Practical experiences with RDz/IDz

GSE EM working group

Rotterdam - 26/04/2017

Gie Indesteege - ABIS Training & Consulting

Welcome

ABIS Training & Consulting

- **provides high-level technological ICT services for large and medium size enterprises**

www.abis.be

Welcome (cont.)

Gie Indesteege

- trainer and consultant
  - application development
  - z/OS - COBOL - PL/I - ISPF - REXX - CICS
  - UNIX/Linux
  - Java SE - EE
- GSE member in working groups on z/OS, CICS, Architecture
  (former president of BeNeLux GSE working group EGL/RBD)

gindesteege@abis.be

gie-indesteege-178a44/

Abstract

Using the IBM Developer for z Systems (IDz) - or RDz - for traditional z/OS application development, requires not only an installation and configuration of a client and a corresponding set of host services. Even more important is the setup of the workbench, defining the right user preferences, a good organisation of local projects, and an efficient way of accessing the host resources.

In this presentation, I will try to hand over my practical experience with the tool to the audience:

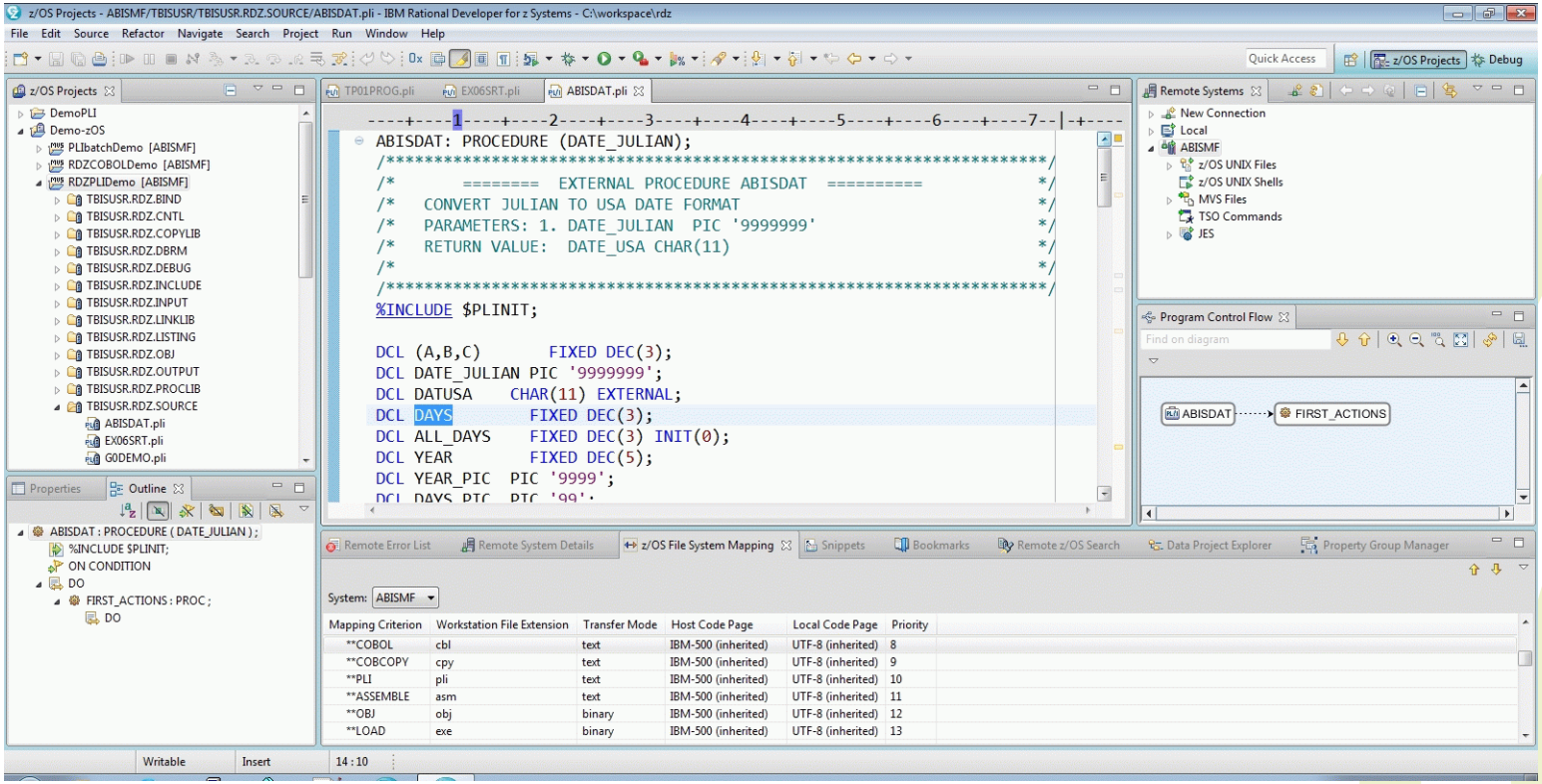
- **I used it for developing batch applications written in COBOL or PL/1, using the editors (LPEX, language specific and JCL), and testing with the integrated debugger;**
- **as well as for CICS application development, with BMS maps, and again the integrated debugger;**
- **but also, using the data perspective, for preparation of SQL queries and stored procedures**

Experience from a 'developer' perspective.

Agenda

- **Workbench configuration and organisation**
- **z/OS project organisation**
- **Development (editor, datasets, procedures, custom menus)**
- **Testing and debugging**
- **CICS (+ BMS)**
- **DB2 (SQL + stored procedures)**
- **Problems encountered**
- **Q & A**

Workbench configuration and organisation



-> avoid superfluous information

-> agree upon common look-and-feel in your team

Workbench configuration and organisation

user preferences - create user defined (default) perspective

- **eliminate unused views**
- **rearrange views**
- **import (company customised) property groups, file system mappings, custom menus, connection definitions, snippets, ...**

===> IDz provides push-to-client facility

useful perspectives - choose the right one!

- **z/OS projects**
- **Debug**
- **Data (data source definition and data projects)**
- **CICS**

Do I use/need Host Connection Emulator (HCE) ?

Agenda

- Workbench configuration and organisation
- z/OS project organisation
- Development (editor, datasets, procedures, custom menus)
- Testing and debugging
- CICS (+ BMS)
- DB2 (SQL + stored procedures)
- Problems encountered
- Q & A

z/OS project organisation - logical

Local resources

z/OS projects
- MVS subprojects

  batch
  batchWithDB2
  CICS
  CICSWithDB2
  DB2 Stored proc
  JCL

Overview/structure

Navigational views
  Outline
  Program flow

Properties view

Editor(s)

Mainframe resources

- UNIX
- MVS files
  private datasets
  project datasets
    (test/acceptance/...)
  common datasets
  ...
- JES
  private jobs
  project jobs
  retrieved jobs
  ...
- TSO command shell

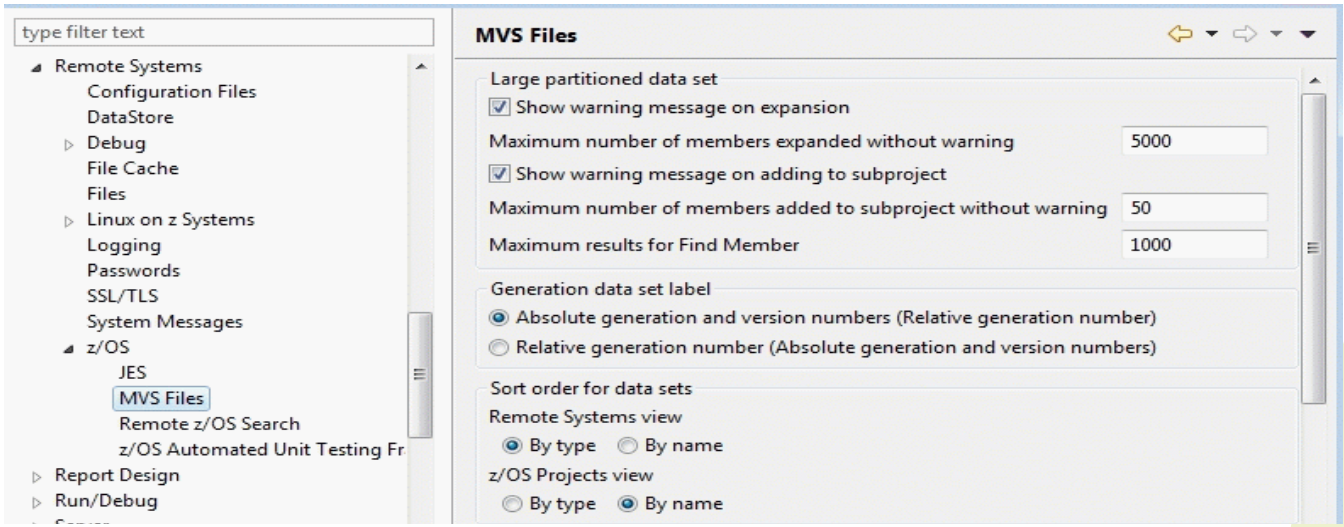
Program development 'tools' views

Search
Problem
Job details table
Snippets

Purpose: how to exchange resources efficiently between local workbench and remote mainframe

z/OS project organisation - Mainframe (MF) ‘side’

- user preferences for dataset visualisation



- MVS datasets/filters

  which datasets do I need for developing my application?

  - source, copybooks, JCL
  - development, test, acceptance, production
  - common datasets - procedure libraries

  usage of member filters in case of large PDS

z/OS project organisation - Mainframe (MF) ‘side’ (cont.)

- **JES filters**

  which jobs do I have to check on the spool?

