

# Developing CICS applications your way, using Visual Studio Code

Drew Hughes

Software Developer – CICS Modernization

[Andrew.Hughes1@ibm.com](mailto:Andrew.Hughes1@ibm.com)

# Agenda

## CICS and APIs

Why do you need them?

What APIs do we have?

## The IDE options for CICS

### Visual Studio Code

Deeper dive!

What extensions are useful?

How does it all work together?

### Give it a try

Tips for your setup

Get in touch and shape the way this works.

# Agenda

## CICS and APIs

Why do you need them?

What APIs do we have?

## The IDE options for CICS

### Visual Studio Code

Deeper dive!

What extensions are useful?

How does it all work together?

### Give it a try

Tips for your setup

Get in touch and shape the way this works.

# CICS – the mixed language application server

You write your business logic

CICS looks after the boring stuff...

- Security, scaling, access to resources
- CICS TS – Transaction Server (commit, rollback)

For CICS to do this, it needs to understand what resources you're accessing and what you're doing.

Enter ... the APIs!

# The orchestra of APIs

A wide choice of languages

- COBOL, PL/I, Assembler, REXX
- C/C++, Java, Node.js

Components written in different languages can call each other and access the same resources

e.g. You can write your business logic in COBOL and write a front end in Java / Node.js

# The orchestra of APIs

COBOL, PL/I, Assembler, REXX

EXEC CICS commands

CICS translator -> a preprocessor

COMMAREA vs CHANNEL/CONTAINER

Async API

EXEC CICS RUN -> starts an async request

EXEC CICS FETCH -> retrieves async result

```
210
211     UPDATE-CUSTOMER-VSAM SECTION.
212     UCV010.
213
214     *
215     *   Position at the matching CUSTOMER record and
216     *   lock it.
217     *
218     MOVE COMM-CUSTNO TO DESIRED-CUSTNO.
219
220     EXEC CICS READ FILE('CUSTOMER')
221           RIDFLD(DESIRED-CUST-KEY)
222           INTO(WS-CUST-DATA)
223           UPDATE
224           RESP(WS-CICS-RESP)
225           RESP2(WS-CICS-RESP2)
226     END-EXEC.
227
```

Developing applications - <https://www.ibm.com/docs/en/cics-ts/6.x?topic=developing-applications>

EXEC CICS command summary - <https://www.ibm.com/docs/en/cics-ts/6.x?topic=reference-cics-command-summary>

ASYNCR API - <https://www.ibm.com/docs/en/cics-ts/6.x?topic=applications-developing-asynchronous-requests>

# The orchestra of APIs

```
62     @GET
63     @Produces("application/json")
64     public Response getCompanyName()
65     {
66         logger.entering(this.getClass().getName(), GET_COMPANY_NAME);
67         // We cache the company name as a static variable. If not set, we jCICS
68         // LINK to a COBOL program to go get it
69         if (companyNameString == null)
70         {
71             Program getCompy = new Program();
72             getCompy.setName("GETCOMPY");
73
74             byte[] companyNameBytes = new byte[40];
75
```

```
// Gets the current CICS Context for the environment we're running in
CICSContext task = CICSContext.getCICSContext();

try {
    // Create a new channel called "charchan", with a CHAR container called
    // "charcont"
    // Add the text to the CHAR container
    CHARContainer charContainer = task.getChannel(CHANNEL_NAME).getCHARContainer(CONTAINER_NAME)
        .put("I'm running under task ");

    // Get the current task number that this unit of work is running under
    Integer taskNumber = task.getTaskNumber();

    // Add the task number to the end of the CHAR container
    charContainer = charContainer.append(taskNumber.toString());

    // Get the full contents of the container and print this
```

## Java

- JCICS API - similar to EXEC CICS
- JCICSX API – remoteable and more “Java native” (provides LINK -> Channel/Container)
- CICS Liberty  
Jakarta, Eclipse MicroProfile, Spring Boot
- Other Java libraries you need!

