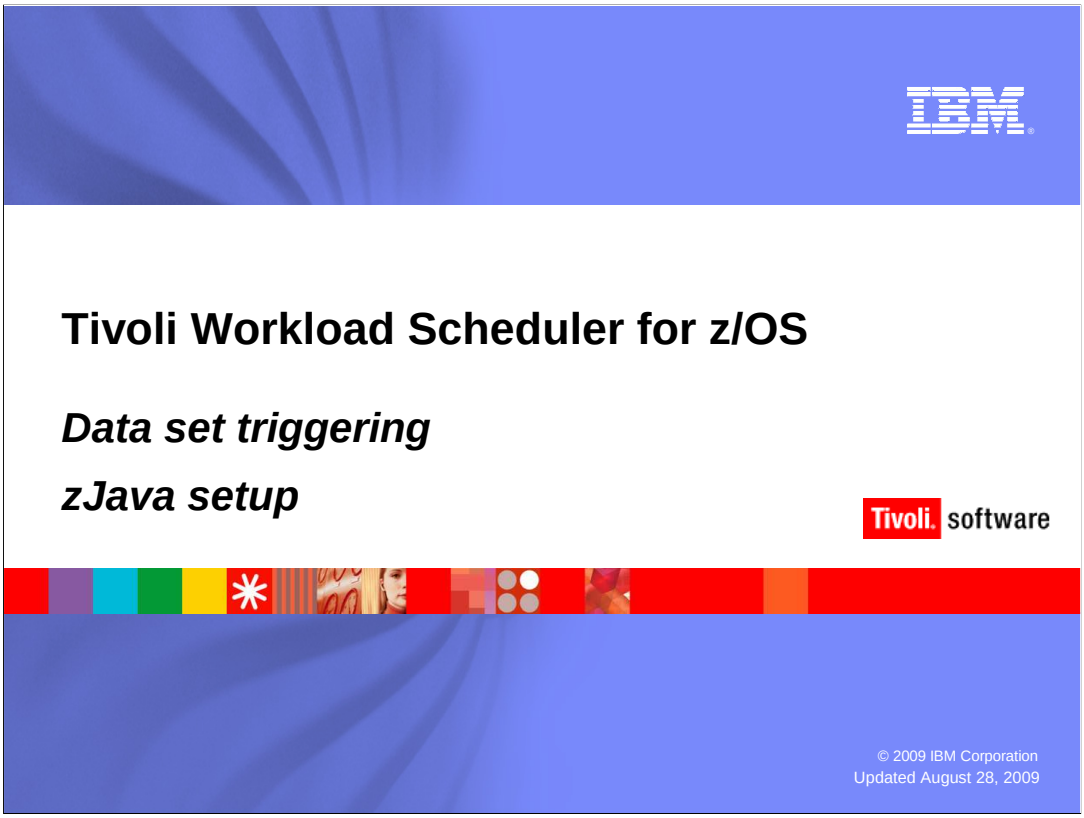
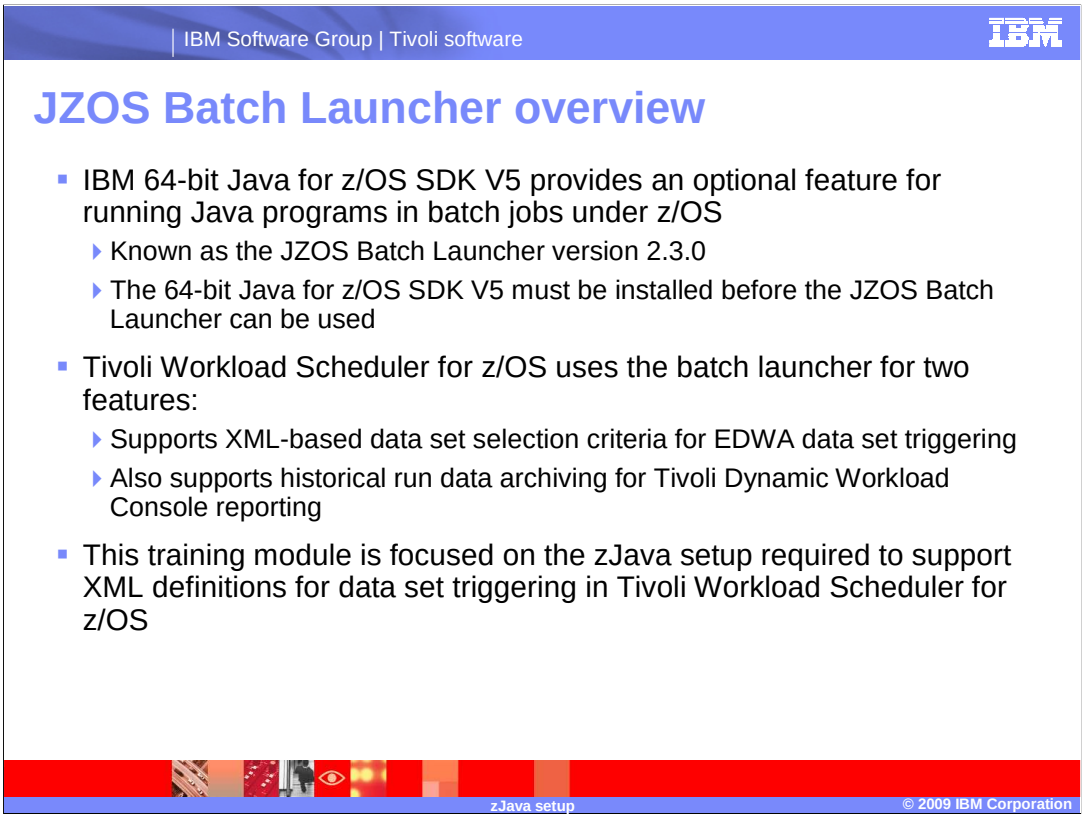


Tivoli Workload Scheduler for z/OS® zJava setup.