  - **personal jobs**
  - **‘common’ project related jobs**

  note: use Remote System details (table) view for advanced filtering

- **z/OS file system mapping**

  standardisation of dataset + member naming allows for correct interpretation/usage of mainframe data/files

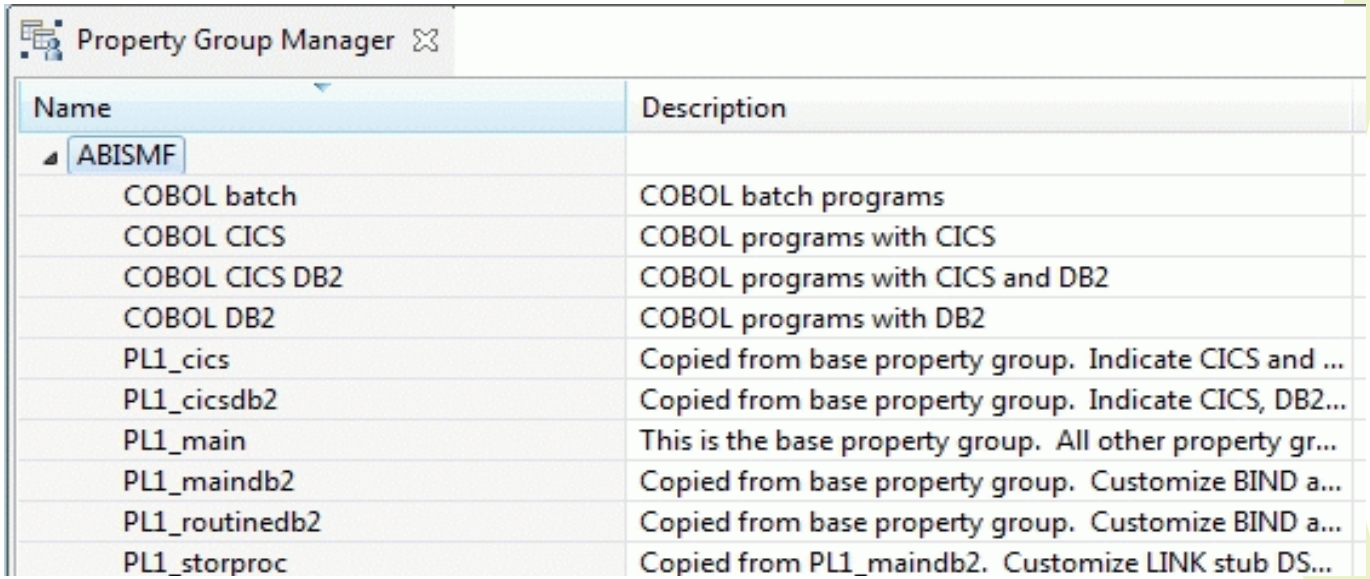
Table 1: z/OS file system mapping

**COBOL, **.COPY	cbl
**PLI, **.INCLUDE	pli
**.CNTL	jcl
**.MAP	bms

z/OS project organisation - Local ‘side’

Local ‘side’ - for editing/manipulation of sources

- z/OS projects
  - grouping of datasets/members related to application
  - subdivided in MVS subprojects, per application type (batch, CICS, DB2, COBOL, PLI, ...) + JCL (for batch run)
  - with ‘corresponding’ property group associated



Name	Description
ABISMF	
COBOL batch	COBOL batch programs
COBOL CICS	COBOL programs with CICS
COBOL CICS DB2	COBOL programs with CICS and DB2
COBOL DB2	COBOL programs with DB2
PL1_cics	Copied from base property group. Indicate CICS and ...
PL1_cicsdb2	Copied from base property group. Indicate CICS, DB2...
PL1_main	This is the base property group. All other property gr...
PL1_maindb2	Copied from base property group. Customize BIND a...
PL1_routinedb2	Copied from base property group. Customize BIND a...
PL1_storproc	Copied from PL1_maindb2. Customize LINK stub DS...

Agenda

- Workbench configuration and organisation
- z/OS project organisation
- **Development (editor, datasets, procedures, custom menus)**
- Testing and debugging
- CICS (+ BMS)
- DB2 (SQL + stored procedures)
- Problems encountered
- Q & A

Development

1. dataset preparation/manipulation -> **MF ‘side’**

   Prepare mainframe resources for application development

   - lookup via dataset filter
   - allocate new dataset
   - create new member, or

     copy prod/accept -> test, devlpt

   optional:

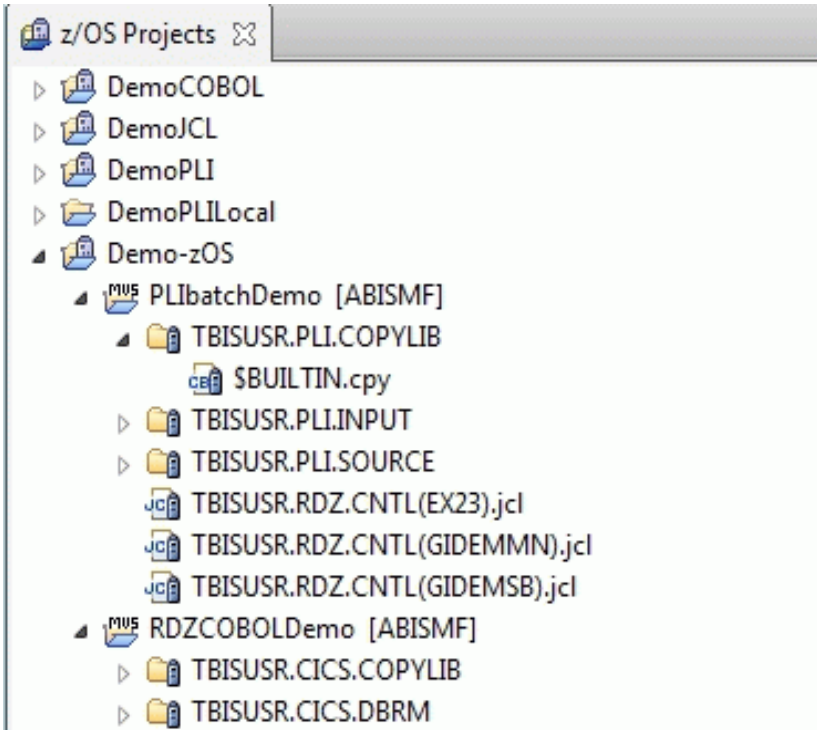
   - member filter -> use find member facility
   - use of z/OS remote searching

Development (cont.)

2. associate dataset/members to z/OS subprojects -> MF 'side'

Transfer mainframe resources to local resources

- DO NOT transfer full PDS, if you need only a few members



- z/OS file system mapping
- property group -> procedures for build: compile/link

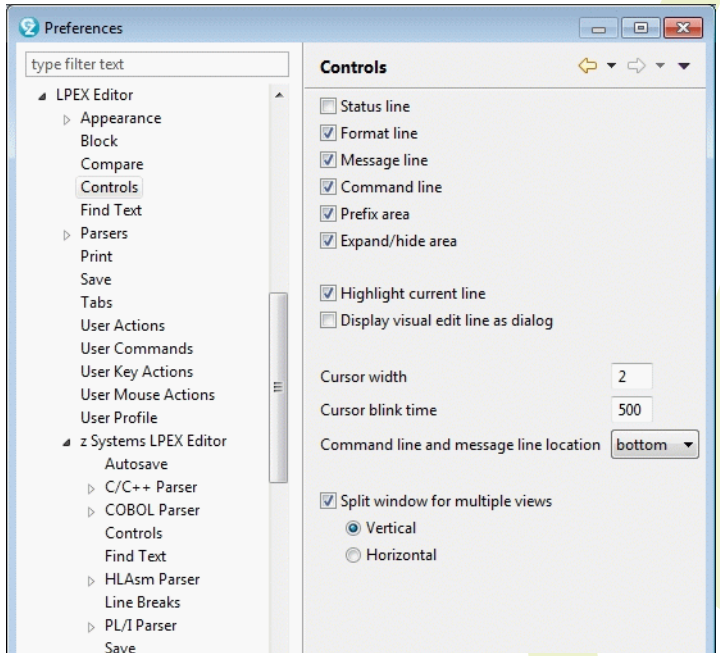
Development - LPEX editor

3. choice of editor -> local ‘side’

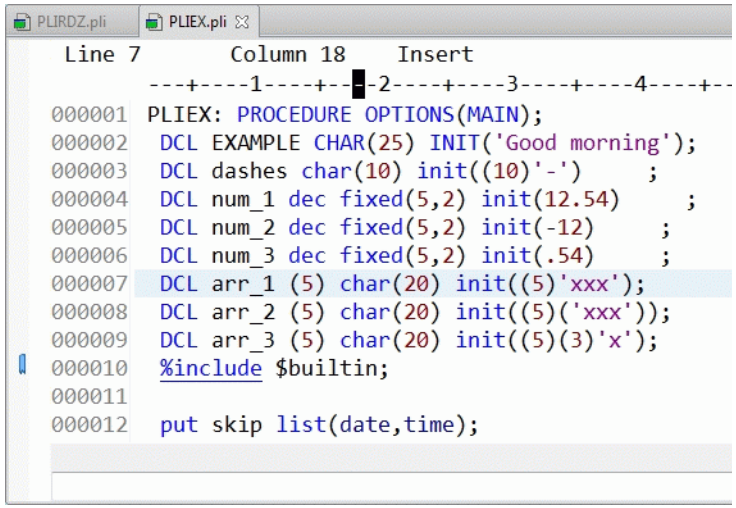
LPEX (3270 - ISPF like support) - line oriented

(Live Parser Extensible Editor)

