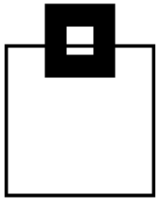
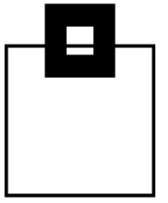


The zGUI (r)evolution -
What is ZOWE going to do for me?

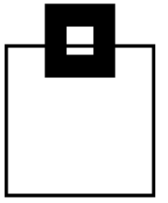
Roy Boxwell, Software Engineering GmbH

Agenda



- GUIs in the past
- Zowe ecosystem overview
- Zowe differentiation to prior GUIs
- Zowe components
- Our Journey
- How it looks
- Live demo

Agenda

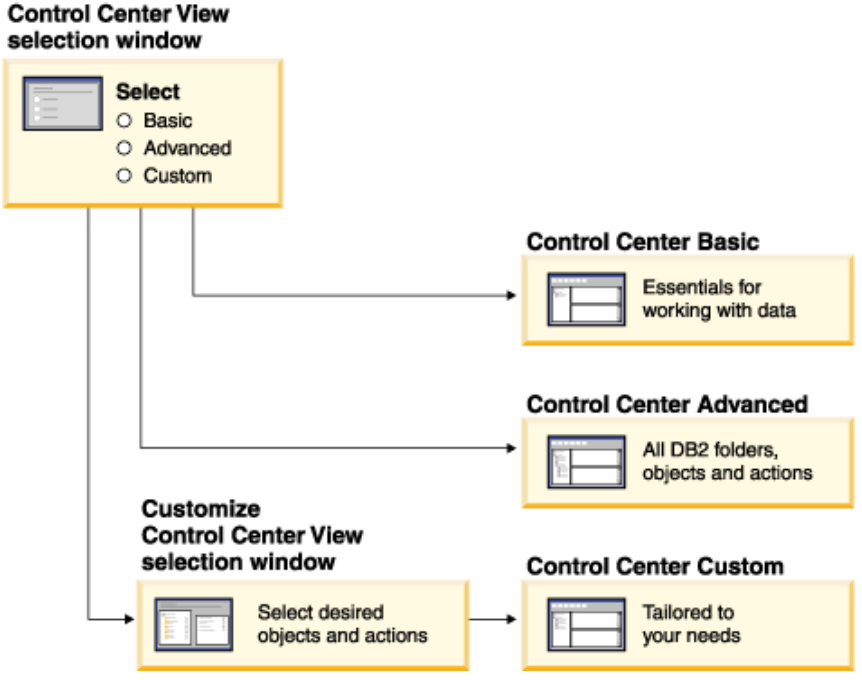


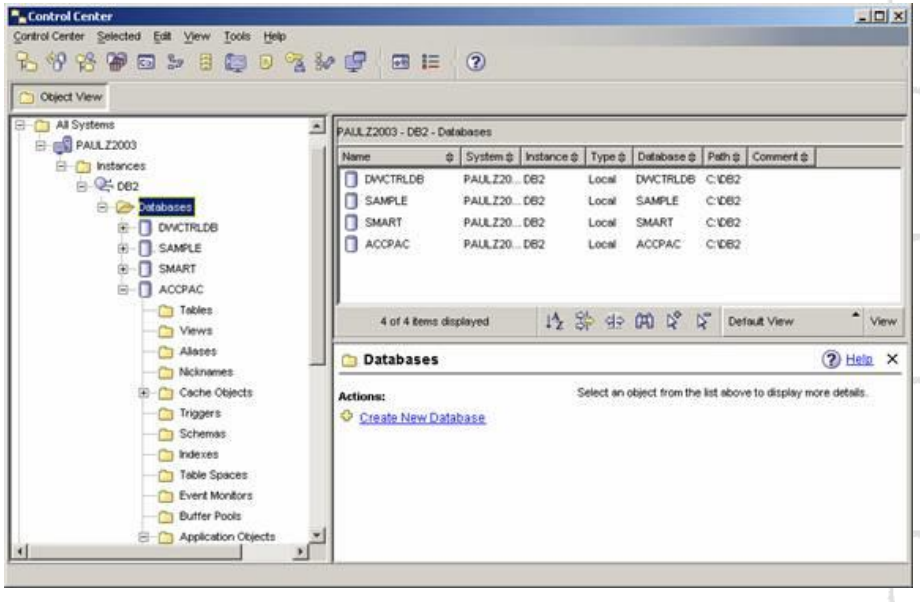
- GUIs in the past
- Zowe ecosystem overview
- Zowe differentiation to prior GUIs
- Zowe components
- Our Journey
- How it looks
- Live demo

GUIs in the past

Db2 Control Center (Db2cc)

- Introduced with Db2 LUW 5, but also able to connect to Db2 z/OS
- A Windows/Linux fat client using Db2 connect and stored procedures
- Manages and administers Db2 systems and objects