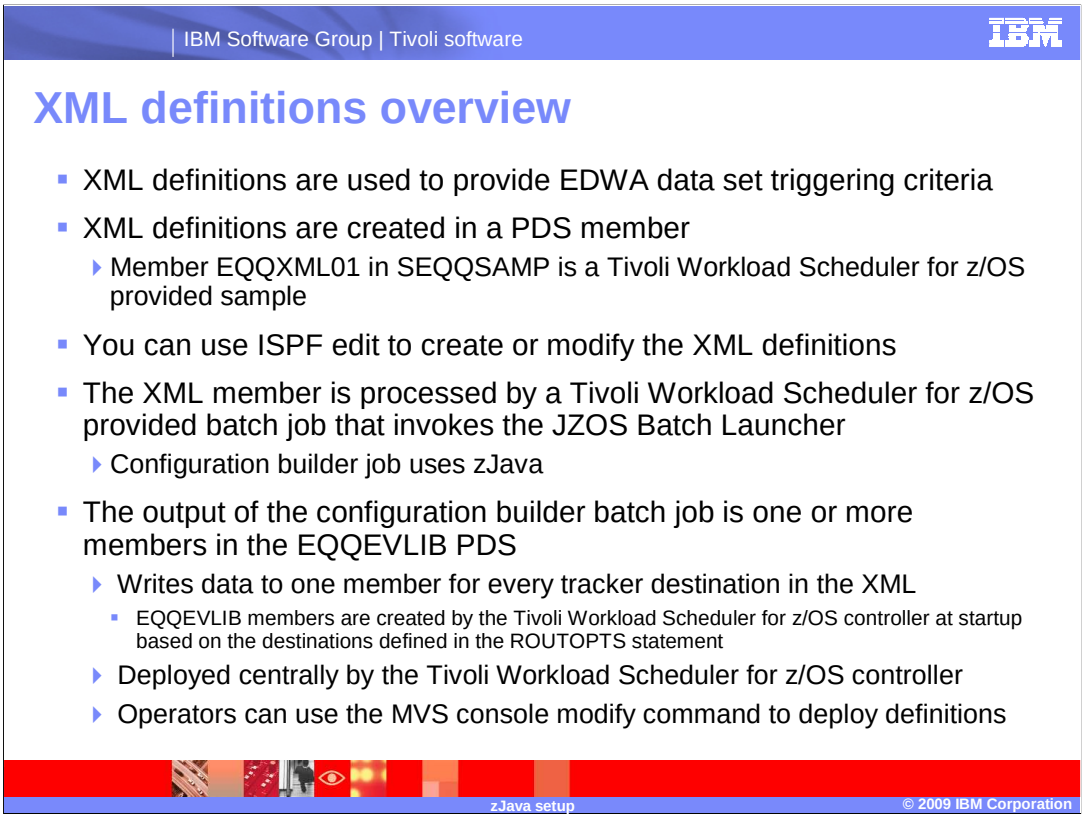
This training module shows you how to set up Tivoli Workload Scheduler for z/OS 8.5 to use Java™ for z/OS. Tivoli Workload Scheduler for z/OS uses the JZOS Batch Launcher feature of Java for z/OS SDK version 5 to support two Tivoli Workload Scheduler for z/OS 8.5 features. Therefore, the Java for z/OS SDK V5 and the JZOS Batch Launcher V2.3.0 must be properly installed and verified before you attempt to implement these two Tivoli Workload Scheduler for z/OS 8.5 features. These two Tivoli Workload Scheduler for z/OS 8.5 features are listed on the next slide. See the training modules *Java for z/OS SDK V5 installation and verification* and *JZOS Batch Launcher implementation* for details on zJava. The Java for z/OS SDK V5 and the JZOS Batch Launcher V2.3.0 are collectively referred to as zJava in this training module.



JZOS Batch Launcher overview.

Tivoli Workload Scheduler for z/OS 8.5 uses the JZOS Batch Launcher to support the use of Java programs in batch mode under z/OS. The Java for z/OS SDK V5 must be installed before you can implement the JZOS batch launcher. Tivoli Workload Scheduler for z/OS 8.5 uses the JZOS Batch Launcher to support two distinct features. Tivoli Workload Scheduler for z/OS 8.5 uses the JZOS Batch Launcher to run Java programs for processing XML and to support historical run data archiving for Tivoli Dynamic Workload Console reporting. The XML is used to define data set selection criteria for data set triggering. See the training module *XML Implementation* for details on using XML.

This training module is focused on the Tivoli Workload Scheduler for z/OS 8.5 zJava setup that is required to support the XML implementation of data set triggering criteria definitions.

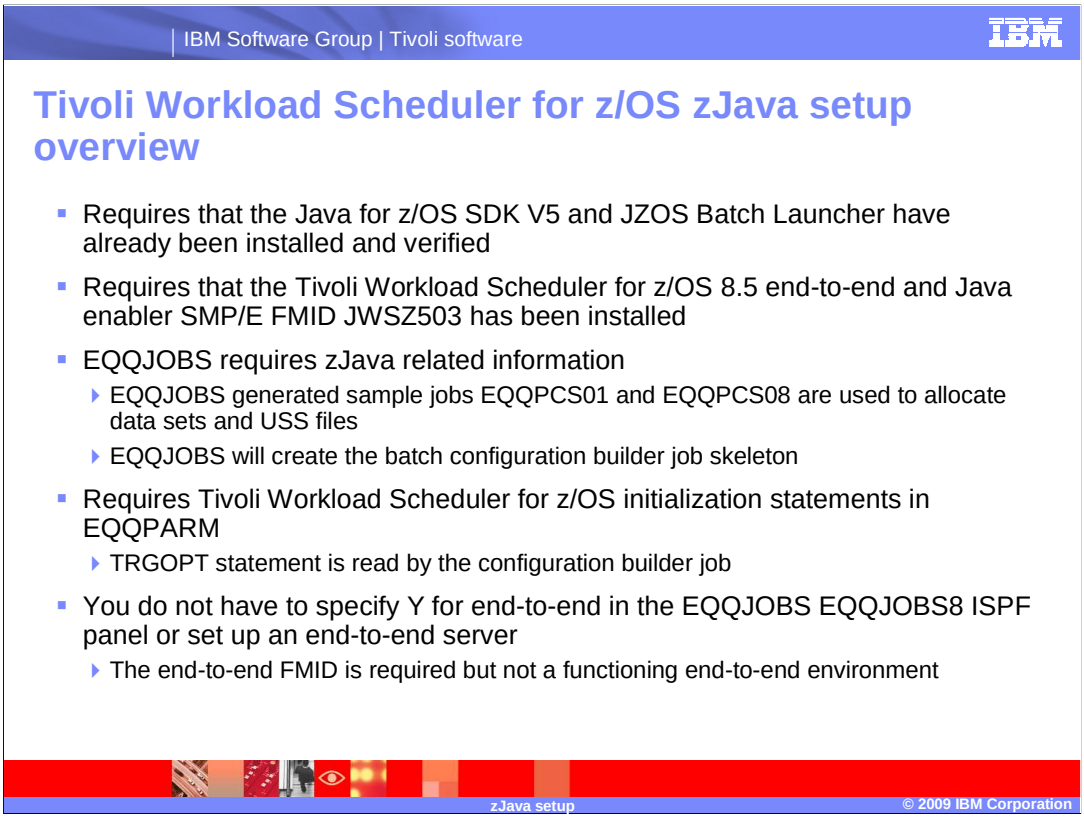


XML definitions overview.