- user preferences



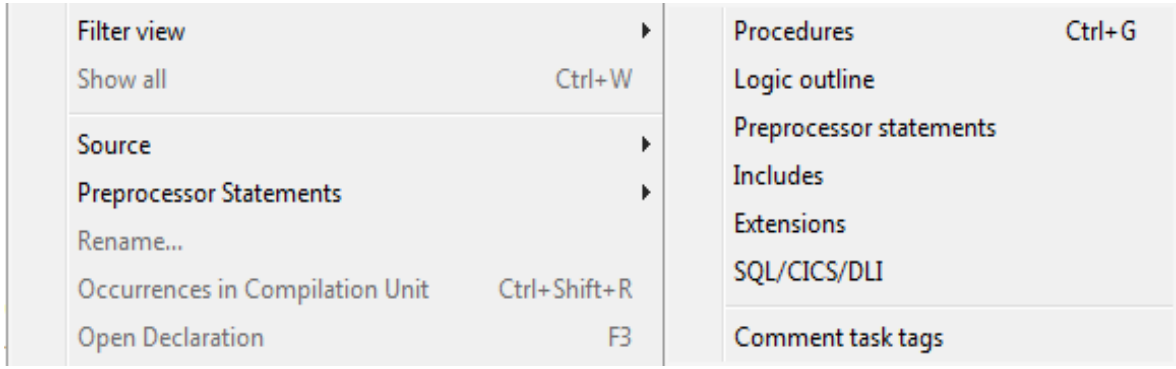
- line and prefix commands



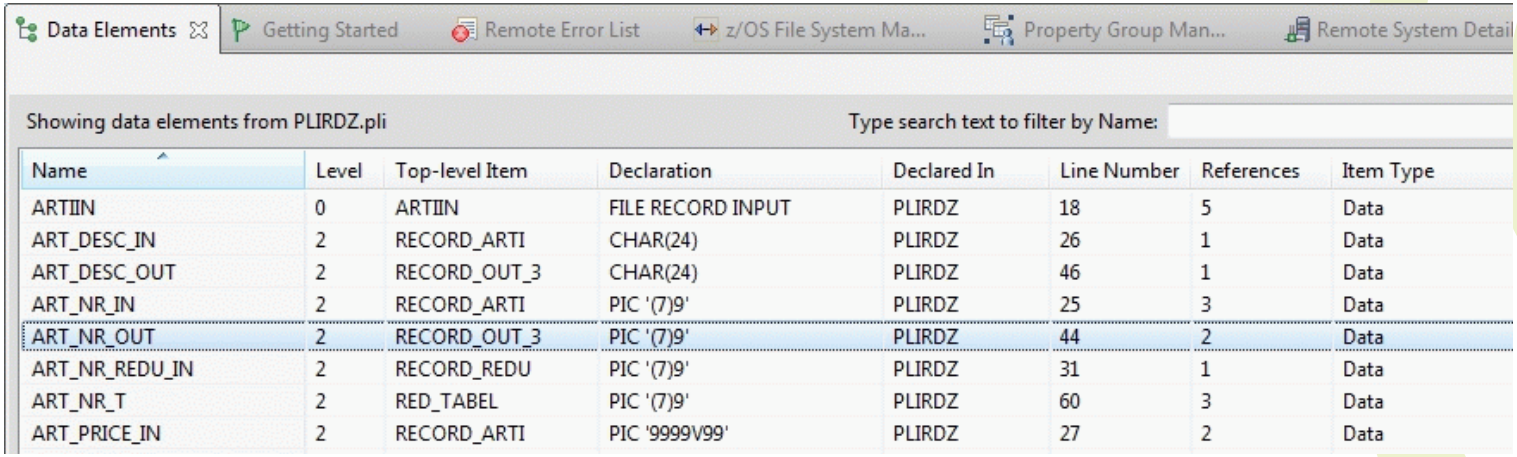
- additional editing (PC) features - find, scrolling, copy/paste

Development - LPEX editor (cont.)

- content assist - quick fix
- filter view



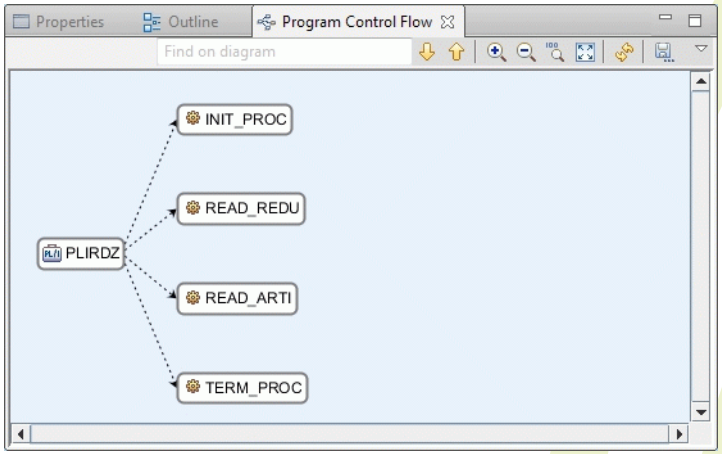
- show in **Data Elements view** to get insight in usage of data fields



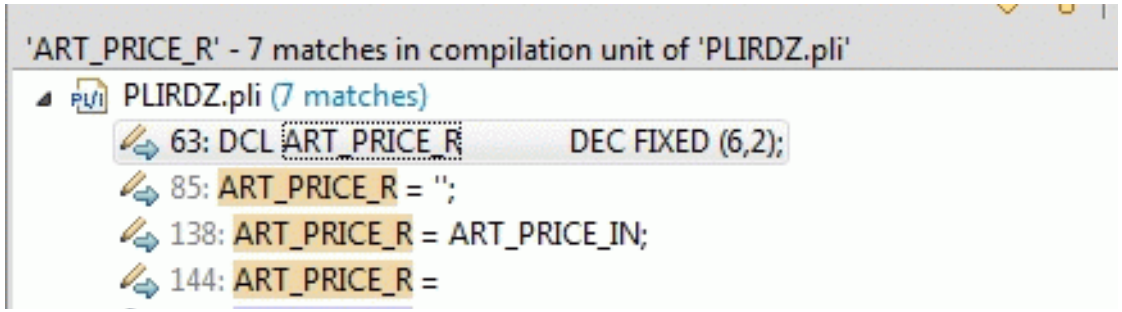
Name	Level	Top-level Item	Declaration	Declared In	Line Number	References	Item Type
ARTIIN	0	ARTIIN	FILE RECORD INPUT	PLIRDZ	18	5	Data
ART_DESC_IN	2	RECORD_ARTI	CHAR(24)	PLIRDZ	26	1	Data
ART_DESC_OUT	2	RECORD_OUT_3	CHAR(24)	PLIRDZ	46	1	Data
ART_NR_IN	2	RECORD_ARTI	PIC '(7)9'	PLIRDZ	25	3	Data
ART_NR_OUT	2	RECORD_OUT_3	PIC '(7)9'	PLIRDZ	44	2	Data
ART_NR_REDU_IN	2	RECORD_REDU	PIC '(7)9'	PLIRDZ	31	1	Data
ART_NR_T	2	RED_TABEL	PIC '(7)9'	PLIRDZ	60	3	Data
ART_PRICE_IN	2	RECORD_ARTI	PIC '9999V99'	PLIRDZ	27	2	Data

Development - LPEX editor (cont.)

- **source navigation**
  - outline view
  - program control flow view



  - occurrences in compilation unit (-> search view)



  - open include file
  - open called routine

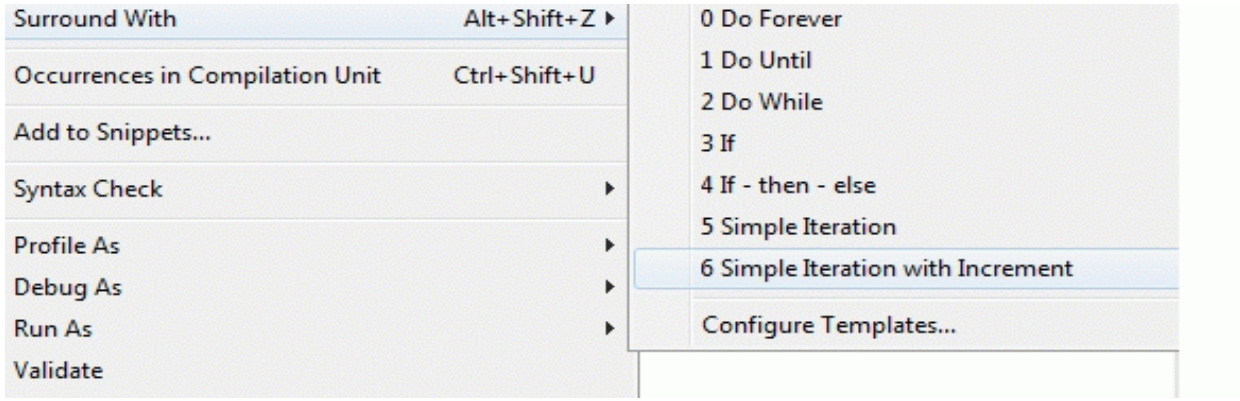
Development - COBOL or PLI editor

- choice of editor -> PC - Eclipse based
  - all Eclipse editor features
  - code collapse
  - extended hover facilities

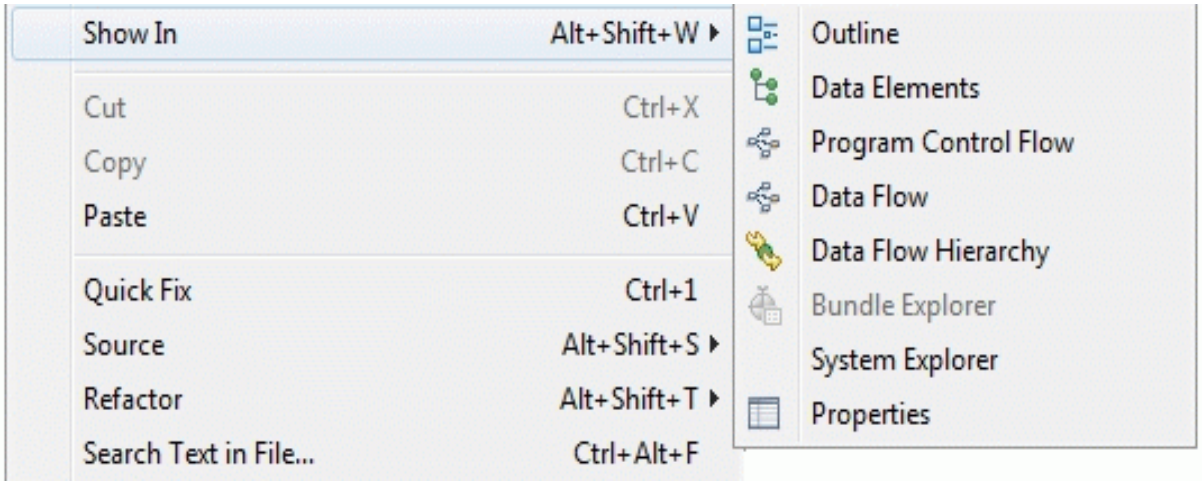
+ language specific editor features
  - code/content assist - quick fix
  - source formatting
  - open declaration

Development - COBOL or PLI editor (cont.)