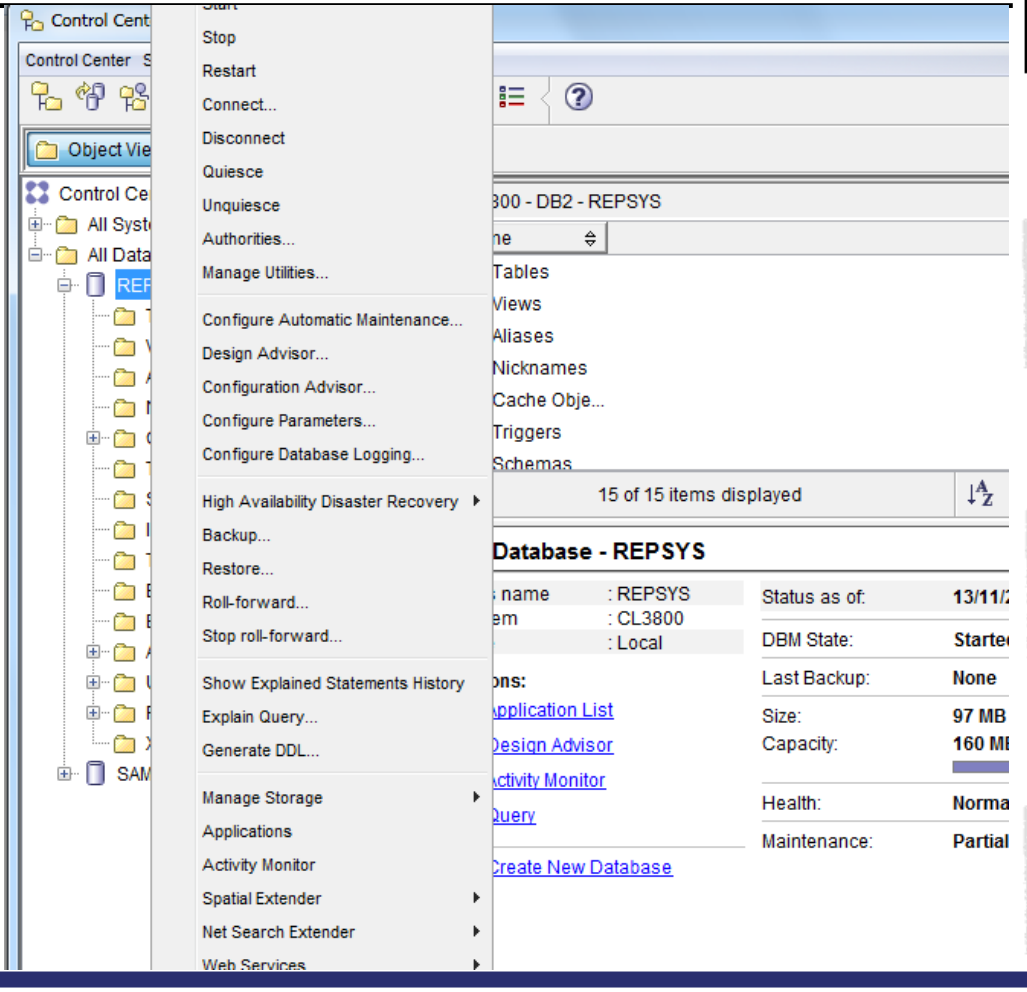




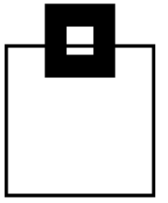
GUIs in the past

Db2 Control Center (Db2cc)

- Can also open other centers to
  - optimize queries, jobs, and scripts
  - perform data warehousing tasks
  - create stored procedures
  - work with DB2 and IMS commands



GUIs in the past



Db2 Control Center (Db2cc)

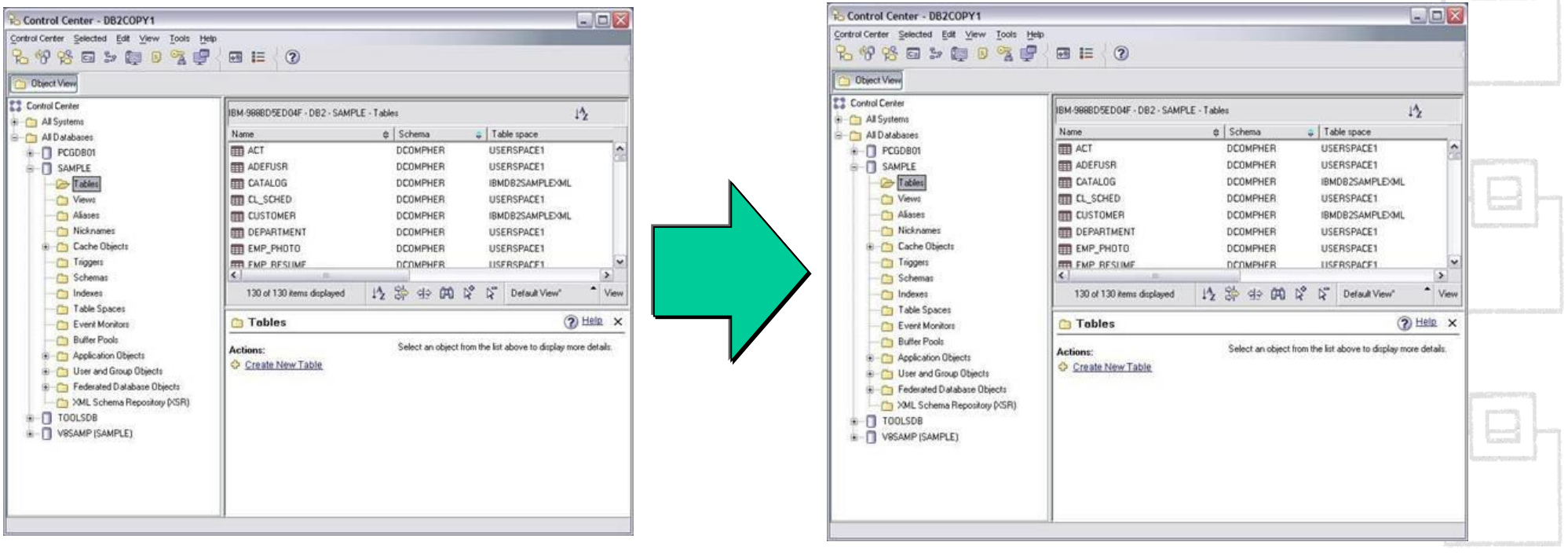
- ...along with wizards and advisors:
  - Control Center and associated wizards and advisors

