



Mainframe Technical Exchange  
PRAGUE, CZECH REPUBLIC | MAY 22-24, 2019



# Zowe Workshop

---

**Petr Weinfurt, Vitezslav Vit Vlcek**

OPE-02



# For Informational Purposes Only

This presentation was based on current information and resource allocations as of **May 2019** and is subject to change or withdrawal by Broadcom at any time without notice. Notwithstanding anything in this presentation to the contrary, this presentation shall not serve to (i) affect the rights and/or obligations of Broadcom or its licensees under any existing or future written license agreement or services agreement relating to any Broadcom software product; or (ii) amend any product documentation or specifications for any Broadcom software product. The development, release and timing of any features or functionality described in this presentation remain at Broadcom's sole discretion. Notwithstanding anything in this presentation to the contrary, upon the general availability of any future Broadcom product release referenced in this presentation, Broadcom will make such release available (i) for sale to new licensees of such product; and (ii) to existing licensees of such product on a when and if-available basis as part of Broadcom maintenance and support, and in the form of a regularly scheduled major product release. Such releases may be made available to current licensees of such product who are current subscribers to Broadcom maintenance and support on a when and if-available basis. In the event of a conflict between the terms of this paragraph and any other information contained in this presentation, the terms of this paragraph shall govern.

Certain information in this presentation may outline Broadcom's general product direction. All information in this presentation is for your informational purposes only and may not be incorporated into any contract. Broadcom assumes no responsibility for the accuracy or completeness of the information. To the extent permitted by applicable law, Broadcom provides this presentation "as is" without warranty of any kind, including without limitation, any implied warranties or merchantability, fitness for a particular purpose, or non-infringement. In no event will Broadcom be liable for any loss or damage, direct or indirect, from the use of this document, including, without limitation, lost profits, lost investment, business interruption, goodwill, or lost data, even if Broadcom is expressly advised in advance of the possibility of such damages. Broadcom confidential and proprietary. No unauthorized copying or distribution permitted.

# Agenda


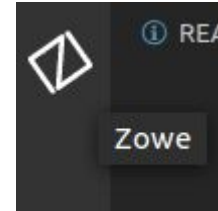
1. Set up development environment
2. Upload software maintenance to USS
3. Run RECEIVE, APPLY CHECK and APPLY jobs
4. Deploy target library to runtime environment
5. Verify maintenance has been successfully installed
6. How we can automate these tasks?
7. Learn about Gulp framework
8. Run individual Gulp tasks
9. Discussion: z/OSMF Software Update REST API

