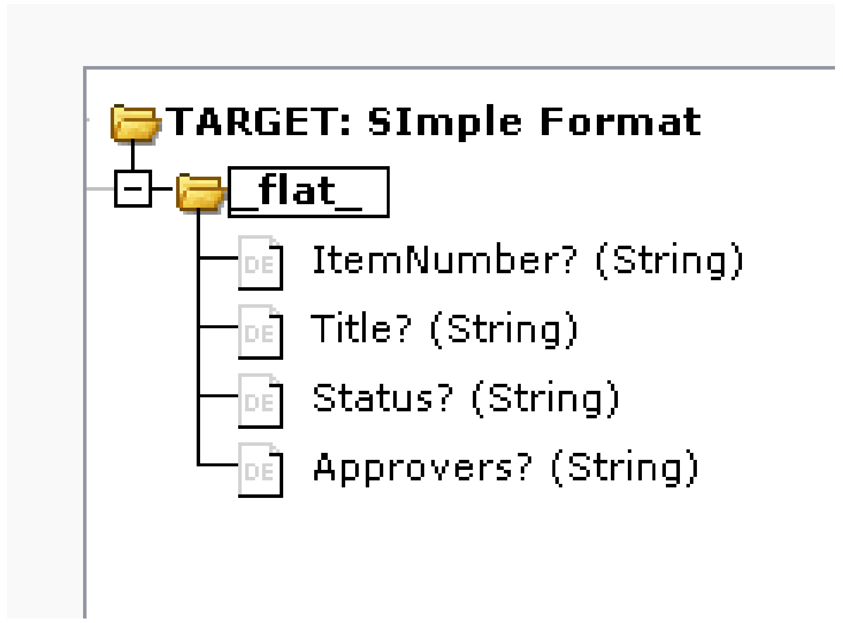
Define Segment Properties

New

#	Field Name	Type	Default	Format	Validation
1	ItemNumber	String			None

Define a Target Data Structure: “Simple” Method (cont.)

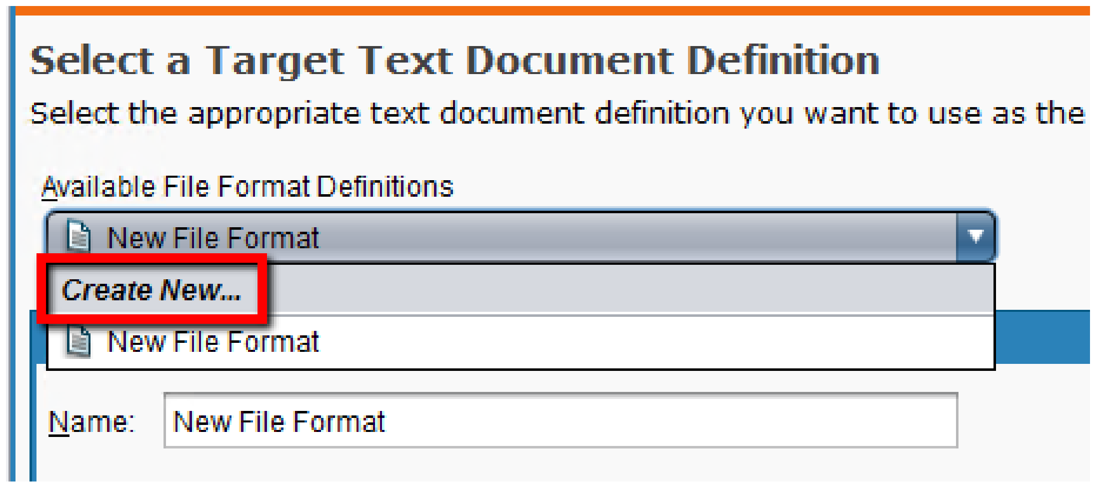
- Review the data structure



TARGET: SImple Format
flat
ItemNumber? (String)
Title? (String)
Status? (String)
Approvers? (String)

Define a Target Data Structure: “Complex” Method

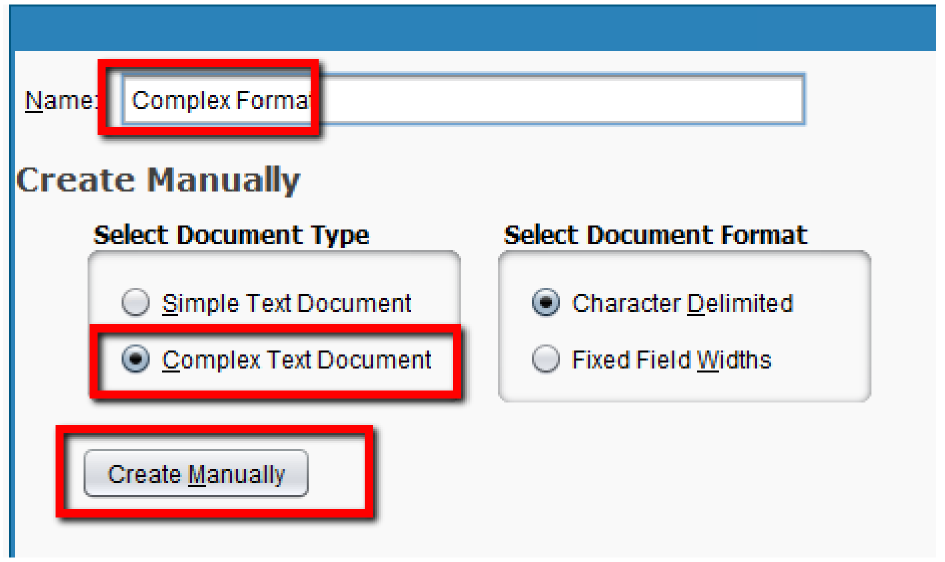
- Select “Create New” from drop down
- Name it. Select COMPLEX and Create Manually


