Thomas Zierer

How to Tame a Dinosaur in 7 Steps: Mainframe Development with Eclipse

November 5, 2015, EclipseCon Europe 2015, Ludwigsburg
Me, myself & I

- Thomas Zierer
- Lead Technical Architect
- BayernLB Dinosaur (since 10/1992)
- Interests: Everything that links the mainframe to the distributed world.
Mainframe: What is it and why do we (still) do it?

Big Iron
- zSeries, System z9, z10, z/Enterprise
- Operating systems MVS, OS/390, z/OS
- Batch processing with JCL (Don‘t panic: No punch cards!)
- Programming Languages PL/I, COBOL
- 55% of all worldwide business applications involve mainframe code (Source: IBM (who else?))
Before we start

Demo

The traditional vs. The Eclipse Way
How to Tame a Dinosaur in 7 Steps

- Search!
- Find!
- Catch!
- Play!
- Train!
- Teach tricks!
- Parade!
Step 1: Search!

- Hardest part indeed!
- z/Series uses very unique file system („VSAM“, „Physically sequential“, „Partitioned“, „Generation data groups“)
- Incomparable to Windows or Unix file systems
- Target Management Plugin (aka Remote Systems Explorer) only offers access to integrated Unix System Services, not native z/Series file system
Step 2: Find!

• Solution: IBM Explorer Family
  • Free of charge, Eclipse based, stand alone clients
  • Access to CICS, IMS, Fault Analyzer, File Manager or z/OS
  • Update Site: http://public.dhe.ibm.com/software/htp/zos/tools/1
Step 2: Find!
Step 3: Catch!

Simply include update site in target definition for your product or Tycho build
Step 4: Play!

Demo

Explorer for z/OS
Step 5: Train!

- Default editor is a notepad-lookalike
- Building a custom editor with Eclipse Text is easy
- Basically implement IPartitionScanner, IToken and Irule or extend default implementations.
- Include new editor as a fragment in z/OS Explorer data sets view
Step 6: Teach tricks!

- Support for code completion or automatic refactoring would require a language server application

- Limitations:
  - All tooling is mainframe based
  - Distributed compilers exist but are expensive and not 100% compatible
  - Compilelisting is meant to be read by humans
Step 6: Teach tricks!

- All IBM mainframe compiler support options for generating *some* compiler information as XML (COBOL: EXIT(ADEXIT(ELAXMGUX)))

- A variable typo appears like this
  ```xml
  <MESSAGE>
  <MSGNUMBER>IGYPS2121-S</MSGNUMBER>
  <MSGLINE>24</MSGLINE>
  <MSGTEXT>&quot;PRINREC&quot; was not defined as a data-name.</MSGTEXT>
  </MESSAGE>
  ```

- This is everything we need to populate the problem view!
Step 6: Teach tricks!

- Allocate temporary dataset, suitable for XML data
- Generate a compile script in Job Control Language
- Transfer it to the mainframe and execute it
- Download the contents of the temporary dataset and parse it
- For every compiler message place a marker into the problem view and add an annotation to the editor
- And of course delete the XML dataset
- ZOSConnectable.getSingleton() does all the magic
Step 7: Parade!

Demo

Remote debugging
Thank you very much for your attention!

Thomas Zierer
Phone:  +49 89 2171-21446
E-Mail: thomas.zierer@bayernlb.de
evaluate the sessions at www.eclipsecon.org