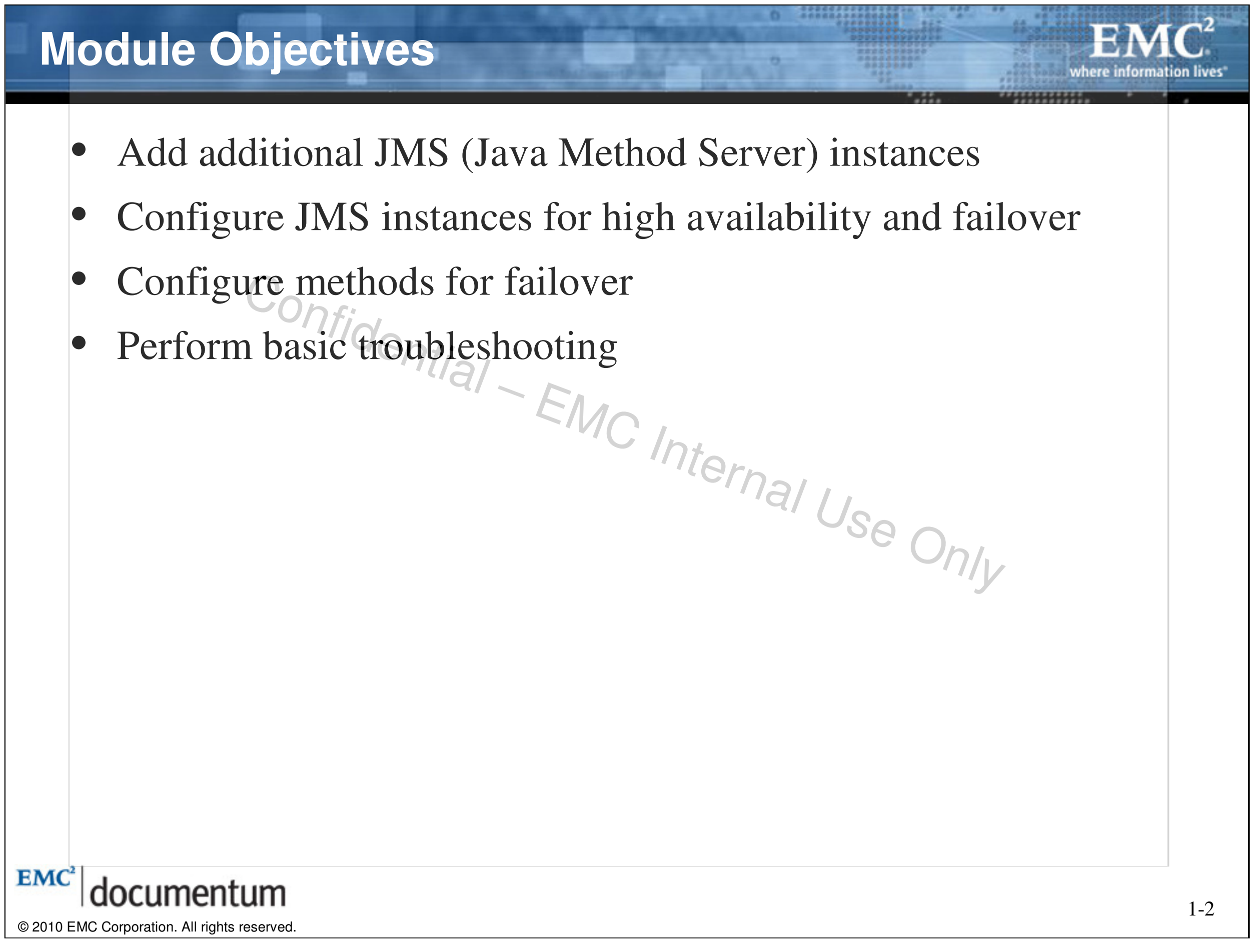
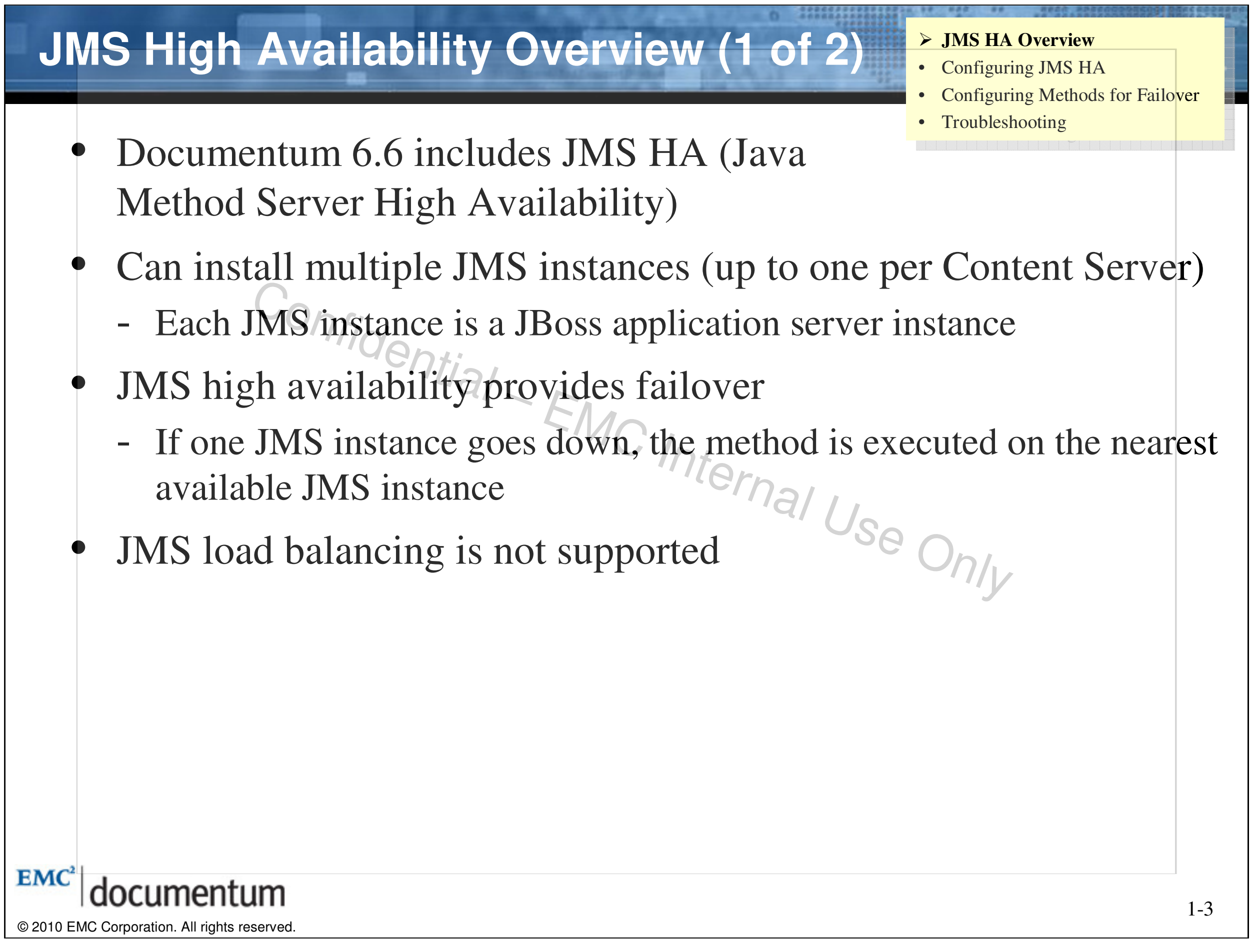
Module X

JMS HA (Java Method Server High Availability)



Module Objectives

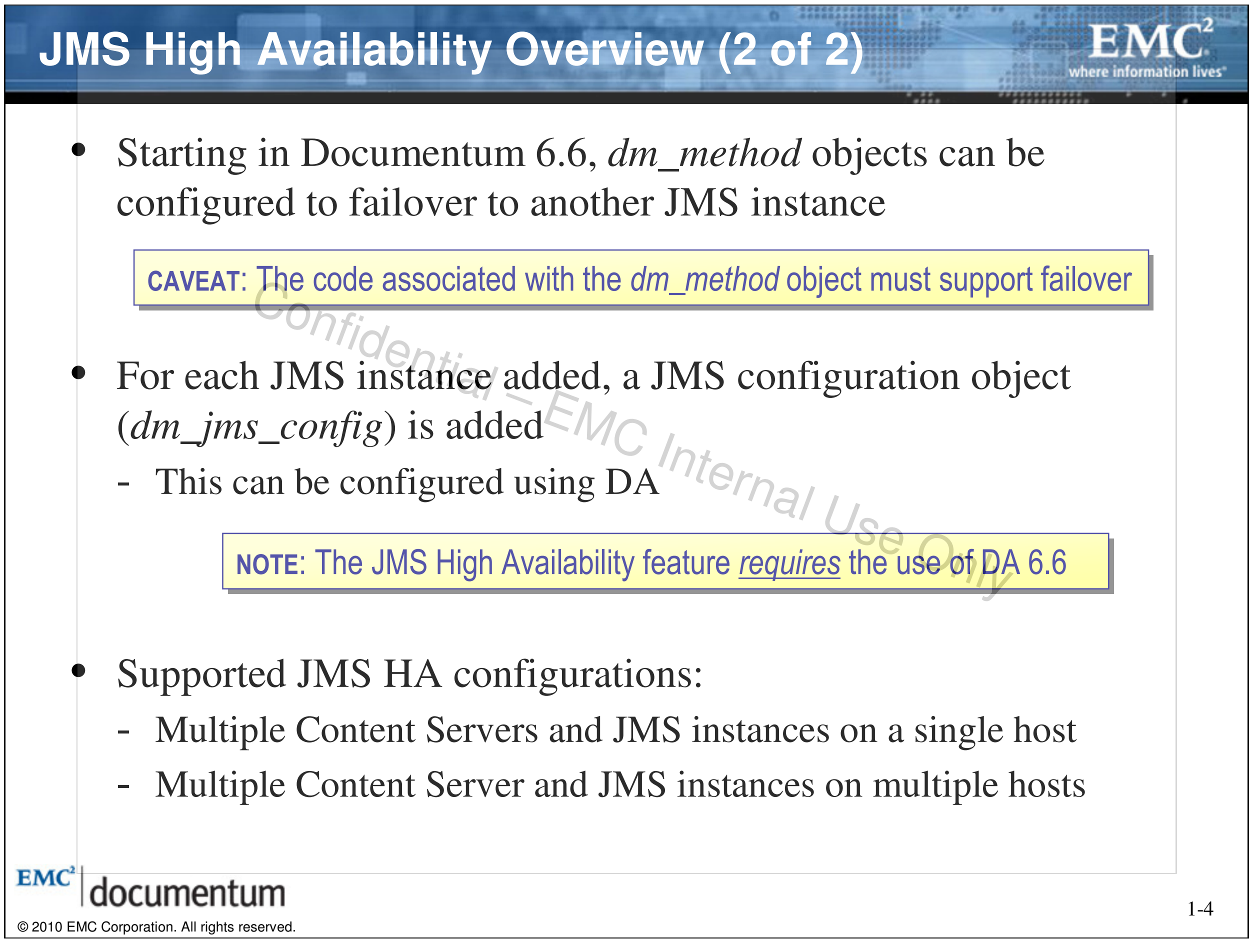
- Add additional JMS (Java Method Server) instances
- Configure JMS instances for high availability and failover
- Configure methods for failover
- Perform basic troubleshooting



JMS High Availability Overview (1 of 2)

- Documentum 6.6 includes JMS HA (Java Method Server High Availability)
- Can install multiple JMS instances (up to one per Content Server)
  - Each JMS instance is a JBoss application server instance
- JMS high availability provides failover
  - If one JMS instance goes down, the method is executed on the nearest available JMS instance
- JMS load balancing is not supported

When multiple Content Servers are installed on a single host, by default, there is only one Java Method Server instance. When the first Content Server is installed, a default Java Method Server instance is installed with it. By default, subsequent Content Server installations on that host use the default Java Method Server instance. If on a single host, each Content Server manages a separate repository, each repository/Content Server can use its own, separate JMS instance. This JMS instance must be installed and configured separately from the Content Server installation. This is described later in the lecture.