- ‘surround with’ feature (PLI example)



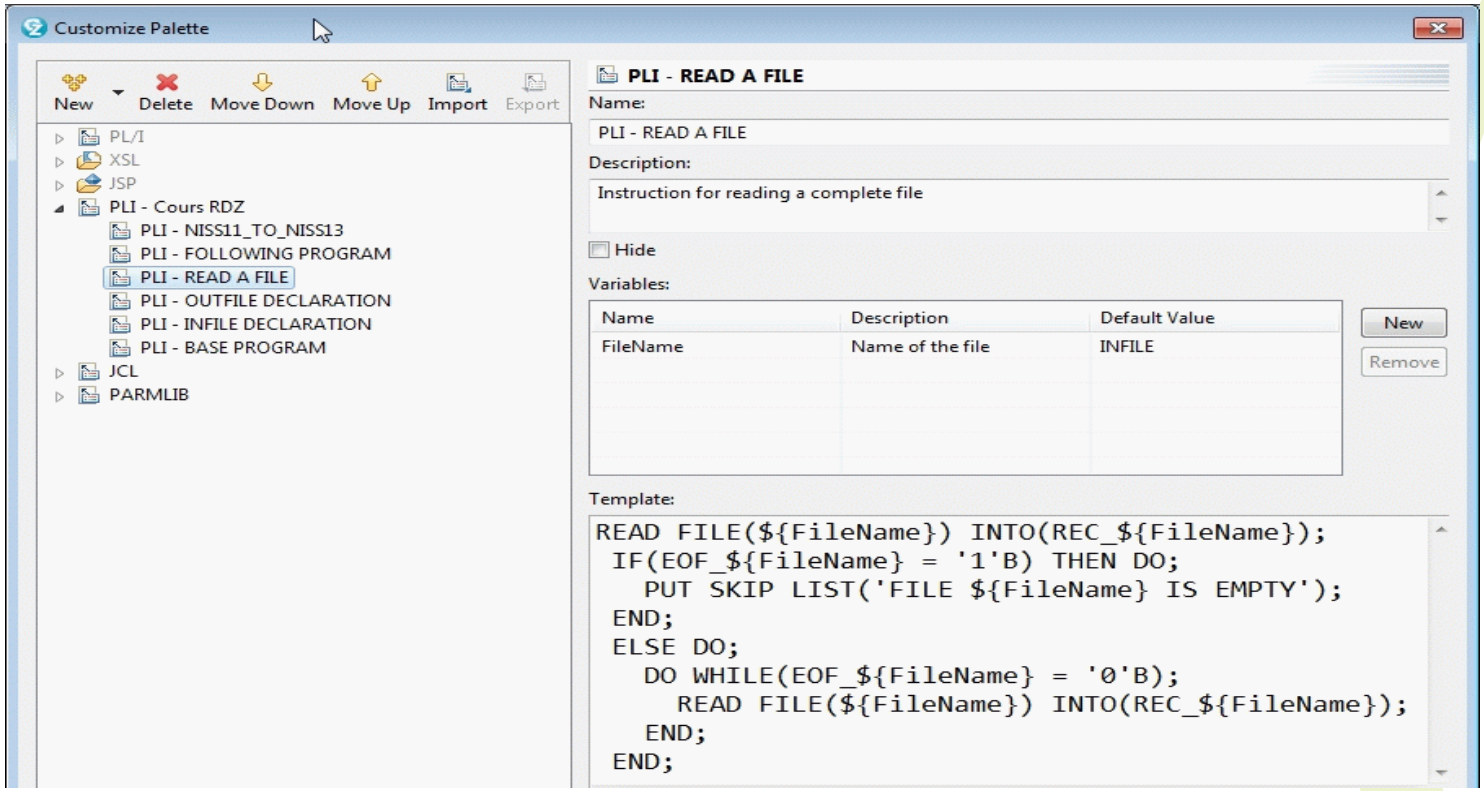
- extended navigation - Show in



Development - editor features

Additional (customisable) features

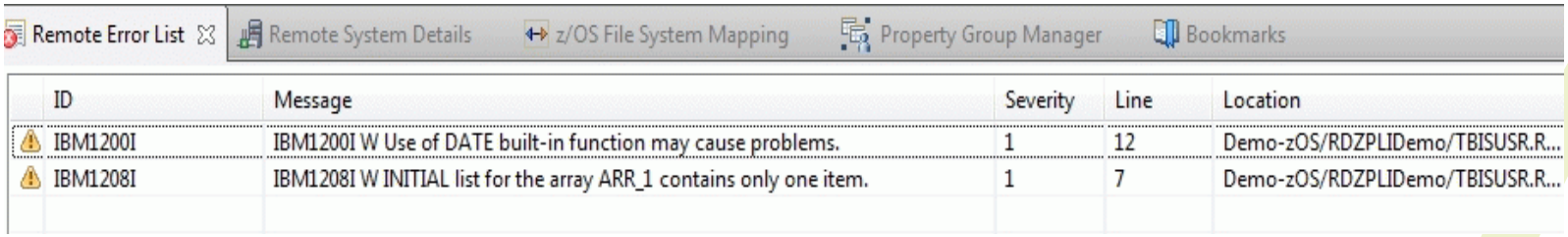
- Using templates (for code assist)
- Snippets



can be exported/imported

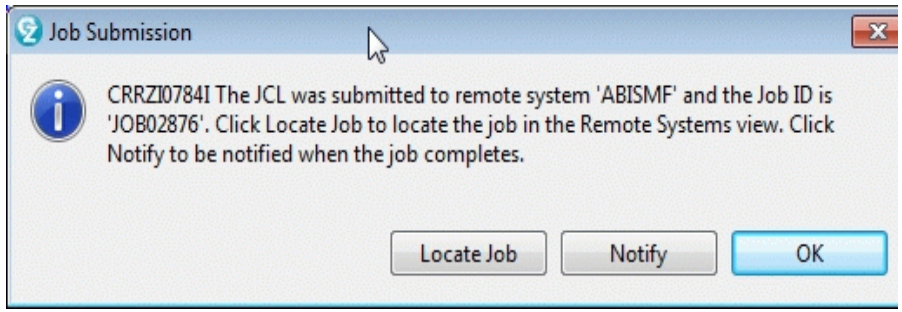
Practical experiences with RDz/IDz

4. Remote syntax check and verify Remote error list view



or

build (via property group) - compile/link



How to find job result (on spool)?

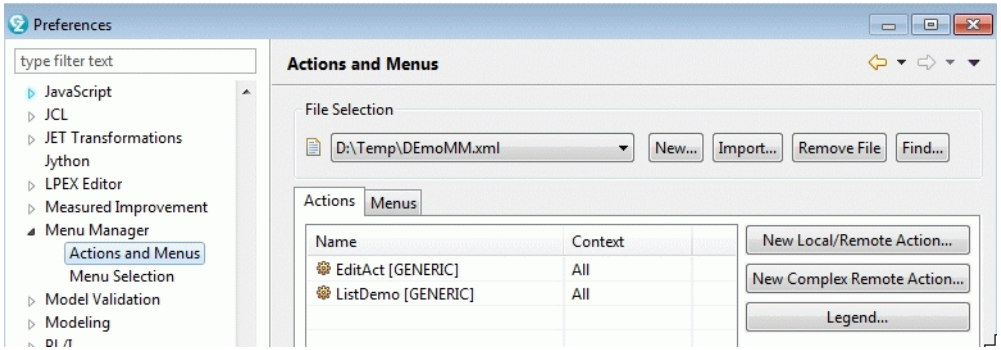
- **Notify -> info in Remote console**
- **Locate job -> JES Retrieved jobs, or**
- **OK -> Remote system details view**

Development - customisation

5. custom menus

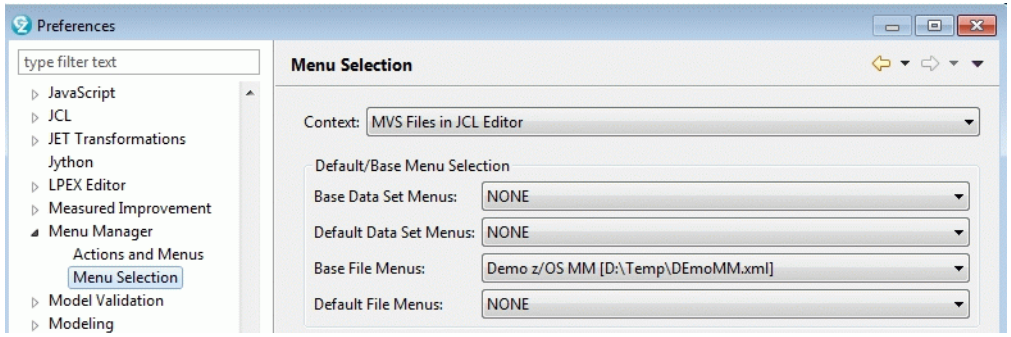
use Preferences -> Menu manager

- define actions and menus



run options with support of REXX

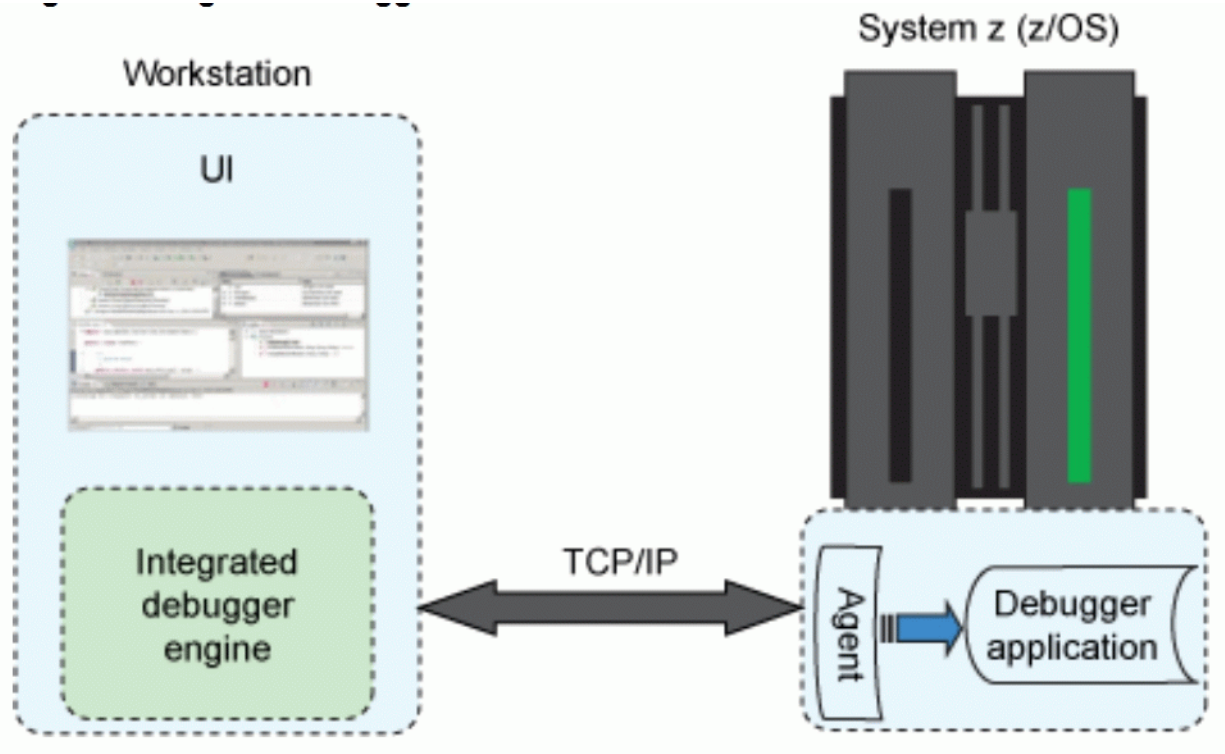
- define menu selection context and options



Agenda

- Workbench configuration and organisation
- z/OS project organisation
- Development (editor, datasets, procedures, custom menus)
- **Testing and debugging**
- CICS (+ BMS)
- DB2 (SQL + stored procedures)
- Problems encountered
- Q & A

Testing and debugging



Integrated debugger

alternative: z/OS Debug Tool, third party debug tools (e.g. Xpediter)

Testing and debugging (cont.)