XML definitions are used to provide EDWA data set triggering criteria. XML definitions are created in a PDS member. The member EQQXML01 in the SEQQSAMP data set is a Tivoli Workload Scheduler for z/OS provided sample. You can use ISPF edit to create or modify the XML definitions. The XML member is processed by a Tivoli Workload Scheduler for z/OS provided batch job that invokes the JZOS batch launcher. The job is referred to as the configuration builder job, and it uses zJava. The output of the configuration builder batch job is one or more members in the EQQEVLIB partitioned data set. The Tivoli Workload Scheduler for z/OS controller started task creates one member in the EQQEVLIB data set for every destination defined in the ROUTOPTS initialization statement. The EQQEVLIB members have the same member name as the destination name.

The configuration builder job writes data set triggering information to a member in EQQEVLIB (or EVLIB) for every tracker destination defined in the XML source PDS member. The member must already exist in EQQEVLIB when the configuration builder job runs. These processed XML definitions are deployed to tracker destinations centrally by the Tivoli Workload Scheduler for z/OS controller. The controller deploys the members from EQQEVLIB to each respective tracker started task, where they are written into the EQQJCLIB data set of the tracker started task as a member name EQQEVLST. Operators can use the MVS console modify command to deploy definitions also. See the training module *XML Implementation* for details.

Note: The historical run data archiving for Tivoli Dynamic Workload Console reporting feature, which also requires zJava, does not use XML definitions on z/OS.

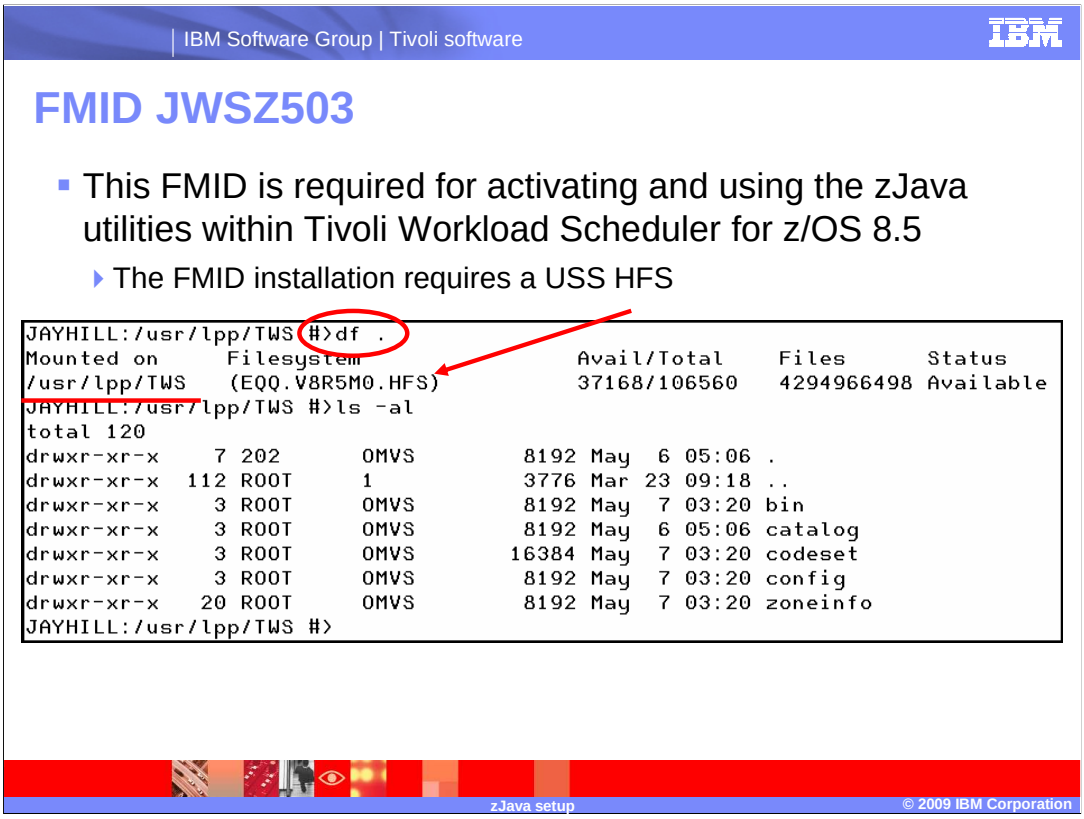


Tivoli Workload Scheduler for z/OS zJava setup overview.

To use XML in Tivoli Workload Scheduler for z/OS data set triggering, the Java for z/OS SDK V5 and JZOS Batch Launcher must be installed and verified. Tivoli Workload Scheduler for z/OS must be set up to include the necessary support for zJava. The support for zJava requires the Tivoli Workload Scheduler for z/OS 8.5 end-to-end and Java enabler SMP/E FMID JWSZ503. You must also provide information and options when running the Tivoli Workload Scheduler for z/OS EQQJOBS utility. The EQQJOBS generated sample jobs EQQPCS01 and EQQPCS08 are used to allocate data sets and USS files that are used to support the XML feature of data set triggering. EQQPCS08 is also used for the historical run data archiving and reporting feature, which requires zJava setup.