JMS High Availability Overview (2 of 2)

- Starting in Documentum 6.6, *dm_method* objects can be configured to failover to another JMS instance

CAVEAT: The code associated with the *dm_method* object must support failover

- For each JMS instance added, a JMS configuration object (*dm_jms_config*) is added
  - This can be configured using DA

NOTE: The JMS High Availability feature *requires* the use of DA 6.6

- Supported JMS HA configurations:
  - Multiple Content Servers and JMS instances on a single host
  - Multiple Content Server and JMS instances on multiple hosts

JMS High Availability and DA 6.6

The JMS configuration object (*dm_jms_config*) contains information about the JMS instance associated with a Content Server. The *server_config_id* of the JMS configuration object is set equal to the *r_object_id* of the associated server configuration object (*dm_server_config*).

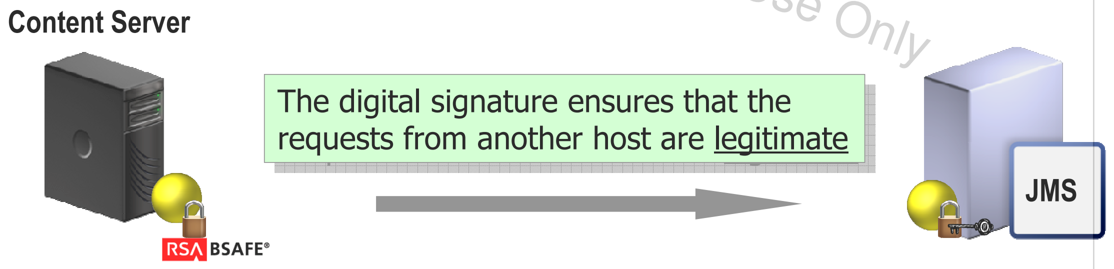
In DA version 6.6, whenever changes are made to the server configuration object, the changes are not versioned. Since the version does not change, the *r_object_id* value of the server configuration object does not change.

In DA versions prior to 6.6, whenever changes are made to the server configuration object, it is checked out and checked back in as a new version. The new version will have a different *r_object_id* value than the previous version. The JMS configuration object's *server_config_id* still contains the *r_object_id* of the previous version. Thus, the relationship between the JMS configuration object and the server configuration object is severed.

That is why it is required to use DA 6.6 when using the JMS high availability feature.

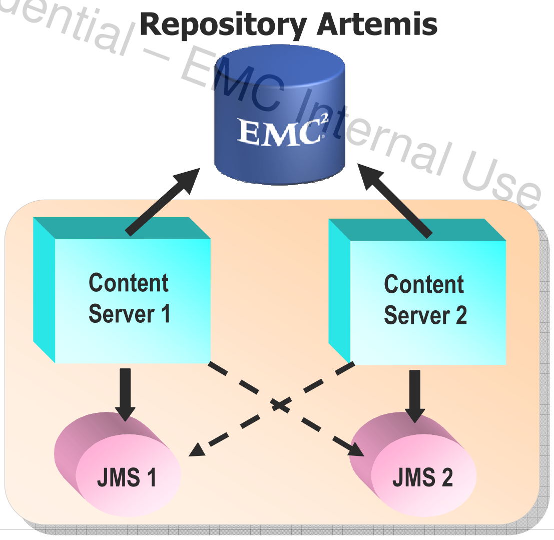
Digital Signatures and Method Requests

- Starting with Documentum 6.6, for each method request sent to the JMS, the Content Server generates a digital signature using the repository's private key
  - RSA BSAFE Crypto-C library is used to sign the method request
- The JMS verifies the signature using the repository's public key
  - Java 2 SDK Security API is used to verify signature

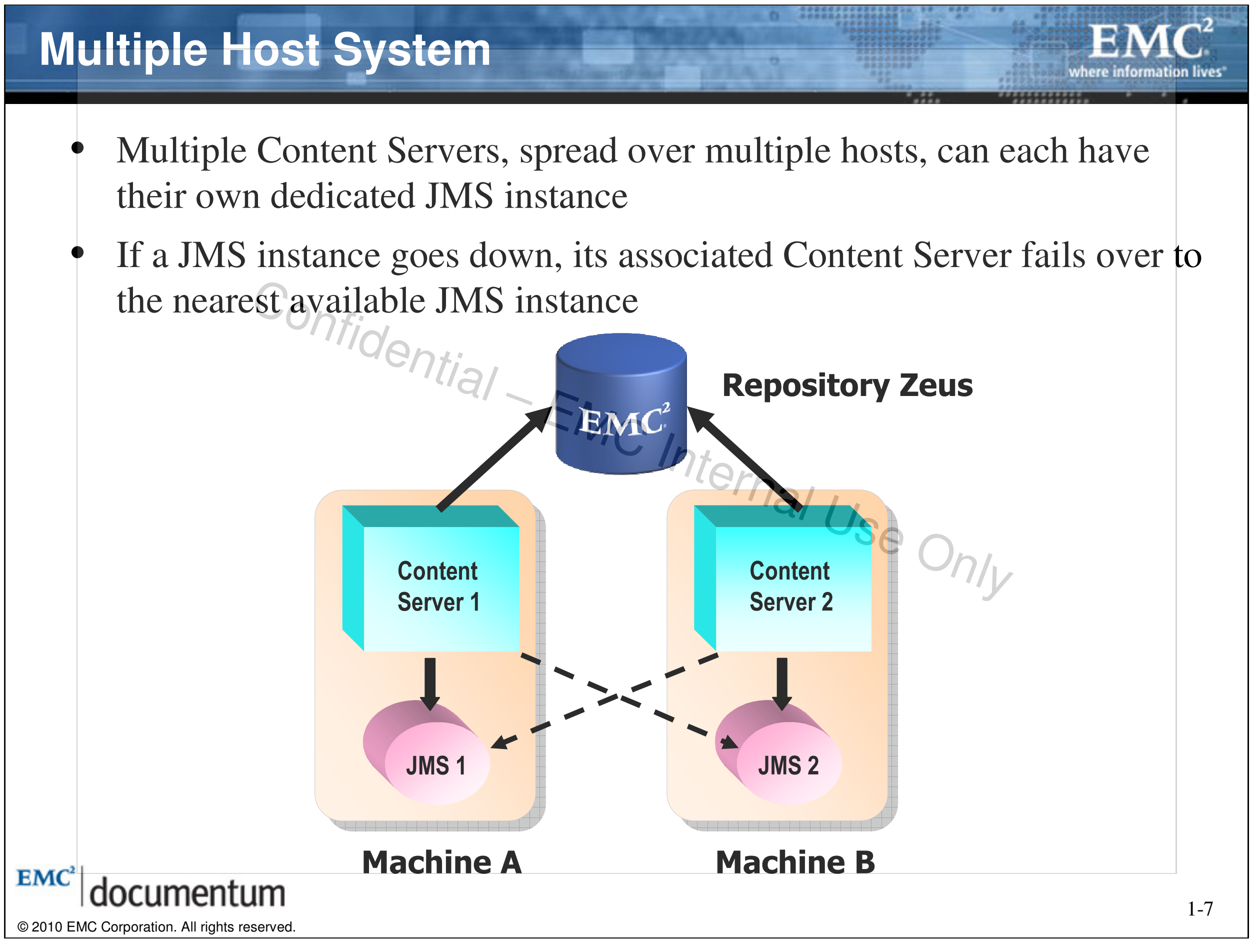

Content Server
The digital signature ensures that the requests from another host are legitimate
RSA BSAFE
JMS

Single Host System

- In a single host system, with one repository, each Content Server can have its own dedicated JMS instance
- If one JMS instance goes down, its associated Content Server automatically fails over to use the other JMS instance

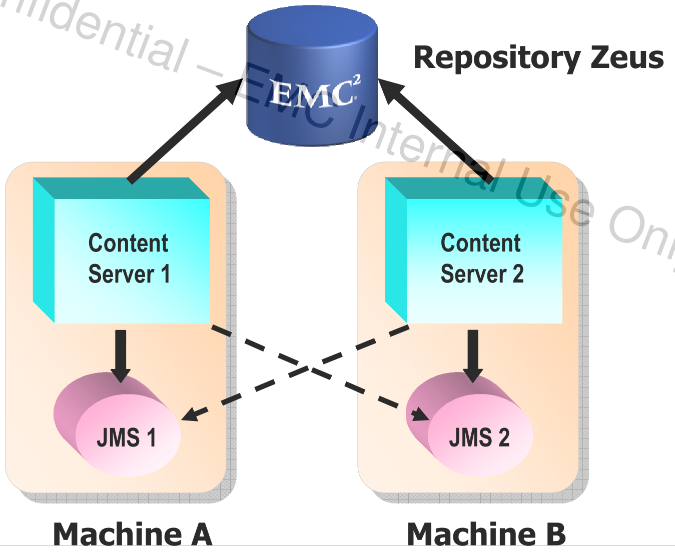


The arrows depict communication *from* the Content Server.

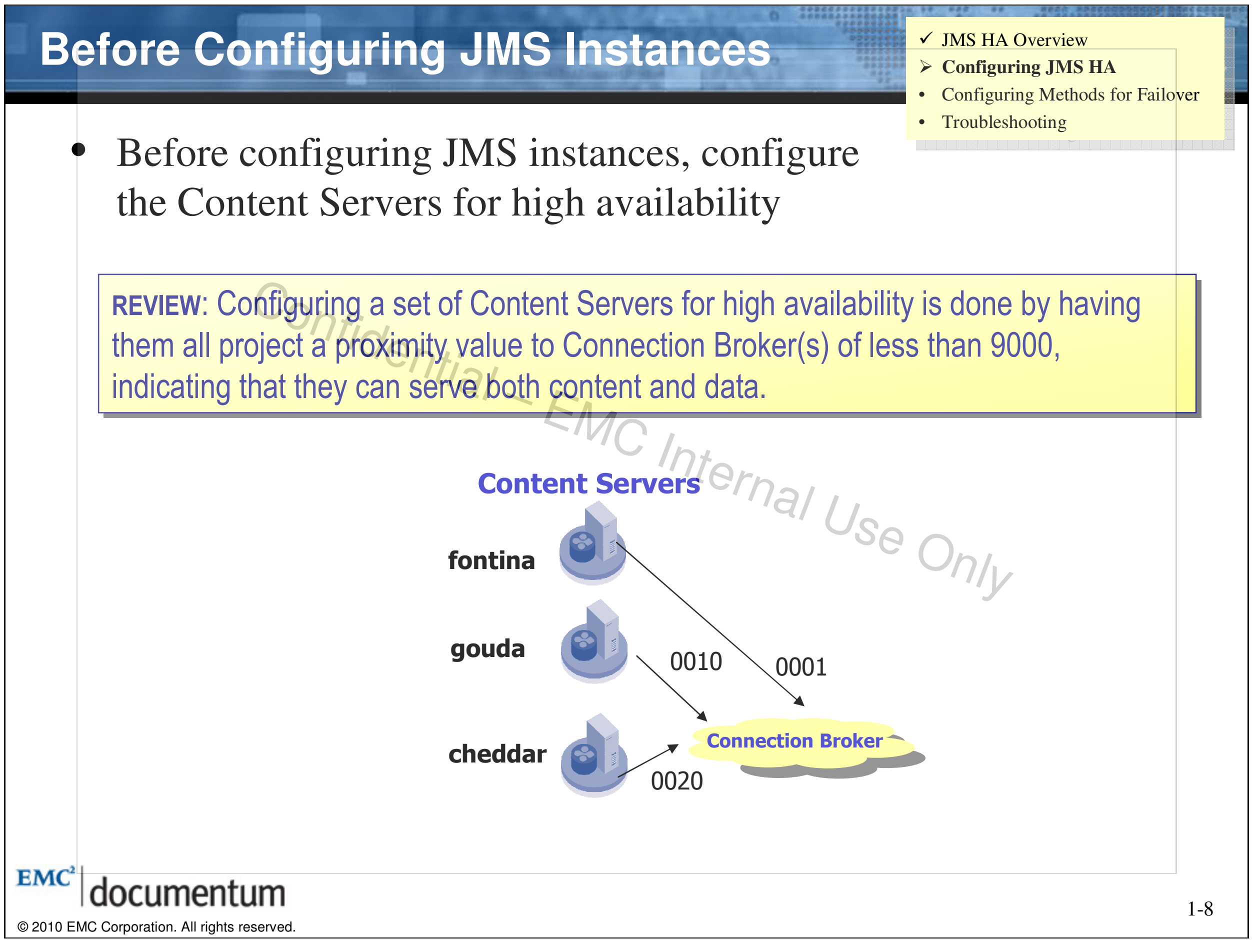


Multiple Host System

- Multiple Content Servers, spread over multiple hosts, can each have their own dedicated JMS instance
- If a JMS instance goes down, its associated Content Server fails over to the nearest available JMS instance



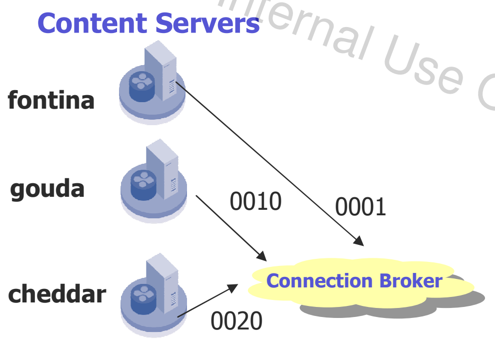
The arrows depict communication *from* the Content Server.