Test preparation

- Compile program with TEST(SEPARATE)

  generates side file - Debug data set - referenced in Property Group

  //SYSDEBUG DD DSN=<HLQ>.SYSDEBUG.PDS,DISP=SHR
- Batch run

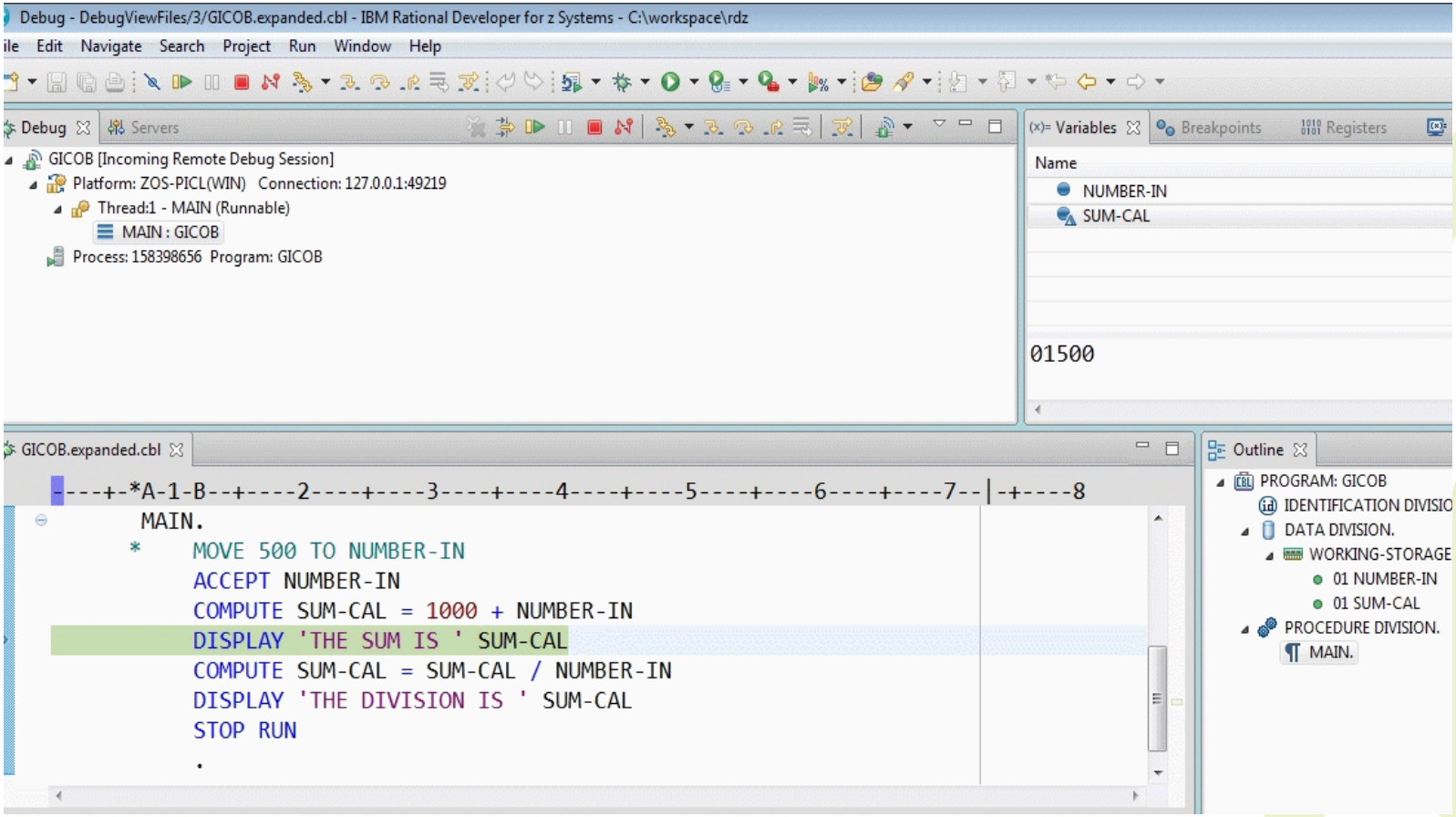
  support of debug manager via Language Environment (LE)

```
//AQEDEBUG DD  DISP=SHR,DSN=<HLQ>.SYSDEBUG.PDS     dataset name of side file
//CEEOPTS     DD  *
  TEST(,,,DBM)
/*
```

or

```
 TEST(,,,TCPIP&ipaddress%8001:)
```

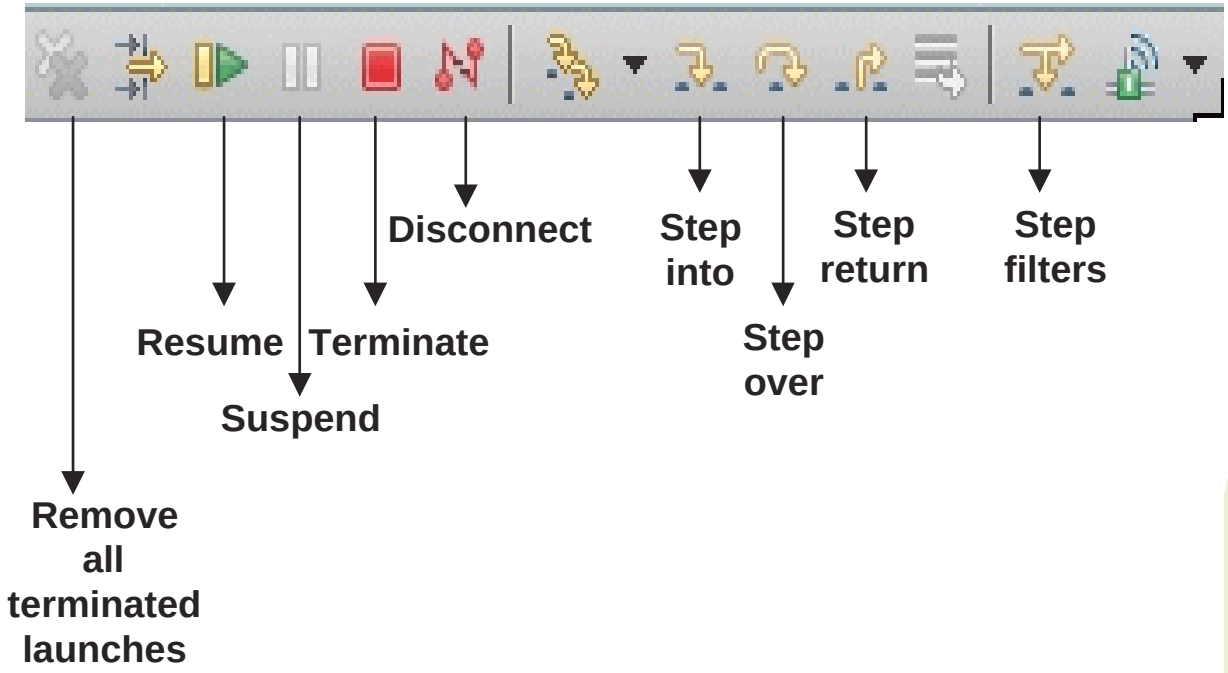
Testing and debugging - Debug perspective



Practical experiences with RDz/IDz

Testing and debugging - Debug features

- navigation

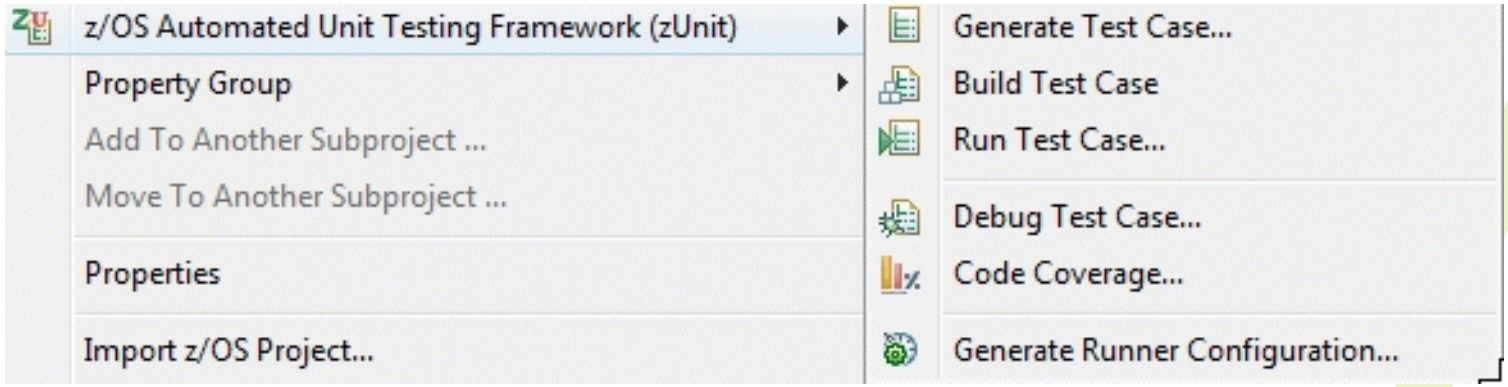


- Run to location

  -> using outline view, or program control flow view
- (qualified - conditional) breakpoints (+ import/export facility)
- variable monitoring and modification

zUnit

based on xUnit framework



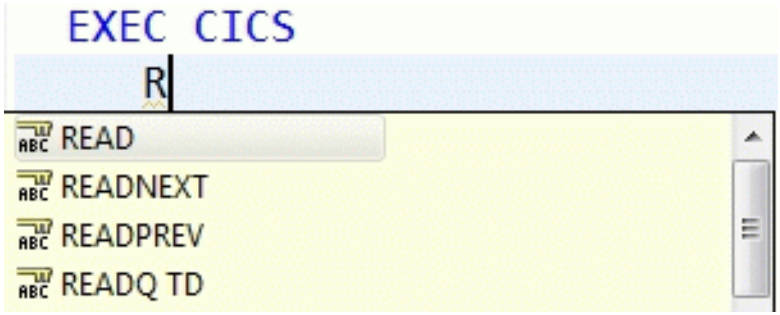
Test case definitions for program functions, providing in/out data

