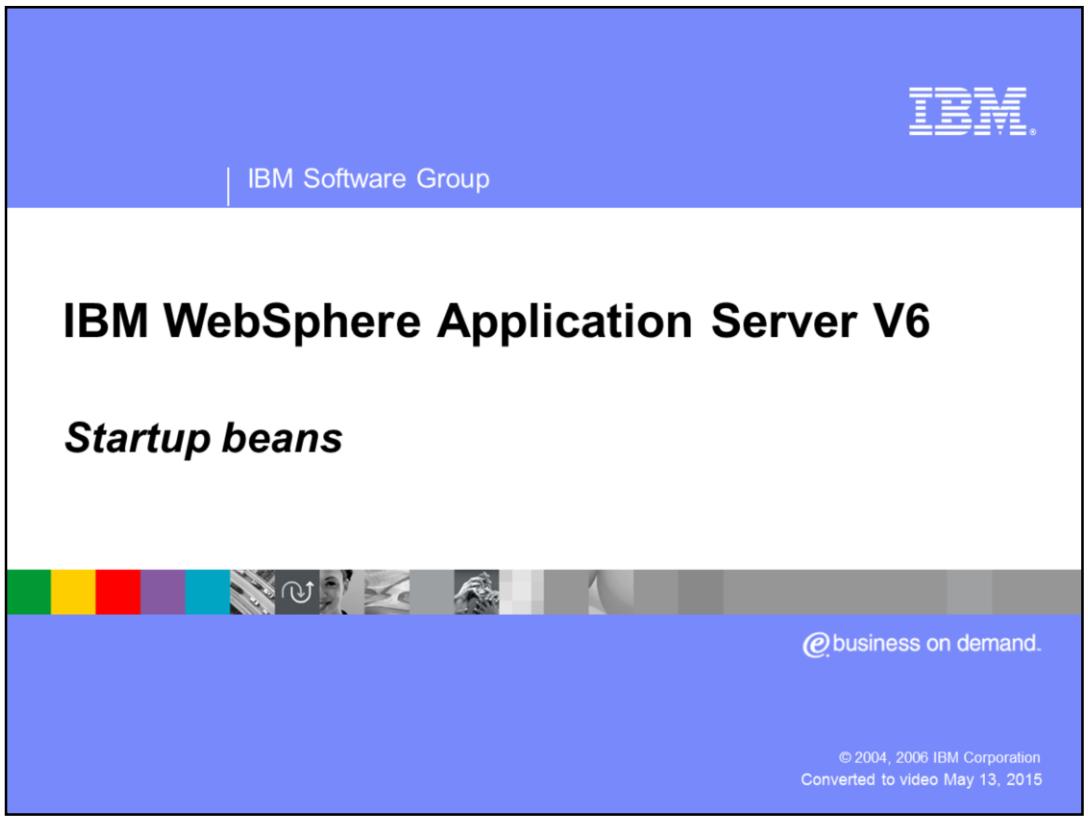
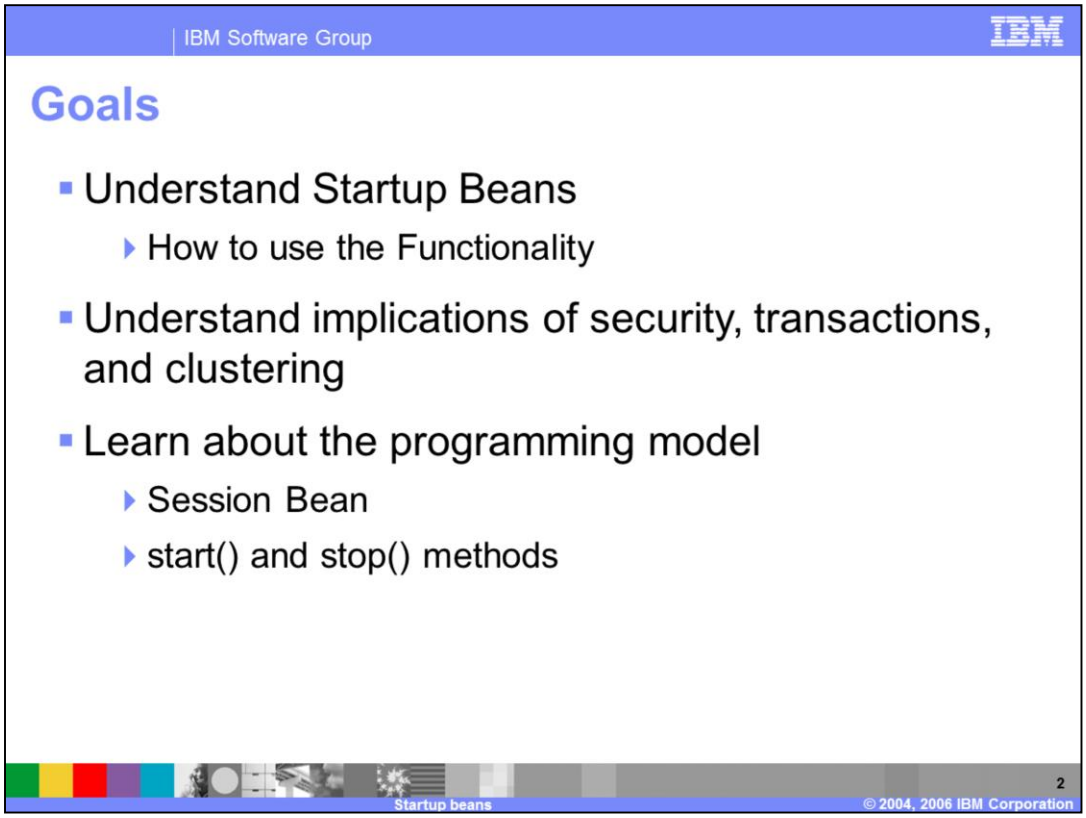