Select a Target Text Document Definition

Select the appropriate text document definition you want to use as the

Available File Format Definitions: New File Format

Name: New File Format


Name: Complex Format

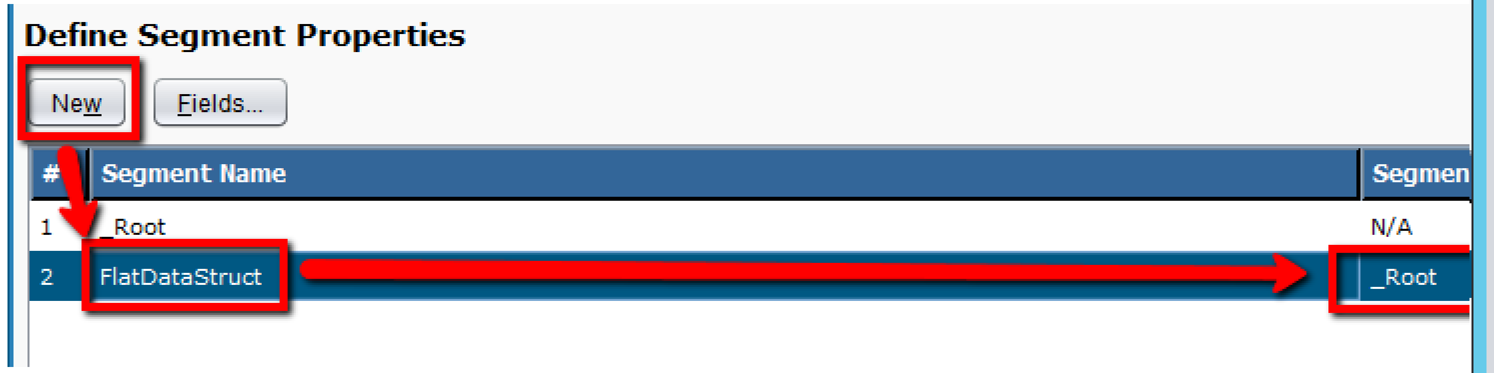
Create Manually

Select Document Type: Simple Text Document [✓ Complex Text Document]

Select Document Format: [✓ Character Delimited] Fixed Field Widths

Define “Flat” Data Structure: “Complex” Method (cont.)

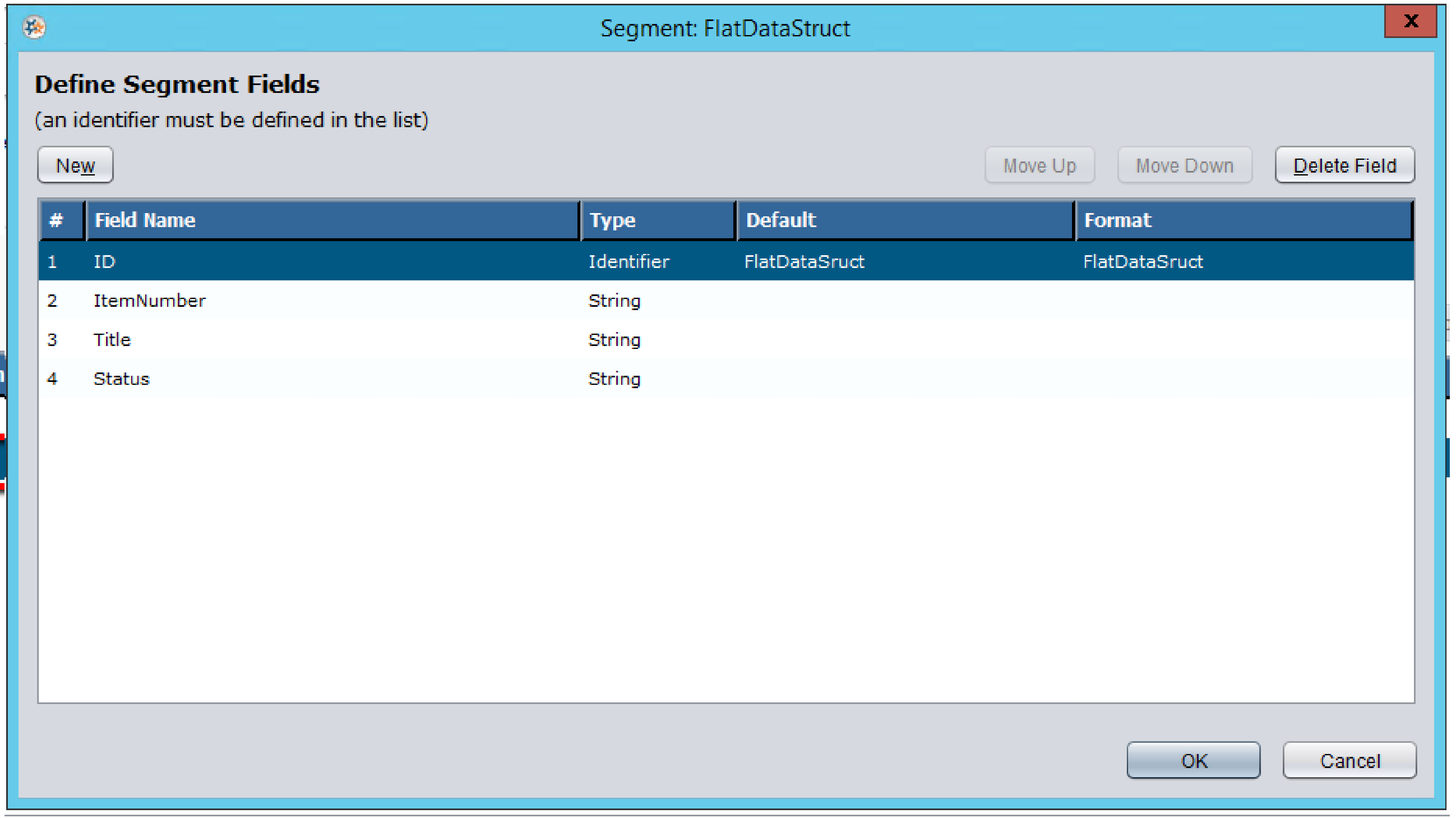
- Select “New” and add a name for the “simple” data segment
- Segment Parent: _Root, Occurrence: Only Once
- Select “Fields” and Field Names for the flat data



Define Segment Properties

New | Fields...