Before Configuring JMS Instances

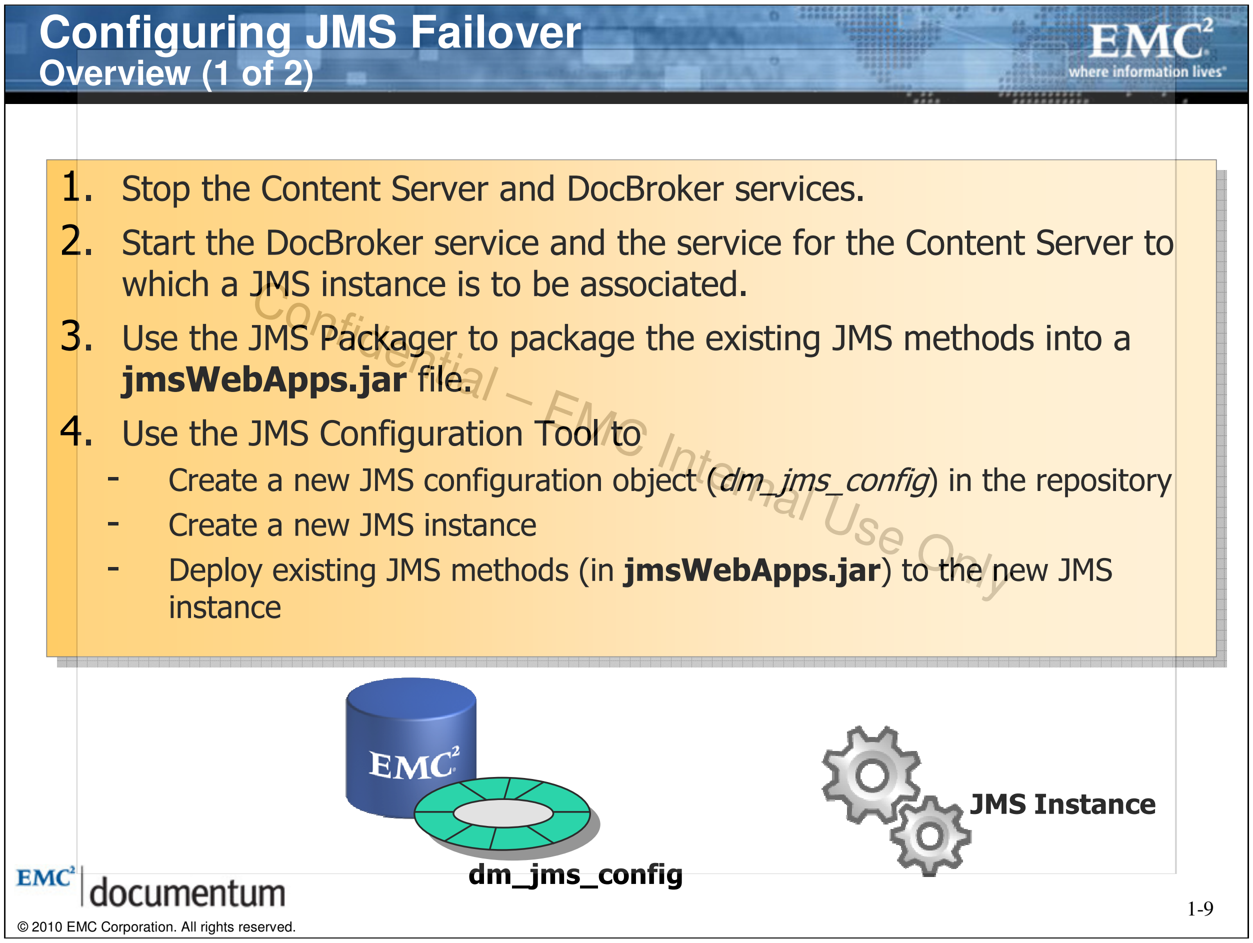
- Before configuring JMS instances, configure the Content Servers for high availability

REVIEW: Configuring a set of Content Servers for high availability is done by having them all project a proximity value to Connection Broker(s) of less than 9000, indicating that they can serve both content and data.



1-8

Recall that when a Content Server projects a proximity of less than 9000, is it implied that it is located near a database. "Near" is a relative term. The important thing is that the Content Server have good connectivity to the database and that communications between and itself and the database are quick.



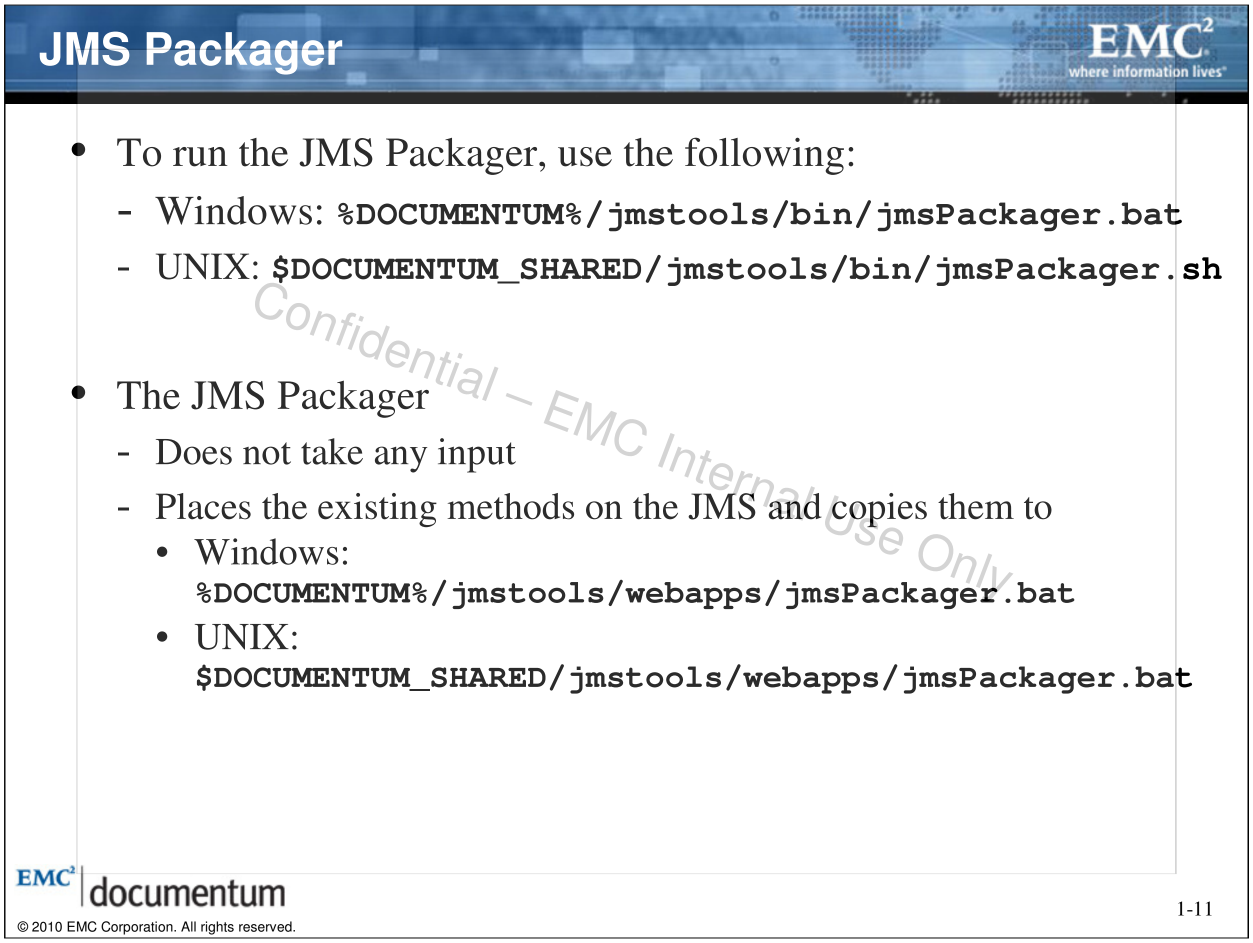
Configuring JMS Failover
Overview (1 of 2)

1. Stop the Content Server and DocBroker services.
2. Start the DocBroker service and the service for the Content Server to which a JMS instance is to be associated.
3. Use the JMS Packager to package the existing JMS methods into a **jmsWebApps.jar** file.
4. Use the JMS Configuration Tool to
   - Create a new JMS configuration object (*dm_jms_config*) in the repository
   - Create a new JMS instance
   - Deploy existing JMS methods (in **jmsWebApps.jar**) to the new JMS instance

The DocBroker service is also referred to as the Connection Broker.

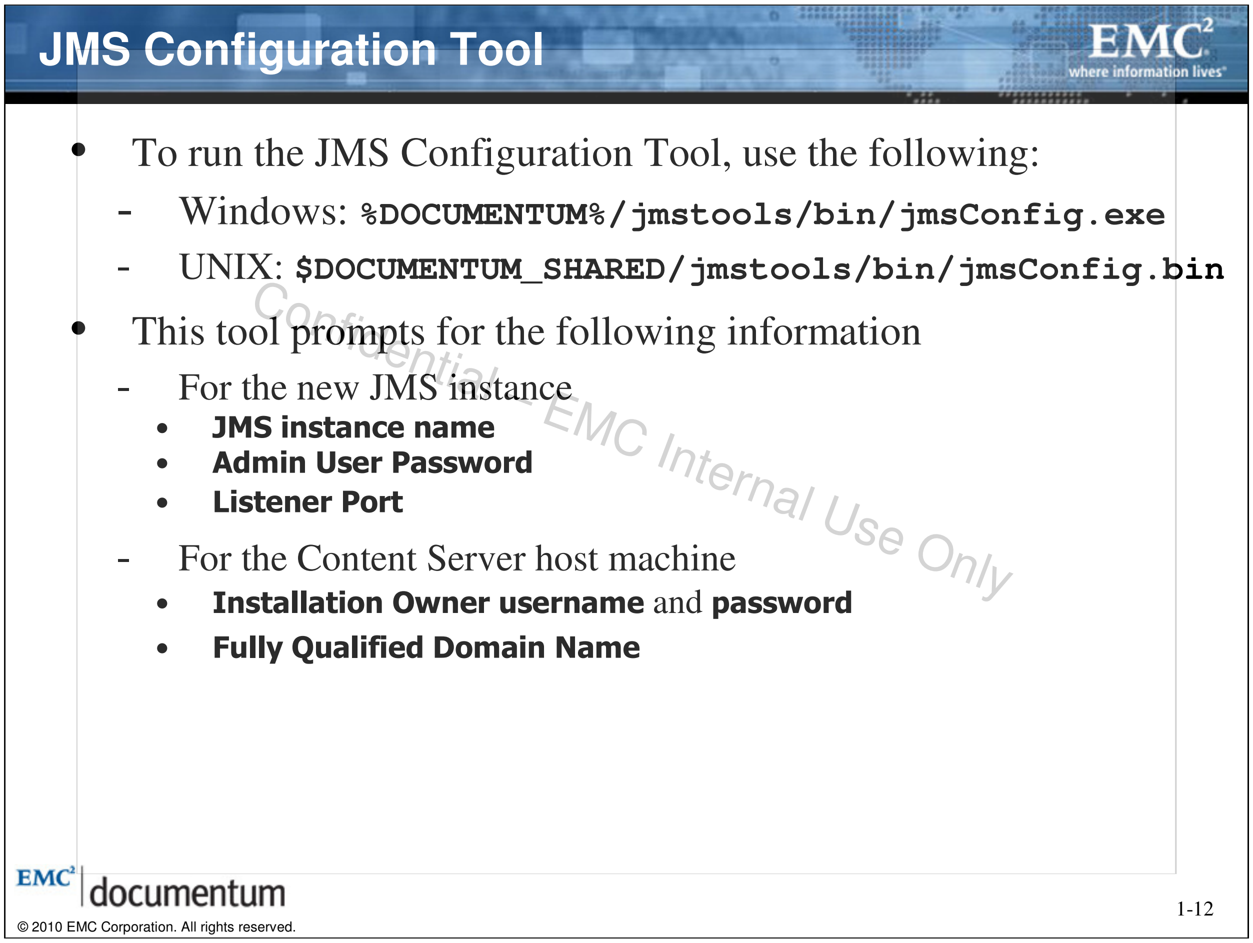
Configuring JMS Failover
Overview (2 of 2)

5. Start the services for the other Content Servers.
6. Use DQL to remove the default *dm_jms_config* object.
7. Use DA to configure the added JMS instance.