## Maven and Gradle plugins available

Java JCICS - <https://www.ibm.com/docs/en/cics-ts/6.1.0?topic=applications-java-development-using-jcics>  
Java JCICSX - <https://www.ibm.com/docs/en/cics-ts/6.1.0?topic=applications-java-development-using-jcicsx>  
cics-bundle-maven - <https://github.com/IBM/cics-bundle-maven>  
cics-bundle-gradle - <https://github.com/IBM/cics-bundle-gradle>

# The orchestra of APIs

```
73
74     let url = catalogServer + '/exampleApp/inquireCatalogWrapper';
75
76     console.log('Get first set of items API request: ' + url);
77     console.dir(inquireRequest1);
78
79     var promise1 = cics.invoke(url, inquireRequest1)
80         .then(function (json) {
81             console.log('Get first set of items API response: ');
82             console.dir(json.inquireCatalogResponse.catalogItem);
83
84             allItemsArray = allItemsArray.concat(json.inquireCatalogRespor
85         });
86
```

## Node.js

- Promises and similar Node.js async API
- Link to other CICS programs
- Other JavaScript libraries you need!

# API samples

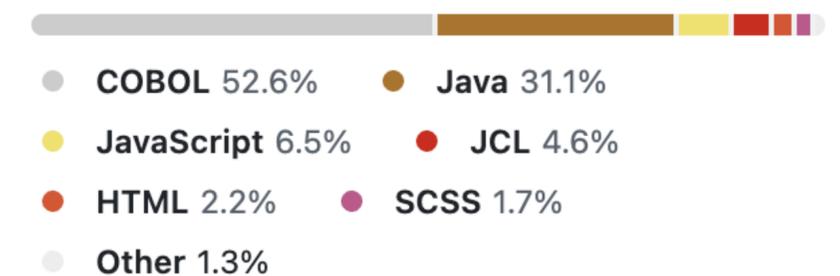
## CICSDev

- A large collection of samples on GitHub
- <https://github.com/cicsdev>

## CICS Banking Sample Application

- Our biggest sample
- Mix of COBOL, Java, JavaScript
- 3270 interface
- Liberty-hosted Web interface

### Languages



EXEC CICS command summary - <https://www.ibm.com/docs/en/cics-ts/6.x?topic=reference-cics-command-summary>

Java JCICS - <https://www.ibm.com/docs/en/cics-ts/6.1.0?topic=applications-java-development-using-jcics>

Java JCICSX - <https://www.ibm.com/docs/en/cics-ts/6.1.0?topic=applications-java-development-using-jcicsx>

CBSA - <https://github.com/cicsdev/cics-banking-sample-application-cbsa>

# Agenda

## CICS and APIs

Why do you need them?

What APIs do we have?

## The IDE options for CICS

### Visual Studio Code

Deeper dive!

What extensions are useful?

How does it all work together?

### Give it a try

Tips for your setup

Get in touch and shape the way this works.