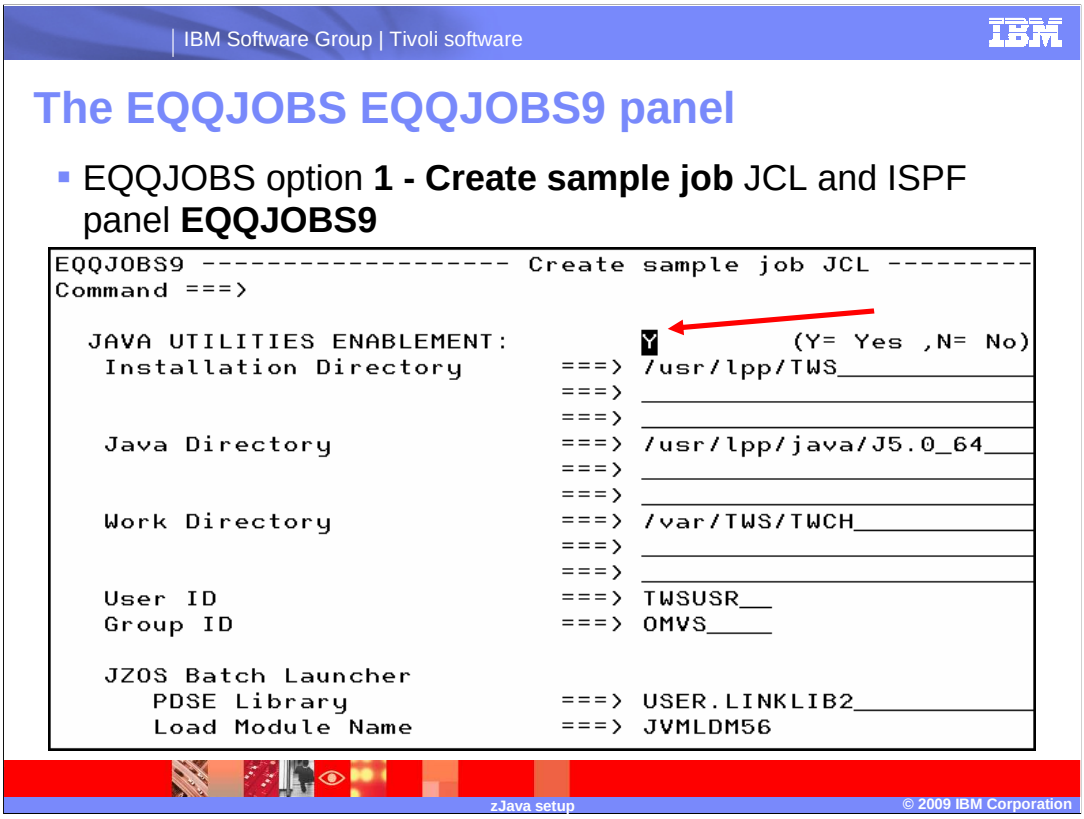
EQQJOBS will also create the batch configuration builder job skeleton. The configuration builder job requires the Tivoli Workload Scheduler for z/OS TRGOPT initialization statement in EQQPARM when you run the job. The TRGOPT statement is read and used by the configuration builder job.

You do not have to specify Y for end-to-end in the EQQJOBS EQQJOBS8 ISPF panel or set up an end-to-end server. The end-to-end FMID is required, but not a functioning end-to-end environment.



FMID JWSZ503.

This FMID is required for activating and using the zJava utilities within Tivoli Workload Scheduler for z/OS 8.5. The FMID installation requires a USS HFS that is populated by SMP/E and then mounted and available in USS. In the example on this slide, the HFS is mounted on the directory /usr/lpp/TWS. This directory will be part of the information that is provided when running the EQQJOBS utility.



The EQQJOBS EQQJOBS9 panel.

The ISPF panel EQQJOBS9 is part of the EQQJOBS dialog when option 1 of EQQJOBS is selected.

Input Fields Descriptions:

JAVA UTILITIES ENABLEMENT: Specify **Y** if you want to enable the reporting feature or the EDWA feature for data set triggering. Both features require a zJava environment.

Installation Directory: Specify the USS directory path where the Tivoli Workload Scheduler for z/OS HFS for FMID **JWSZ503** is mounted. This HFS contains various directories and files, including the /bin directory where the Tivoli Workload Scheduler for z/OS executables are installed.

Java Directory: Specify the directory path where the Java for z/OS SDK V5 HFS is mounted. This HFS contains the Java for z/OS SDK V5 installed directories and files.

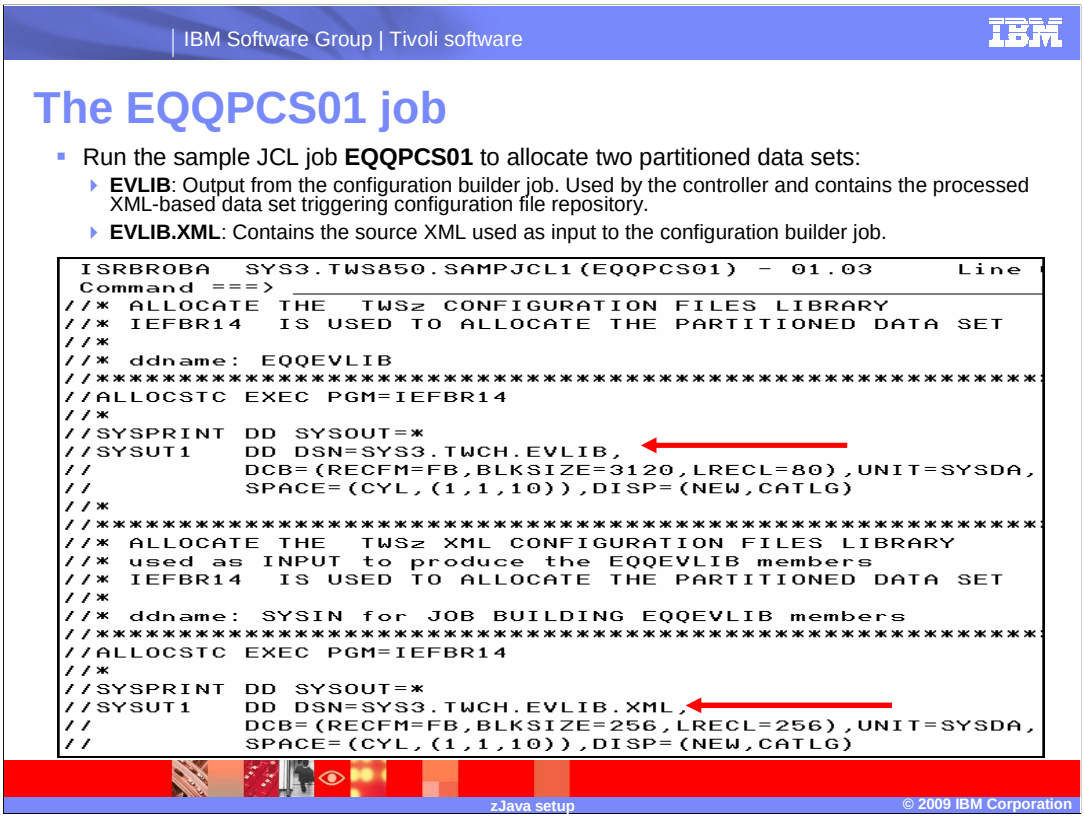
Work Directory: Specify where the subsystem-specific USS files are located. Each Tivoli Workload Scheduler for z/OS controller subsystem must have its own work directory.

User ID: Specify the Tivoli Workload Scheduler for z/OS user ID defined in RACF®.

Group ID: Specify the group ID for the user ID.

JZOS Batch Launcher PDSE Library: Specify the name of the PDSE that contains the JZOS Batch Launcher JVMLDM56 load module.

JZOS Batch Launcher Load Module Name: Specify the JZOS Batch Launcher load module name. The name is JVMLDM56 for the 64-bit Java for z/OS SDK V5.

