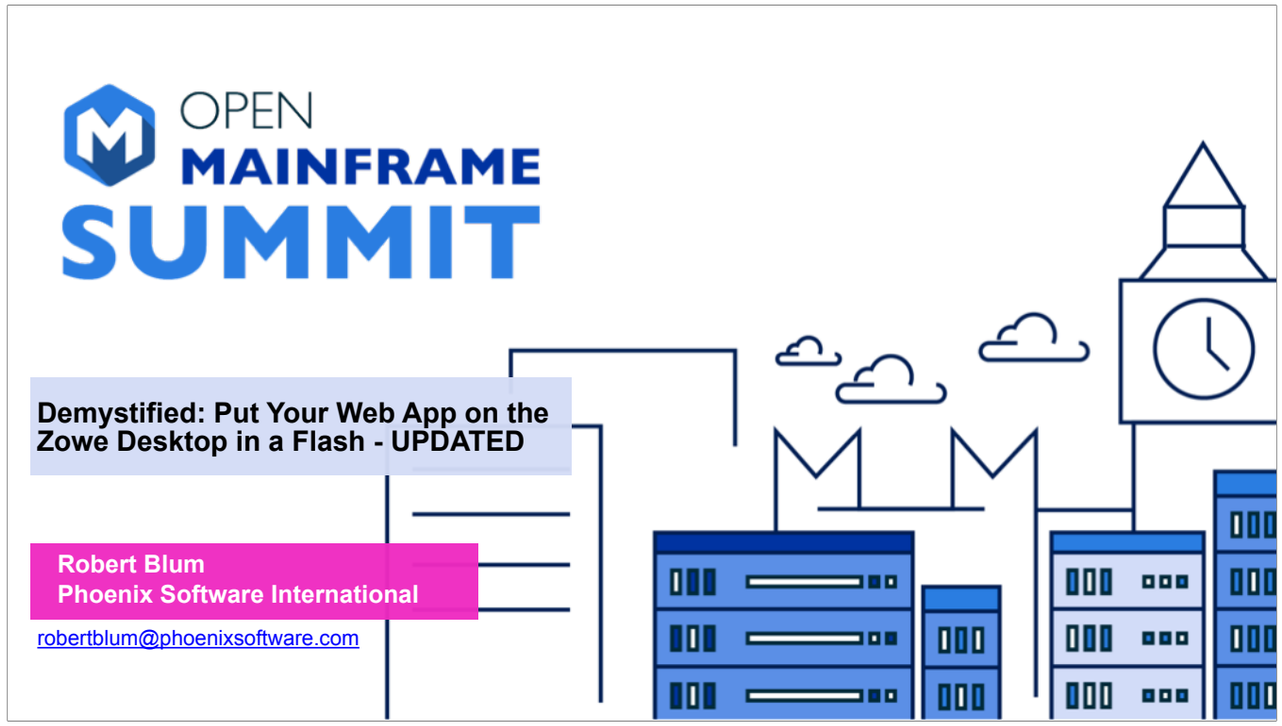
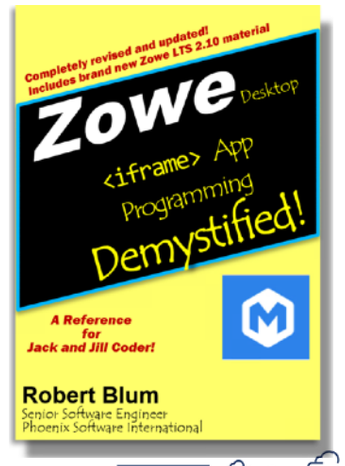


I'm Robert Blum, a developer at Phoenix Software International.

Experience It, Don't Transcribe It

- Think of this as a 10 minute book discussion
  - The conference makes this presentation available for download
  - Includes GitHub repository links
  - Includes Zowe Documentation links
  - My talk covers only highlights
  - Take notes for questions or important concepts
  - Refer to presentation later to do actual work



In this talk, I want to DE-mystify taking a web app
AND
making it display on the Zowe desktop.
It's actually quite easy!
It’s that
the Zowe documentation
is reference material, not a tutorial.
I spent weeks figuring out what turned out to be pretty trivial.
I'm not a z/OS unix programmer. I'm a web developer AND I sometimes use the mainframe.
I've analyzed what's needed and what YOU can ignore.
If you walk away with one thing, I want it to be that setting up your web app as an iFrame web app is easy.
The slides, which are available from the conference, are filled with details.
Think of the slides as a book.
This talk gives you the Cliffs Notes version.

Lessons Learned

- Your web app on Zowe Desktop without learning the Zowe API
- Minimum you need to know to succeed
  - z/OS unix command essentials
  - GitHub and cloning
  - Finding and cycling the Zowe desktop STC job
  - Finding necessary paths
  - The **zwe** command and installation
  - What to modify or remove
- Learn what I learned the easy way
  - No reading reference manuals written for the already expert
  - No experimenting to figure out what you missed

I'm going to cover only what you need, including:

A list of z/OS unix commands
Using GitHub and cloning the projects
Finding and cycling the Zowe desktop STC job
Finding the necessary paths needed for commands
Explaining the z-w-e command
AND
Explaining what to modify and what to remove.

Why is the Zowe Desktop Attractive?

- Securely available anywhere corporate access to a web address is available
- Strategic direction for many z/OS clients
- Management may want (or policy may dictate) existing web apps be available on the new platform

Some reasons you'll port to Zowe.

Reduce, Reuse, Recycle

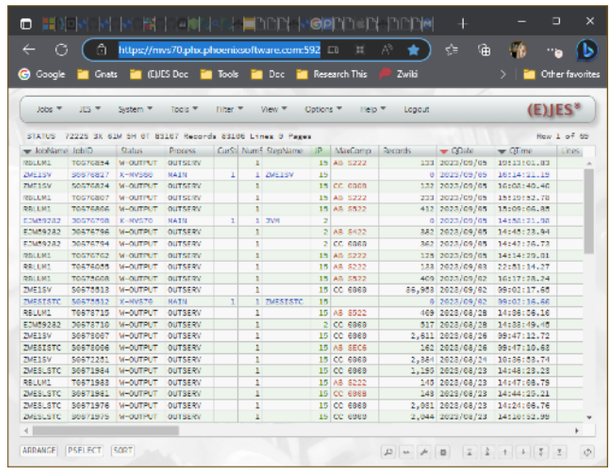
- "Learn how reducing, reusing, and recycling can help you, your community, and the environment by saving money, energy, and natural resources." – EPA website
  - Getting your program running in a flash saves:
    - Your company money
    - Energy powering your computers
    - Resources (i.e., *you*) for more difficult problems.

- Why do this?
  - Existing apps are tested and iterated for the corporate environment
  - Don't reinvent the wheel: No need to implement for a new API if old one provides solution
  - The less time spent porting, the more resources for other projects
  - No need to program in Typescript; use what you used for your web app

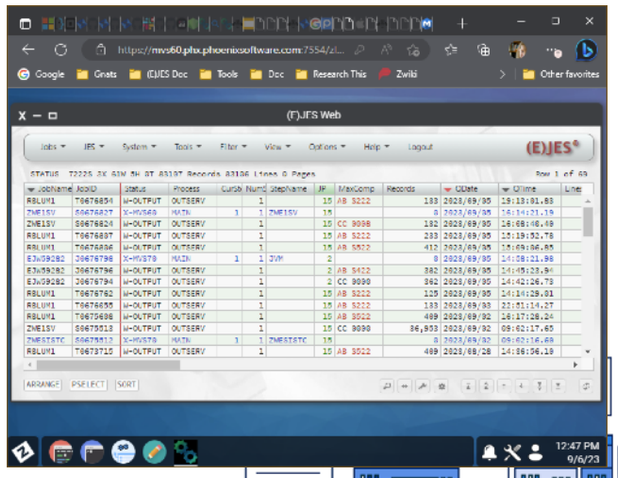
This whole presentation boils down to, "Don't reinvent the wheel."