JMS Packager

- To run the JMS Packager, use the following:
  - Windows: **`%DOCUMENTUM%/jmstools/bin/jmsPackager.bat`**
  - UNIX: **`$DOCUMENTUM_SHARED/jmstools/bin/jmsPackager.sh`**
- The JMS Packager
  - Does not take any input
  - Places the existing methods on the JMS and copies them to
    - Windows: **`%DOCUMENTUM%/jmstools/webapps/jmsPackager.bat`**
    - UNIX: **`$DOCUMENTUM_SHARED/jmstools/webapps/jmsPackager.bat`**



JMS Configuration Tool

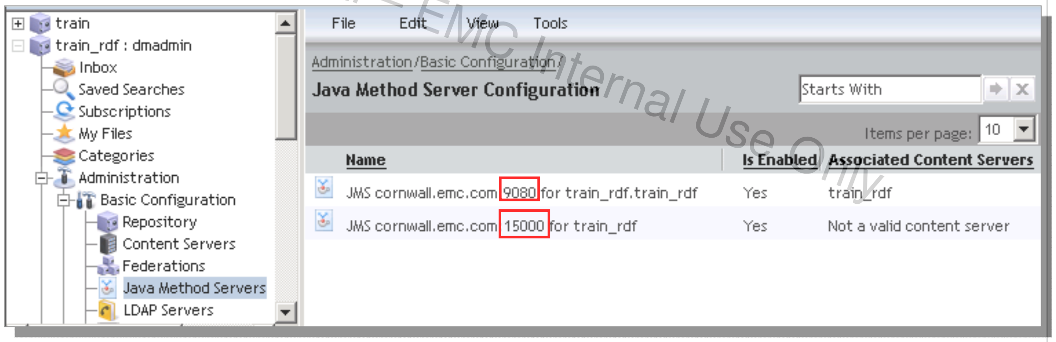
- To run the JMS Configuration Tool, use the following:
  - Windows: `%DOCUMENTUM%/jmstools/bin/jmsConfig.exe`
  - UNIX: `$DOCUMENTUM_SHARED/jmstools/bin/jmsConfig.bin`
- This tool prompts for the following information
  - For the new JMS instance
    - **JMS instance name**
    - **Admin User Password**
    - **Listener Port**
  - For the Content Server host machine
    - **Installation Owner username** and **password**
    - **Fully Qualified Domain Name**

FQDN (Fully Qualified Domain Name)

An FQDN is the complete domain name for a specific host. It has two parts: a hostname and a domain name. For example, for the FQDN cornwall.emc.com, the first part is the host name, which is cornwall. The second part specifies the domain where this host is located, emc.com.

JMS Configuration Objects
Overview (1 of 2)

- In the left frame of DA, select **Administration > Basic Configuration > Java Method Servers**
  - Note that this shows that there are *two* JMS configuration objects
  - One refers to the default JMS; the other refers to the additional JMS instance created by the JMS Configuration Tool



Administration/Basic Configuration/

Java Method Server Configuration

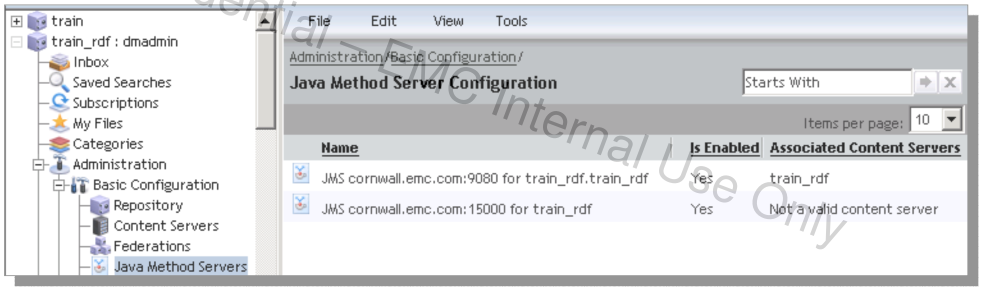
Name	Is Enabled	Associated Content Servers
JMS cornwall.emc.com 9080 for train_rdf.train_rdf	Yes	train_rdf
JMS cornwall.emc.com 15000 for train_rdf	Yes	Not a valid content server

TIP: Note that the Name includes the port number on which the JMS runs. Use this to help identify the JMS instance to which a JMS configuration object corresponds.

EMC² documentum

JMS Configuration Objects
Overview (2 of 2)

- A given Content Server should only have one JMS configuration object associated with it
- Therefore, to associate a new JMS instance, delete the configuration object pertaining to the default embedded JMS



Java Method Server Configuration

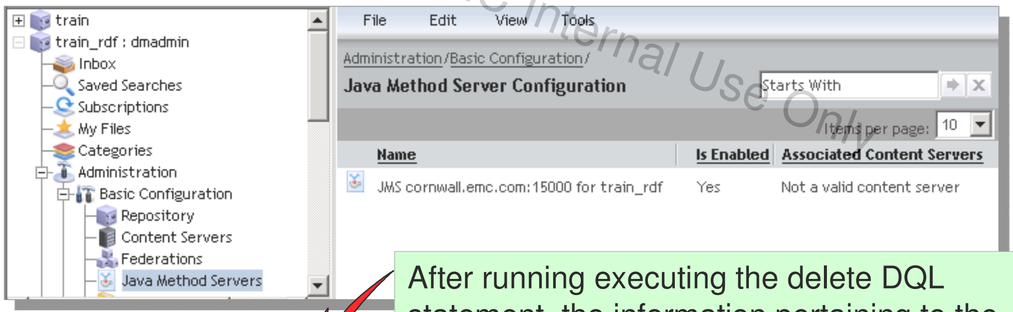
Name	Is Enabled	Associated Content Servers
JMS cornwall.emc.com:9080 for train_rdf.train_rdf	Yes	train_rdf
JMS cornwall.emc.com:15000 for train_rdf	Yes	Not a valid content server

- The JMS added by the JMS Configuration Tool must be properly configured for the Content Server
  - Note that the Associated Content Servers for this JMS presently shows as "Not a valid content server"

Deleting the Default JMS Configuration Object

- The default JMS configuration object must be deleted manually – for example, using DQL, the IAPI tool, or programmatically
- For example:

```
delete dm_jms_config object where object_name like '%9080%'
```



Name	Is Enabled	Associated Content Servers
JMS cornwall.emc.com:15000 for train_rdf	Yes	Not a valid content server

After running executing the delete DQL statement, the information pertaining to the default JMS configuration object is removed

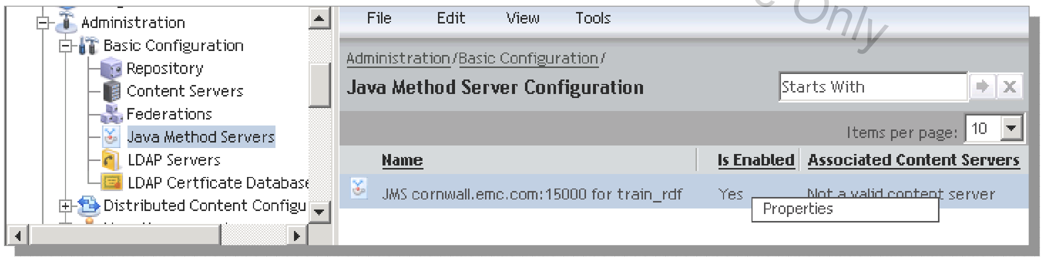
EMC² documentum

Recall that the *object_name* of the JMS configuration object (*dm_jms_config*) object includes the port number that the associated JMS instance uses for communication. This can be used to delete this *dm_jms_config* object, as shown in the example in the yellow box on the slide.

Configuring the Added JMS Instance (1 of 4)

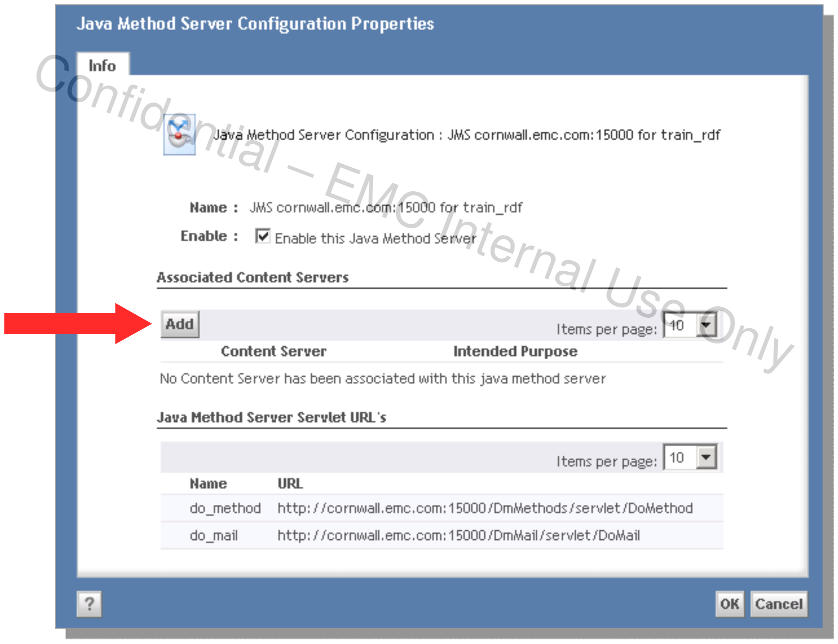
- Use DA to configure the added JMS instance
  - Associated it with a Content Server
  - Configure a proximity value

1. In the left frame of DA, select **Administration > Basic Configuration > Java Method Servers**
2. In the main frame of DA, right-click the entry pertaining to the added JMS instance



Configuring the Added JMS Instance (2 of 4)

3. Click the **Add** button to associate the added JMS instance with a Content Server.



Java Method Server Configuration Properties

Info

Java Method Server Configuration : JMS cornwall.emc.com:15000 for train_rdf

Name : JMS cornwall.emc.com:15000 for train_rdf

Enable : ☑ Enable this Java Method Server

Associated Content Servers

Content Server	Intended Purpose

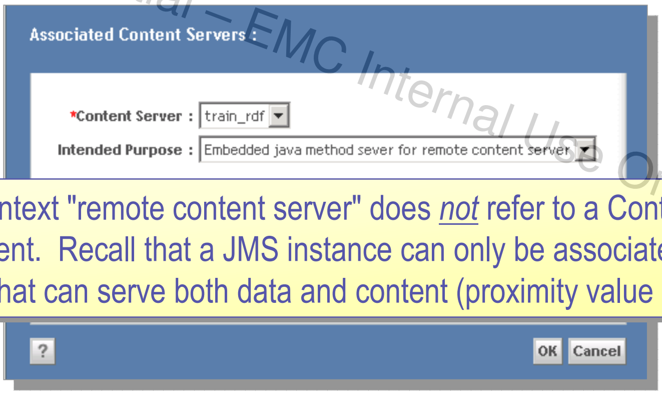
No Content Server has been associated with this java method server

Java Method Server Servlet URL's

Name	URL
do_method	http://cornwall.emc.com:15000/DmMethods/servlet/DoMethod
do_mail	http://cornwall.emc.com:15000/DmMail/servlet/DoMail

Configuring the Added JMS Instance (3 of 4)

4. Select the **Content Server** with which this JMS instance should be associated.
5. For the **Intended Purpose**, select **Embedded java method server for remote content server**.
6. Click **OK**.