# Our 3 contestants

## 3270 (ISPF, CEMT)

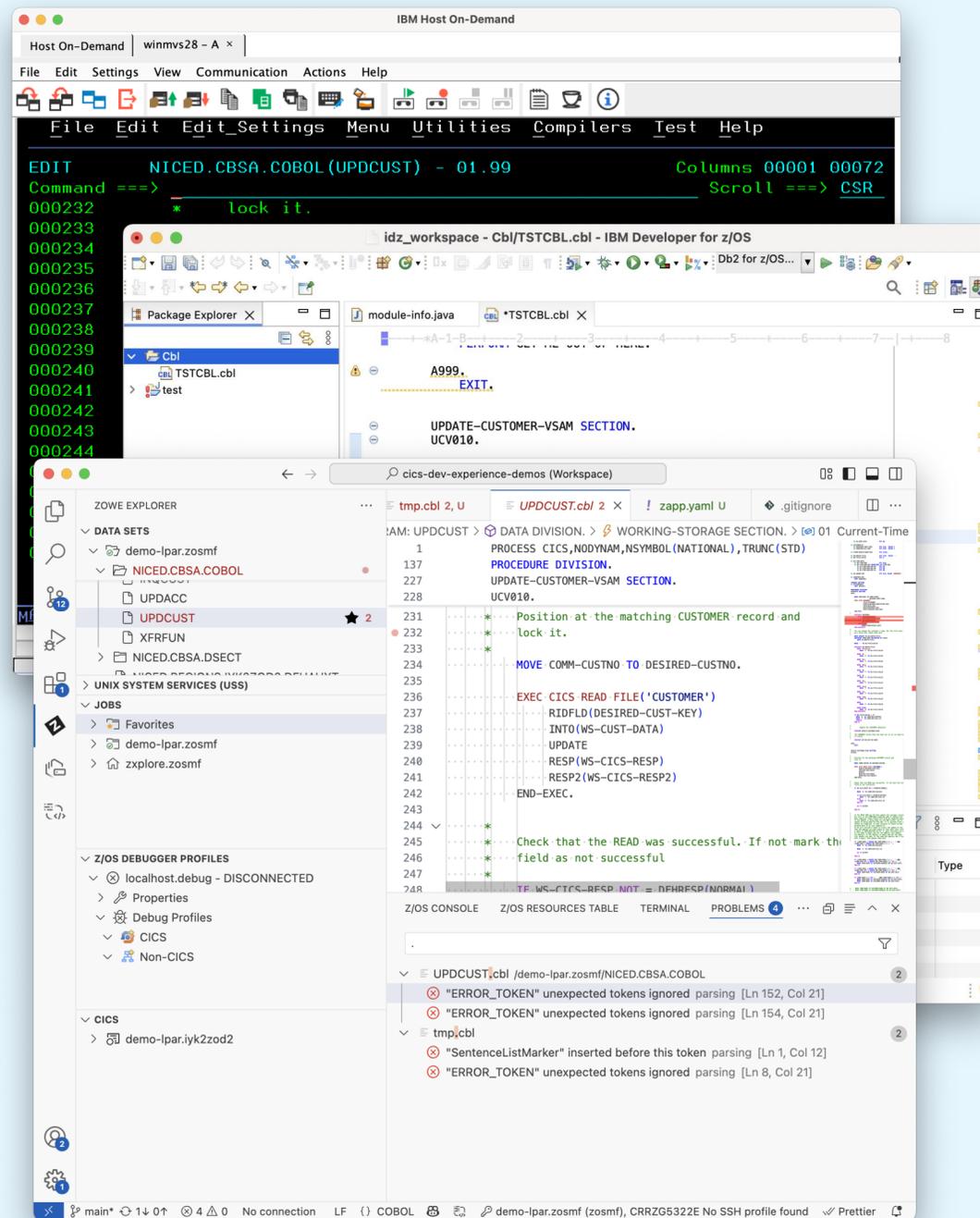
- Traditional choice
- Keyboard, keyboard, keyboard!

## Eclipse

- Mature, stable, feature-rich

## Visual Studio Code (VS Code)

- The new(ish) kid on the block
- Lightweight
- Modular / extension-oriented



# Our 3 contestants

- Many tasks can be completed in any environment, so it's a preference call.
- What your colleagues do doesn't have to be what you do!
- Some options will fit better with your company's SCM and any mandated workflows
- Today we'll focus on VS Code!

# Agenda

## CICS and APIs

Why do you need them?

What APIs do we have?

## The IDE options for CICS

### **Visual Studio Code**

Deeper dive!

What extensions are useful?

How does it all work together?

## Give it a try

Tips for your setup

Get in touch and shape the way this works.

# VS Code

Local file manipulation  
Git tools

VS Code

Data sets, z/OS Unix, Jobs, z/OS Console Profiles (connection details)

Zowe Explorer

Managing CICS resources in a CICS region

IBM CICS for Zowe Explorer

COBOL, PL/I, Assembler  
syntax highlighting, code complete

IBM Z Open Editor

The screenshot displays the VS Code interface with the following components:

- Zowe Explorer (Left Panel):** Shows a tree view of data sets and jobs. The 'UPDCUST' data set is highlighted with a star and the number 7. Below it, the 'CICS' section shows a list of programs including UPDACC, UPDACCT, and UPDCUST.
- Code Editor (Center):** Displays the source code for 'UPDCUST.cbl'. The code is in COBOL and includes sections for PROCEDURE DIVISION, UPDATE-CUSTOMER-VSAM SECTION, and UCV010. A tooltip is visible over the 'UPDATE' keyword, showing options for 'UPDATE', 'COBOL CICS: READ UPDATE', 'COBOL Embedded SQL: Update Where Current ...', and 'COBOL Embedded SQL: Set-Level SQL Update ...'. The status bar at the bottom indicates 'Ln 223, Col 20 Spaces: 3 UTF-8 LF {} COBOL'.
- Problem List (Bottom Panel):** Shows several error messages, including '"READOptions" is invalid parsing [Ln 223, Col 17]' and multiple 'Unreachable code symbol' warnings.
- Outline (Right Panel):** Shows a hierarchical view of the program structure, including sections like IDENTIFICATION DIVISION, ENVIRONMENT DIVISION, DATA DIVISION, and WORKING-STORAGE SECTION.