- Alter Database Partition Group wizard
- Backup wizard
- Configuration advisor
- Configure Database Logging wizard
- Configure Multisite Update wizard
- Create Cache Table wizard
- Create Database wizard
- Create Federated Objects wizard (Also known as Create Nicknames wizard)
- Create Table Space wizard
- Create Table wizard
- Design advisor
- Drop Partition launchpad
- Health Alert Notification
- Health Indicator Configuration launchpad
- Load wizard
- Recommendation advisor
- Redistribute Data wizard
- Restore wizard
- Set Up Activity Monitor wizard
- Set Up High Availability Disaster Recovery (HADR) Databases wizard
- Storage Management Setup launchpad
- Troubleshooting wizard

GUIs in the past

Db2 Control Center (Db2cc)

- Deprecated with Db2 LUW 9.7 and Db2 z/OS 10.1
- Db2cc successor: Data Studio



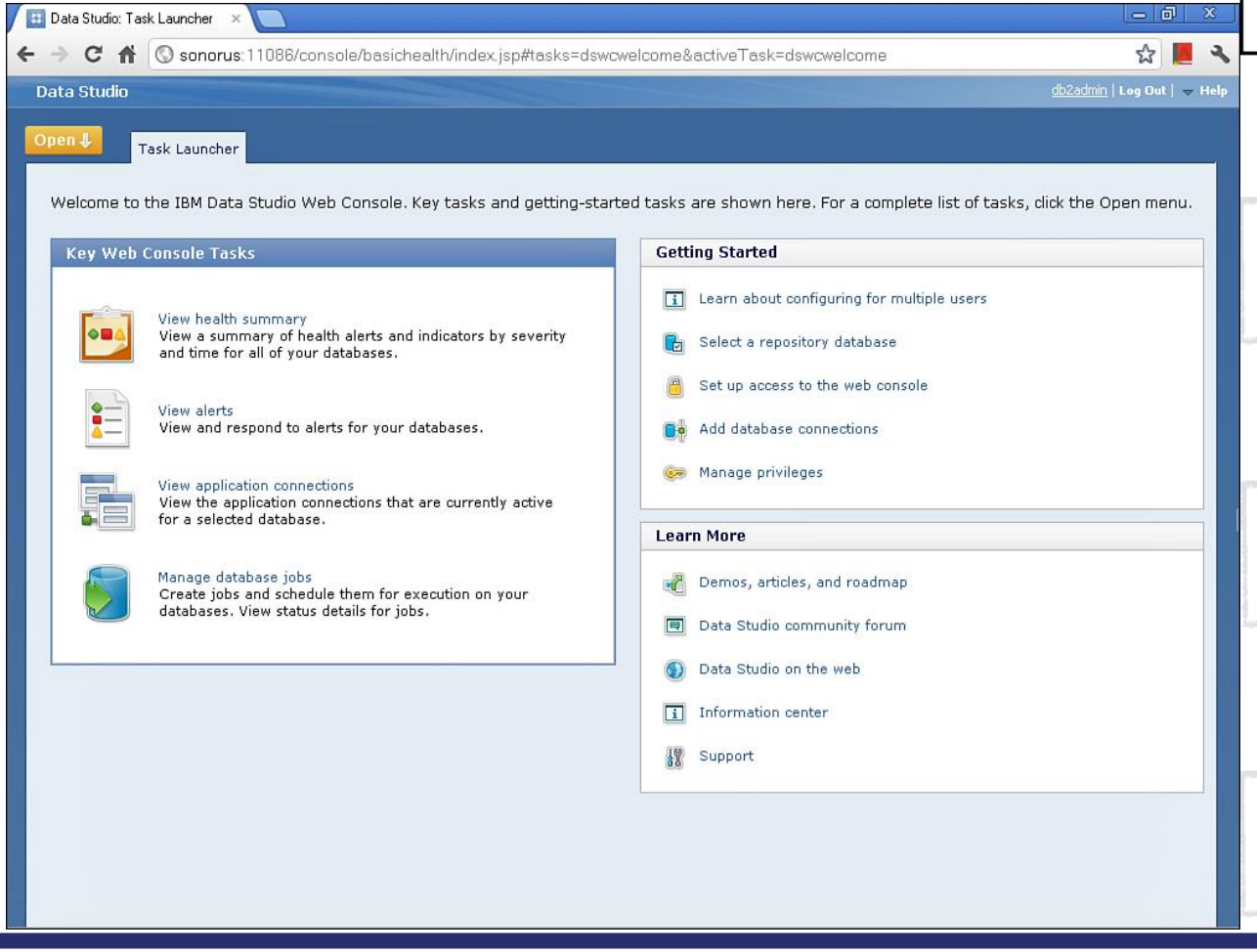
GUIs in the past

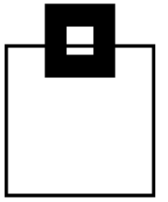
- Db2 Data Studio (Db2DS)