Associated Content Servers :

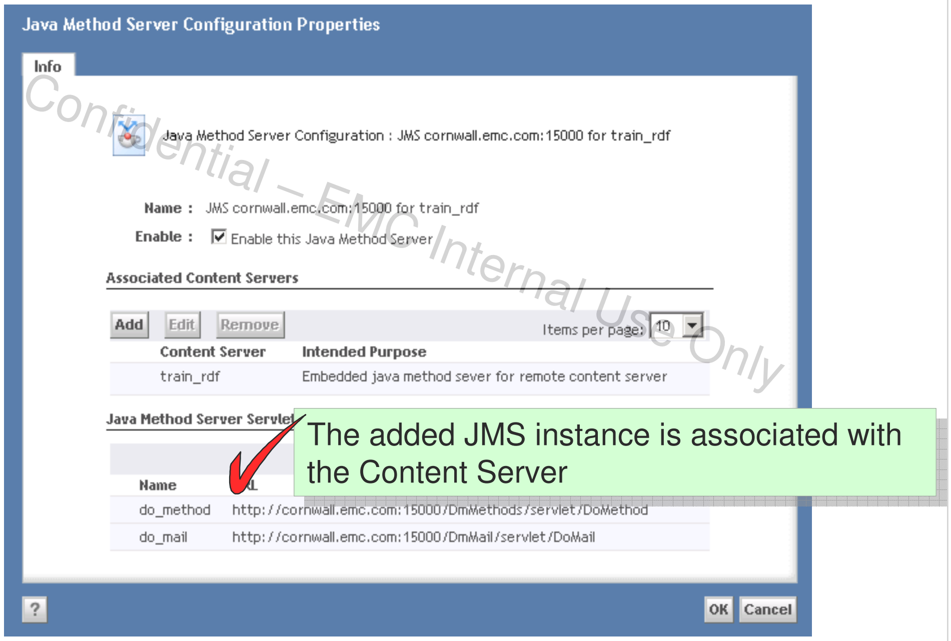
*Content Server : train_rdf

Intended Purpose : Embedded java method sever for remote content server

NOTE: In this context "remote content server" does *not* refer to a Content Server that serves only content. Recall that a JMS instance can only be associated with a Content Server that can serve both data and content (proximity value less than 9000)

Configuring the Added JMS Instance (4 of 4)

7. Click **OK**.



Java Method Server Configuration Properties

Info

Java Method Server Configuration : JMS cornwall.emc.com:15000 for train_rdf

Name : JMS cornwall.emc.com:15000 for train_rdf

Enable : ☑ Enable this Java Method Server

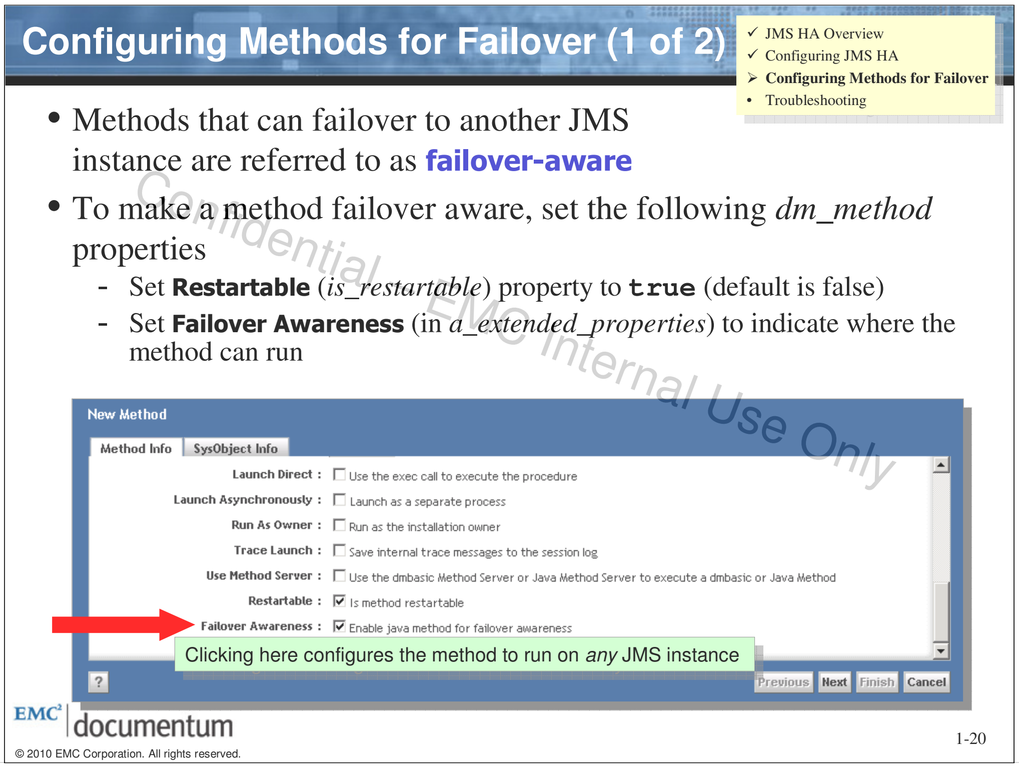
Associated Content Servers

Content Server	Intended Purpose
train_rdf	Embedded java method sever for remote content server

Java Method Server Servlet

Name	URL
do_method	http://cornwall.emc.com:15000/DmMethods/servlet/DoMethod
do_mail	http://cornwall.emc.com:15000/DmMail/servlet/DoMail

The added JMS instance is associated with the Content Server



Configuring Methods for Failover (1 of 2)

- Methods that can failover to another JMS instance are referred to as **failover-aware**
- To make a method failover aware, set the following *dm_method* properties
  - Set **Restartable** (*is_restartable*) property to **`true`** (default is false)
  - Set **Failover Awareness** (in *a_extended_properties*) to indicate where the method can run

New Method

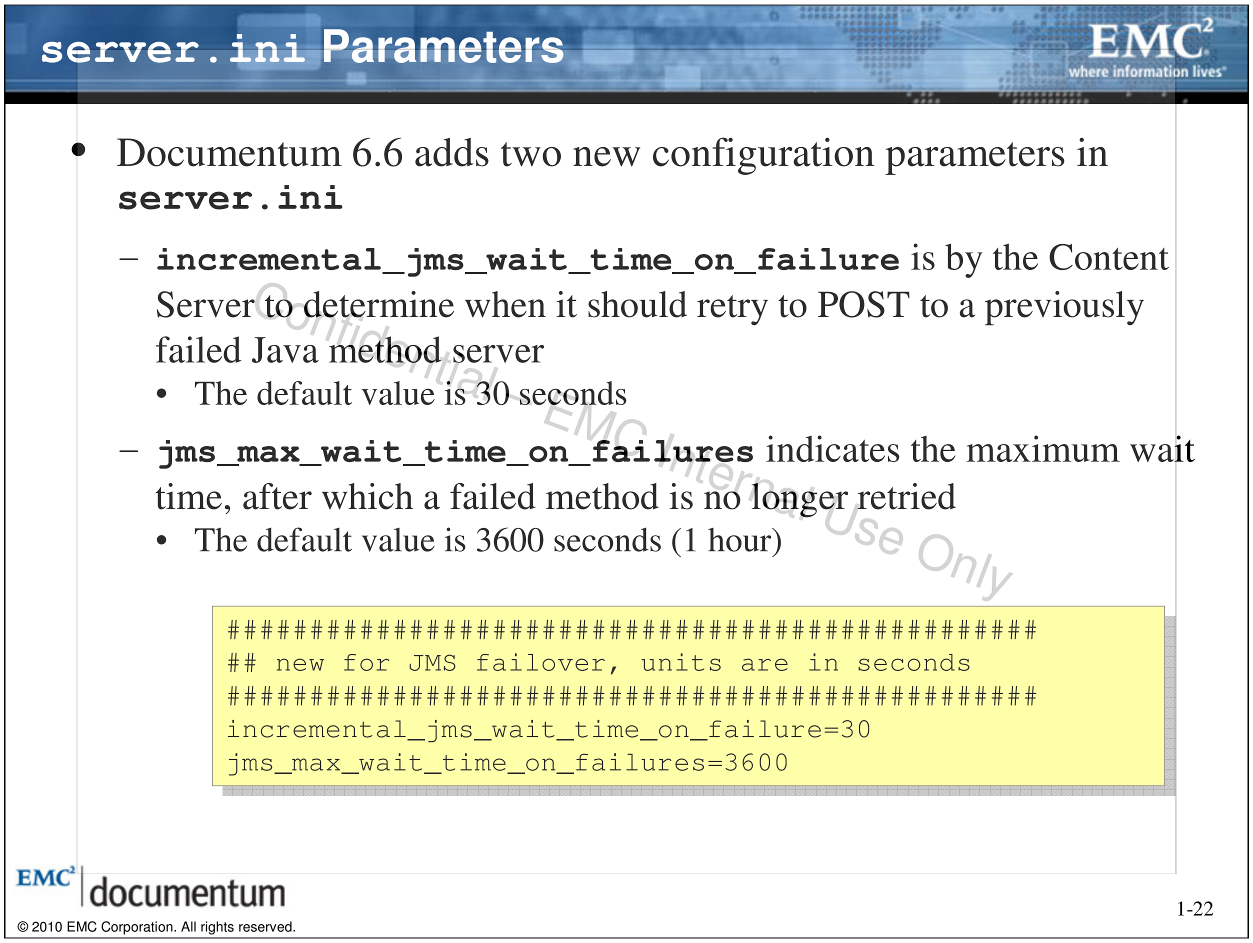
Launch Direct :	☐ Use the exec call to execute the procedure
Launch Asynchronously :	☐ Launch as a separate process
Run As Owner :	☐ Run as the installation owner
Trace Launch :	☐ Save internal trace messages to the session log
Use Method Server :	☐ Use the dmbasic Method Server or Java Method Server to execute a dmbasic or Java Method
Restartable :	☑ Is method restartable
Failover Awareness :	☑ Enable java method for failover awareness

Clicking here configures the method to run on *any* JMS instance

EMC² documentum

Configuring Methods for Failover (2 of 2)

- *a_extended_properties*
  - Use this property to set the location (**`JMS_LOCATION`**) where it is valid to execute this method
  - If the method must be run on the same host as originating Content Server, then set *a_extended_properties*[0] to **`JMS_LOCATION=ORIGINAL`**
  - If method can be run on any JMS instance (default), then
    - Set *a_extended_properties*[0] to **`JMS_LOCATION=ANY`** ***or***
    - Leave *a_extended_properties*[0] at its default value (blank)