#	Segment Name	Segmen
1	_Root	N/A
2	FlatDataStruct	_Root



Segment: FlatDataStruct

Define Segment Fields

(an identifier must be defined in the list)

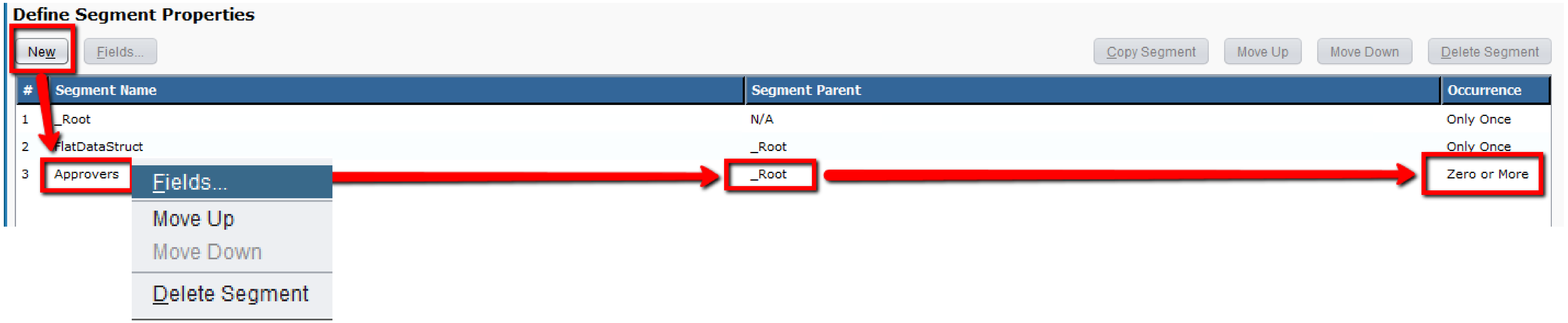
New | Move Up | Move Down | Delete Field

#	Field Name	Type	Default	Format
1	ID	Identifier	FlatDataSruct	FlatDataSruct
2	ItemNumber	String		
3	Title	String		
4	Status	String		

OK | Cancel

Define a Target Data Structure: “Complex” Method (cont.)

- Select “New”, enter a name for the “complex” data segment
- Segment Parent: _Root, Occurrence: Zero or More
- RMC on complex data name and chose “Fields”.

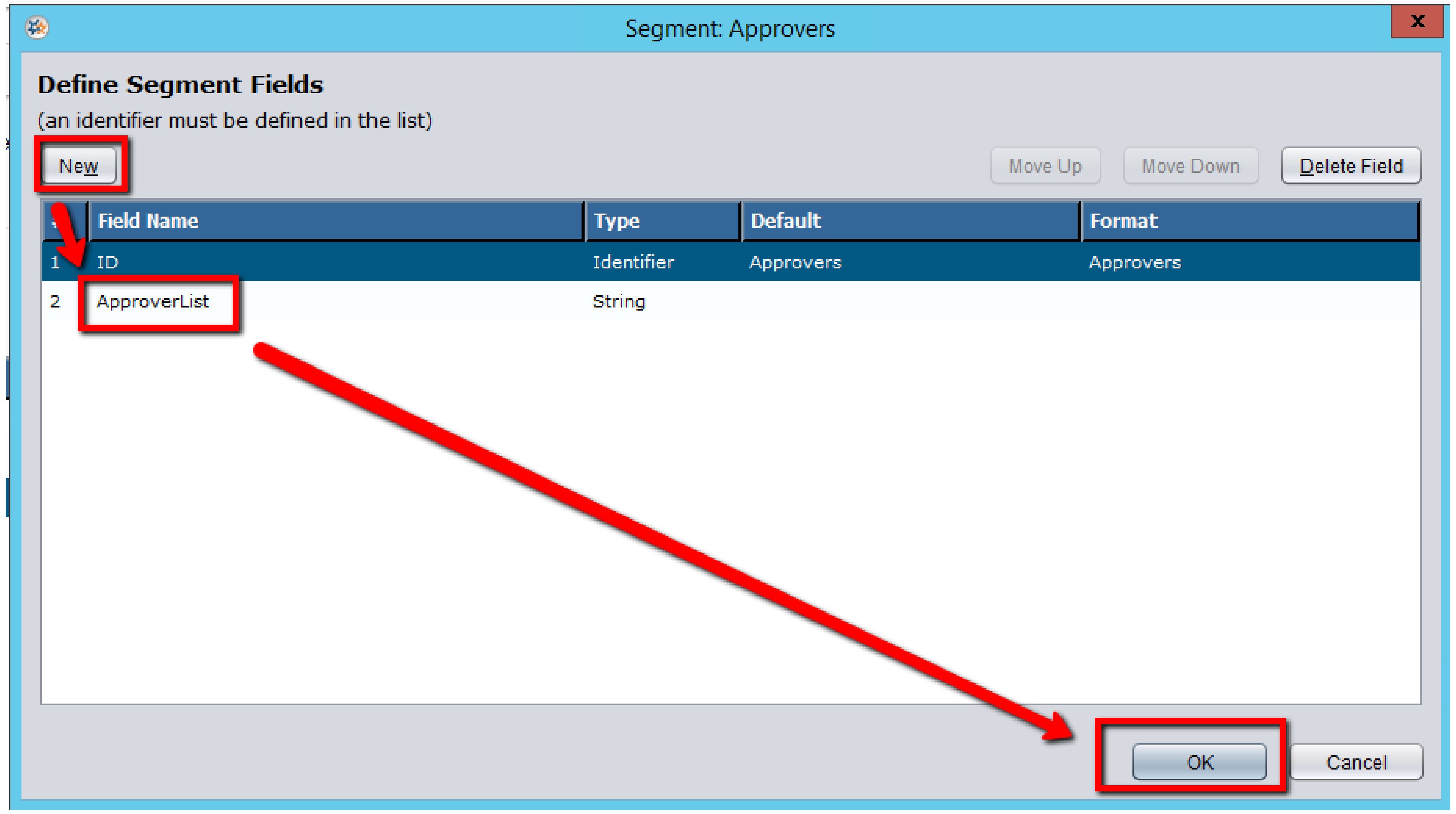


Define Segment Properties

#	Segment Name	Segment Parent	Occurrence
1	_Root	N/A	Only Once
2	FlatDataStruct	_Root	Only Once
3	Approvers	_Root	Zero or More

Define a Target Data Structure: “Complex” Method (cont.)