Agenda

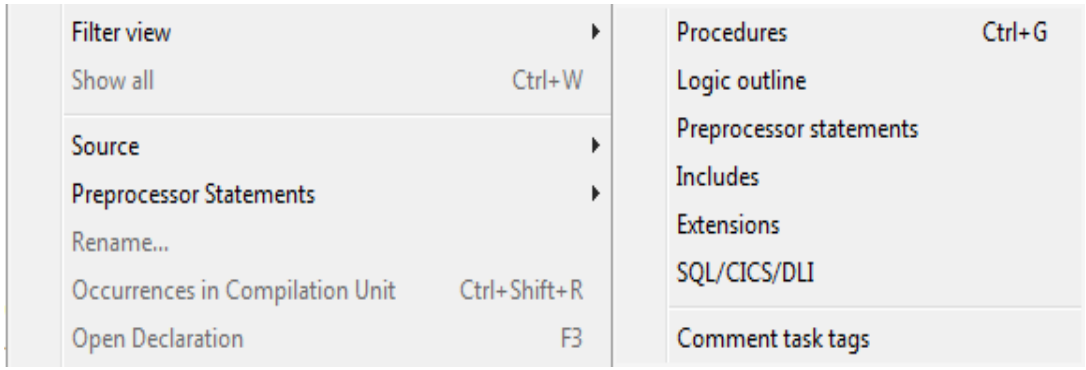
- Workbench configuration and organisation
- z/OS project organisation
- Development (editor, datasets, procedures, custom menus)
- Testing and debugging
- **CICS (+ BMS)**
- DB2 (SQL + stored procedures)
- Problems encountered
- Q & A

CICS program editor

Language (COBOL or PLI) editor provides support for EXEC CICS calls



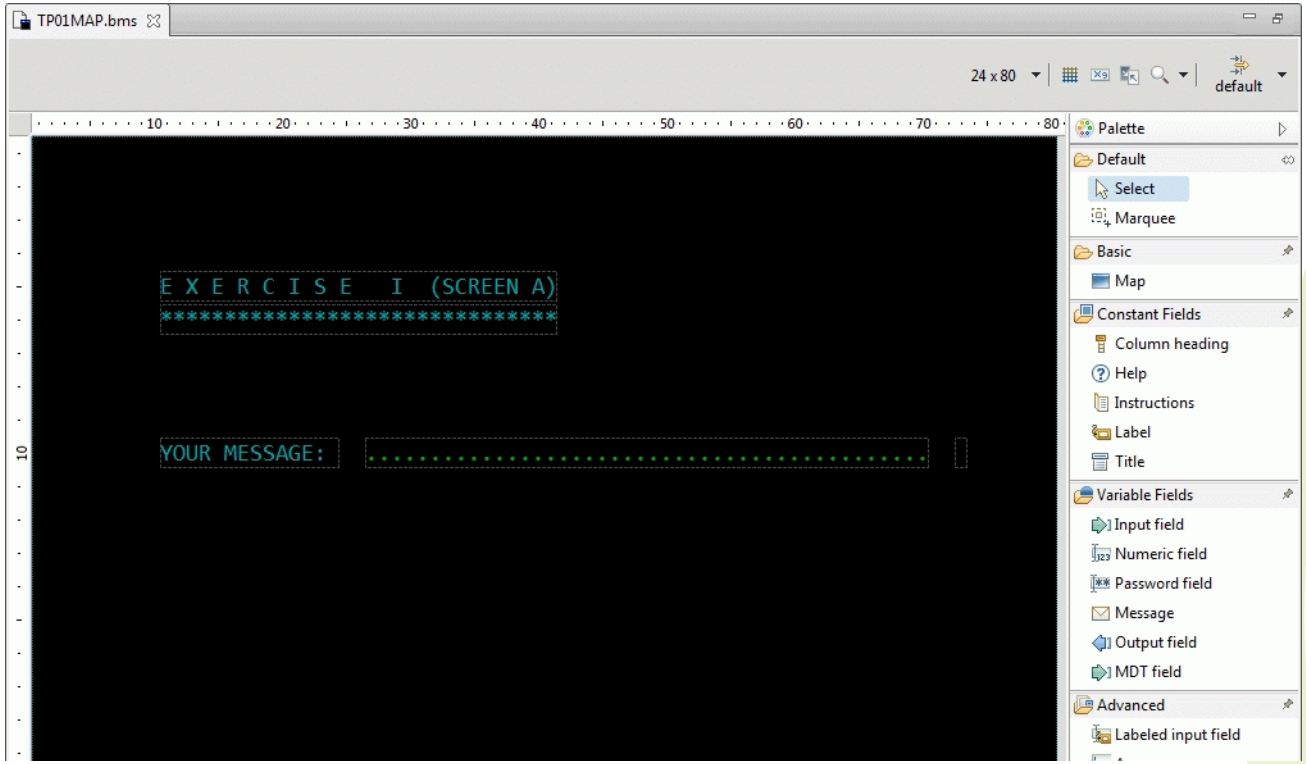
LPEX editor -> Filter view on EXEC (SQL/CICS) statements



BMS map editor

Special (default) editor with 3 tabs

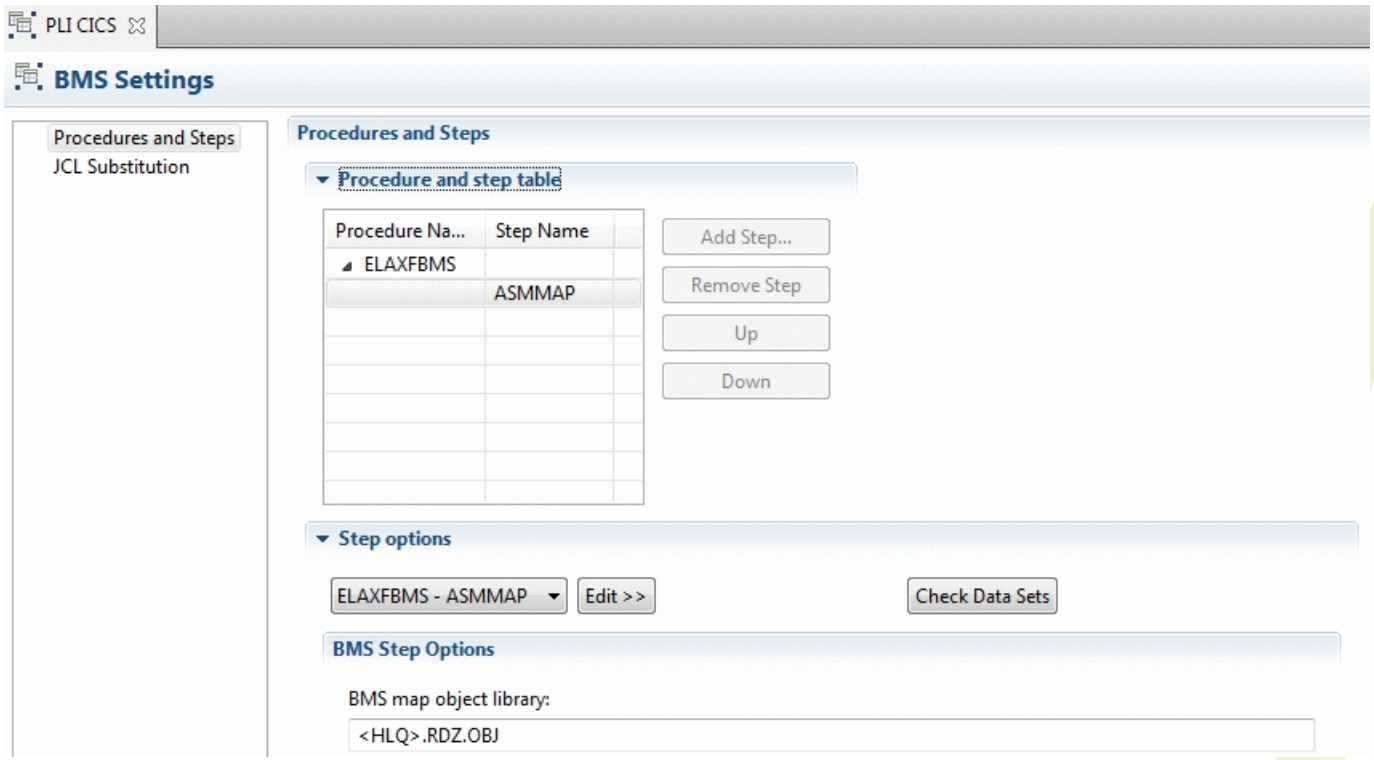
- design: graphical WYSIWYG editor - palette



- source: assembler editor + content assist
- preview: of resulting 3270 map (optional web view)

BMS generation

Generation of physical and symbolic maps via procedures defined in associated property group



CICS - Test and debug

CICS debugging configuration for integrated debugger

- Compile CICS program with TEST(SEPARATE) option
- Start **Host Connection Emulator** (HCE)
- Define CICS debugging profile in CADP

```
CADP      -    CICS Application Debugging Profile Manager   -     CICSTTBT

All Debugging Profiles       (A=Activate,I=Inactivate,D=Delete,C=Copy)

  Owner     Profile  S Tran Program  Compile Unit  Applid    Userid    Term Type
_ $EXAMPLE  COMP1    I T*   P*       *             CICSREG1  PANDREWS  TTT1 Comp
_ $EXAMPLE  COMP2    I TR   *        SAMPCOMPUN +  CICSREG2  DRBEARD*  TTT2 Comp
_ $EXAMPLE  COMP3    I TRN3 PROG3    *             CICSREG3  *         TTT2 Comp
_ $EXAMPLE  CORBA    I T*                          *         IORWERTH       Corb
_ TB00127   DEMOCOB  A TC56 TC56PROG *             CICSTTBT  TB00127   *    Comp
_ TB00127   DEMOPLI  A TP56 TP56PROG *             CICSTTBT  TB00127   *    Comp
```