`server.ini` Parameters

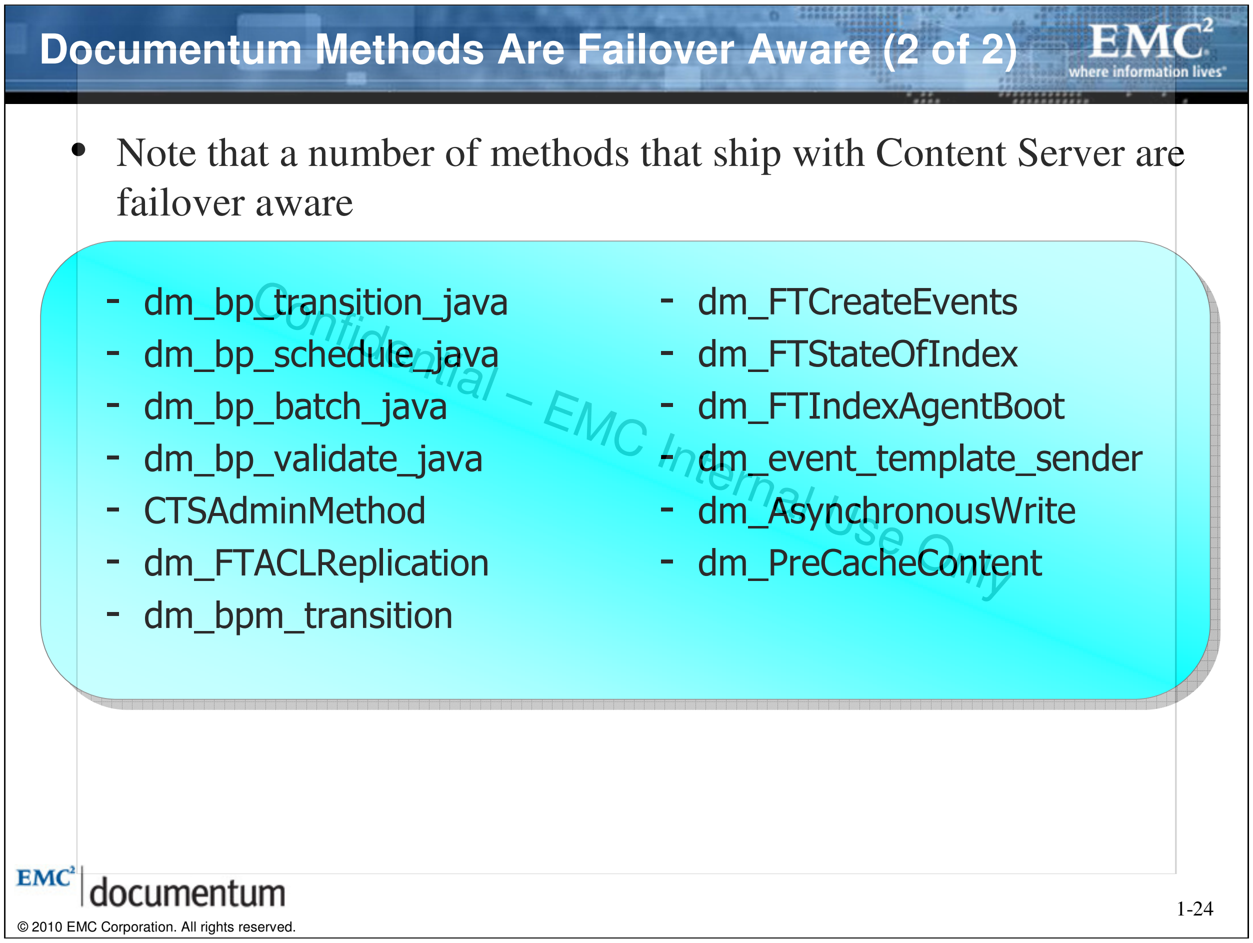
- Documentum 6.6 adds two new configuration parameters in **`server.ini`**
  - **`incremental_jms_wait_time_on_failure`** is by the Content Server to determine when it should retry to POST to a previously failed Java method server
    - The default value is 30 seconds
  - **`jms_max_wait_time_on_failures`** indicates the maximum wait time, after which a failed method is no longer retried
    - The default value is 3600 seconds (1 hour)

```
#################################################
## new for JMS failover, units are in seconds
#################################################
incremental_jms_wait_time_on_failure=30
jms_max_wait_time_on_failures=3600
```

Documentum Methods Are Failover Aware (1 of 2)

- The following DQL query shows the methods that are failover aware

```
select r_object_id, object_name, a_extended_properties from dm_method
where any a_extended_properties like '%JMS_LOCATION%'
```



Documentum Methods Are Failover Aware (2 of 2)

- Note that a number of methods that ship with Content Server are failover aware
  - dm_bp_transition_java
  - dm_bp_schedule_java
  - dm_bp_batch_java
  - dm_bp_validate_java
  - CTSAdminMethod
  - dm_FTACLReplication
  - dm_bpm_transition
  - dm_FTCreateEvents
  - dm_FTStateOfIndex
  - dm_FTIndexAgentBoot
  - dm_event_template_sender
  - dm_AsynchronousWrite
  - dm_PreCacheContent

General Troubleshooting Tips (1 of 3)

- Using DA, make sure that
  - No more than one JMS is associated to a Content Server
  - The **Intended Purpose** is set to **Default embedded java method server**
- Turn on debug tracing
  - In the JMS instance's **`web.xml`** file, set the value of the **`trace`** parameter to **`t`** and then re-start the JMS instance

```
<init-param>
      <param-name>trace</param-name>
      <param-value>t</param-value>
</init-param>
```

  - This file can be found in the following location
    `DOCUMENTUM\jboss4.3.0\server\DctmServer_MethodServer\depl oy\ServerApps.ear\DmMethods.war\WEB-INF`

EMC² documentum

General Troubleshooting Tips (2 of 3)

- Turn on trace launch
  - This can be done at the Content Server level – for example, in DA, using the **`SET_OPTIONS`** administration method
  - It can also be done
    - By setting *trace_launch* property on an individual *dm_method* object to **`T`**

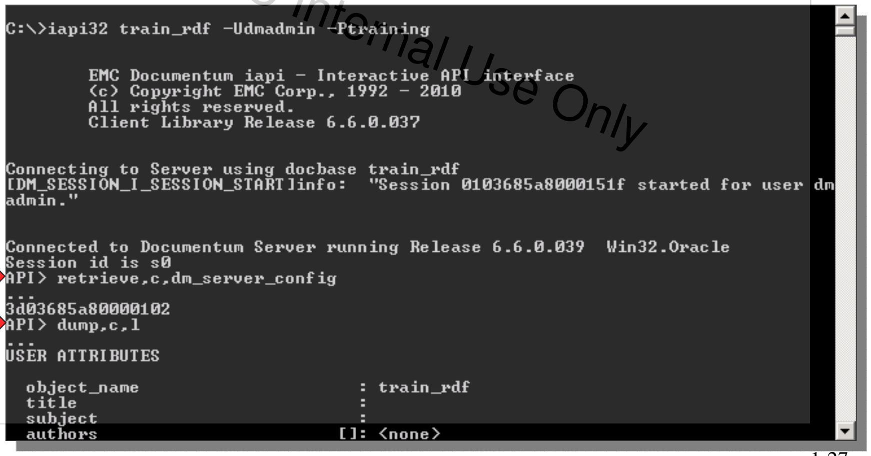
    or
    - When launching the method
- Output from the trace launch can be found in **`DOCUMENTUM\dba\log\`*`repository_id`*`\MethodServer\MethodServer\`*`server_config_name`*`.log`**
  - Check for information about why the method or methods did not execute properly

General Troubleshooting Tips (3 of 3)

- Dump the following repository objects:
  - *dm_docbase_config*
  - *dm_server config* objects
  - All *dm_jms_config* objects
  - The failed *dm_method* object

TIP: To dump repository objects, at the Windows or UNIX command line, use iapi32

First, **`retrieve`** the object's *r_object_id*, and then **`dump`** its value



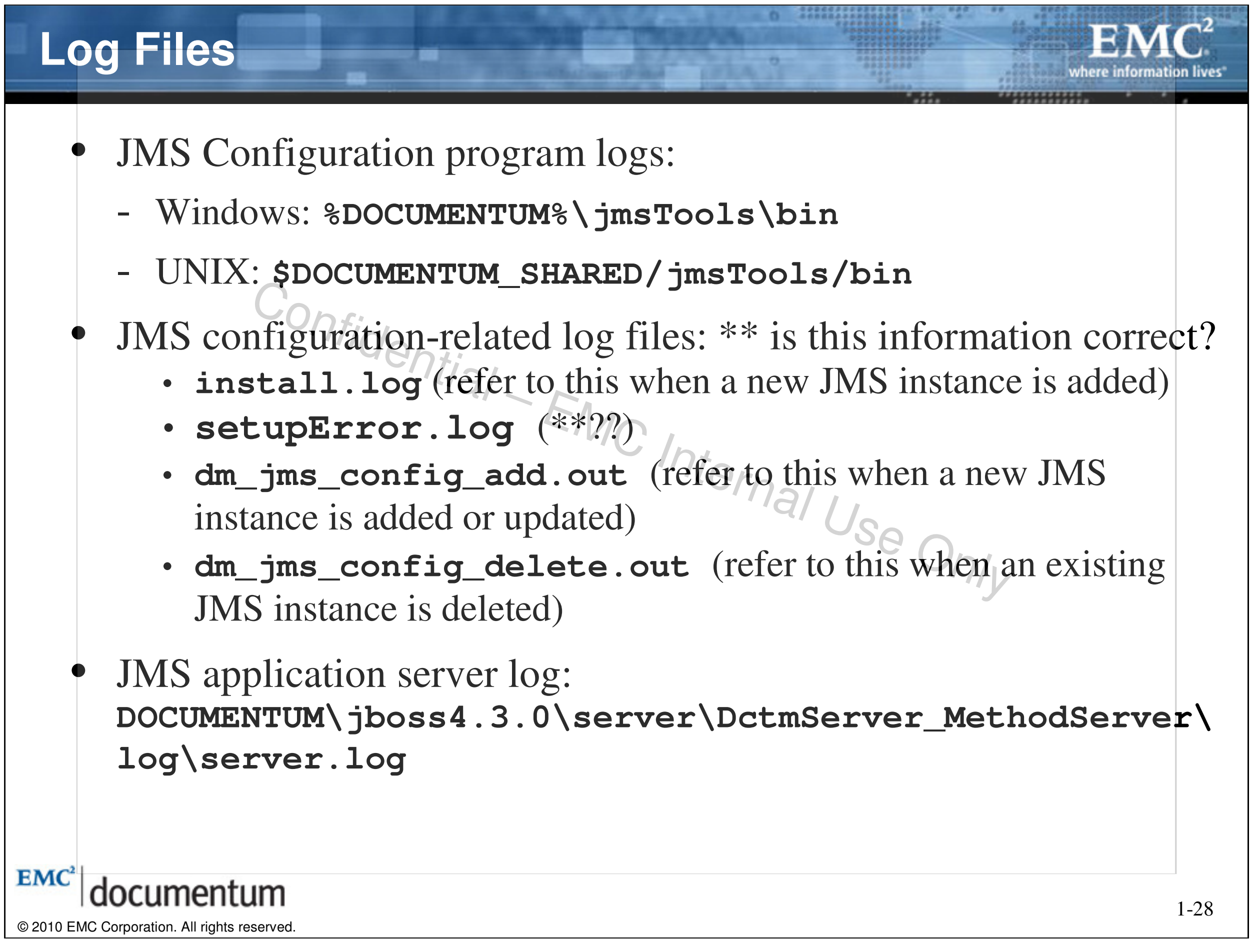
```
C:\>iapi32 train_rdf -Udmadmin -Ptraining


        EMC Documentum iapi - Interactive API interface
        (c) Copyright EMC Corp., 1992 - 2010
        All rights reserved.
        Client Library Release 6.6.0.037


Connecting to Server using docbase train_rdf
[DM_SESSION_I_SESSION_START]info:  "Session 0103685a8000151f started for user dm
admin."


Connected to Documentum Server running Release 6.6.0.039  Win32.Oracle
Session id is s0
API> retrieve,c,dm_server_config
...
3d03685a80000102
API> dump,c,l
...
USER ATTRIBUTES

  object_name                     : train_rdf
  title                           :
  subject                         :
  authors                       []: <none>
```