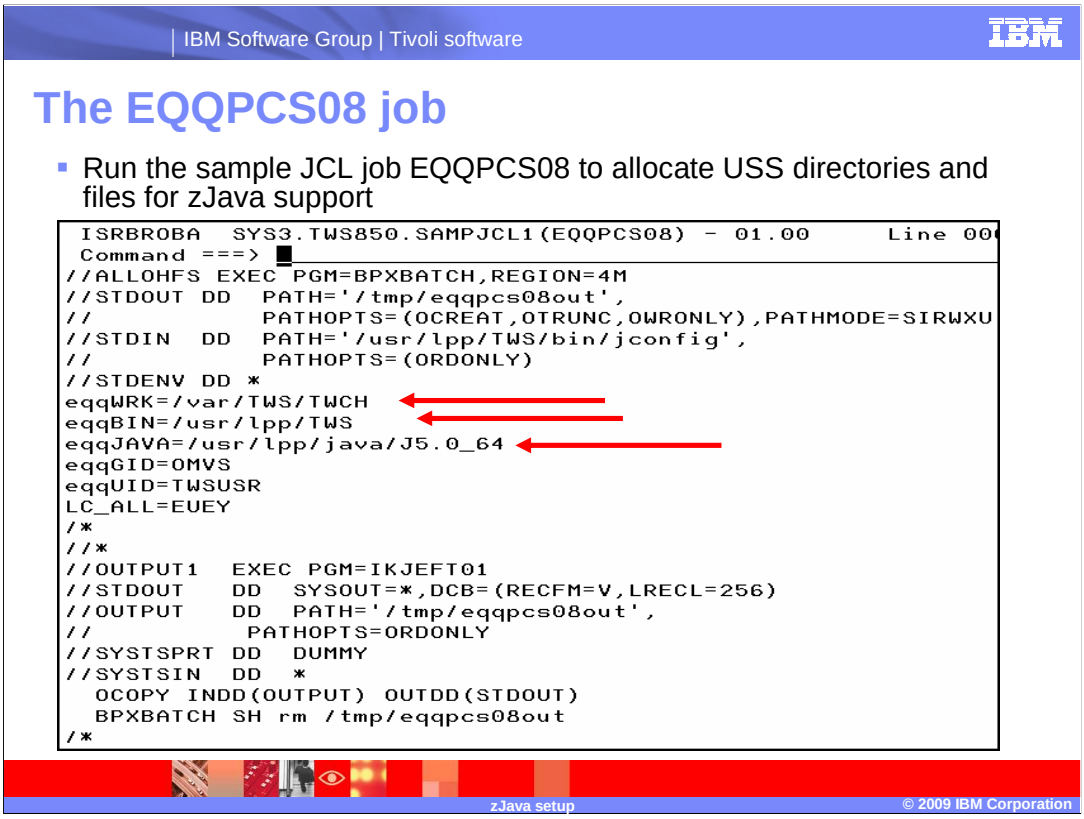


The EQQPCS01 job.

Run the sample JCL job EQQPCS01 to allocate two partitioned data sets. Both of these data sets are used by the configuration builder job.

The first data set is the EVLIB data set. EVLIB is used to contain the output from the configuration builder job. The output represents the processed XML data set triggering criteria definitions. There will be one PDS member in EVLIB for every tracker destination specified in the ROUTOPTS initialization statement. There will also be a member with the name $$$$$$$$, which represents the local tracker destination. EVLIB is used by the controller for deploying the processed XML definitions to the tracker started tasks. The DD name used by the controller for this data set is EQQEVLIB. When the controller deploys the members in EQQEVLIB to the tracker started tasks, the members are written into the tracker started task EQQJCLIB data set with a member name of EQQEVLST.

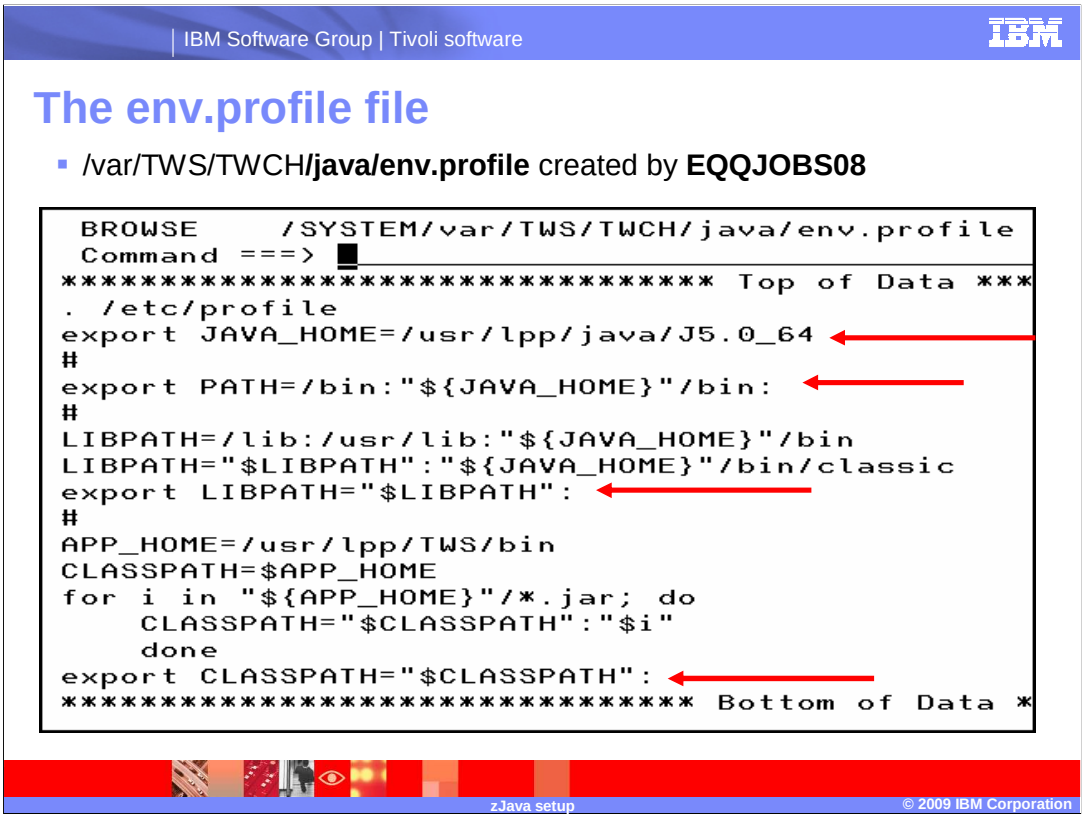
The second data set is EVLIB.XML. This data set will contain the source XML members used as input to the configuration builder job. You can copy the Tivoli Workload Scheduler for z/OS 8.5 sample member EQQXML01 to this data set and use it as a starting point for your XML definitions.