- Select “New”
- Add fields and data types



Segment: Approvers

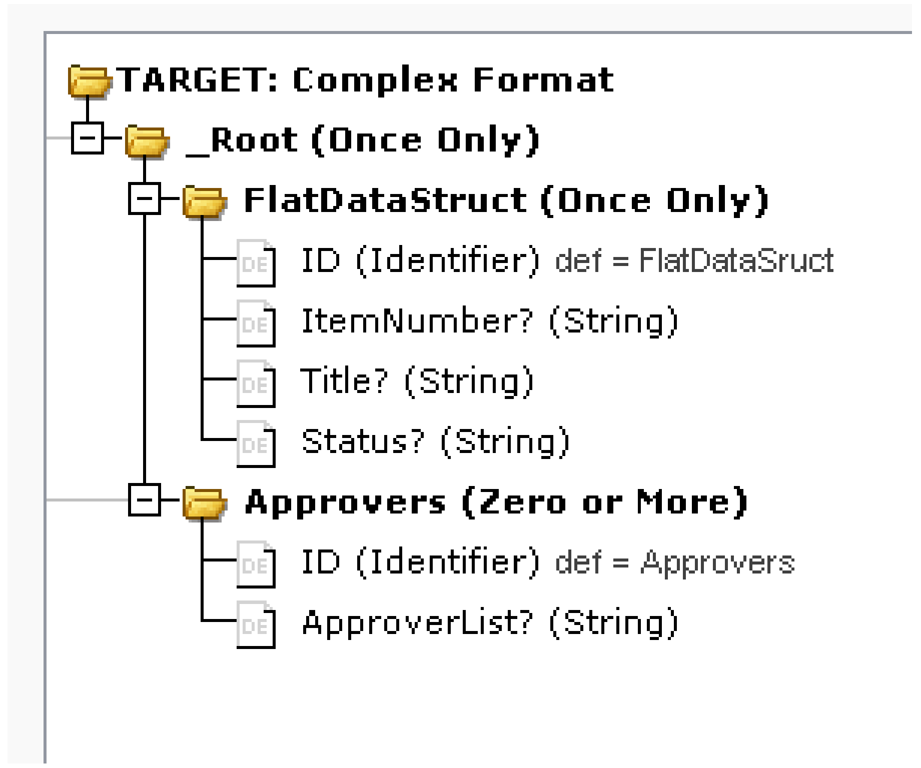
Define Segment Fields

(an identifier must be defined in the list)

	Field Name	Type	Default	Format
1	ID	Identifier	Approvers	Approvers
2	ApproverList	String		

Define a Target Data Structure: "Complex" Method (cont.)

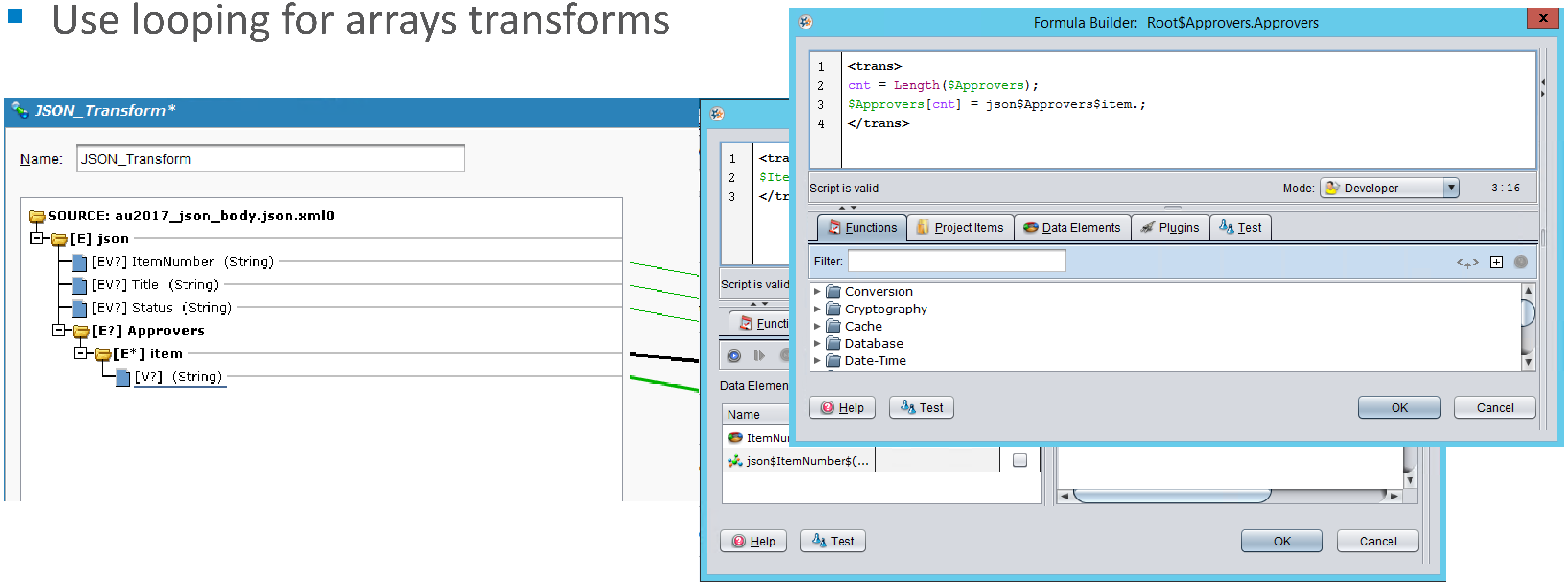
- Review the transformation structure



```
TARGET: Complex Format
  _Root (Once Only)
    FlatDataStruct (Once Only)
      ID (Identifier) def = FlatDataSruct
      ItemNumber? (String)
      Title? (String)
      Status? (String)
    Approvers (Zero or More)
      ID (Identifier) def = Approvers
      ApproverList? (String)
```