Case in Point #1: Hugely Complex

Browser (E)JES® Web



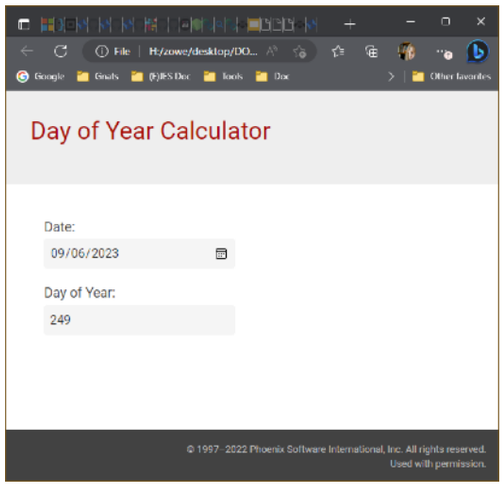
Zowe Plug-in (E)JES Web



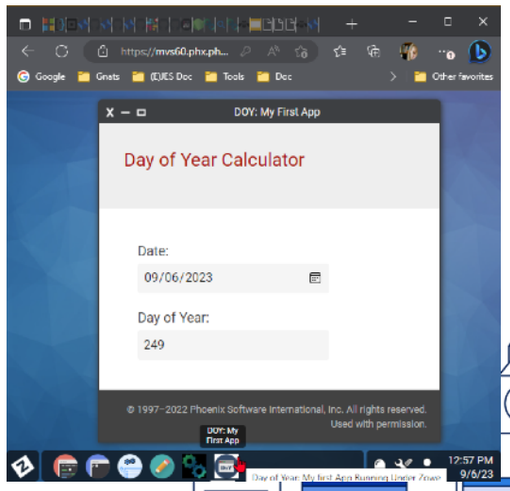
Case in point.
On the left is a very complex web app,
AND
on the right
The same app running on the Zowe desktop.

Case in Point #2: Simple

Browser Day of Year

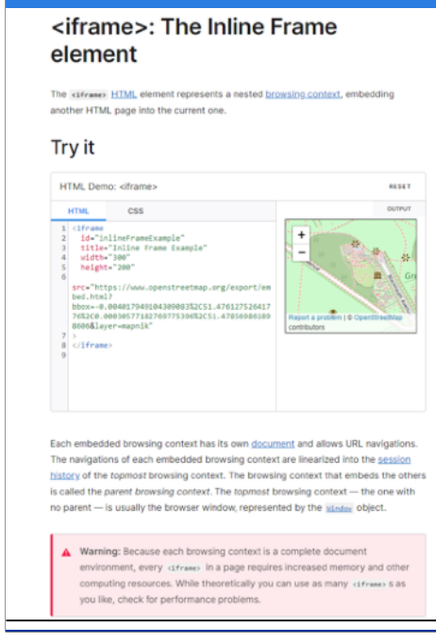


Zowe Plug-in Day of Year

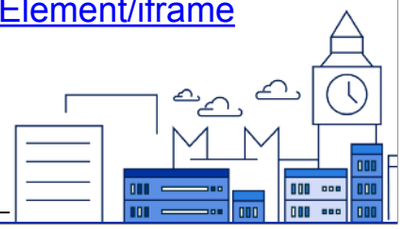


Another case in point. It can be any web app, even a simple one.

Basics: <iframe>



- A Zowe iframe app puts your web app into an isolated environment so that it is unable to compromise Zowe security in any way
- Cross-domain URL restrictions apply
- Image to left is from
  - https://developer.mozilla.org/en-US/docs/Web/HTML/Element/iframe



Why an iframe?
Putting your web app inside an iframe tag,
you isolate it from the Zowe desktop
with no chance of compromising Zowe security.

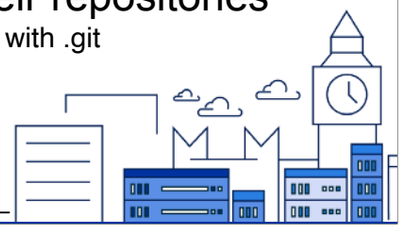
z/OS Unix Essentials

- Unix primer for the non-cognoscenti (me)
  - Upper and lowercase matter; get it correct!
  - Ask a colleague how to copy and paste in your Linux terminal shell
  - Linux commands to know:
    - cat *file* - Display a file
    - cd - Change directory, just like in PowerShell or the DOS prompt
    - chtag - Change z/OS tag attributes to iso8859-1 or binary
    - find *root-path* -name "*text*" -exec ls -ld {} \ - Find text in a directory and subdirectories
    - ls - List directory
      - ls -laF - Full information
      - ls -laT - Tag information
    - man *command* - Display manual page, documentation for a command
    - more *file* - Display a file one screen at a time, using **q** to quit
    - pwd - Display current directory path
  - More at:
    - https://github.com/IBMRedbooks/USSCheatSheet/blob/main/USSCheatSheet.md

If you are new to z/OS Unix, use this cheat sheet.

Git Essentials

- Popular multiplatform version control program
  - Samples stored in GitHub repositories (PSI's and Zowe's)
  - Install GIT on your machine (https://git-scm.com)
  - Install Nodejs (https://nodejs.org/)
- Clone (download project to your file system)
  - git clone https://github.com/phoenixsoftware/day-of-year-zowe-desktop-plugin
  - git clone https://github.com/zowe/sample-iframe-app
  - git clone https://github.com/phoenixsoftware/ejes-web-zowe-desktop-plugin
- You are working with your own copy of their repositories
  - Delete the hidden .git directory and other things that start with .git
  - What remains are your project files



Install Git if you haven’t already.
I'll be discussing these three projects.
Download them when you start.

Optional: Initialize a New Git Repository

- In the root of your project
  - git init
  - git add . There is a space before the period
  - git status
  - git commit -m "Initial commit of iframe app source"
- Tutorial
  - https://kbroman.org/github_tutorial/pages/init.html

```
C:\Users\Robert\Projects\scratch\my-first-app>git init
Initialized empty Git repository in C:/Users/Robert/Projects/scratch/my-first-app/.git/
C:\Users\Robert\Projects\scratch\my-first-app>git add .
C:\Users\Robert\Projects\scratch\my-first-app>git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   .gitattributes
        new file:   .gitignore
        new file:   CHANGELOG.md
        new file:   CONTRIBUTING.md
        new file:   LICENSE
        new file:   README.md
        new file:   manifest.yaml
        new file:   pluginDefinition.json
        new file:   sample-iframe-app.ppf
        new file:   sonar-project.properties
        new file:   web/assets/.gitkeep
        new file:   web/assets/icon.png
        new file:   web/css/iframe.css
        new file:   web/html/index.html
        new file:   web/js/main.js

C:\Users\Robert\Projects\scratch\my-first-app>git commit -m "Initial commit of iframe app source"
[master (root-commit) 4569f57] Initial commit of iframe app source
 15 files changed, 1513 insertions(+)
 create mode 100644 .gitattributes
 create mode 100644 .gitignore
 create mode 100644 CHANGELOG.md
 create mode 100644 CONTRIBUTING.md
 create mode 100644 LICENSE
 create mode 100644 README.md
 create mode 100644 manifest.yaml
 create mode 100644 pluginDefinition.json
 create mode 100644 sample-iframe-app.ppf
 create mode 100644 sonar-project.properties
 create mode 100644 web/assets/.gitkeep
 create mode 100644 web/assets/icon.png
 create mode 100644 web/css/iframe.css
 create mode 100644 web/html/index.html
 create mode 100644 web/js/main.js
```