  A Windows/Linux EclipsePlugin using Java Db2 connection

- Db2 Data Studio Web Console (Db2DSWC)

  A Client/Server architecture, that enables web browser access





Db2cc successor: Data Studio

- True for most of the Db2cc tools, except:
  - Activity Monitor, Event Analyzer, Health Center, Web Console, Memory Visualizer, Query Patroller Center
    → InfoSphere Optim Performance Manager
  - Configuration Assistant
    → InfoSphere Optim Configuration Manager
- With more complex licensing associated:
  - InfoSphere Optim Performance Manager Extended Insight is a separately priced feature for InfoSphere Optim Performance Manager (part of InfoSphere Optim Performance Manager EE)
  - Data Studio consists of three components
  - The Index Advisor and Query Advisor require an InfoSphere Optim Query Workload Tuner license
  - Db2 Data Studio (Db2DS) renamed and bundled into Optim in 2009

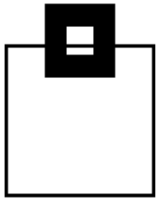
GUIs in the past

Then Db2 Data Server Manager was introduced* and customers were confused whether this is a DS successor/replacement

- Some IBMers said yes, some insisted they address different people:
  - DS is intended for developers
  - DSM is intended for DBAs
- Unfortunately some DS features are not maintained with Db2 12 CD
- Digging deeper indicates lots of the prior GUI Eclipse stuff and components "borrowed" from Db2DSWC
- However, the labs are saying it is "very much a rewrite of the front end, but the smarts have been passed onto this next generation”

* in July 2010 also z/OS Management Facility for system programmers

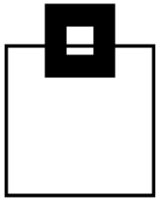
GUIs in the past



Bottom line/downside for ISVPs and customers:

- Familiar UIs continue to be changed
- Used features deprecated, or slightly shifted into other UIs
- No single/common point of control
  - → ISPF still the one and only true (Db2) z/OS UI that stays reliably solid over the years
  - → ISPF still the one and only true (Db2) z/OS UI that is supported by IBM AND ISVs

Agenda



- GUIs in the past
- Zowe ecosystem overview
- Zowe differentiation to prior GUIs
- Zowe components
- Our Journey
- How it looks
- Live demo

Zowe ecosystem overview

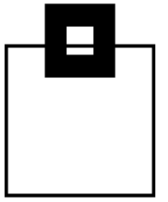
At the SHARE 2018 conference, IBM, Rocket Software and CA Technologies (now BROADCOM) announced Zowe – THE z ecosystem

- Open source project licensed under EPL 2.0
- Extensible framework
- Fuses and unites „old“, solid mainframe UI (tn3270, VT) with latest UI (HTML5, JS, TS, CLI)
- Based on and exploiting proven, rock solid technology (RLF, SAF, USS)
- Introduces REST APIs, ESM microservices, discovery services, …

Addresses

- Application Developers
- System Programmers
- DBAs
- DevOps Architects

Zowe ecosystem overview



Zowe is four major components:

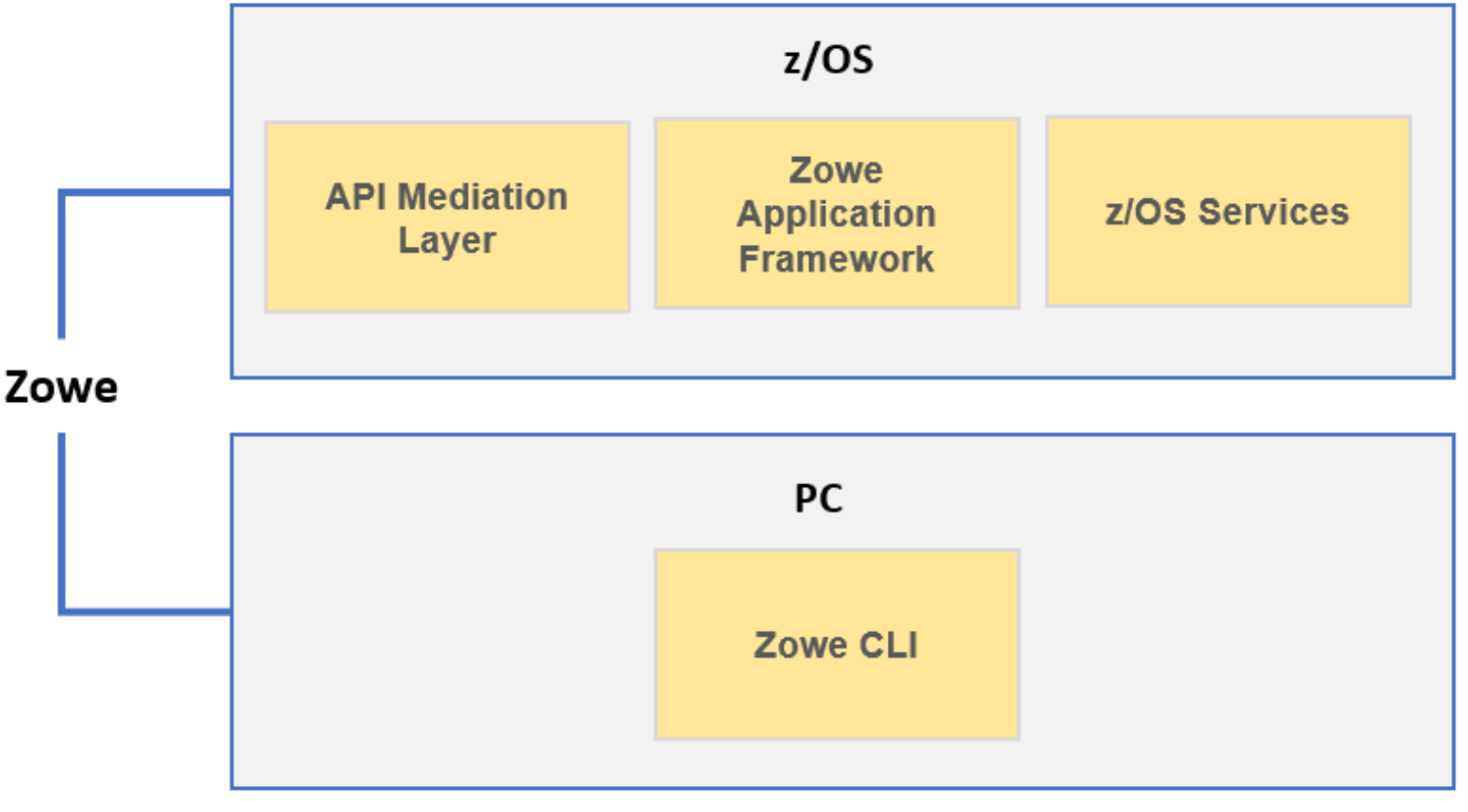
1. **Application Framework**
   The web UI that works with the underlying REST APIs presenting and bundling information in a modern, powerful full screen mode

2. **z/OS Services**
   Providing z/OS RESTful web service and deployment architecture for z/OS microservices

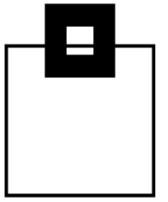
3. **Zowe CLI**
   Allowing to interact with the mainframe to efficiently build z/OS applications

4. **API Mediation Layer**
   Central point for all mainframe service REST APIs of the ecosystem

Zowe ecosystem overview

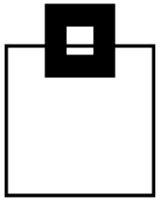


Agenda



- GUIs in the past
- Zowe ecosystem overview
- **Zowe differentiation to prior GUIs**
- Zowe components
- Our Journey
- How it looks
- Live demo

Zowe differentiation to prior GUIs



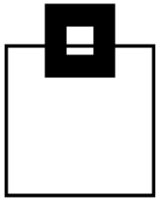
Zowe is

- the very first open source project on z/OS
- an extensible, common framework for existing and new applications
- designed to make the mainframe an agile, integrated platform
- a common UI for senior mainframe staff and new workforce
- a unified framework that merges proven and latest technology

...to

- demystify the mainframe and attract new people
- reduce the learning curve and improve productivity
- enhance integration and consumability
- simplify the architecture and reduce operational costs
- improve co-existence with a modern, platform-neutral interface

Zowe differentiation to prior GUIs



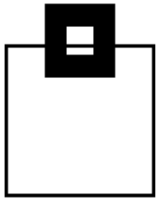
Zowe is

- the very first open source project on z/OS
- an extensible, common framework for existing and new applications
- designed to make the mainframe an agile, integrated platform
- ~~THE~~ common UI for senior mainframe staff and new workforce
- a unified framework that merges proven and latest technology

...to

- demystify the mainframe and attract new people
- reduce the learning curve and improve productivity
- enhance integration and consumability
- simplify the architecture and reduce operational costs
- improve co-existence with a modern, platform-neutral interface

Zowe differentiation to prior GUIs

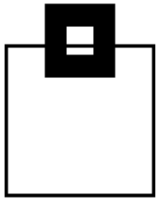


Zowe is vendor independent:

- Open source project under the Open Mainframe Project
- Free to be used under the Eclipse Public License 2.0
- Open, extensible interfaces of the code
- IBM, Rocket and BROADCOM (fka. CA) are founding members

→ Use, change and contribute

Zowe differentiation to prior GUIs



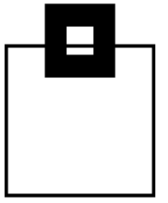
Zowe integrates nicely into an existing environment:

- Security management: SAF – System Authorization Facility
  - Controlling access by RACF, or other security products, like ACF2
- Resource management: RLF – Resource Limit Facility
  - Control processor usage of Db2 queries

- z/OS and USS support:
  - Explore JES, MVS, USS
  - Access and interact with subsystems like Db2, CICS
  - Browse and edit data sets
  - Execute JCL, Shell and z/OS commands, bash and z/OS scripts