Note: definition of profile also possible via **CICS Explorer**

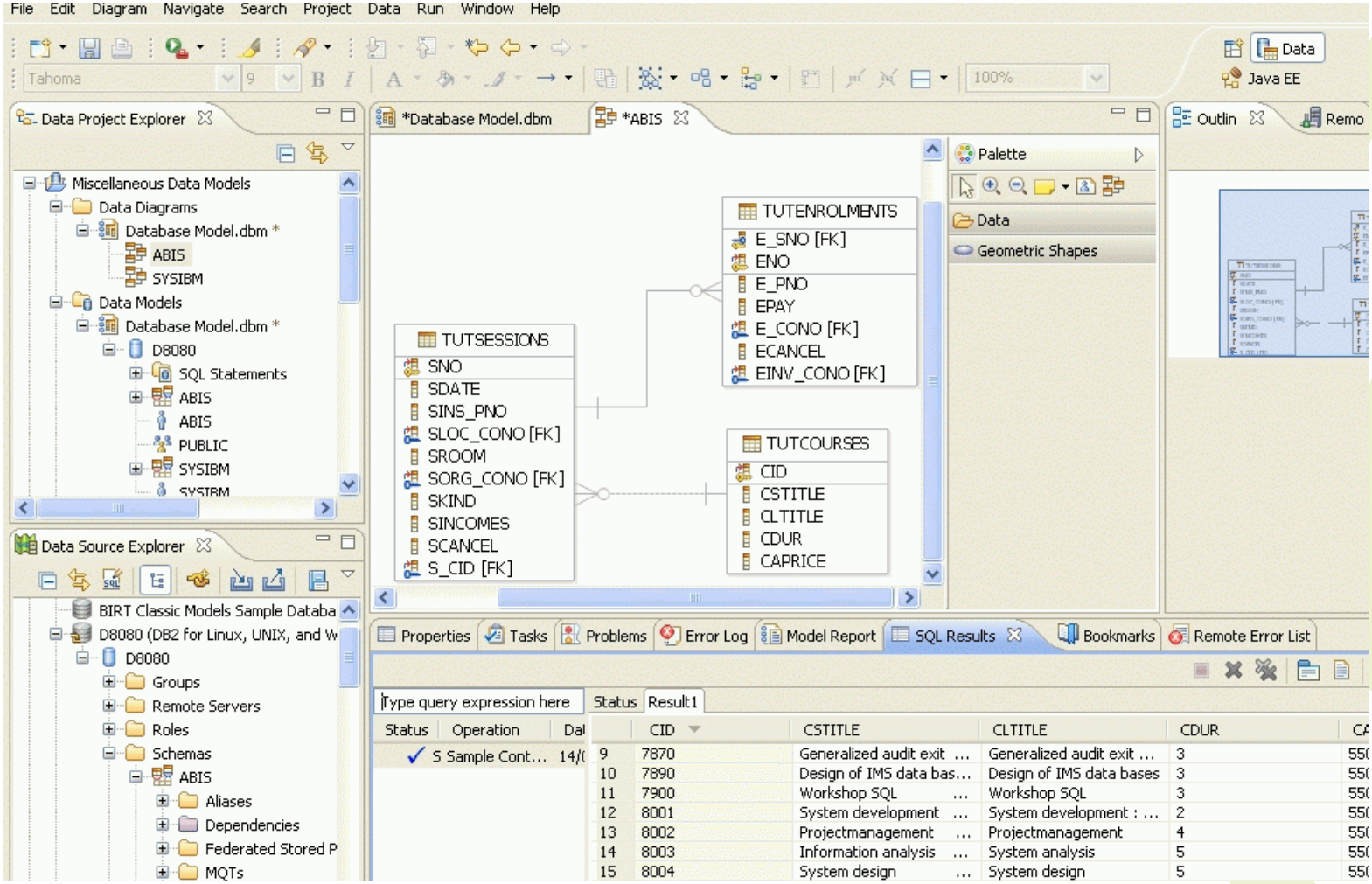
- run CICS transaction

  debug manager intercepts the execution and activates Debug mode -> RDz switches to Debug Perspective

Agenda

- Workbench configuration and organisation
- z/OS project organisation
- Development (editor, datasets, procedures, custom menus)
- Testing and debugging
- CICS (+ BMS)
- **DB2 (SQL + stored procedures)**
- Problems encountered
- Q & A

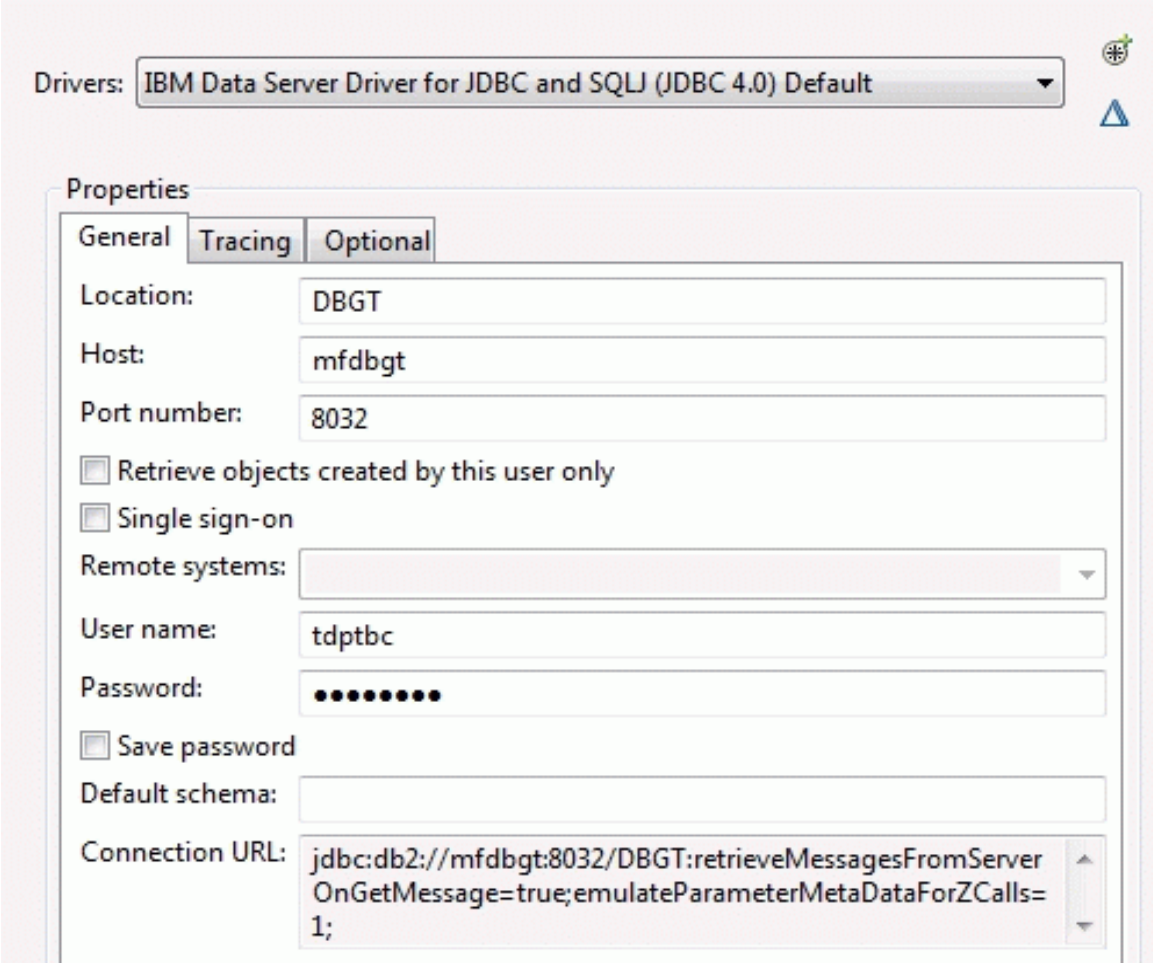
switch to data perspective (integrated Data Studio)