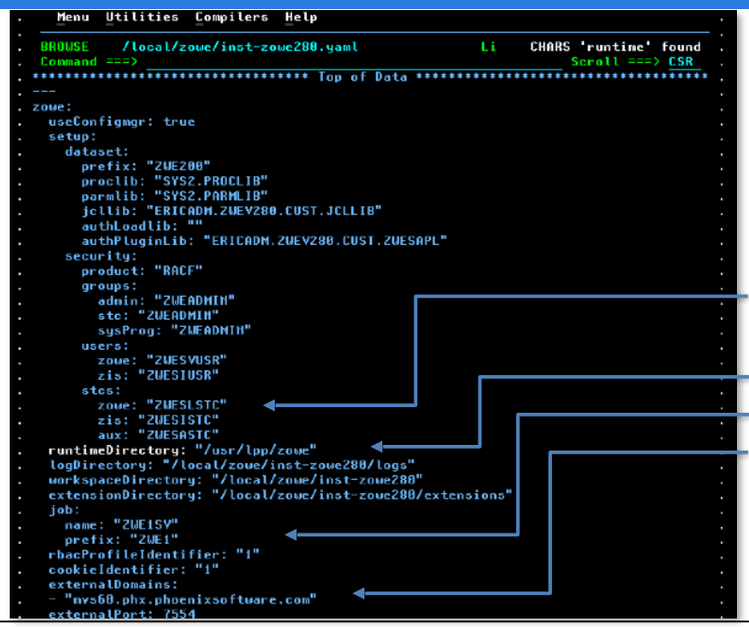
If you want version control for your custom iframe project, this slide will show you how.

Find Essential Zowe Information

- Find the Zowe STC job in JES
  - ZWESLSTC or ZWE1SV if vanilla parameters used
- Find ZWEL0023I in the job output
  - Provides full path on zFS
  - Typical config path: /local/zowe/zowe.yaml
- Browse the zowe.yaml file.
  - Record these directories
    - **runtimeDirectory**: Where Zowe is installed, including all preinstalled apps
    - **workspaceDirectory**: Root where your apps and data can be found
    - **extensionDirectory**: Symbolic link to installed apps
    - **logDirectory**: Log directory where components write error information

You need to know where Zowe keeps its configuration file.
This information is needed to cycle Zowe and install apps.
Find the Zowe STC job in the JES queues.
Look for ZWESLSTC or ZWE1SV.
Find the ZWEL0023I message in the job output.
Browse the yaml file indicated.
Write down these four directories.

zowe.yaml Example



Here's what the yaml file looks like.
Note the directories we need.

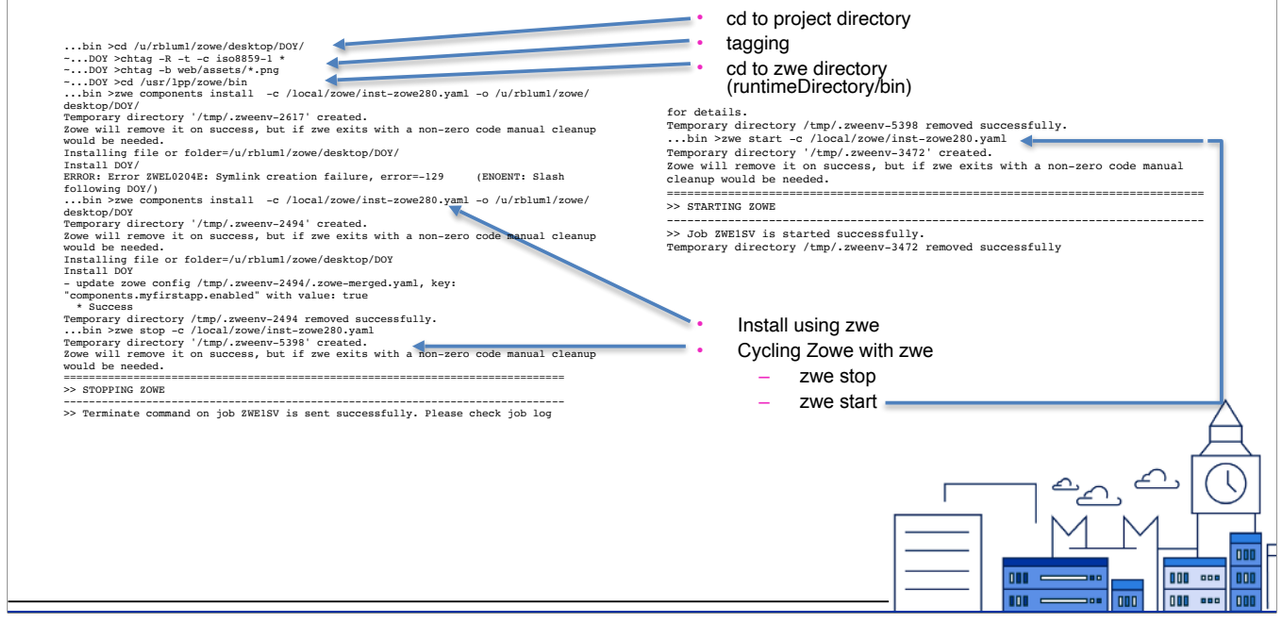
zwe Command Primer

- https://docs.zowe.org/stable/appendix/zwe_server_command_reference/zwe/
- cd to and execute all zwe commands from *runtimeDirectory*/bin
  - Cycle Zowe (enter both commands)
    - zwe stop -c /path/zowe.yaml
    - zwe start -c /path/zowe.yaml
  - Install the app
    - zwe components install -c /path/zowe.yaml -o /path/myapp
  - Run after changing app (recycling Zowe only if metadata changed)
    - zwe components upgrade -c /path/zowe.yaml -o /path/myapp
  - Uninstall app completely
    - zwe components uninstall -c /path/zowe.yaml -o /path/myapp

Note: /path/myapp is the path to your app's directory that contains manifest.yaml

You use the z-w-e command to do all the work.
This slide describes the minimum five commands you'll need:
stop and start to cycle Zowe,
AND install,
uninstall, and upgrade to maintain Zowe.

Sample z/OS Output



```
...bin >cd /u/rbluml/zowe/desktop/DOY/
~...DOY >chtag -R -t -c iso8859-1 *
~...DOY >chtag -b web/assets/*.png
~...DOY >cd /usr/lpp/zowe/bin
...bin >zwe components install  -c /local/zowe/inst-zowe280.yaml -o /u/rbluml/zowe/
desktop/DOY/
Temporary directory '/tmp/.zweenv-2617' created.
Zowe will remove it on success, but if zwe exits with a non-zero code manual cleanup
would be needed.
Installing file or folder=/u/rbluml/zowe/desktop/DOY/
Install DOY/
ERROR: Error ZWEL0204E: Symlink creation failure, error=-129     (ENOENT: Slash
following DOY/)
...bin >zwe components install  -c /local/zowe/inst-zowe280.yaml -o /u/rbluml/zowe/
desktop/DOY
Temporary directory '/tmp/.zweenv-2494' created.
Zowe will remove it on success, but if zwe exits with a non-zero code manual cleanup
would be needed.
Installing file or folder=/u/rbluml/zowe/desktop/DOY
Install DOY
- update zowe config /tmp/.zweenv-2494/.zowe-merged.yaml, key:
"components.myfirstapp.enabled" with value: true
  * Success
Temporary directory /tmp/.zweenv-2494 removed successfully.
...bin >zwe stop -c /local/zowe/inst-zowe280.yaml
Temporary directory '/tmp/.zweenv-5398' created.
Zowe will remove it on success, but if zwe exits with a non-zero code manual cleanup
would be needed.
================================================================================
>> STOPPING ZOWE
--------------------------------------------------------------------------------
>> Terminate command on job ZWE1SV is sent successfully. Please check job log
for details.
Temporary directory /tmp/.zweenv-5398 removed successfully.
...bin >zwe start -c /local/zowe/inst-zowe280.yaml
Temporary directory '/tmp/.zweenv-3472' created.
Zowe will remove it on success, but if zwe exits with a non-zero code manual
cleanup would be needed.
================================================================================
>> STARTING ZOWE
--------------------------------------------------------------------------------
>> Job ZWE1SV is started successfully.
Temporary directory /tmp/.zweenv-3472 removed successfully
```

- cd to project directory
- tagging
- cd to zwe directory (runtimeDirectory/bin)
- Install using zwe
- Cycling Zowe with zwe
  - zwe stop
  - zwe start

Here's a z/OS Unix session.
I've pointed out tagging, which we'll get to later.
I've also installed my sample app.
Since it is an install, I've cycled Zowe by stopping and starting it.