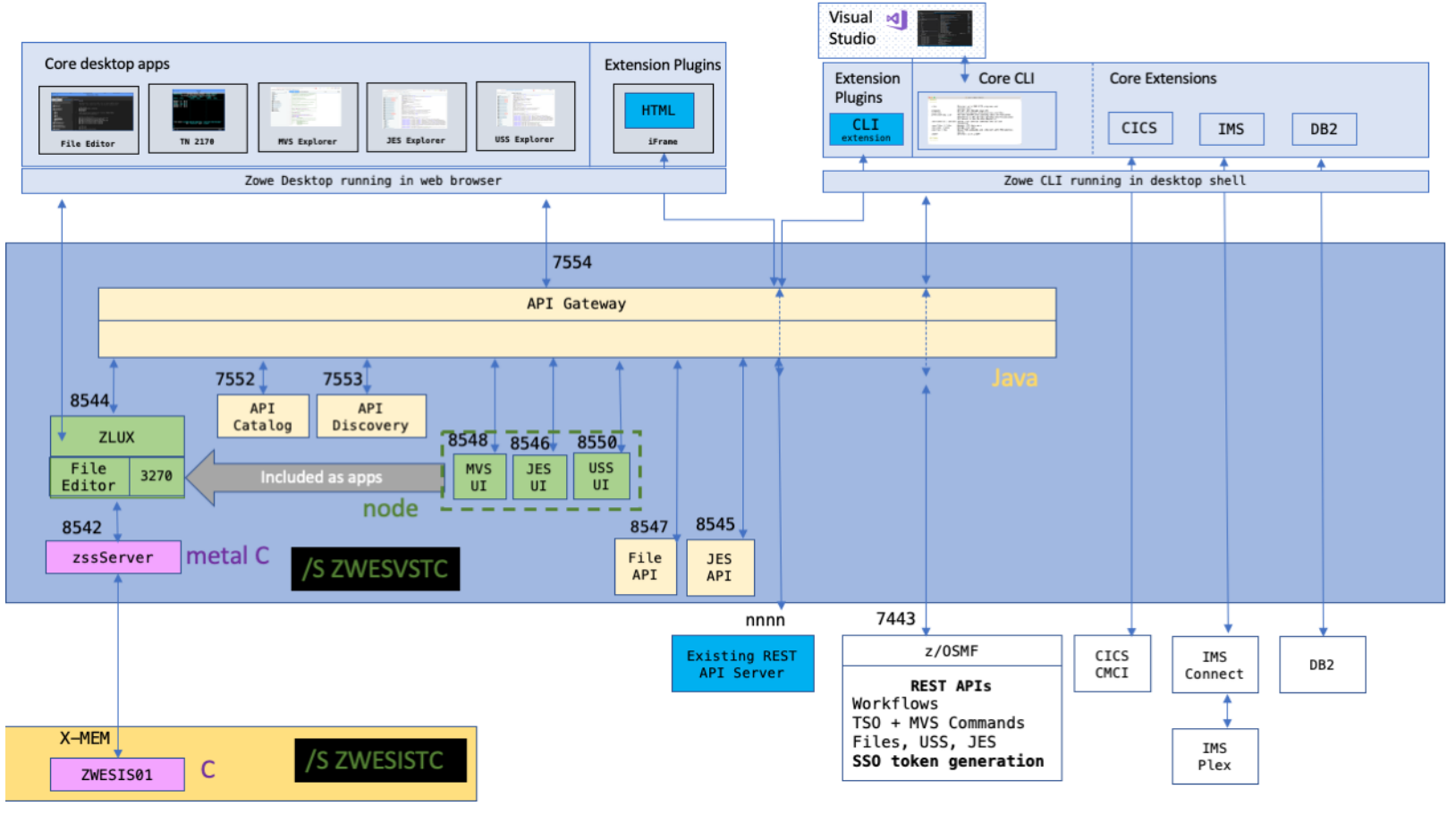
- Platform independent browser technology:
  - HTML5, CSS, JS, TS, …
- Platform independent CLI
  - Node.js, npm, IDEs, Jenkins, TravisCI, …

Agenda

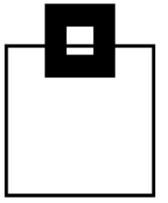


- GUIs in the past
- Zowe ecosystem overview
- Zowe differentiation to prior GUIs
- Zowe components
- Our Journey
- How it looks
- Live demo

Zowe components



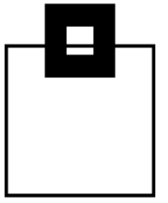
Zowe components



Zowe Application framework is four major components

1. **Desktop**
   Browser based web desktop
2. **Application Server**
   Web services framework plus proxy applications that communicates with z/OS services and components
3. **ZSS Server**
   REST services to support the Application Server
4. **Application plug-ins**
   Included and addable applications to access the mainframe and to perform various tasks, e.g.
   - Dataset editor and browser (z/OS and USS)
   - Workflows
   - z/OS subsystem browser (JES, CICS, Db2, IMS, …)
   - …

Zowe components



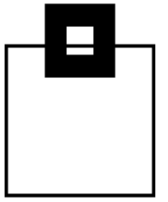
Zowe z/OS services contain the following core components

1. z/OS dataset services
   list, browse, edit, create, delete, … datasets and members
2. z/OS job services
   list, browse, submit jobs

A full list of capabilities of the RESTful API can be listed via the API catalog

- The Open API Specification describes the APIs and allows to use any standard-based REST API developer tool, or API management process
- APIs can be used by any application
- z/OS services are running as microservices with a Spring Boot embedded Tomcat stack