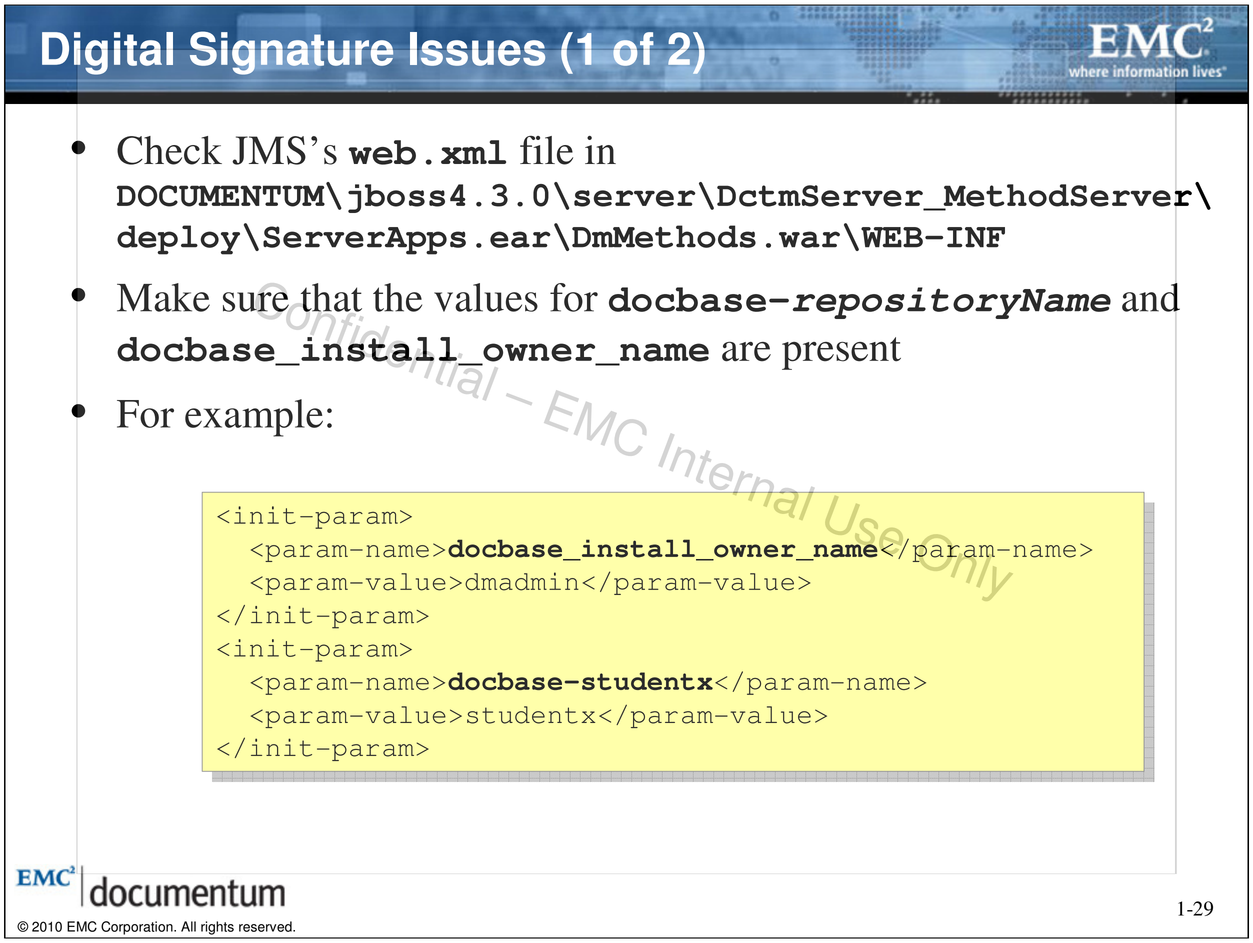


Log Files

- JMS Configuration program logs:
  - Windows: **`%DOCUMENTUM%\jmsTools\bin`**
  - UNIX: **`$DOCUMENTUM_SHARED/jmsTools/bin`**
- JMS configuration-related log files: ** is this information correct?
  - **`install.log`** (refer to this when a new JMS instance is added)
  - **`setupError.log`** (**??)
  - **`dm_jms_config_add.out`** (refer to this when a new JMS instance is added or updated)
  - **`dm_jms_config_delete.out`** (refer to this when an existing JMS instance is deleted)
- JMS application server log:
  `DOCUMENTUM\jboss4.3.0\server\DctmServer_MethodServer\log\server.log`

EMC² documentum

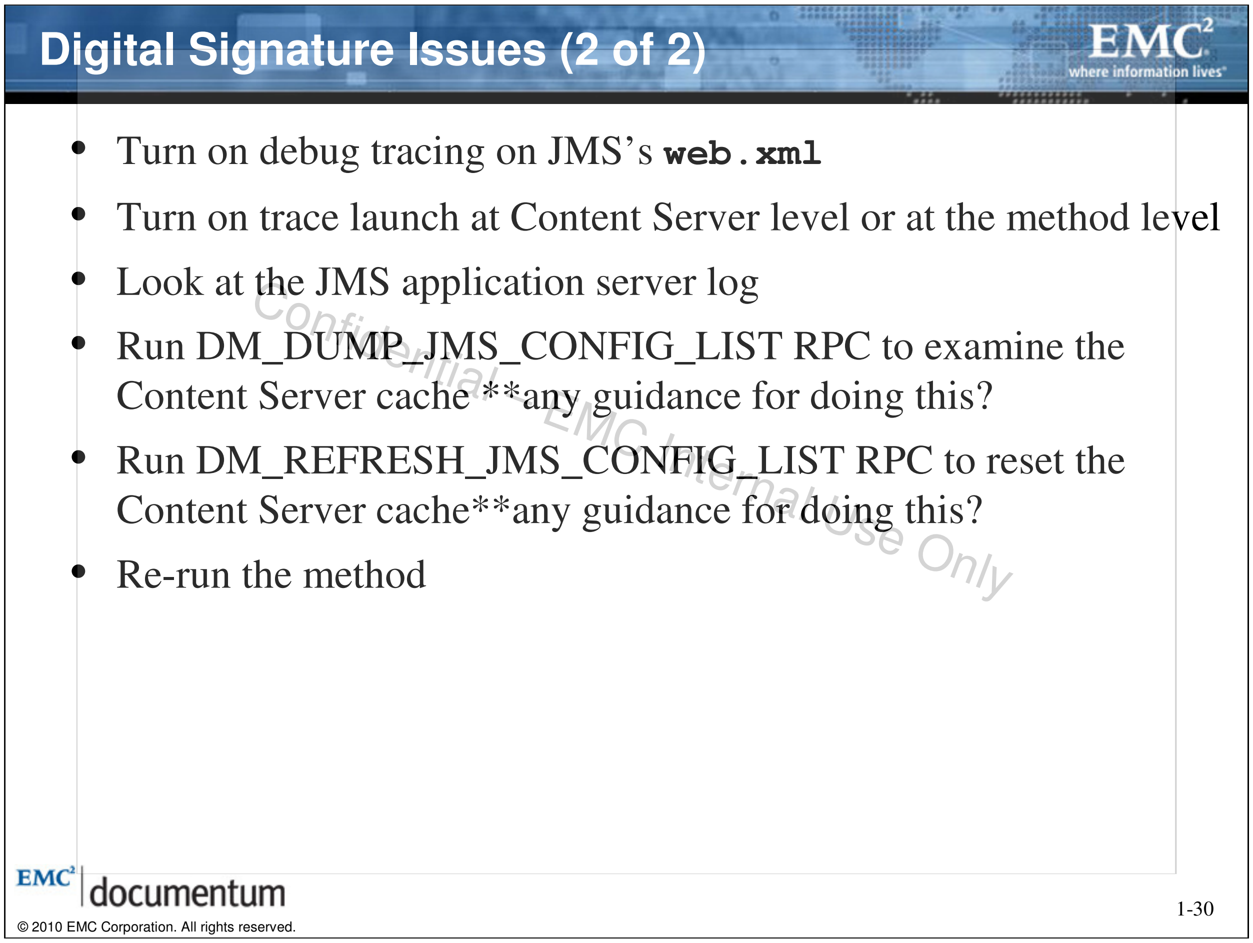
1-28



Digital Signature Issues (1 of 2)

- Check JMS's **`web.xml`** file in **`DOCUMENTUM\jboss4.3.0\server\DctmServer_MethodServer\deploy\ServerApps.ear\DmMethods.war\WEB-INF`**
- Make sure that the values for **`docbase-`*`repositoryName`*** and **`docbase_install_owner_name`** are present
- For example:

```
<init-param>
  <param-name>docbase_install_owner_name</param-name>
  <param-value>dmadmin</param-value>
</init-param>
<init-param>
  <param-name>docbase-studentx</param-name>
  <param-value>studentx</param-value>
</init-param>
```



Digital Signature Issues (2 of 2)

- Turn on debug tracing on JMS's **`web.xml`**
- Turn on trace launch at Content Server level or at the method level
- Look at the JMS application server log
- Run DM_DUMP_JMS_CONFIG_LIST RPC to examine the Content Server cache **any guidance for doing this?
- Run DM_REFRESH_JMS_CONFIG_LIST RPC to reset the Content Server cache**any guidance for doing this?
- Re-run the method

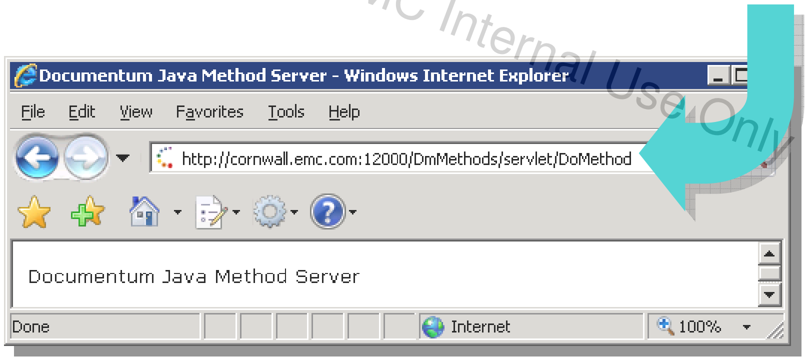
JMS Failover Issues (1 of 3)

- Check the Content Server log to see if there are any errors
- Get dumps of the *dm_jms_config* objects and make sure that:
  - A value is defined for the *do_method base_uri* in each of *dm_jms_config* object
  - The *base_uri* values have correct host, port and path
  - The *projection_proximity_value* is set to 1 (use DA to assign correct **Intended Purpose**, if needed)
  - The *server_config_id* matches the *r_object_id* in the corresponding *dm_server_config* object
  - There is exactly one *server_config_id* associated to one *dm_jms_config* object
  - The *server_config_id* is not NULL (for example, 0000000000000000)

JMS Failover Issues (2 of 3)