Map and Transform the Data

- Map data
- Assign variables and cast/transform as needed
- Use looping for arrays transforms



JSON_Transform*

Name: JSON_Transform

```
SOURCE: au2017_json_body.json.xml0
[E] json
  [EV?] ItemNumber (String)
  [EV?] Title (String)
  [EV?] Status (String)
  [E?] Approvers
    [E*] item
      [V?] (String)
```

Formula Builder: _Root$Approvers.Approvers

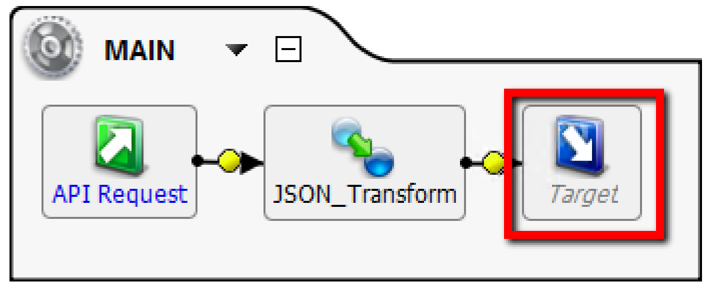
```
<trans>
cnt = Length($Approvers);
$Approvers[cnt] = json$Approvers$item.;
</trans>
```

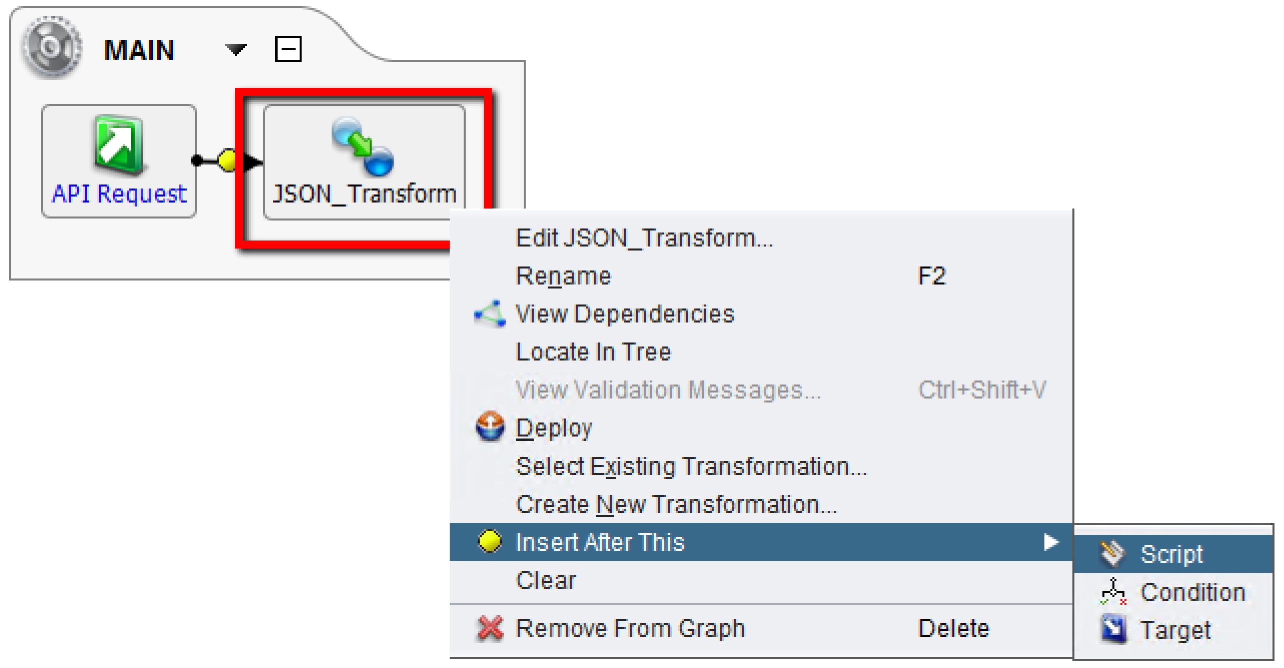
Script is valid

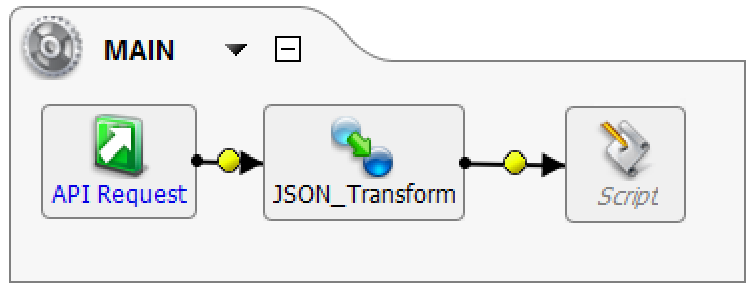
Mode: Developer 3:16

Modify Process and Create a Script

- RMC and delete the default target
- Select the transformation>RMC>Insert After This>Script