# Microsoft Visual Studio Code IDE

# Set up Development Environment

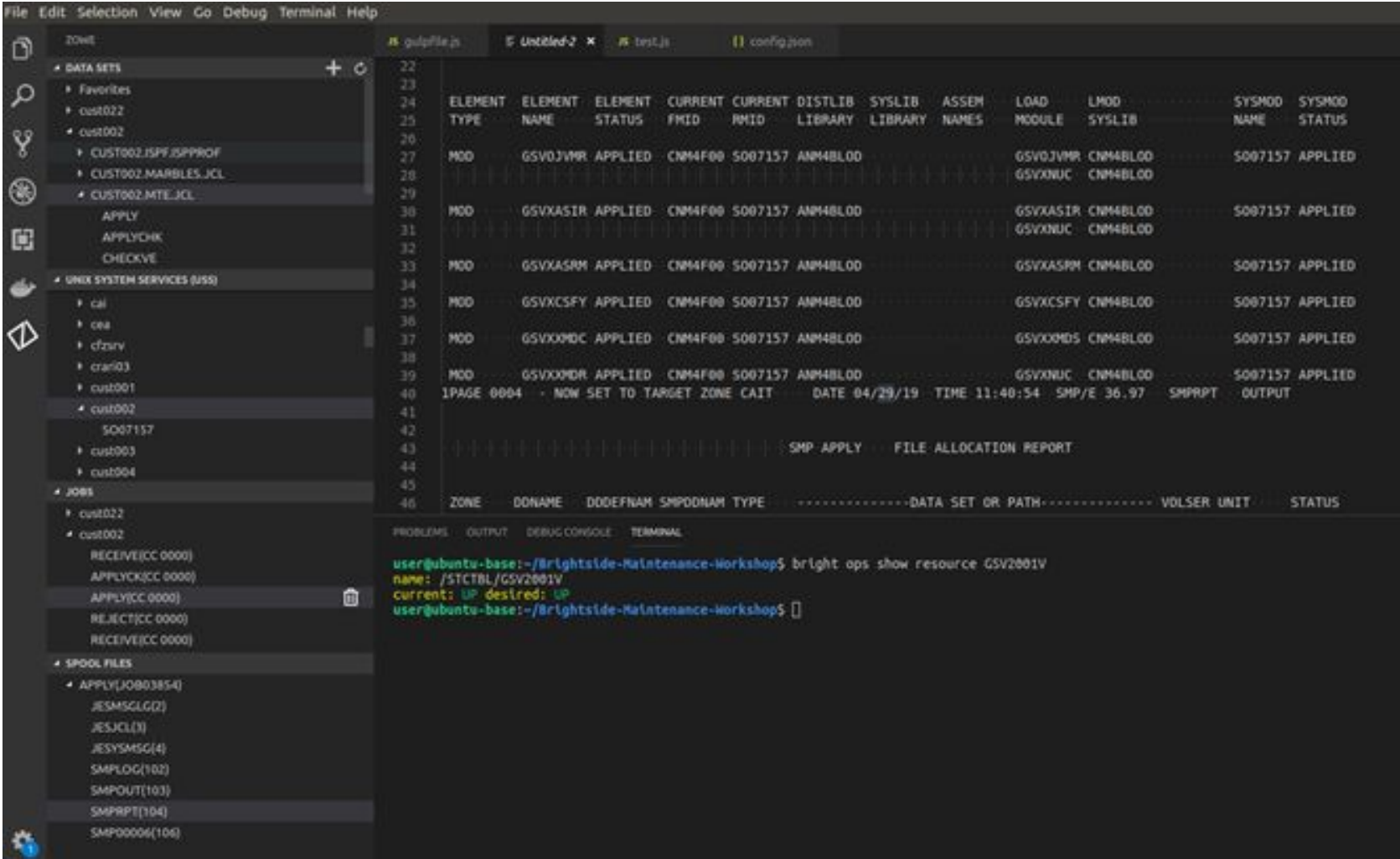
- For today's lab we will be using Visual Studio Code, a popular lightweight text editor. We will be working within a folder on our Desktop named "Zowe-Maintenance-Workshop".
- We will also be making use of the Zowe VSCode Extension and Zowe CLI.
- All participants have been assigned a user id for today's lab. Please replace references to n with your assigned id. For example, if your ID is 4, use CUST004 instead of CUST00n.

# Set up Development Environment

1. Open VSCode 
2. File -> Open Folder -> Select the “**Zowe-Maintenance-Workshop**” folder from your Desktop
3. In your VSCode editor, notice the Zowe VSCode extension icon on the left task bar. Click on the icon. 
4. Notice the UNIX SYSTEM SERVICES tree provided by the Zowe VSCode Extension
5. Click the magnifying glass on that tab to search USS for “**/u/users**”
6. Also notice the DATA SETS tree provided by the Zowe VSCode Extension
7. Click the magnifying glass on that tab to search for “**CUST00n.\***”
8. Lastly, we will make use of the integrated terminal by clicking View -> Terminal.
9. Please verify the CLI is operational by issuing the “**zowe**” command.



# Set up Development Environment



## Use CLI to upload software maintenance patch to USS

Inside the “bin” folder, there should be a patch named SO07157. This is the software maintenance that needs to be applied to SYSVIEW. We will use the Zowe CLI to upload the file to our home directory in USS.

1. In the integrated terminal, issue:

```
zowe files upload ftu "bin/SO07157" "/u/users/cust00n/SO07157" -b
```

2. Please confirm the file has been uploaded by refreshing your UNIX SYSTEM SERVICES tab and expanding the CUST00n directory under /u/users.



# Run RECEIVE, APPLY CHECK, and APPLY Jobs

All JCL for this workshop is in the **CUST00n.MTE.JCL** PDS. You can easily view each member in VSCode by clicking it in the DATA SETS tree. You can also enable syntax highlighting by clicking the "Plain Text" language selection in the bottom right of the editor and keying in JCL.

1. Submit the RECEIVE job by issuing the following command in the integrated terminal:

```
zowe jobs submit ds "CUST00n.MTE.JCL(RECEIVE) "
```

2. You can follow up on the status of the job by issuing:

```
zowe jobs view jsbj <JOBID>
```

where <JOBID> is available from the output of the first command

3. You can list all available spool files by issuing:

```
zowe jobs list sfbj <JOBID>
```

4. Finally, you can access a particular spool file with:

```
zowe jobs view sfbi <JOBID> <SPOOLID>
```

where <SPOOLID> is the first column of output of the previous command.