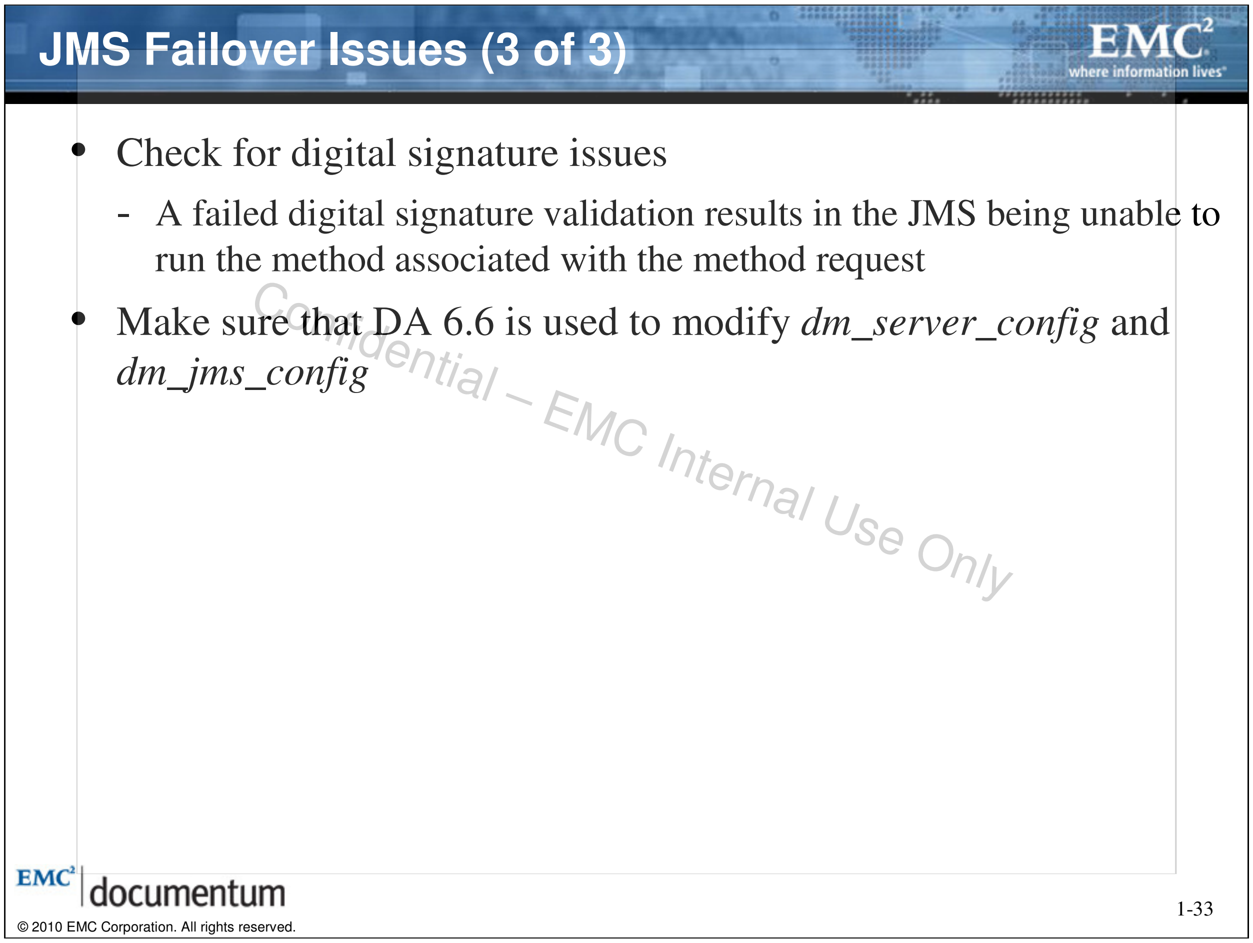
- Using a Web browser, access the URL values defined in the *dm_jms_config* object for the failover JMS
  - Make sure the failover JMS is indeed up and running at the defined host, port and path
  - For example:

```
base_uri                      [0]: http://cornwall.emc.com:15000/DmMethods/servlet/DoMethod
```


Documentum Java Method Server - Windows Internet Explorer
File Edit View Favorites Tools Help
Done
Internet
100%

http://cornwall.emc.com:12000/DmMethods/servlet/DoMethod

Documentum Java Method Server

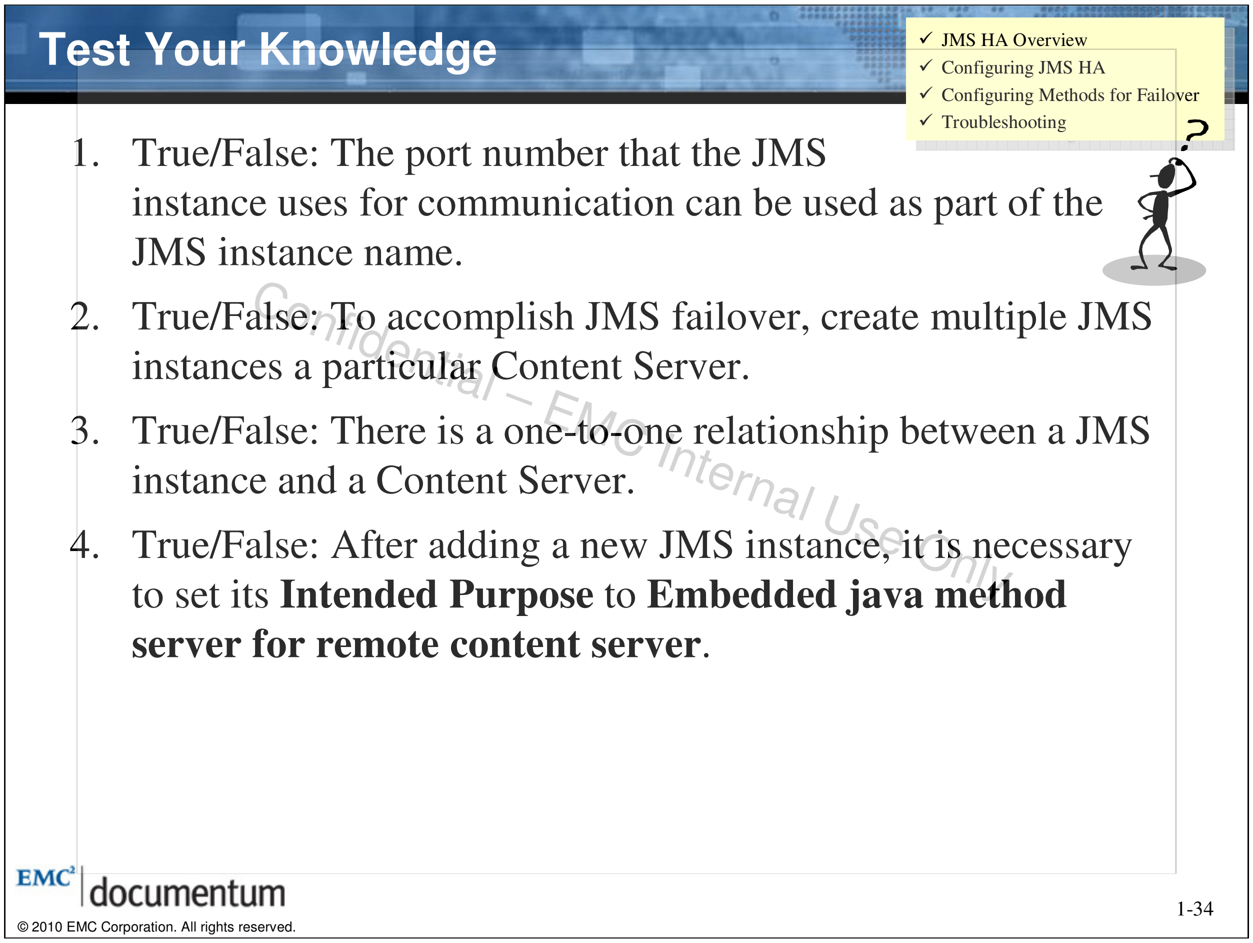


JMS Failover Issues (3 of 3)

- Check for digital signature issues
  - A failed digital signature validation results in the JMS being unable to run the method associated with the method request
- Make sure that DA 6.6 is used to modify *dm_server_config* and *dm_jms_config*

Recall that the *dm_jms_config.server_config_id* is equal to the *dm_server_config.r_object_id*.

When the *dm_server_config* object is edited in Documentum Administrator (DA) versions prior to 6.6, the object is checked-out/checked-in. This results in a new version of the *dm_server_config* object being generated. The new version has a different *r_object_id* than the previous version. That breaks the link that the *dm_server_config* object has with the *dm_jms_config* object.

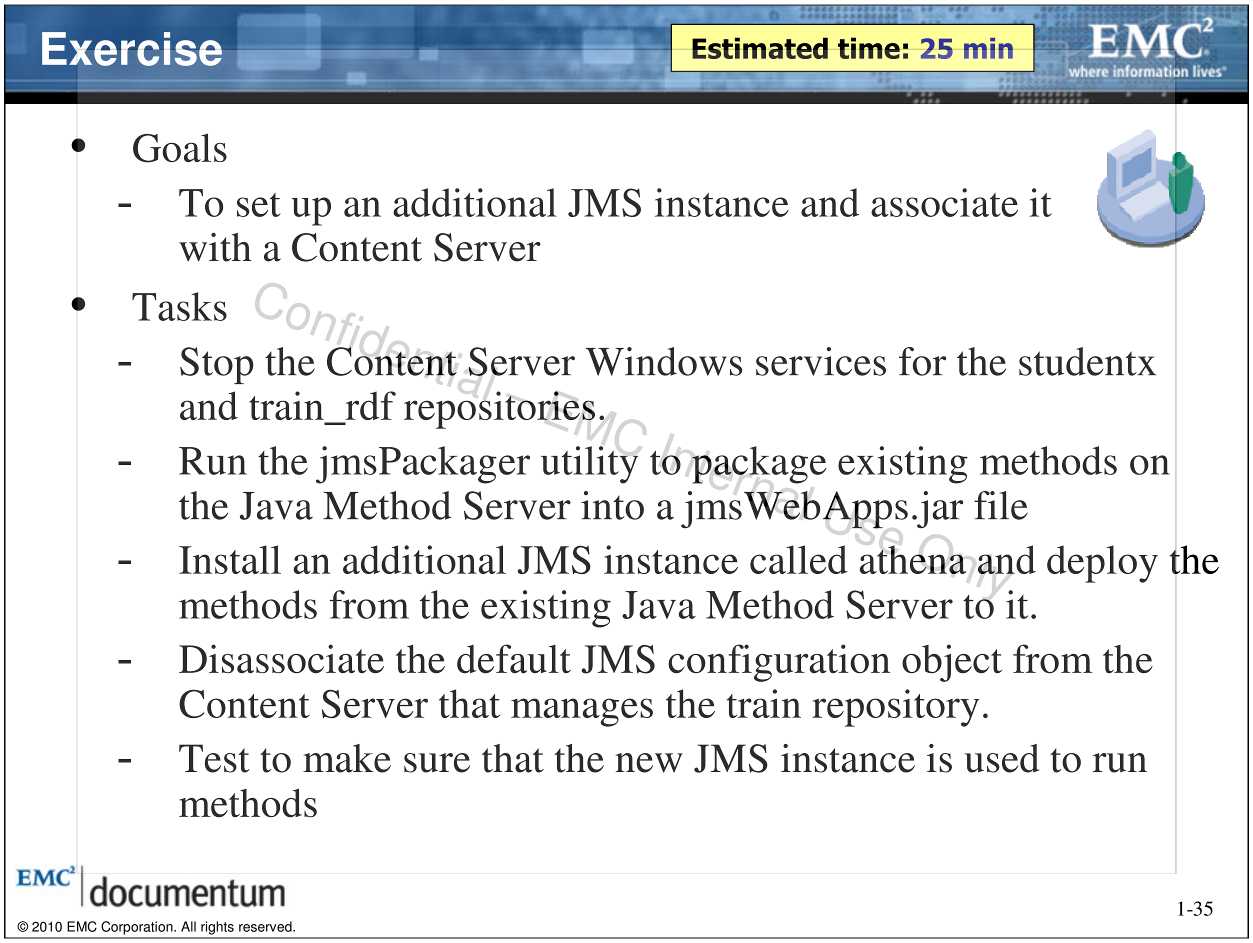


Test Your Knowledge

✓ JMS HA Overview
✓ Configuring JMS HA
✓ Configuring Methods for Failover
✓ Troubleshooting

1. True/False: The port number that the JMS instance uses for communication can be used as part of the JMS instance name.
2. True/False: To accomplish JMS failover, create multiple JMS instances a particular Content Server.
3. True/False: There is a one-to-one relationship between a JMS instance and a Content Server.
4. True/False: After adding a new JMS instance, it is necessary to set its **Intended Purpose** to **Embedded java method server for remote content server**.

1. True. The port number is part of the JMS instance name.
2. False. JMS failover is not supported.
3. True. Only one JMS instance is supported per Content Server.
4. True. This sets the proximity from the Content Server to the JMS instance equal to 1, which is required in Documentum 6.6.



Exercise

Estimated time: 25 min

EMC² where information lives

- Goals
  - To set up an additional JMS instance and associate it with a Content Server
- Tasks
  - Stop the Content Server Windows services for the studentx and train_rdf repositories.
  - Run the jmsPackager utility to package existing methods on the Java Method Server into a jmsWebApps.jar file
  - Install an additional JMS instance called athena and deploy the methods from the existing Java Method Server to it.
  - Disassociate the default JMS configuration object from the Content Server that manages the train repository.
  - Test to make sure that the new JMS instance is used to run methods

EMC² documentum