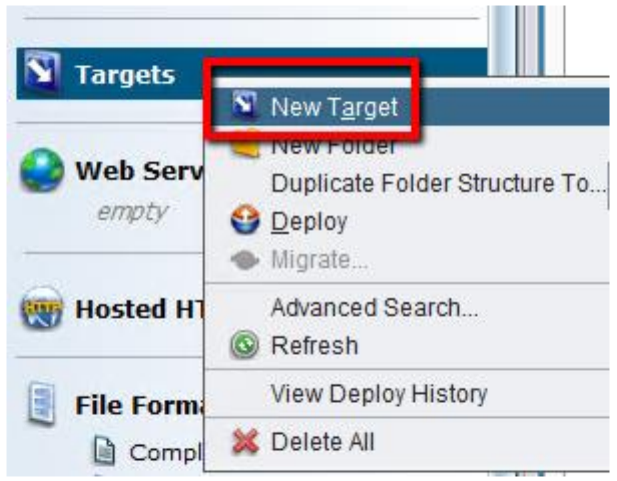


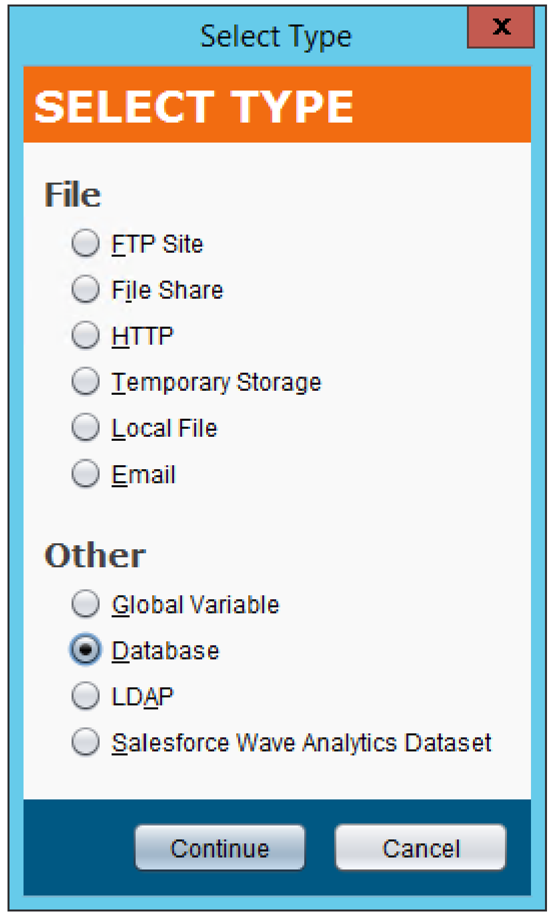




SQL: Run a Stored Procedure

- Create a database connection: Locate Targets>New Target
- Select Database
- Create the connection

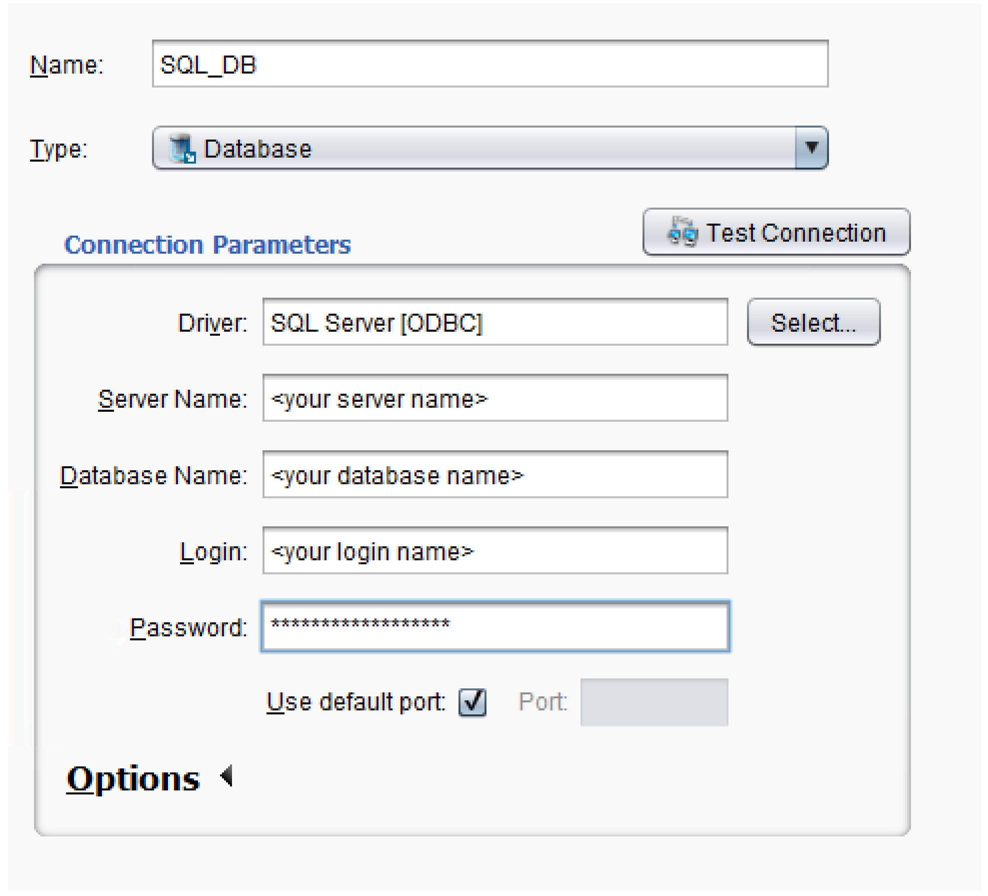



Select Type

SELECT TYPE

File
- ( ) FTP Site
- ( ) File Share
- ( ) HTTP
- ( ) Temporary Storage
- ( ) Local File
- ( ) Email

Other
- ( ) Global Variable
- (●) Database
- ( ) LDAP
- ( ) Salesforce Wave Analytics Dataset


Name: SQL_DB

Type: Database

Connection Parameters

Driver: SQL Server [ODBC]