DB definition

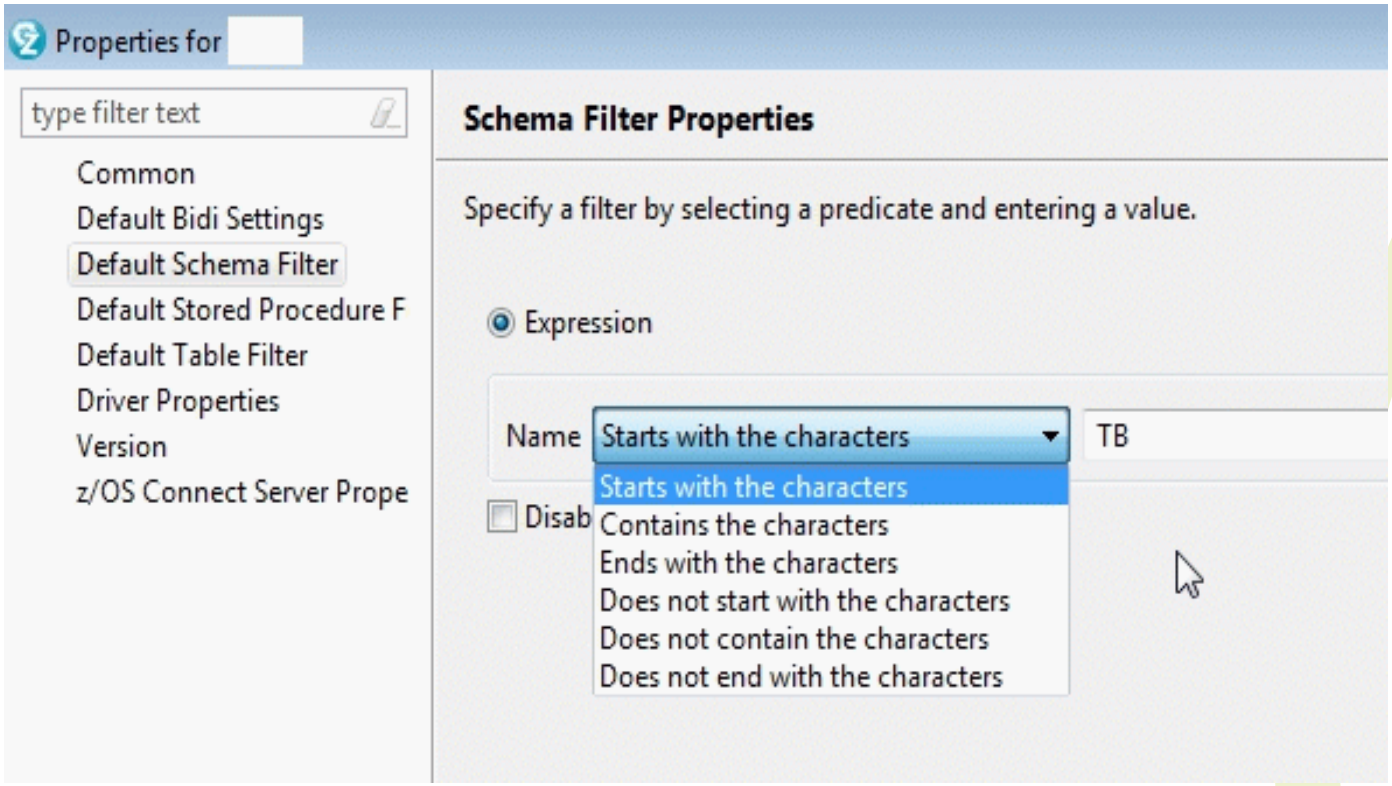
Data source explorer

- **connection (setup)**



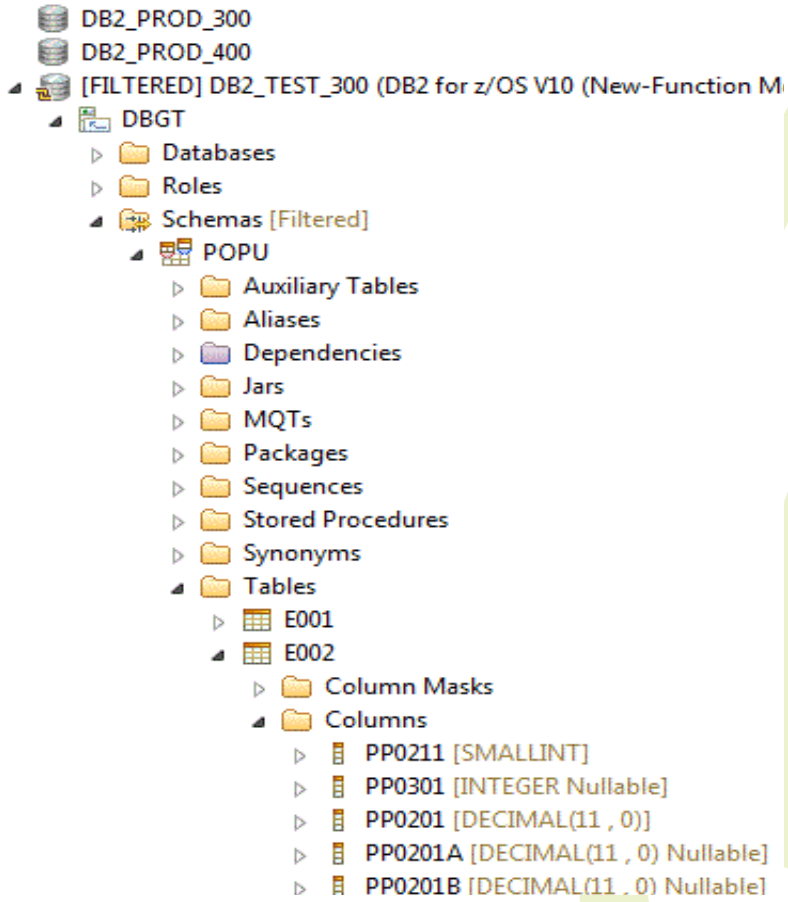
DB definition (cont.)

- **apply filters (schema - table)**



Preparation of data access

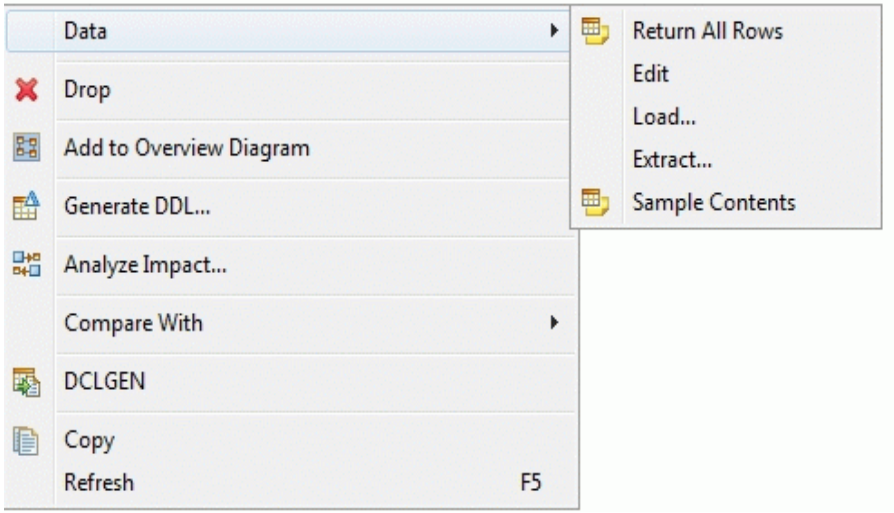
- **overview of DB objects**
  - **database**
  - **schema**
  - **table**
    - columns
    - constraints
    - triggers
    - indexes
    - …
  - **views**
  - **…**



Note: interesting information for individual object is visible in properties view!

Data manipulation

- **create + manipulate DB objects (+ DDL)**



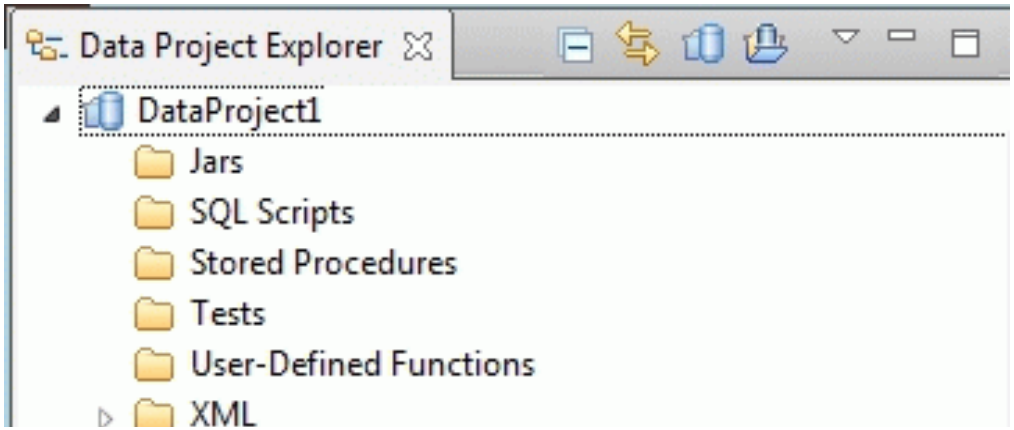
- **manipulate data**

DCLGEN -> COBOL copybook / PLI include

Preparing SQL

Data project explorer

- create Data development project
- create programs/scripts in project



- SQL queries
  - SQL scrapbook
  - SQL query builder

Test (run) SQL

run SQL from context menu (F5)

result shown in SQL result view

SQL Results

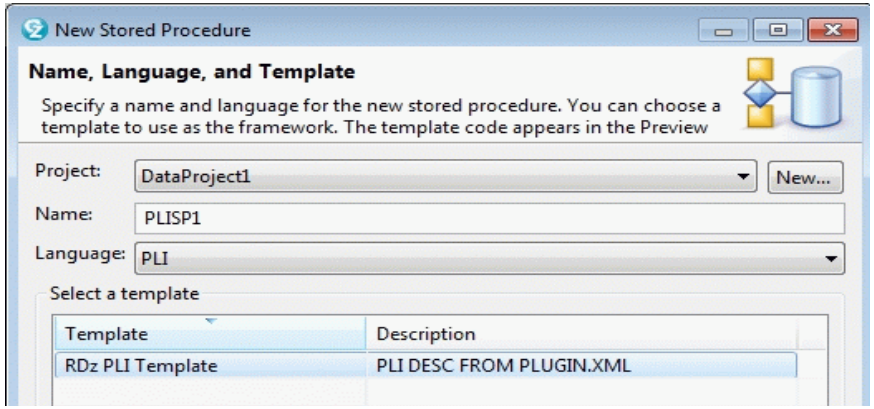
PP0211	PP0201	PP0201A	PP0201B	PP0201C	PP0208	PP0206	PP0301	PP0204
304	0	NULL	NULL	0	0	F	NULL	1901-01-01
309	1	166018	220797	1	1982120332533	N	8500	1982-12-03
309	2	NULL	NULL	726459	1956092533062	N	8500	1956-09-25
309	6	NULL	NULL	6	-6	N	NULL	1915-03-02
322	10	3262	951293	3262	1987070510668	N	NULL	1987-07-05
311	33	NULL	NULL	33	1946121334920	N	NULL	1946-12-13
305	41	NULL	NULL	41	1950110340847	F	NULL	1950-11-03
315	44	NULL	NULL	44	1941070408724	F	NULL	1941-07-04
304	57	3282	1714521	57	1984061633620	N	NULL	1984-06-16
309	58	1300	1973808	58	1973012026494	N	NULL	1973-01-20

Use the created SQL in COBOL / PL/I program

Stored procedures - creation

- native (SQL PL)
  - uses SQL commands for program logic
  - source code in CREATE PROCEDURE
  - define to DB2 (with run)
- external (language specific)
  - uses host language (PL/I, COBOL, ...) for program logic
  - load module and package
  - define to DB2 (with separate CREATE PROCEDURE)

Data project -> Stored procedures folder -> new



Test stored procedures

- Test via calling program

  EXEC SQL CALL ...
- For SQL PL stored procedures:

  direct test is possible via **Run** option

Debugging

- for COBOL/PLI
  - debugging possible if compiled with TEST option!
- for SQL PL
  - add to procedure definition

    ALLOW DEBUG MODE
  - specify preference **Enable Debugging** in deploy routine options

    (see Window -> preferences -> session manager)

  Debug via context menu -> Debug

Problems encountered - performance

- PC capacity - Windows (7, 8, 10)
  - 32 - 64 bit
  - 2.5 GB disk (install)
  - memory 2 GB
- Host capacity - z/OS (1.13, 2.x)
  - RSE daemon heap size
  - private address space size
  - define enough initiators (blocked during batch integrated debug)

  depends on number of users
- network capacity
  - ???

Problems encountered - bugs

- editor features not enabled
- include processing is slow
- BMS (graphical) editor manipulation -> source code not correct
- syntax checking not always correct
  - includes not found (even if SYSLIB definition OK)
- synchronisation lost, in case of network/connection problems
- push-to-client feature not immediately activated?
- special characters not always represented correctly

  e.g. PLI not sign

Possible tool improvements

- same features in LPEX and language specific editors
  - filter view
  - show in ...
  - surround with ...
- ‘filtering’ of views
  - show only ‘relevant’ views/options according to ‘working set’
    e.g. do not show UNIX features if only working with z/OS
  - option: ‘hide’ instead of ‘show disabled’ feature
- clean up of terminated projects
- ‘modular’ property groups
  - base COBOL batch
  - additional options for CICS
  - additional options for DB2
- missing info about remote activity/status -> waiting for ....

Additional information

- ABIS training course on IDz 

  `http://www.abis.be/html/en1312.html`

- IDz product site

  `http://www-03.ibm.com/software/products/en/ibm-developer-for-z-systems-enterprise-edition`

- IBM Knowledge center

  `https://www.ibm.com/support/knowledgecenter/SSQ2R2`

- IBM Developer works - Rational Developer for System z Hub

  `https://www.ibm.com/developerworks/community/groups/service/html/communityview?communityUuid=df67969e-ba40-44c7-a1ca-ef4a2aa99e01`

Q & A

Thank you

Gie Indesteege

Trainer and Consultant

gindesteege@abis.be

thanks you