## Run RECEIVE, APPLY CHECK, and APPLY Jobs

We can also submit a job, wait for it to complete, and view all spool content with a single command.

For the APPLYCHK job, try issuing:

```
zowe jobs submit ds "CUST00n.MTE.JCL(APPLYCHK)" --vasc
```

After confirming the previous job completes without error, try submitting the APPLY job.

## COPY LOADLIB to runtime environment

To copy the SMPE target lib to the runtime environment, we will submit COPY job by issuing:

```
zowe jobs submit ds "CUST00n.MTE.JCL(COPY)" --vasc
```

- If CA FileMaster Plus is available then Zowe can be used with File Master Plus plugin like this:

```
zowe fmp copy ds <source data set> <target data set>
```

## Verify maintenance has been successfully applied

To verify the maintenance has been applied, we will view the fix level of the particular module of interest, GSVXASRM. In SYSVIEW, this can be accomplished by using the LISTDIR command:

```
listdir PRODUCT.SVRUNn.MSTRBRS.CNM4BLOD(GSVXASRM),,,modid
```

- We have a job that will run this command in batch, **CUST00n.MTE.JCL(CHECKVE)**. Submit this job and verify the FixLevel of GSVXASRM is SO07157.

# How can we automate these tasks?

# How can we automate?

- Shell scripts, JavaScript, Python, etc.
- Modern task runners like Gulp.js
  - <https://gulpjs.com/>
- Modern test frameworks like Mocha.js
  - <https://mochajs.org/>



## Gulp tasks

Gulp tasks can be run from the command line, try issuing:

**gulp help**

Gulp help should show you the available tasks in our project.

Before moving on, please issue:

**gulp reset**

This task will restore, reject and delete from USS the maintenance you manually applied earlier in the lab. We need it so that we can receive and apply the maintenance again.

Next, issue:

**gulp upload**

## Gulp tasks

Gulp tasks can also be run from the Gulp VSCode extension:

- try clicking on the **GULP TASKS** tab in the VSCode editor in the file explorer and running the **receive** task and then **apply-check** gulp task

These gulp tasks are exercising the Zowe CLI behind the scenes. Open the **gulpfile.js** file located at the project's root. Notice line 5:

```
var config = require('./config.json');
```

config.json (also located at the project's root) contains the configuration information that we would customize depending upon the maintenance we are applying.

Further down in gulpfile.js, starting on line 96, you will see the definitions for each of our gulp tasks.

## Code your own gulp task to apply maintenance

From the **apply-check task** definition starting on line **100**, try to fill in the **apply task** definition starting on line **96**.

- Run the gulp task you created by issuing:

```
gulp apply
```

## Run all tasks together

Before moving on, please run:

**gulp reset**

task again which sets us to initial state.

Now that you have filled in the apply step you can run the deploy task to deploy the maintenance in a single task.

Review the deploy task on line 137 of gulpfile.js to view how multiple tasks are combined into a single flow.

Issue:

**gulp deploy**

## Run the test suite to ensure the maintenance was applied

In order to verify the maintenance was applied successfully, please issue:

**npm test**

Npm test will look in the package.json file located at the root directory. In the package.json, notice:

```
"scripts": {  
  "test": "mocha --reporter mochawesome"  
}
```

- Mocha is the modern test framework being used by this project. Modern test frameworks have assertion libraries that make testing quick and easy.
- The tests are located in test/test.js from the project's root. The general test suite starts on line 111.

## Review the report that was generated from the run

A report was generated from the final test for audit purposes. It is located within the project at [mochawesome-report/mochawesome.html](#).

Please review.



# **z/OSMF Software Update ... without UI**

# What is a REST API and how can it enable automation?

A REST API can offer greater flexibility - no need to follow the path set out in a UI

Some examples..

- schedule an automated maintenance process to run on a set cadence
- choose to run only part of the maintenance operation e.g. run an APPLY CHECK only, don't APPLY (create a flat file from the results of the APPLY CHECK so that maintenance can be applied separately)
- APPLY maintenance across multiple zones in 1 CSI or even across multiple CSIs

## 2 Global approaches to building the REST API:

End-to- end approach that Mirrors UI

or

Modular approach with no direct correlation to the UI



**BROADCOM<sup>®</sup>**

connecting everything<sup>®</sup>