Tagging Files Tutorial

- Files on z/OS must be properly tagged
  - Editing files, especially the icon file, may remove or modify tags
  - Tag the icon file as binary
  - Tag all text files as iso8859-1
  - Use the Zowe desktop **Editor** program to tag files
  - Alternatively use a command in project root (scriptable)
    - chtag -R -t -c iso8859-1 *
    - chtag -b web/assets/*.png

There is a space before the asterisk

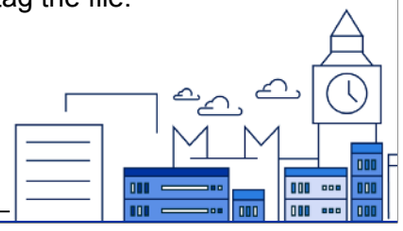
  - Check result
    - ls -laT

Tagging is important.
If you get the file attributes wrong,
Zowe will see garbage.
Worse, the desktop may fail to come up.

Sample z/OS Output (tags)

```
~...DOY >pwd
/u/rblum1/zowe/desktop/DOY
~...DOY >la -laT
total 96
                drwxr-xr-x   4 RBLUM1   DEV        8192 Sep  5 14:07 ./
                drwxr-xr-x   5 RBLUM1   DEV        8192 Sep  5 14:07 ../
t ISO8859-1   T=on  -rwxr-xr-x   1 RBLUM1   DEV         618 Sep  5 15:26 manifest.yaml*
t ISO8859-1   T=on  -rwxr-xr-x   1 RBLUM1   DEV        1243 Sep  5 15:40 pluginDefinition.json*
                drwxr-xr-x   2 RBLUM1   DEV        8192 Sep  5 14:07 schemas/
                drwxr-xr-x   6 RBLUM1   DEV        8192 Sep  5 14:07 web/
~...DOY >cd web/assets/
~...assets >la -laT
total 48
                drwxr-xr-x   2 RBLUM1   DEV        8192 Sep  5 14:07 ./
                drwxr-xr-x   6 RBLUM1   DEV        8192 Sep  5 14:07 ../
- untagged    T=off -rwxr-xr-x   1 RBLUM1   DEV        4621 Sep 16  2022 icon.png*
~...assets >chtag -b icon.png
~...assets >la -laT
total 48
                drwxr-xr-x   2 RBLUM1   DEV        8192 Sep  5 14:07 ./
                drwxr-xr-x   6 RBLUM1   DEV        8192 Sep  5 14:07 ../
b binary      T=off -rwxr-xr-x   1 RBLUM1   DEV        4621 Sep 16  2022 icon.png*
```

- Because I changed icon.png, I had to retag the file.



Here's what tags look like
AND
what they look like when they aren't there.

Install and Run the Day of Year Sample

- We've learned enough to run the sample app on the Zowe desktop
- Procedure summary:
  - Tag files
  - Install using **zwe**
  - Cycle Zowe using **zwe**
  - Run the app
  - Uninstall the app because we plan to customize its identifying metadata

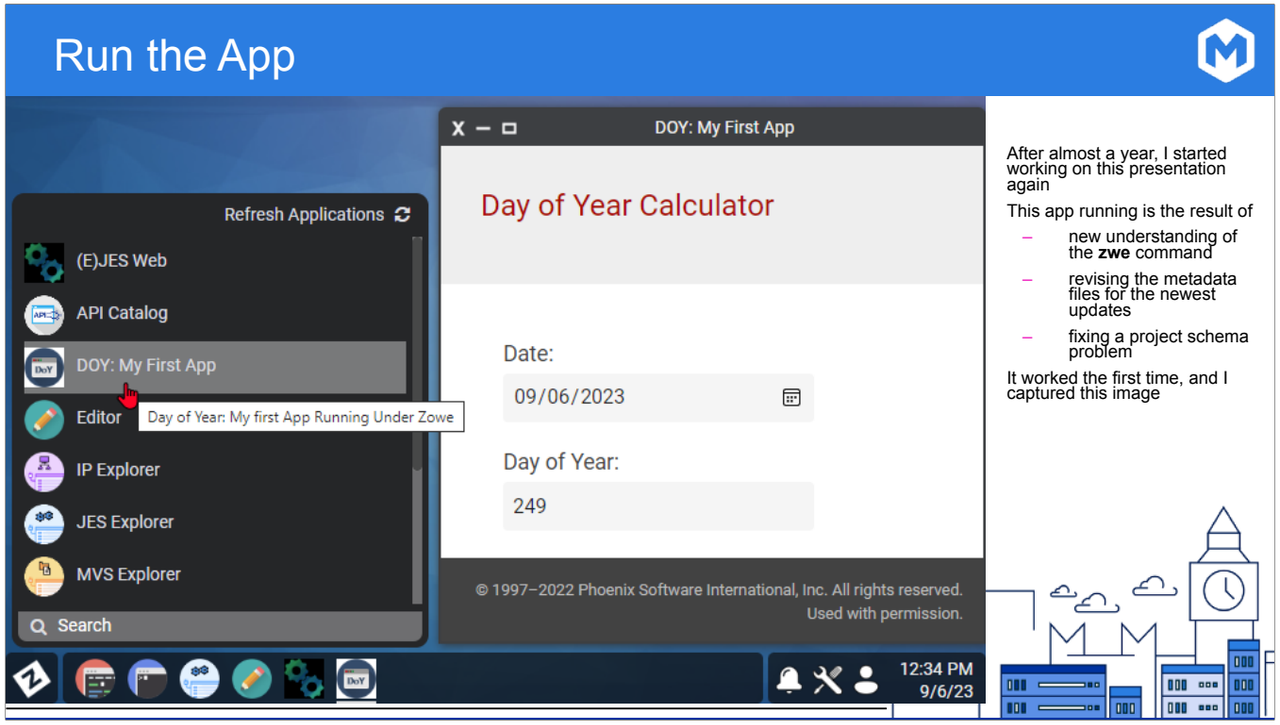
The Day of the Year app project comes ready to install.
Here's a summary of the procedure.
It's really simple.

Tag files,
install using z-w-e,
cycle Zowe,
run the app,
then uninstall it

Install

- Project directory must be accessible to Zowe
- Change directory to the zwe directory discussed on a prior slide
  - cd *runtimeDirectory*/bin
- Install the app
  - zwe components install -c *config* -o *plugin*
    - *config* is the fully-qualified path to the previously discussed zowe.yaml file
    - *plugin* is the path to the directory containing your manifest.yaml file
- Cycle Zowe
  - zwe stop -c *config*
  - zwe start -c *config*
- Logon to Zowe and find the app in the Zowe start menu

In more detail:
Goto the z-w-e command directory.
Install the app using the **z-w-e components instal**l command.
Cycle Zowe.
AND
Run the new app from the
Zowe
start
menu.



Believe it or not, I ended up reading this presentation
AND
following all the procedures I've just outlined.
I hadn't memorized the material.
I found the job,
found the yaml file,
got the paths,
then installed my sample app.
I captured this image when it ran the first time.
I was a happy camper.