# Demo

Hey, can you take a look at the *update customer* feature of CBSA? It's not working properly...

Yeah sure!



Demo uses **CBSA** installed in a CICS region  
z/OSMF available on the host  
CMCI connection through an SMSS region with the CMCI JVM server

# Agenda

## CICS and APIs

Why do you need them?

What APIs do we have?

## The IDE options for CICS

### Visual Studio Code

Deeper dive!

What extensions are useful?

How does it all work together?

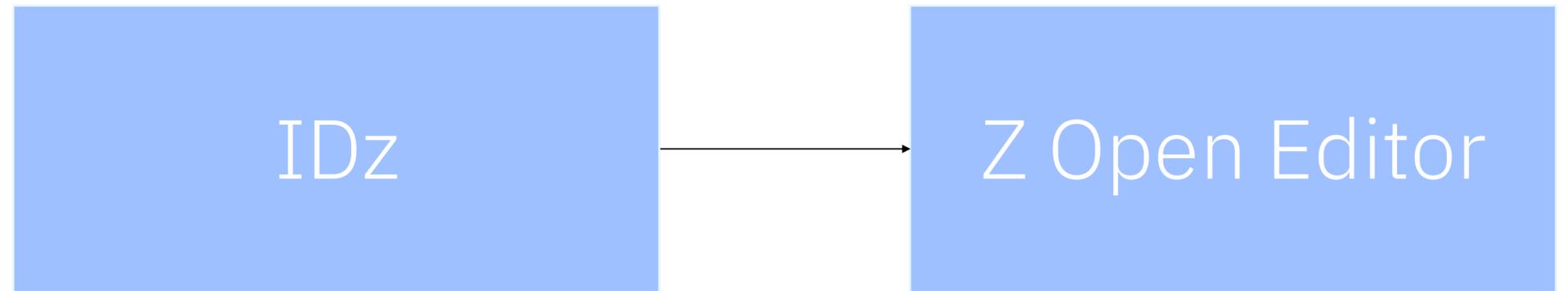
### **Give it a try**

Tips for your setup

Get in touch and shape the way this works.

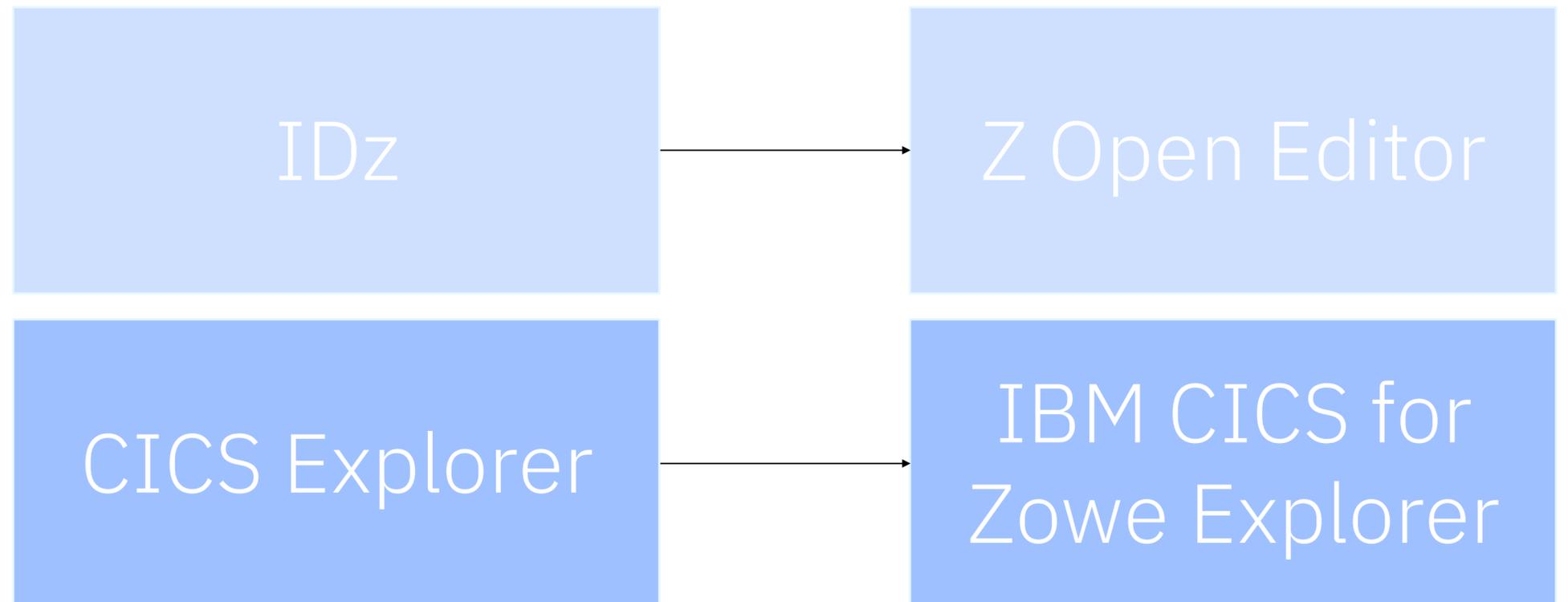
# What are you currently using?