Server Name: <your server name>

Database Name: <your database name>

Login: <your login name>

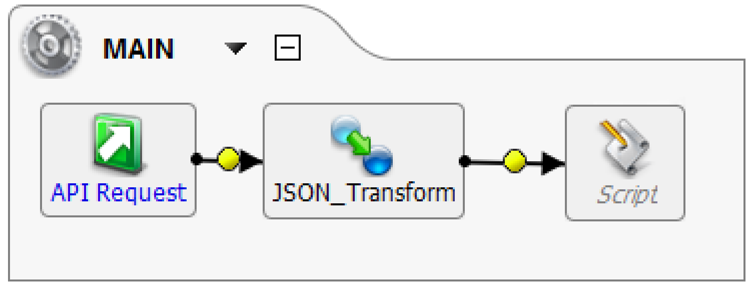
Password: ******************

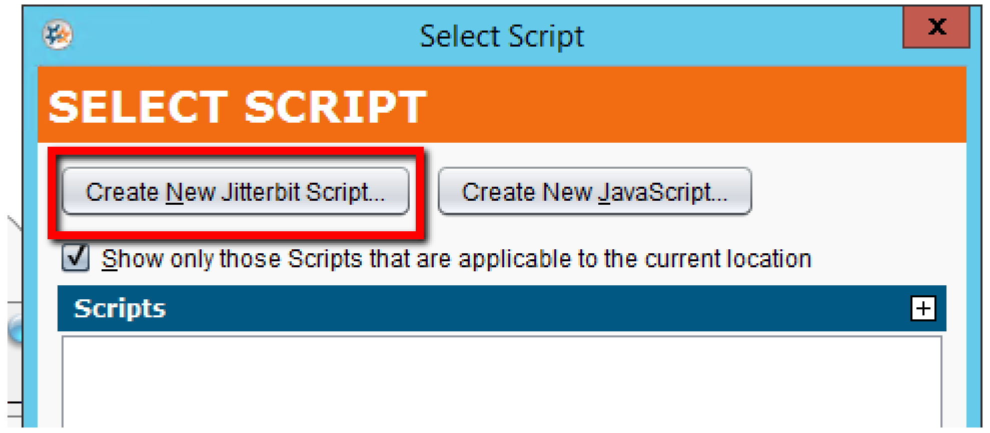
Use default port: ✓ Port:

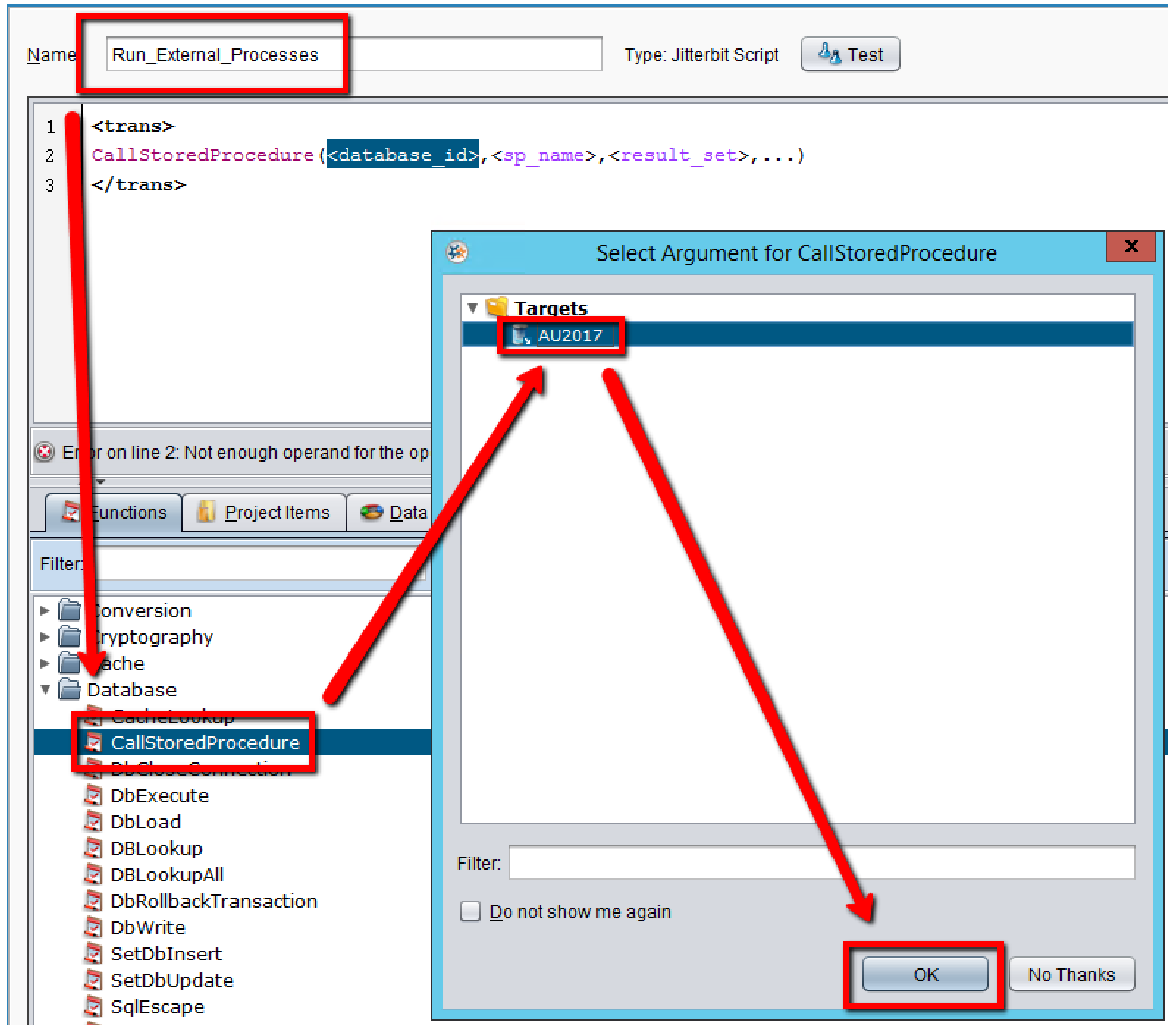
Options

SQL: Run a Stored Procedure (cont.)