Uninstall

- Project directory must be accessible to Zowe
- Change directory to the zwe directory discussed on a prior slide
  - cd *runtimeDirectory*/bin
- Uninstall the app
  - zwe components uninstall -c *config* -o *plugin*
    - *config* is the fully-qualified path to the previously discussed zowe.yaml file
    - *plugin* is the path to the directory containing your manifest.yaml file
- Cycle Zowe to remove Zowe's metadata references
  - zwe stop -c *config*
  - zwe start -c *config*
- App should not appear in the start menu

Because metadata identifies the app to Zowe,
if you are planning on modifying the metadata, you have to uninstall the app.
Think of it like installing Robert.exe on Windows.
If I rename Robert.exe to Rose.exe, Windows will no longer be able to uninstall it.
You're welcome to keep the Day of the Year app, of course!
Just copy the project to a new directory and change the metadata…

Customization Overview

- Applicable to all Zowe iframe apps
- Tips:
  - Customizing Metadata files and web files are related but separate topics
  - You may wish to customize only the metadata before installing, then subsequently customize the web files
    - This minimizes the errors you may have to correct at one time
  - Metadata changes are visible:
    - On the start menu
    - By looking at the properties of your app
  - Metadata errors will display in the Zowe STC job output
  - Learn at your own rate

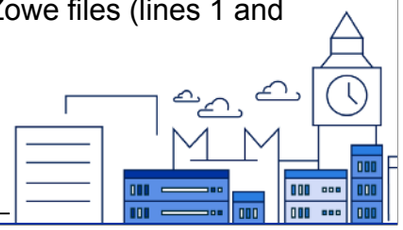
A Zowe iframe project consists of two sets of files:
Metadata, which I'll cover next,
AND
web app files.
I suggest modifying the metadata first rather than all of it at once.
Less problems to troubleshoot that way.
Changes to metadata show up in the start menu and app properties.

Essential YAML File Notation

- The following is valid YAML

```
schemas:
  configs: schemas/trivial-schema.json
schemas.configs: schemas/trivial-schema.json
```

- Lines 1 and 2 above are the same as line 3 alone
- Whitespace indentation is meaningful and signifies levels
  - Inserting a line 3 of " enable: true" is the same as specifying "schemas.enable: true".
- Zowe documentation uses dot notation for brevity, as do I
- I suggest you use the same notation style you find in the Zowe files (lines 1 and 2) for consistency
- This link is one of many YAML primers you can find
  - https://getopentest.org/reference/yaml-primer.html



I use YAML dot-notation in my slides.
AS DOES
Zowe documentation.

Line 3 of the illustration is "dot-notation"
AND
is the same as lines 1 and 2.

Metadata vs Web App

- The iframe app bundle consists of metadata and your web app
- The **zwe** command installs and manages both
- Your metadata
  - Configures Zowe to use your app
  - It identifies the app to Zowe, so you need to uninstall before modifying the metadata that identifies the app to Zowe
  - You must cycle Zowe to see changes
- Your web app is
  - A web app!
  - You must refresh the Zowe desktop in the browser to see changes, exactly like any other web app

Metadata describes your app to Zowe.
You have to cycle Zowe anytime you change metadata.
I hope you know what a web app is…
NEXT SLIDE

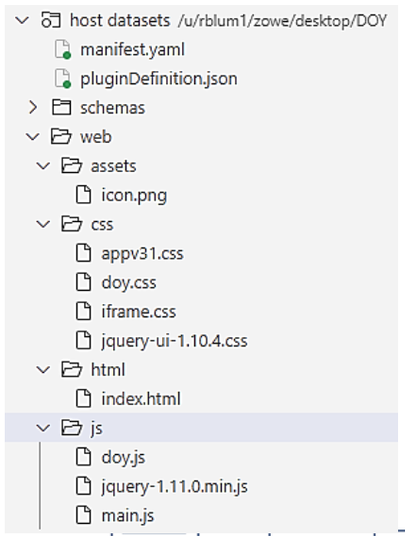
Metadata vs Web App

- The iframe app bundle consists of metadata and your web app
- The **zwe** command installs and manages both
- Your metadata
  - Configures Zowe to use your app
  - It identifies the app to Zowe, so you need to uninstall before modifying the metadata that identifies the app to Zowe
  - You must cycle Zowe to see changes
- Your web app is
  - A web app!
  - You must refresh the Zowe desktop in the browser to see changes, exactly like any other web app

I've highlighted this line because it isn't obvious.
Your Zowe iframe app is just a web app.

Day of Year File Layout

- Metadata
  - manifest.yaml
  - pluginDefinition.json
  - schemas (folder)
- Canonical web app (may be customized)
  - icon.png
  - iframe.css
  - index.html
  - main.js
- Supporting web app files
  - Everything else you add
  - Note that I've added jquery files for my own use
  - The **doy.js** is the core Javascript functionality of my app
- This is the layout used by Zowe
  - Use it, unless you have a good reason not to
  - I keep my repository files on a different filesystem
  - I edit on the host, but use Git and GitHub on Windows
  - I need to synchronize the filesystems
  - Remember you may need to tag files



Here's the canonical layout of the iframe app.
Unless you have a good reason NOT TO, stick to it.

Managing Metadata

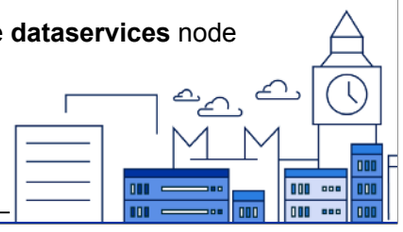
- The repositories I've provided are already customized
  - Use this information to understand and customize your app
  - Use **zwe stop** and **zwe start** any time you modify metadata
- manifest.yaml
  - "The Zowe server component manifest file defines the name and purpose of the component. It also provides information about how this component should be installed, configured, and started." From https://docs.zowe.org/stable/appendix/server-component-manifest/
  - You need to change only: *name*, *id*, *version*, and *title*
  - Modifying *title*, *description*, and *repository* is recommended
  - The property ***schema.configs*** is critical
    - It must point to schema/trival-schema.json
    - It causes any valid JSON in pluginDefinition.json to validate without error
    - Validation errors prevent app installation or Zowe startup
    - The **schemas** folder appears in the DOY repository and other sample repositories for this reason

Anytime you modify metadata, you need to run z-w-e stop and z-w-e start.
The Manifest.yaml file describes your app to Zowe.
Here's the things you'll change; read them later.

DO
NOT
change the value of schema.configs.
If you do, chances are your app won't validate and the desktop will ABEND.