The EQQPCS08 job.

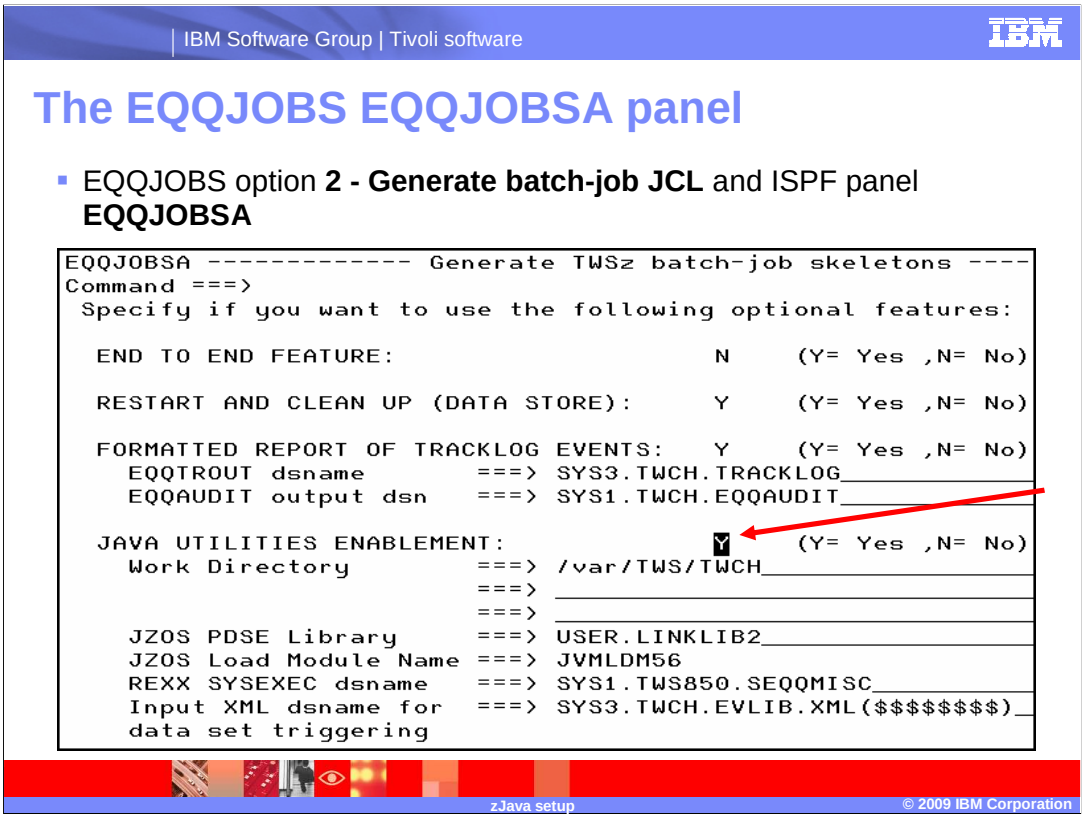
Run the sample JCL job EQQPCS08 to identify and allocate USS directories and files for zJava support. This JCL must be executed by a user with a USS uid equal to 0, or by a user with permissions to the RACF BPX.SUPERUSER FACILITY class profile, or by the user specified in the eqqUID parameter that is part of the group specified in eqqGID. If eqqGID and eqqUID were not specified in EQQJOBS, specify them in the STDENV DD input stream before you execute this JCL. Ensure that the '/bin/sh' login shell is used by the user executing this JCL.

eqqWRK specifies the work directory that is controller subsystem specific. eqqBIN specifies the directory where the Tivoli Workload Scheduler for z/OS binary executables are located. eqqJAVA specifies the directories where the Java for z/OS SDK V5 is installed.



The env.profile file.

The env.profile file will be executed by the configuration builder job to establish the zJava runtime environment that is required. The env.profile file executes four different USS export commands to set up access paths to required zJava directories and files. The env.profile file is created by the EQQPCS08 sample job in the java directory under the Tivoli Workload Scheduler for z/OS controller subsystem-specific work directory path.



The EQQJOBS EQQJOBSA panel.

The ISPF panel EQQJOBSA is part of the EQQJOBS dialog when option 2 of EQQJOBS is selected.

Input Fields Descriptions:

JAVA UTILITIES ENABLEMENT: Specify **Y** if you want to enable the reporting feature or the EDWA feature for data set triggering. Both features require a zJava environment.

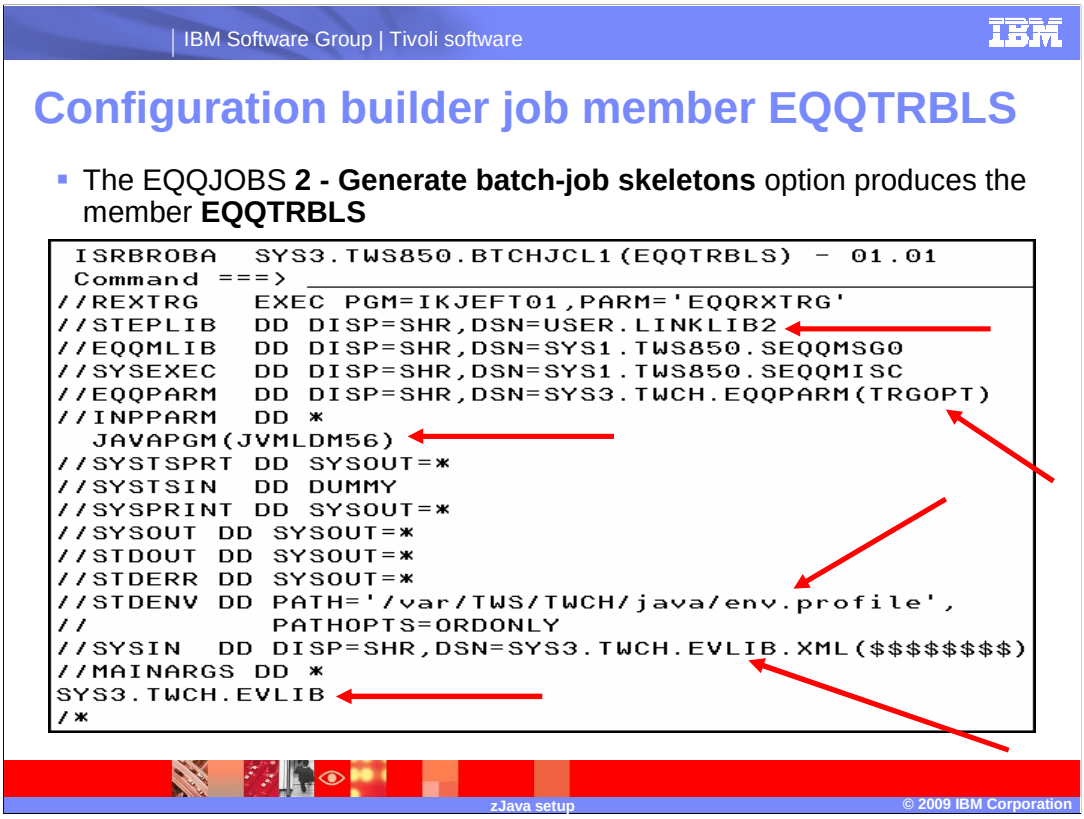
Work Directory: Specify where the controller subsystem-specific USS files are located. Each Tivoli Workload Scheduler for z/OS controller subsystem must have its own work directory to enable the zJava features.