- 😞 If you have an IDz license you are supported to use all Z Open Editor features
- ✅ Without an IDz license you can still use most Z Open Editor features



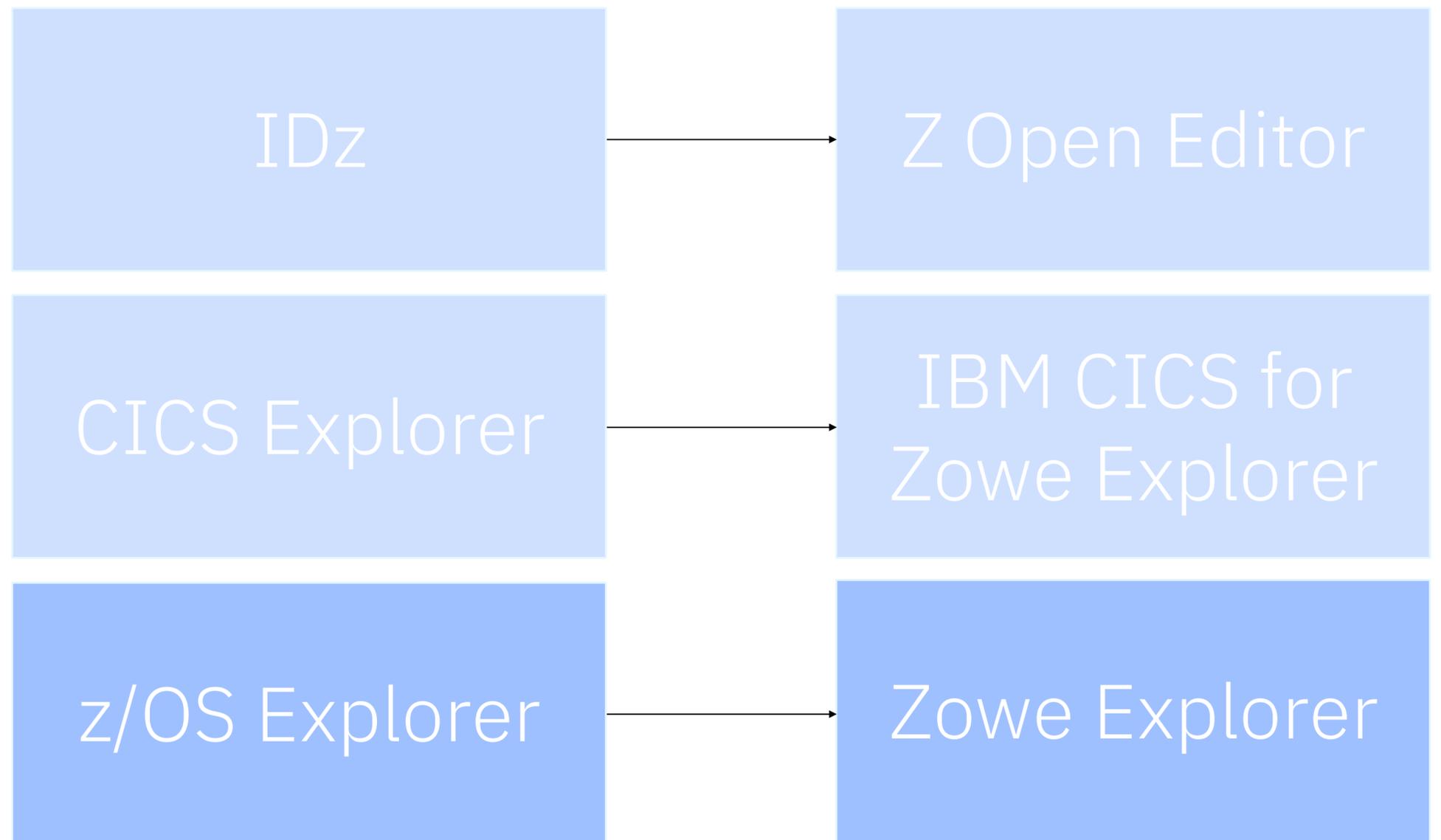
# What are you currently using?