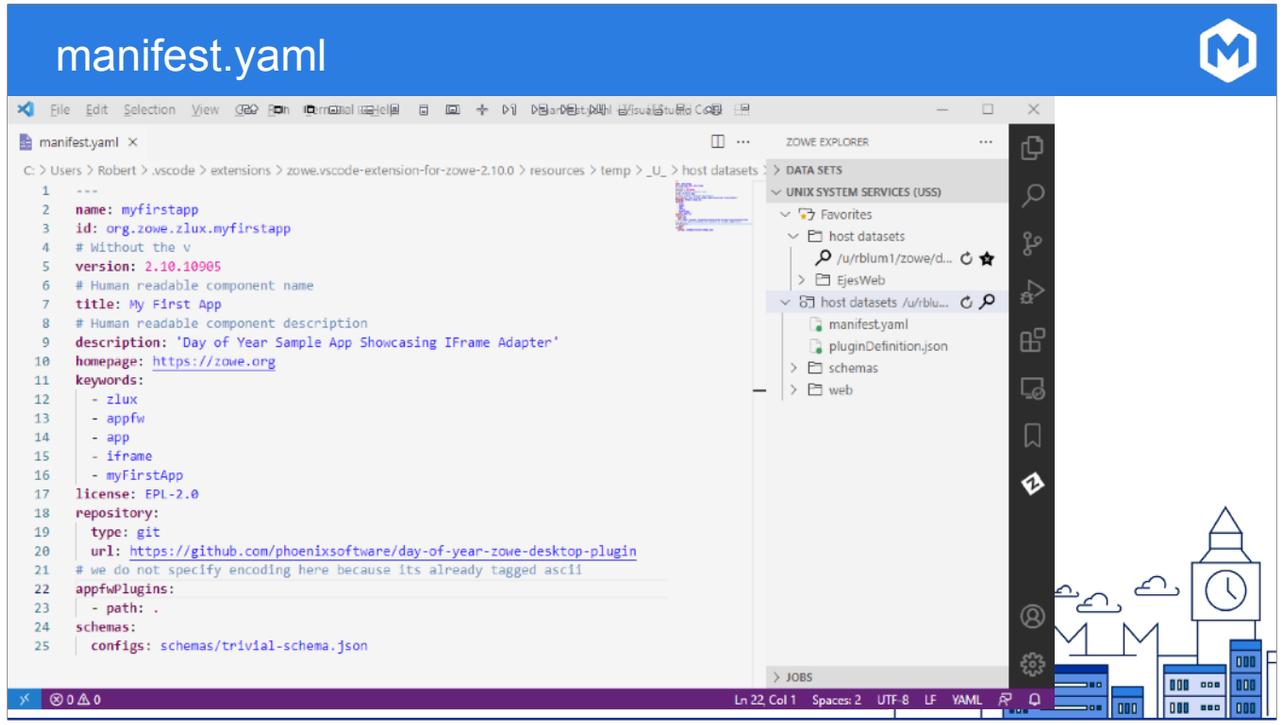
Managing Metadata continued

- pluginDefinition.json
  - This file describes an application Plugin to the Zowe Application Server. From https://docs.zowe.org/v2.8.x/extend/extend-desktop/mvd-plugindefandstruct/
  - You need to change only these properties:
    - identifier
    - pluginVersion (zoweMajor.zoweMinor.yourVersion, e.g., 2.10.0906)
    - webContent.launchdefinition.pluginShortNameKey (short tooltip text)
    - webContent.launchdefinition.pluginShortNameDefault
    - webContent.descriptionkey
    - webContent.descriptionDefault (long tooltip text)
  - Depending on the sample, you'll probably want to remove the **dataservices** node
    - I'll mention more about this in advanced topics



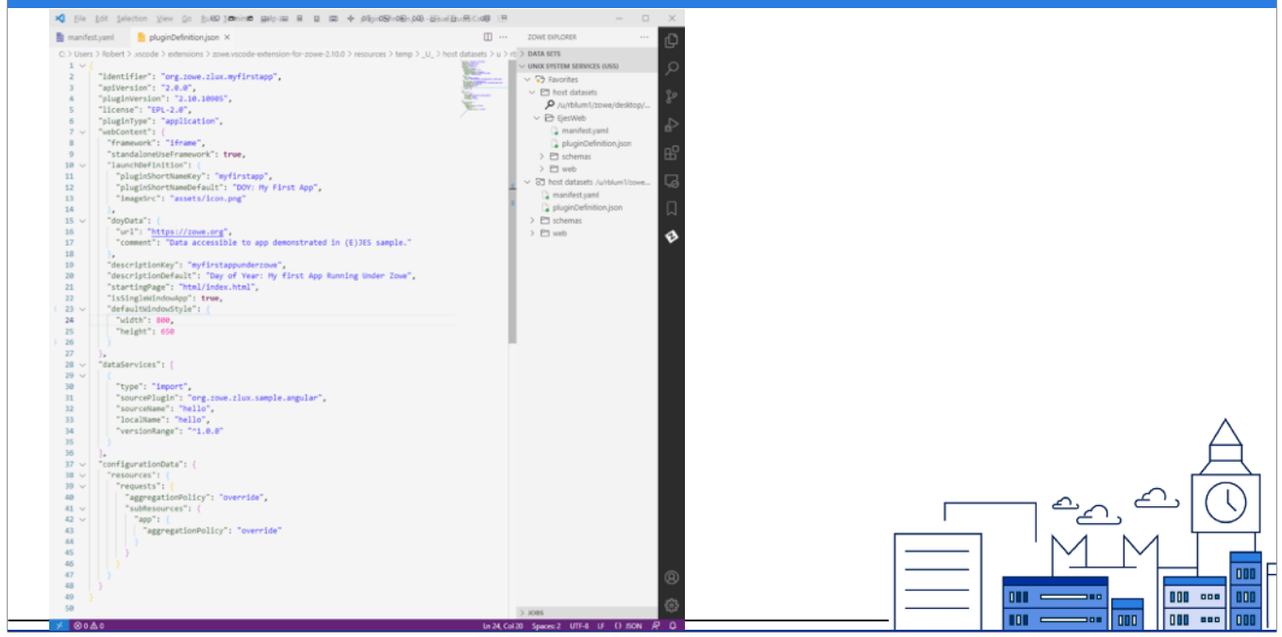
The pluginDefinition.JSON files describes the app and its properties to Zowe.
Here's a list of things to change. Read it later.
Dataservices is a bit arcane
AND I'll discuss it later.

manifest.yaml



Here's a changed manifest.yaml file.

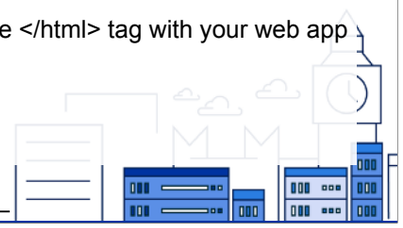
pluginDefinition.json



Here's a changed pluginDefinition.json file.

Managing the Web App

- Use Zowe app file layout (see earlier slide)
- Copy supporting files to /web/assets, /web/css, /web/html, and /web/js
- Modify
  - /web/html/index.html as discussed below to run your code
  - /web/css/iframe.css if necessary for your code
  - /web/assets/icon.png
- Specific changes
  - The DOY sample includes support files
    - If you remove the DOY code in index.html, you can remove everything but iframe.css in /web/css and main.js in /web/js
  - In the index.html file, find the </head> element
    - Replace the DOY code spanning from the </head> tag to the </html> tag with your web app
    - Change the <title> tag
    - Add <link> tags for your CSS
    - Add <script> tags to include your Javascript files



Let's talk web apps.
Copy your web app files into these standard directories.
Modify the index.html file as shown.
You'll probably want to change the .css file or link your own.
Save modifying the icon until you've gotten the app up and running.
The rest is a list of specific changes to the index.html

Managing the Web App continued

- Additional specific changes
  - The icon supplied is a placeholder
    - Insert your own graphic, optionally putting your image in a round cutout as suggested by this spec: https://docs.zowe.org/stable/contribute/guidelines-ui/appicon/
  - I've left **<!-- UNIT TEST -->** functions from the original iframe sample app that exercise features like notification for reference
    - Remove them if you don't want them
  - If the location of your starting HTML page is not the default **html/index.html**
    - Change the **webContent.startingPage** property in pluginDefinition
  - If the location of your app icon is not the default **web/assets/icon.png**
    - Change **webContent.launchDefinition.imageSrc** in pluginDefinition.json

Same on this slide.

Changes Made to index.html

Add <script> and <link> elements

```
<html>
  <head>
    <title>My First App</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="../css/iframe.css">
    <script>
      if(exports){
        var ZoweZLUX_tempExports = exports;
      }
      var exports = {"__esModule": true};

    </script>
    <script type="text/javascript" src="../../../../../lib/org.zowe.zlux.logger/0.9.0/logger.
    js"></script>
    <script type="text/javascript" src="../../../org.zowe.zlux.bootstrap/web/iframe-adapter.
    js"></script>
    <script type="text/javascript" src="../js/main.js"></script>
    <!-- Demystified Presentation: Added metadata for DOY web app. -->
    <meta name="theme-color" content="#ffffff">
    <link rel="stylesheet" href="../css/jquery-ui-1.10.4.min.css" type="text/css">
    <link rel="stylesheet" href="../css/appv31.css" type="text/css">
    <link rel="stylesheet" href="../css/doy.css" type="text/css">
    <script charset="utf-8" src="../js/jquery-1.11.0.min.js"></script>
    <script charset="utf-8" src="../js/doy.js"></script>
    <!-- End of added elements -->
```

Add html between <body> and </html>

```
  </head>
    <!-- Demystified presentation: Added DOY HTML for Web App
         Note that much of the Zowe iframe code above isn't needed, but it
         it is there in case you need to  leverage it with your own code.
    -->
    <body style="overflow:hidden;" id="doy">
      <div id="webapp"></div>
      <header class="hero" >
          <h1 >Day of Year Calculator</h1>
      </header>
      <main id="content" class="grid-container" style="margin-left:50px;">
        <form>
          <label>Date:</label>
          <input type="date" autofocus>
          <label>Day of Year:</label>
          <input type="number">
        </form>
      </main><!-- div id="content" -->
      <footer id="doyfooter" >
        <div id="doycopyright" class="cell large-5 copyright">
          &copy; 1997&ndash;2022 Phoenix Software International, Inc. All rights reserved.
          <br>Used with permission.
        </div>
      </footer>
    <!-- End of added DOY content. -->
    </body>
</html>
```

Here's a changed index.html file.