JZOS PDSE Library: Specify the name of the PDSE that contains the JZOS Batch Launcher JVMLDM56 load module.

JZOS Load Module Name: Specify the JZOS Batch Launcher load module name. The name is JVMLDM56 for the 64-bit Java for z/OS SDK V5.

REXX SYSEXEC dsname: Specify the SEQQMISC library name where the EQQRXARC, EQQRXAR1, EQQRXTRG, and EQQRXTR1 members are located. These are REXX execs used by the historical archiving feature (EQQRXARC and EQQRXAR1) and EDWA data set triggering feature (EQQRXTRG and EQQRXTR1).

Input XML dsname for data set triggering: Specify the input XML data set name that will contain the data set triggering criteria definition members. These criteria definition members will be used to build the data set triggering configuration members to be sent to the tracker started tasks.



Configuration builder job member EQQTRBLS.

You will invoke this job using the Tivoli Workload Scheduler for z/OS 8.5 ISPF dialog option 1.7.3. The job will process the XML criteria definitions. Make sure that the REXX/370 runtime libraries are available to TSO/E and that the IRXCMPTM table is defined, or if you want to use the REXX/370 alternate libraries make sure they are available to TSO/E. If you use the REXX/370 alternate libraries, change the EQQRXTRG parameter value to EQQRXTR1 in the EXEC statement of the JCL.

The STEPLIB DD statement allocates the PDSE data set where the JZOS Batch Launcher zJava runtime load module JVMLDM56 is located.

The SYSEXEC DD statement allocates the Tivoli Workload Scheduler for z/OS SEQQMISC data set where EQQRXTRG and EQQRXTR1 REXX execs are located.

The EQQPARM DD statement allocates the parameter member that contains the TRGOPT initialization statement.

The SYSIN DD statement allocates the input data set member containing the data set triggering event rules in XML format. The XML is processed and used to populate the output data set members with the configuration builder data to be sent to the tracker started tasks.

The MAINARGS DD statement provides the name of the output PDS data set EQQEVLIB. This partitioned data set is used to store the output members of configuration builder. This data set is allocated by the Tivoli Workload Scheduler for z/OS controller started task and the members are deployed to the tracker started tasks by the controller.

The INPPARM DD statement provides the parameters that are passed to EQQRXTRG or EQQRXTR1. In this example the zJava runtime load module name JVMLDM56 is being passed.

The syntax for the possible parameters is **PARAMETER(VALUE)**.

To comment out a parameter, use an asterisk prefix. For example: *PARAMETER(VALUE).

The valid parameters are:

•JAVAPGM: This is the Java virtual machine load module name. The name is JVMLDM56 for the 64-bit Java for z/OS SDK V5.

•JVPARMS: These are the optional parameters for the Java batch launcher. For example, specify JVPARMS(+T) to set trace level messages for the Java batch launcher.

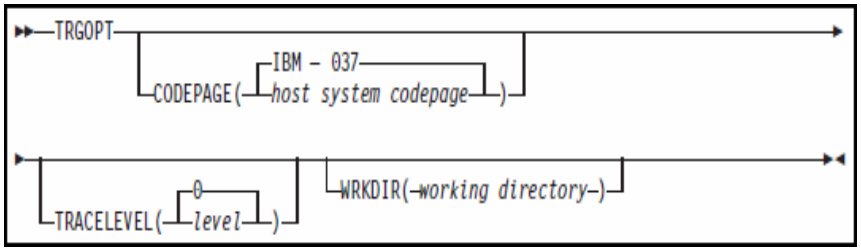
If you get the message **JVMJZBL1044E The Java virtual machine aborted** when running the job, increase the region memory size and rerun the job.

An example of running this job is available in the training module *XML Implementation*.

The historical run data archiving for Tivoli Dynamic Workload Console reporting also uses a similar job skeleton, which is located in the member EQQDBARS. The EQQDBARS job requires the zJava environment but runs a different REXX exec, called EQQRXARC. You can run the EQQDBARS job from Tivoli Workload Scheduler for z/OS 8.5 ISPF panel option 3.6. EQQDBARS does not process XML.

The TRGOPT initialization statement

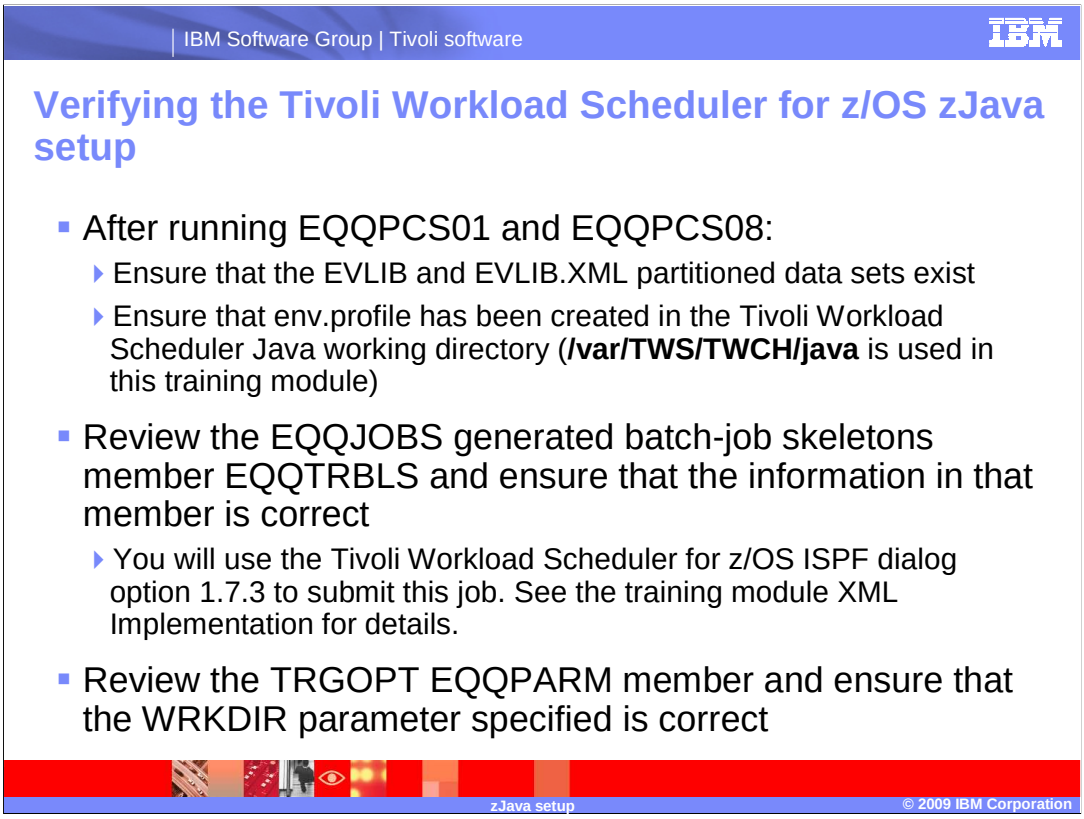
- Used by the Java program that creates configuration files for data set triggering
- Syntax