- CICS Explorer uses the same **CMCI connection** as IBM CICS for Zowe Explorer



# What are you currently using?

- z/OS Explorer with z/OSMF or FTP – you can use this with Zowe Explorer (and IBM z/OS FTP for Zowe Explorer)
- z/OS Explorer with RSE – install the additional [RSE API](#) to use Zowe Explorer



# ID/z EE extension pack

One click to install:

- Z Open Editor
- Z Open Debug
- Zowe Explorer
- IBM CICS for Zowe Explorer

The screenshot shows the Visual Studio Marketplace page for the 'IBM Developer for z/OS on VS Code Extension Pack'. The page includes the following information:

- Extension Pack Name:** IBM Developer for z/OS on VS Code Extension Pack
- Publisher:** IBM (ibm.com)
- Installs:** 140
- Rating:** 5 stars (0 reviews)
- Price:** Free
- Description:** Extension pack for mainframe application developers entitled to IBM Developer for z/OS Enterprise Edition.
- Buttons:** Install, Trouble Installing?
- Navigation:** Overview, Version History, Q & A, Rating & Review
- Categories:** Extension Packs
- Tags:** ASM, CICS, COBOL, Debug, HLASM, IBM Z, IDE, IDz, IDzEE, JCL, language-server, LSP, Mainframe, PL/I, PL1, PLI, REXX, Z Software, z/OS, zOS, Zowe
- Works with:** Universal, Web
- Resources:** Issues, Repository, Homepage, License, Changelog
- Project Details:** IBM/zopeneditor-about, No Pull Requests, Last Commit: 4 weeks ago, 118 Open Issues
- More Info:** Version 5.4.0, Released on 25/04/2025, 23:37:00, Last updated 25/04/2025, 23:41:08, Publisher IBM, Unique Identifier IBM.developer-for-zos-on-vscode-extension-pack, Report Report a concern

# Zowe profiles

- JSON based files that contain your connection details
- Updated significantly with Zowe v3 in Oct 2024
- 😊 Shared with zowe-cli
- 😊 Can be stored in workspaces to share with colleagues
- 😞 Can be a bit painful to get set up with
- New in v3 are nested profiles ... you can associate profiles for a host together and share properties. Nested is the way to go! 🎉

Zowe profiles doc: <https://docs.zowe.org/stable/user-guide/ze-profiles>

# Zowe profiles

- Profiles are nested **by host**
- **Common properties** for a host
- **Connection profiles** inheriting common properties
- **Save passwords** to **secure** storage

```
{
  "$schema": "./zowe.schema.json",
  "profiles": {
    "host1": {
      "properties": {
        "host": "host1.com",
        "rejectUnauthorized": false
      },
      "secure": ["user", "password"],
      "profiles": {
        "zosmf": {
          "type": "zosmf",
          "properties": {
            "port": 32070,
            "protocol": "https"
          }
        },
        "cicsplex1": {
          "type": "cics",
          "properties": {
            "protocol": "http",
            "port": 12345,
            "cicsPlex": "MYPLEX",
            "regionName": "MYREGION"
          }
        }
      }
    }
  },