- Double click the script icon
- Create a new script
- Apply function CallStoredProceedure




Select Script
SELECT SCRIPT
Create New Jitterbit Script...
Create New JavaScript...
Show only those Scripts that are applicable to the current location
Scripts

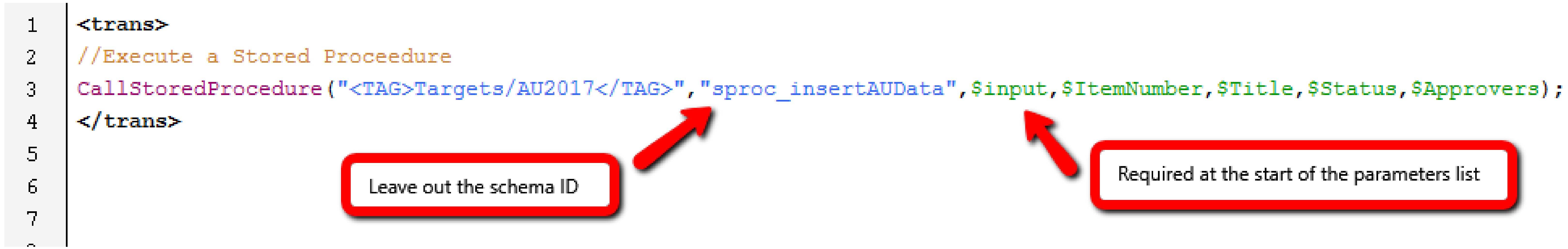


```
<trans>
CallStoredProcedure(<database_id>,<sp_name>,<result_set>,...)
</trans>
```

Error on line 2: Not enough operand for the op

SQL: Run a Stored Procedure (cont.)

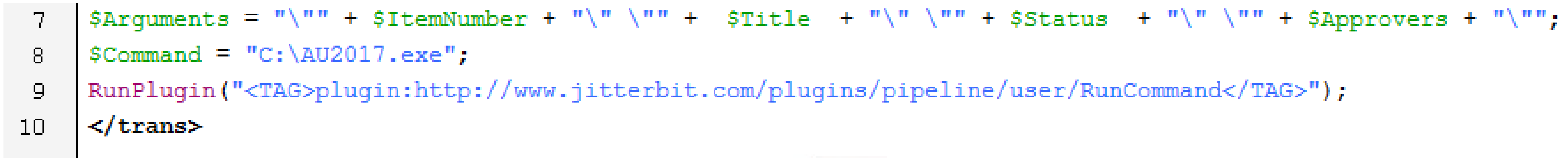
- Write the code for the sproc
- Leave out the schema ID when naming the stored procedure
- Insert the variable “$input” and then list the variables/params



```
<trans>
//Execute a Stored Proceedure
CallStoredProcedure("<TAG>Targets/AU2017</TAG>","sproc_insertAUData",$input,$ItemNumber,$Title,$Status,$Approvers);
</trans>
```

C#: Run a Console Application

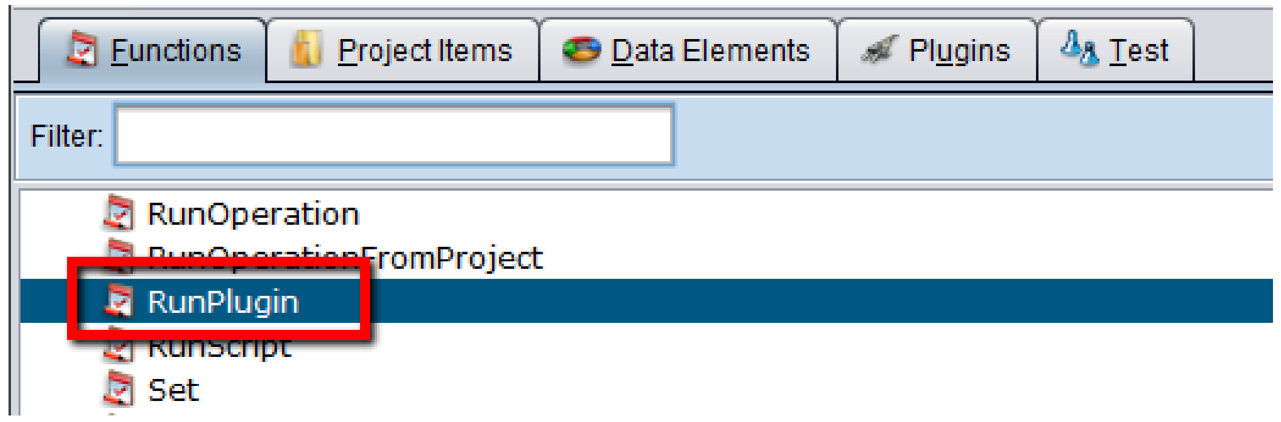
- Build “$Arguments” string
  - Encapsulate argument data in quotes (use escape characters)
  - Command is simply the path to the EXE
- Build “$Command” string
  - The string is simply the path to the EXE - Avoid white spaces
  - The Jitterbit service account should have access to the location

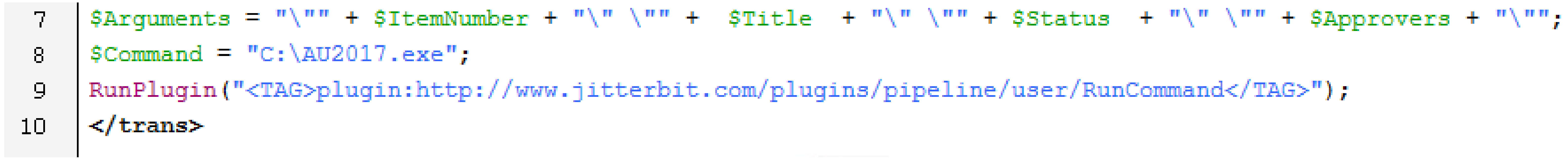


```
$Arguments = "\"" + $ItemNumber + "\" \"" +  $Title  + "\" \"" + $Status  + "\" \"" + $Approvers + "\"";
$Command = "C:\AU2017.exe";
RunPlugin("<TAG>plugin:http://www.jitterbit.com/plugins/pipeline/user/RunCommand</TAG>");
</trans>
```

C#: Run a Console Application

- From the Functions tab locate General>RunPlugin
- Insert into the code





```
$Arguments = "\"" + $ItemNumber + "\" \"" +  $Title  + "\" \"" + $Status  + "\" \"" + $Approvers + "\"";
$Command = "C:\AU2017.exe";
RunPlugin("<TAG>plugin:http://www.jitterbit.com/plugins/pipeline/user/RunCommand</TAG>");
</trans>
```

Trigger Script from Fusion Lifecycle

- Hook the action script up to a triggering event in FLC
  - Workflow transition
  - On edit or create script
  - Push button script

QUESTIONS