- Example: TRGOPT CODEPAGE(IBM-280)
  TRACELEVEL(1)
  WRKDIR(/var/TWS/TWCH)
- WRKDIR will contain files used by the configuration builder job:
  - **TriggersFormatter.log** and **env.profiles**

The TRGOPT initialization statement.

The TRGOPT initialization statement should be coded and placed in a member of the EQQPARM data set. The member containing this statement will be read by the Java program that is executed as part of the configuration builder job shown on the previous slide. The configuration builder job will use the WRKDIR parameter to locate the Java subdirectory and execute the env.profile file.

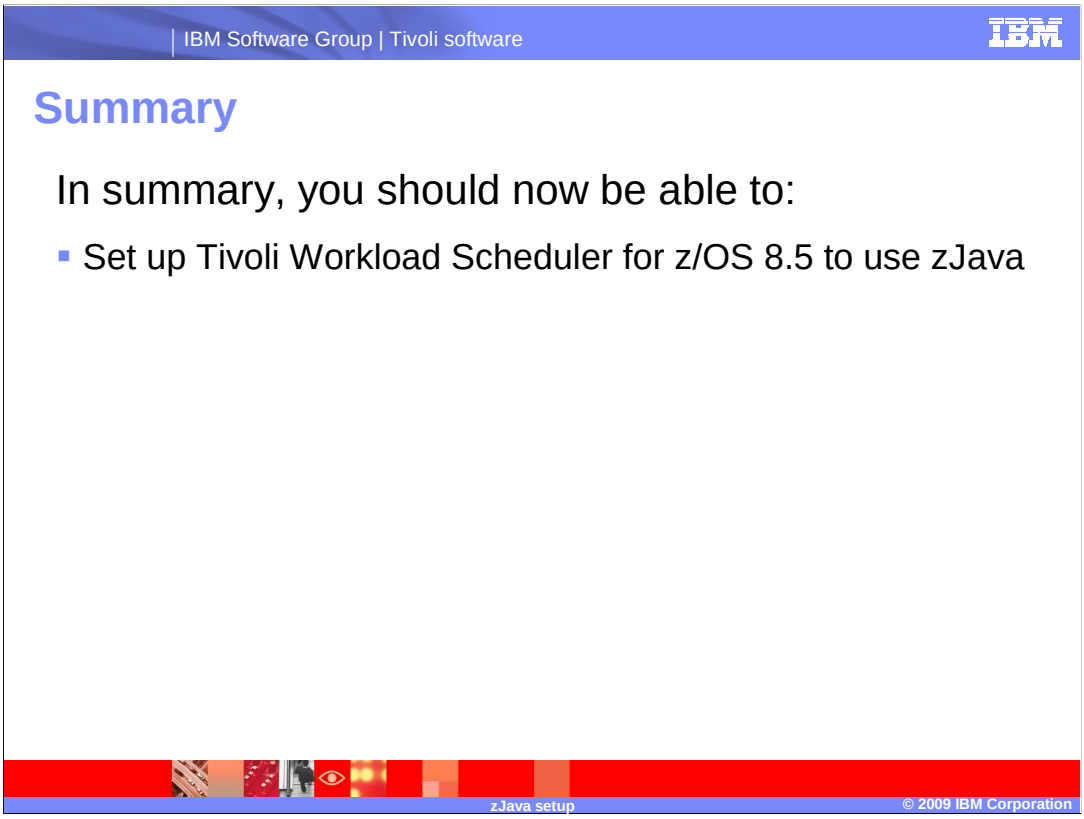


Verifying the Tivoli Workload Scheduler for z/OS zJava setup.

After running EQQPCS01 and EQQPCS08, ensure that the EVLIB and EVLIB.XML partitioned data sets exist. Remember that the EVLIB data set is used by both the configuration builder job and the Tivoli Workload Scheduler for z/OS controller started task. The EVLIB data set is allocated by the controller using the DD statement EQQEVLIB.

Ensure that env.profile file has been created in the Tivoli Workload Scheduler for z/OS zJava working directory. The directory path /var/TWS/TWCH/java is used in this training module. Review the EQQJOBS generated batch-job skeletons member EQQTRBLS and ensure that the information in that member is correct. Modify the member if necessary.

You will use the Tivoli Workload Scheduler for z/OS ISPF dialog option 1.7.3 to submit the configuration builder job. See the training module *XML Implementation* for details. Review the TRGOPT EQQPARM member and ensure that the WRKDIR parameter specified is correct.



Summary.

In summary, you should now be able to set up Tivoli Workload Scheduler for z/OS 8.5 to use zJava.

Feedback

Your feedback is valuable

You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send e-mail feedback:

mailto:iea@us.ibm.com?subject=Feedback_about_tws-for-zos-zjava-setup.ppt

This module is also available in PDF format at: ../tws-for-zos-zjava-setup.pdf

You can help improve the quality of IBM Education Assistant content by providing feedback.

Trademarks, copyrights, and disclaimers

IBM, the IBM logo, ibm.com, and the following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

RACF Tivoli z/OS

If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of other IBM trademarks is available on the Web at "Copyright and trademark information" at http://www.ibm.com/legal/copytrade.shtml

Java, Java runtime environment, and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements or changes in the products or programs described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (for example, IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products.

IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.