```

```
    "host2": {
      "properties": {
        "host": "host2.com",
        "rejectUnauthorized": false
      },
      "secure": ["user", "password"],
      "profiles": {
        "zosmf": {
          "type": "zosmf",
          "properties": {
            "port": 32070,
            "protocol": "https"
          }
        },
        "cicsplex3": {
          "type": "cics",
          "properties": {
            "protocol": "https",
            "port": 12347
          }
        }
      }
    },
    "defaults": {
      "zosmf": "host1.zosmf",
      "tso": "host1.tso",
      "ssh": "host1.ssh",
      "rse": "host1.rse",
      "cics": "host1.cicsplex1"
    }
  },
  "autoStore": true
}
```

An example: <https://gist.github.com/davenice/0cd4623ea3386ab522eac0999b8f7aaa>

# ZAPP files

- YAML files that help set out rules for your Z application
- Referenced by Z Open Editor, IDz, and build tools
- The most basic reason is to help resolve copybooks if you're editing code remotely

## Steps:

- (1) Set up the ZAPP file
- (2) Tell Z Open Editor which connection to use

ZAPP files doc: <https://ibm.github.io/zopeneditor-about/Docs/zapp.html>

Property groups doc (for include resolution): [https://ibm.github.io/zopeneditor-about/Docs/setting\\_propertygroup.html](https://ibm.github.io/zopeneditor-about/Docs/setting_propertygroup.html)

# ZAPP files

- Copybooks unresolved!
- Use quick fix to jump into the ZAPP file
- Add entry to resolve from MVS (can use variables so these can be shared with your app code)
- Use status bar widget to choose connection

The screenshot shows the IBM Z Open Editor interface. The top panel displays a COBOL program named 'UPDCUST.cbl' with the following code:

```
1          PROCESS CICS,NODYNAM,NSYMBOL(NATIONAL),TRUNC(STD)
38
39          DATA DIVISION.
40          WORKI
41
42          COPY SORTCODE.
43          SORTCODE.
44
45          01 Current-Time.....PIC X(10).
46
```

A red arrow points to the 'Quick Fix... (⌘.)' option in the context menu that appears over the 'COPY SORTCODE.' statement. The bottom panel shows the configuration file 'zowe.config.json' with the following content:

```
9  author: IBM Z Open Editor
10 propertyGroups:
11   - name: search-all
12     libraries:
13       - name: syslib
14         type: local
15         locations:
16           - '**'
17       - name: syslib
18         type: mvs
19         locations:
20           - 'NICED.CBSA.DSECT'
21
```

Red boxes highlight the 'syslib' entry with 'type: mvs' and the 'locations' array containing 'NICED.CBSA.DSECT'. The status bar at the bottom shows the connection 'demo-lpar.zosmf (zosmf), CFRZG5322E No'.

# Get involved!

## Contribute

- Zowe Explorer, IBM CICS for Zowe Explorer are both open source projects!
- Raise your 🐛, feature requests, PRs :-D
- We'd ❤️ to talk to more CICS developers whatever their current development workflow

## Connect with CICS!

- LinkedIn - <https://www.linkedin.com/groups/4304249/>
- Stack overflow - <https://stackoverflow.com/questions/tagged/cics>
- Github - <https://github.com/zowe/cics-for-zowe-client/discussions>

# Thank you

© Copyright IBM Corporation 2025

IBM and the IBM logo are trademarks of IBM Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on [ibm.com/trademark](http://ibm.com/trademark). This document is current as of the initial date of publication and may be changed by IBM at any time.

THE INFORMATION IN THIS DOCUMENT IS PROVIDED "AS IS" WITHOUT ANY WARRANTY, EXPRESS OR IMPLIED, INCLUDING WITHOUT ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND ANY WARRANTY OR CONDITION OF NON-INFRINGEMENT.

Examples presented are illustrative only. Actual results will vary based on client configurations and conditions and, therefore, generally expected results cannot be provided.

Not all offerings are available in every country in which IBM operates.

Statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

**IBM**