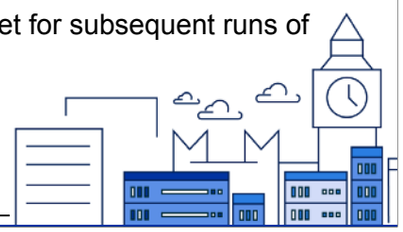
Install the Customized App

- Follow the tagging and installation instructions
- Deal with errors
  - Zowe desktop didn't come up?
    - Check the Zowe STC job output for validation errors caused by bad metadata
    - Did you try to change the schema? Don't do that!
    - Search the Zowe STC job output for your app's name
    - Are you receiving error messages about loading a file like manifest.yaml, e.g. ZWEL0066W?
      - Tag your files again
  - Your app isn't in the Zowe start menu?
    - Check the Zowe STC job output for minor errors
    - Your issue is most likely the metadata files, but it could be misplaced or missing web app files
- Note
  - Once you've finished customizing your metadata, you no longer need to uninstall it to before you modify it; you can upgrade instead

Installing the custom app is pretty much the same as what we did before.
Note that I've included a list common errors and what to do about them on this slide.
I learned what to do the hard way, of course.

Debugging

- Use the JavaScript debugger built into your browser, exactly like for any other web app
  - Run your application
  - Display developer tools and then the debugger
  - Open the **webpack** folders (typically at the bottom) until you find your app
  - Set break points
    - At least in Firefox, breakpoints, watches, etc., stay set for subsequent runs of a Zowe iframe app



I was surprised to learn that debugging a Zowe iframe app was
almost
identical to debugging my original web app.
The only major difference was finding my sources.
There's also the issue of inspecting an iframe,
but you'll figure that out pretty quickly.
The big takeaway here is
you
don't
need
to worry.

Updating a Modified App

- Change directory to the zwe directory discussed on a prior slide
  - cd *runtimeDirectory*/bin
- Upgrade the app
  - zwe components upgrade -c *config* -o *plugin*
    - *config* is the fully-qualified path of the previously discussed zowe.yaml file
    - *plugin* is the path to the directory containing your manfest.yaml file
- Changed metadata?
  - Cycle Zowe to update Zowe's metadata references
    - zwe stop -c *config*
    - zwe start -c *config*
- Web app files changed but not metadata?
  - Logoff in the Zowe desktop browser tab
  - Refresh the tab
  - Log on

When you make a change to your Zowe app,
you issue the **z-w-e components upgrade** command,
logoff the Zowe desktop,
refresh the browser tab,
logon to Zowe again,
AND
test.

Advanced: Insert a Web App from a Server URL

- Provide your server URL so it is available in index.html
  - Embed your server URL in the **webContent** property of pluginDefinition.json, or hard-code in index.html
  - This is fully implemented in the (E)JES Web GitHub repository
- Change tag id values as necessary to position your web app
- Add JavaScript that creates an iframe that fills the Zowe window:

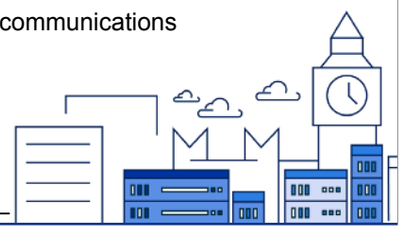
```
if ( configuredDomainEjesWeb ) {
  let node = document.createElement('iframe');
  node.setAttribute('style', 'position: fixed;top: 0px;bottom: 0px;right: 0px;width: 100%;
  border: none;margin: 0;padding: 0;overflow: hidden;z-index: 999999;height: 100%;');
  node.setAttribute('id', 'ejesweb_iframe');
  node.setAttribute('src', configuredDomainEjesWeb + '/EjesWeb/');
  document.getElementById('ejesweb_reference_frame').appendChild(node);

  __ejesZlux = document.getElementById('ejesweb_iframe').contentWindow;
```

You can also insert a web app from a server URL.
I'll leave that for you to read.
It's what I did for that hugely complex company web app I showed at the beginning.

Advanced: Customize the Official Zowe iframe Sample

- Procedure
  - Remove the .github, build, and dco-signoff directories. You'll make your own repository.
    - Delete the .gitattributes .gitignore, CHANGELOG.md, CONTRIBUTING.md, sample-iframe-app.ppf, and sonar-project.properties files
  - If you wish to run the official sample
    - Install the required **angular sample app** that is provided but not preinstalled with Zowe 2.x
      - zwe components install -c *config* -o *runtimeDirectory*/components/app-server/share/sample-angular-app
      - Follow the tagging and installation steps
      - Cycle Zowe
      - Run the angular sample from the start menu
  - Modify the README.md and LICENSE files as necessary
  - Follow previous steps to change the metadata files and to insert your own web code
  - The angular sample above demonstrates iframe inter-application communications
    - If you don't plan to use this feature, remove:
      - The **dataServices** property from pluginDefinition.json
      - The data services code from index.html



Now that we gotten the Day of the Year app running,
I've extended the procedure to the official Zowe iframe sample.
There's a gotcha, however.
It requires the **angular sample app** be installed, but it isn't explained.
Fortunately, that app is included with the Zowe distro.
This slide explains what you need to install it.

The official sample includes "unit tests."
These demonstrate things like notifications,
Zowe-style menus,
AND
Zowe inter-application communication.
You may later become interested in these things,
OR NOT
I'm betting your web app already has it all, which is the point of this talk

Advanced: (E)JES Sample

- (E)JES Web iframe app framework Zowe plug-in
  - https://github.com/phoenixsoftware/ejes-web-zowe-desktop-plugin.git
  - **Note:** *Running* this plug-in requires (E)JES and its REST API, *however*, the code is still usable as a starting point for your application!
  - This plug-in uses **postMessage** to message between Zowe and iframe code (web/js/webapp.js and index.html). The served up web app knows it runs under Zowe and behaves differently. It accesses persistence data and includes a function to request the Zowe plug-in to do work.
  - Functions in main.js are modified to return data via callback functions. instead of inserting data in HTML tags. Console log messages aid tracing.
  - Requires (E)JES V6R3, expected GA September 29, 2023.
  - It is in active development—I will be updating with further features in the near future!