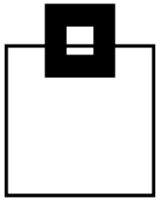
Zowe components



Zowe CLI comes with the following capabilities

- **Interact with files:**
  Create, edit, download, and upload data sets
- **Submit jobs:**
  Submit JCL from data sets or local storage, monitor the status, and view/download the output
- **Execute commands:**
  Issue TSO, or z/OS console commands
- **Integrated scripts:**
  Define scripts that do both mainframe and local tasks
- **Return JSON documents:**
  Return the data in JSON format to be used in other programming languages

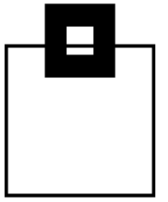
Zowe components



Zowe API mediation layer consists of the following components

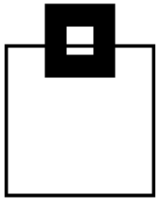
- API gateway
  - Clients interact with microservices behind a reverse proxy forwarding requests to the appropriate service
  - The gateway is built on Netflix Zuul and Spring Boot technology
- Discovery services
  - Accepts the REST service announcements and serves active ones
  - The service is built on Netflix Eureka and Spring Boot technology
- API catalog
  - UI catalog of published APIs along with their documentation (Swagger) and status
  - Services can be implemented by multiple instances for high-availability or scalability
- ESM microservices
  - Authenticates and authorizes users with mainframe credentials

Agenda



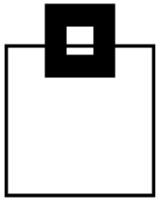
- GUIs in the past
- Zowe ecosystem overview
- Zowe differentiation to prior GUIs
- Zowe components
- Our Journey
- How it looks
- Live demo

ZOWE Our Journey (1:5)



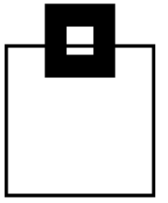
- ZOWE for SE started with version 0.9.5 and node.js on the PC – That's all you need!
  - Why did we choose ZOWE?
    - Modernize the Mainframe – The crowd is greying out there…
    - Multi-Factor Authentication (MFA) required – Nearly all of our customers require MFA these days and Eclipse based support is not being delivered by IBM
    - Much better GUI – HTML5 is way better than Eclipse with Jasper

ZOWE Our Journey (2:5)



- We chose our WorkloadExpert as the first of our Eclipse based plug-ins to be migrated to ZOWE
  - Largest of our GUIs with the most extensive use of graphics and reports
  - For this the API mediation layer and Application Framework from ZOWE were important
  - The Application Framework is what we call ZOWE desktop
    - It provides a virtual GUI but it can be accessed by browser
    - It serves as a starting point for preinstalled and external apps
    - It runs as a webserver on the mainframe

ZOWE Our Journey (3:5)



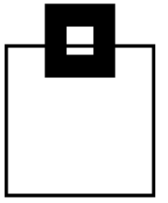
- To add your app to the ZOWE desktop you must use the ZOWE desktop technology stack
  - It consists of HTML, CSS and JavaScript for the GUI, Node.js for the data services and Java for the API Mediation layer

- ZOWE supports three frameworks for apps:
  - Angular
  - React
  - Iframe

- Angular and React are JavaScript Frameworks which allow you to develop highly interactive web apps also allowing the use of Typescript. This is then compiled into JavaScript.

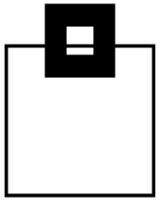
- The ZOWE desktop allows you to include Angular and React apps directly

ZOWE Our Journey (4:5)



- Iframe is basically a website within a website
  - If you configure your app in ZOWE as iframe, it just takes the HTML and associated files (JavaScript, CSS, Images and other files) and serves as a website
  - If you take this approach you can include apps in ZOWE which have been built without even having ZOWE in mind
  - You can also still access functions provided by ZOWE because they are provided as a global variable
    - Eg URIBroker to avoid hard coding URLs or preference storage

ZOWE Our Journey (5:5)



- What we did and Why we did it!
  - Front-end: The GUI – user interaction and graphics. We chose Angular and we compile it with the angular-cli
  - Back-end: Mainframe with Db2 database. We chose to use an API via the API Mediation Layer instead of a data service
    - Why? The APIs have a wider applicability than data services as they are not only for the ZOWE desktop but for all ZOWE services
    - This could be seen as „overkill“ for our WLX App but the fact that the API can be written in Java allows us to re-use code from the original Eclipse plug-in
      - For this we use the Spring Boot framework

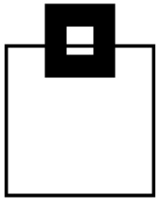
ZOWE Gotcha's

- We are a mainframe shop and so we had a little bit to learn…
  - z/OS Unix System Services: This must be installed and running well as it is an absolutely critical core requirement for ZOWE
  - Configuration and Profile files:
    - In Language Environment we found out that we had to raise the HEAP64 quite a bit:

      HEAP64(512M,4M,KEEP,256M,4M,KEEP,0K,0K,FREE)
    - The MEMLIMIT of the OMVS users must also be raised
    - In the profile datasets the important thing is to set up your ASCII:

```
#ASCII support the environment variables
export _BPXK_AUTOCVT=ON
export _CEE_RUNOPTS='FILETAG(AUTOCVT,AUTOTAG) POSIX(ON)'
export _TAG_REDIR_ERR=txt
export _TAG_REDIR_IN=txt
export _TAG_REDIR_OUT=txt
```