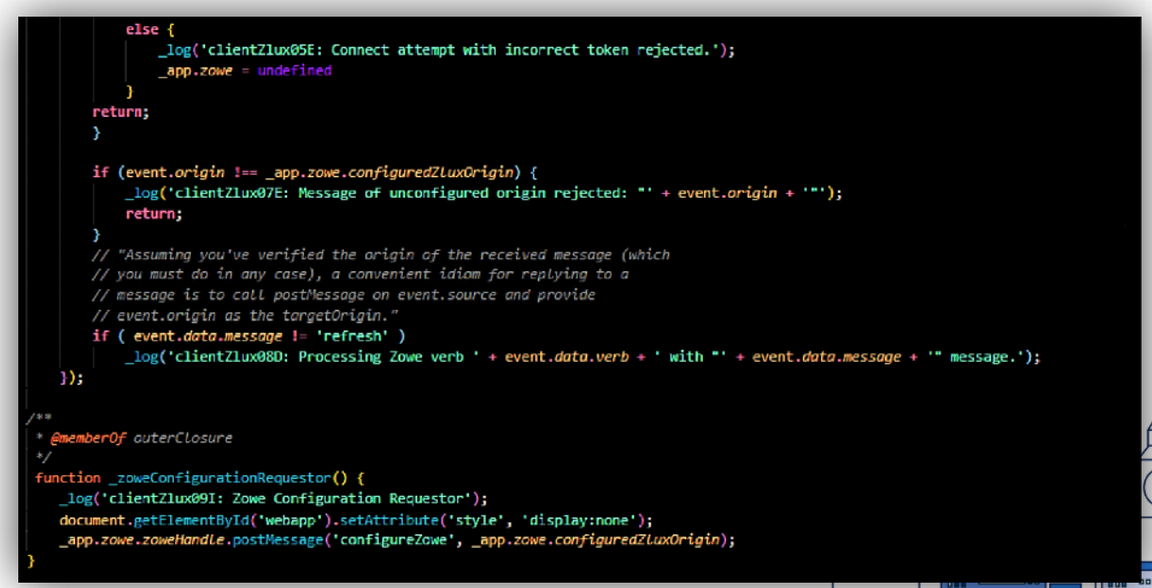
At Phoenix Software, beyond our Zowe CLI offerings,
we decided we wanted our (E)JES product on the Zowe desktop.
You're welcome to examine the Zowe interface part of that.
To actually run it, you'll require (E)JES installed on z/OS.
Even not running (E)JES,
the repository download demonstrates communicating between the iframe
AND
a server app,
which you can modify and use.
The following two images are snapshots from the licensed part of (E)JES web,
showing only the code that pertains to communicating with Zowe.
Match that with the code in the (E)JES Web Zowe desktop app repository to paint yourself a full picture.

`webapp.js` Part One

```
// Zowe-Listener: Called sometime after index.html loads in a Zowe iframe (if ever).
if ( window.self !== window.top ) // Don't install Zowe listener unless in an iframe.
    $(window).on('message onmessage', function (e) {
        // Do we trust the sender of this message?
        var event = e.originalEvent; // https://stackoverflow.com/questions/9904490/jquery-doesnt-support-postmessage-event
        _log('clientZlux01D: Origin is ' + event.origin + '.  Data is ' + JSON.stringify(event.data) + '".');

        if ( ! _app.zowe ) {
            _app.zowe = {
                configuredZluxOrigin: 'https://mvs60.phx.phoenixsoftware.com:8544', // TODO: From persistence data on host.
                configuredZluxOrigin: 'https://mvs70.phx.phoenixsoftware.com:7556', // TODO: From persistence data on host.
                isZowefied: false,
                zoweHandle: null,
                key: 'ZoweTabData',
                token: 'zoweZluxEL'
            }
            if (event.origin !== _app.zowe.configuredZluxOrigin) {
                _log('clientZlux03E: Message of unconfigured origin rejected: "' + event.origin + '"');
                _app.zowe = undefined
                return;
            }
            if ( event.data.token == _app.zowe.token && event.data.verb == 'intialize' || event.data.verb == 'refresh' ) {
                _log('clientZlux04I: (E)JES Web running under Zowe.  Connector initialized.  Zowe handle set. ' + event.data.message )
                event.source.postMessage('ejesweb', event.origin);
                _app.zowe.isZowefied = true;
                _setSessionStorage(_app.zowe.key, JSON.stringify(_app.zowe));
                _app.zowe.zoweHandle = event.source; // Handle cannot be included is local storage.  Zowe posts this regularly to us.
            }
            else {
```

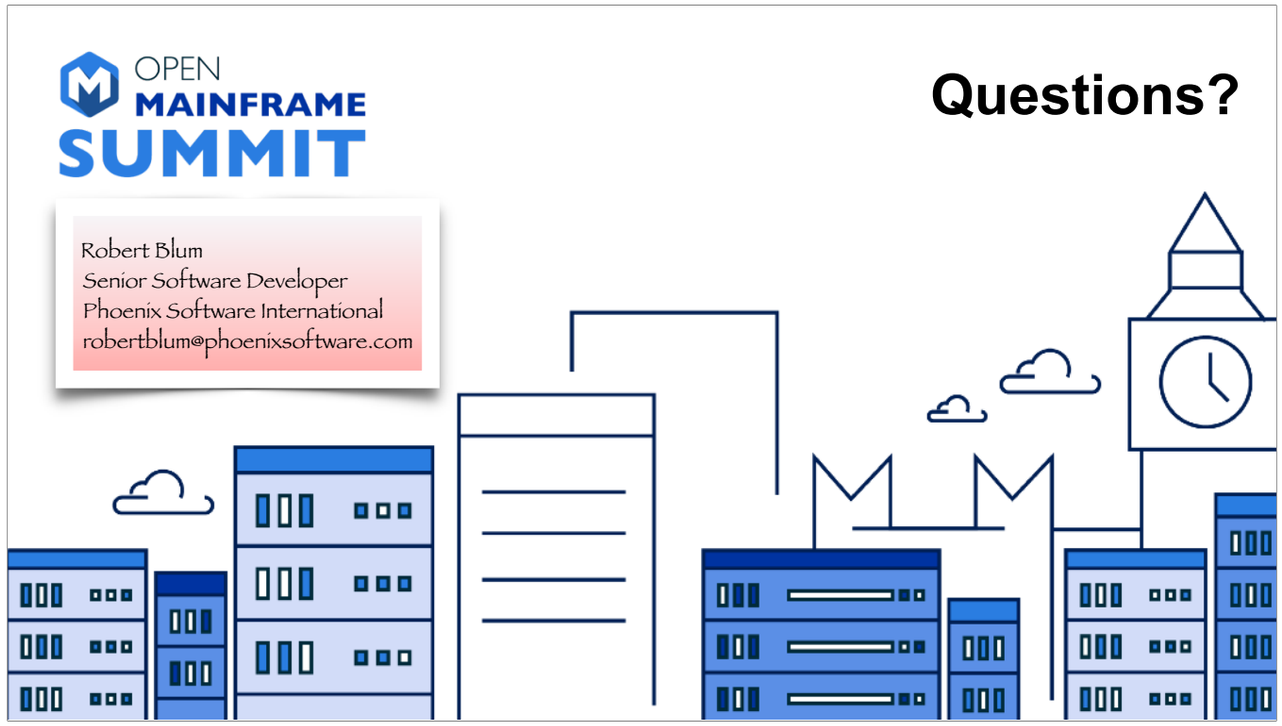
webapp.js Part Two



```
            else {
                _log('clientZlux05E: Connect attempt with incorrect token rejected.');
                _app.zowe = undefined
            }
        return;
        }

        if (event.origin !== _app.zowe.configuredZluxOrigin) {
            _log('clientZlux07E: Message of unconfigured origin rejected: "' + event.origin + '"');
            return;
        }
        // "Assuming you've verified the origin of the received message (which
        // you must do in any case), a convenient idiom for replying to a
        // message is to call postMessage on event.source and provide
        // event.origin as the targetOrigin."
        if ( event.data.message != 'refresh' )
            _log('clientZlux08D: Processing Zowe verb ' + event.data.verb + ' with "' + event.data.message + '" message.');
    });

/**
 * @memberOf outerClosure
 */
function _zoweConfigurationRequestor() {
    _log('clientZlux09I: Zowe Configuration Requestor');
    document.getElementById('webapp').setAttribute('style', 'display:none');
    _app.zowe.zoweHandle.postMessage('configureZowe', _app.zowe.configuredZluxOrigin);
}
```



And that's all folks.