Agenda



- GUIs in the past
- Zowe ecosystem overview
- Zowe differentiation to prior GUIs
- Zowe components
- Our Journey
- How it looks
- Live demo

How it looks

- Here‘s how the json Iframe link to ZOWE looks:

```
VIEW       /Z23A/usr/lpp/seg/vnext/wlx/0.0.1/pluginDefinition.json
Command ===>
****** ***************************************************** Top
000001 ä
000002   "identifier": "de.seg.wlx.gui",
000003   "apiVersion": "0.0.1",
000004   "pluginVersion": "0.0.1",
000005   "pluginType": "application",
000006   "webContent": ä
000007     "framework": "iframe",
000008     "launchDefinition": ä
000009       "pluginShortNameKey": "WLXgui",
000010       "pluginShortNameDefault": "WLX GUI",
000011       "imageSrc": "assets/icon.png"
000012     ü,
000013     "descriptionKey": "WLX GUI",
000014     "descriptionDefault": "WLX GUI",
000015     "startingPage": "index.html",
000016     "isSingleWindowApp": true,
000017     "defaultWindowStyle": ä
000018       "width": 1028,
000019       "height": 768,
000020       "x": 20,
000021       "y": 20
```

How it looks

- Here‘s the contents of the assets directory where our SEG logo lives:

```
Command ===>

Pathname . : /Z23A/usr/lpp/seg/vnext/wlx/0.0.1/web/assets
EUID . . . : 3
Command  Filename        Message          Type Permission
---------------------------------------------------------
_________ .                               Dir  rwxr-x---
_________ ..                              Dir  rwxrwx--x
_________ _variables.scss                 File rw-r-----
_________ icon.png                        File rw-r-----
_________ i18n                            Dir  rwxr-x---
_________ veil-loaderrota                 File rw-r-----
*********************************************************
```

How it looks

- Here‘s how the index.html looks:

```
VIEW        /Z23A/usr/lpp/seg/vnext/wlx/0.0.1/web/index.html
Command ===>
****** ***************************************************** Top of Data ********
000001 <Üdoctype html>
000002 <html lang="en">
000003 <head>
000004   <meta charset="utf-8">
000005   <title>SQL WorkloadExpert for Db2 z/OS</title>
000006   <base href="/ZLUX/plugins/de.seg.wlx.gui/web/">  <meta name="viewport" con
000007   <link rel="icon" type="image/x-icon" href="favicon.ico">
000008 <link rel="stylesheet" href="styles.c3f4c4c5a52bb89c8ce1.css"></head>
000009 <body>
000010   <app-root></app-root>
000011 <script src="runtime-es2015.40d02d0fde87355ce70a.js" type="module"></script>
000012 </html>
****** ***************************************************** Bottom of Data ******
```

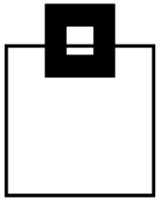
How it looks

- And here's the beginning of that runtime:

```
VIEW       /Z23A/usr/lpp/seg/vnext/wlx/0.0.1/web/runtime-es2015.40d02d0fde87355ce70a.js
Command ===>
****** ***************************************************** Top of Data ***********
000001 Üfunction(e)äfunction r(r)äfor(var n,c,u=rÝ0",i=rÝ1",f=rÝ2",d=0,p=Ý";d<u.length;
****** ***************************************************** Bottom of Data ********
```

```
  Menu  Utilities  Compilers  Help
 BROWSE     /Z23A/usr/lpp/seg/vnext/wlx/0.0.1/web/8- Line 0000000000 Col 001 080
 Command ===>                                          Scroll ===> CSR
**************************** Top of Data ***********************************
(window.webpackJsonp=window.webpackJsonp!!Ý").push(ÝÝ8",ä"+FGM":function(e,t)äe.
exports=function(e)ävar t=e.COMMENT("--","$"),a="BIGINT INT8 BIGSERIAL SERIAL8 B
IT VARYING VARBIT BOOLEAN BOOL BOX BYTEA CHARACTER CHAR VARCHAR CIDR CIRCLE DATE
 DOUBLE PRECISION FLOAT8 FLOAT INET INTEGER INT INT4 INTERVAL JSON JSONB LINE LS
EG!10 MACADDR MACADDR8 MONEY NUMERIC DEC DECIMAL PATH POINT POLYGON REAL FLOAT4
SMALLINT INT2 SMALLSERIAL!10 SERIAL2!10 SERIAL!10 SERIAL4!10 TEXT TIME ZONE TIME
```

Agenda



- GUIs in the past
- Zowe ecosystem overview
- Zowe differentiation to prior GUIs
- Zowe components
- Our Journey
- How it looks
- Live demo

Give and Take Program, Germany 2020

The 2020 Program offers:

January – March (1Q): Db2 11 + 12 Audit + SIEM (Security Information Event Management) with optional framework Eclipse or ZOWE

April – June (2Q): Access Path Recovery – The first participant of this program is the FiduciaGAD IT AG

July – September (3Q): Space Assurance – K-no-w your limits

October – December (4Q): Zowe and SQL Workload Performance for Db2 11 & 12

Questions???

Many thanks for your attention and now....