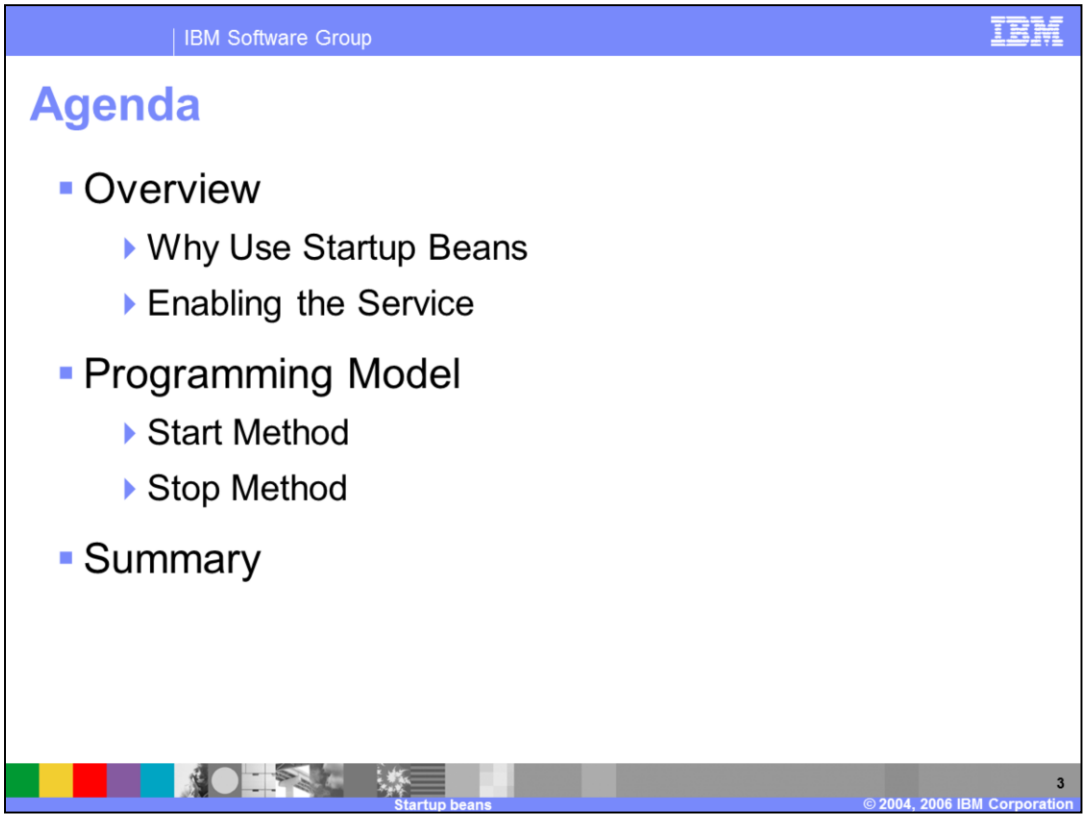


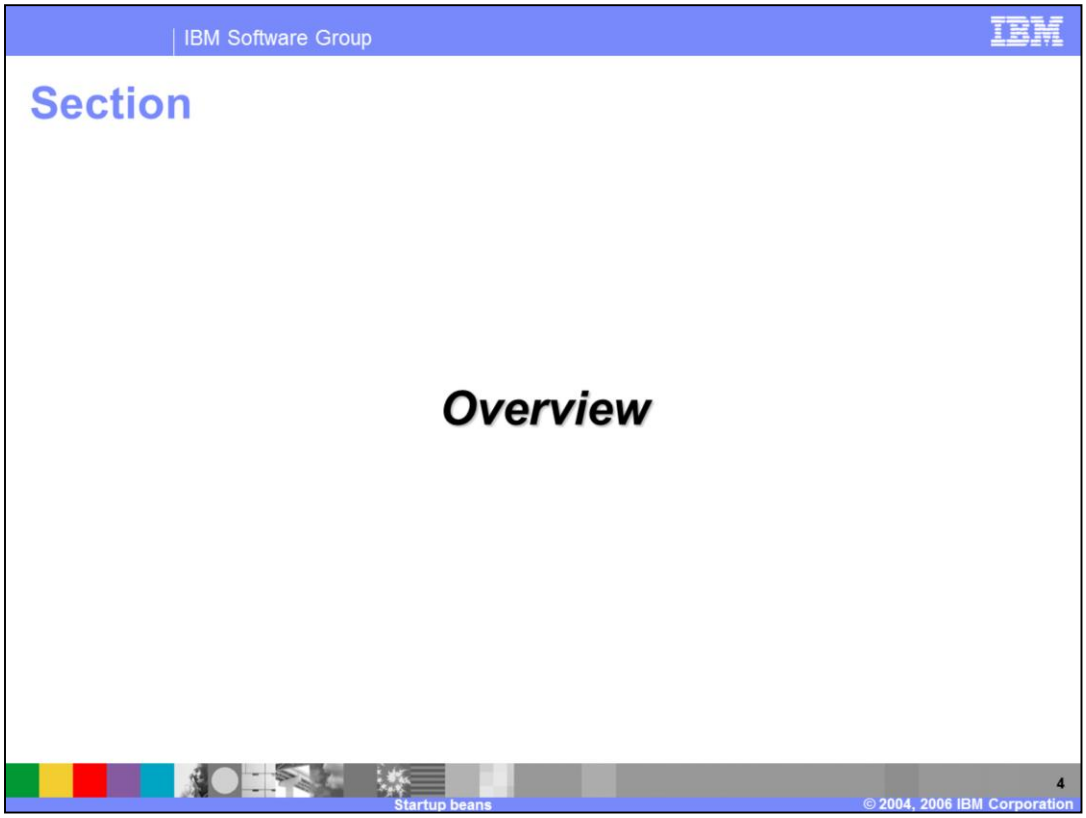
This presentation will discuss the Startup Beans programming model extension available in WebSphere® Application Server V6.



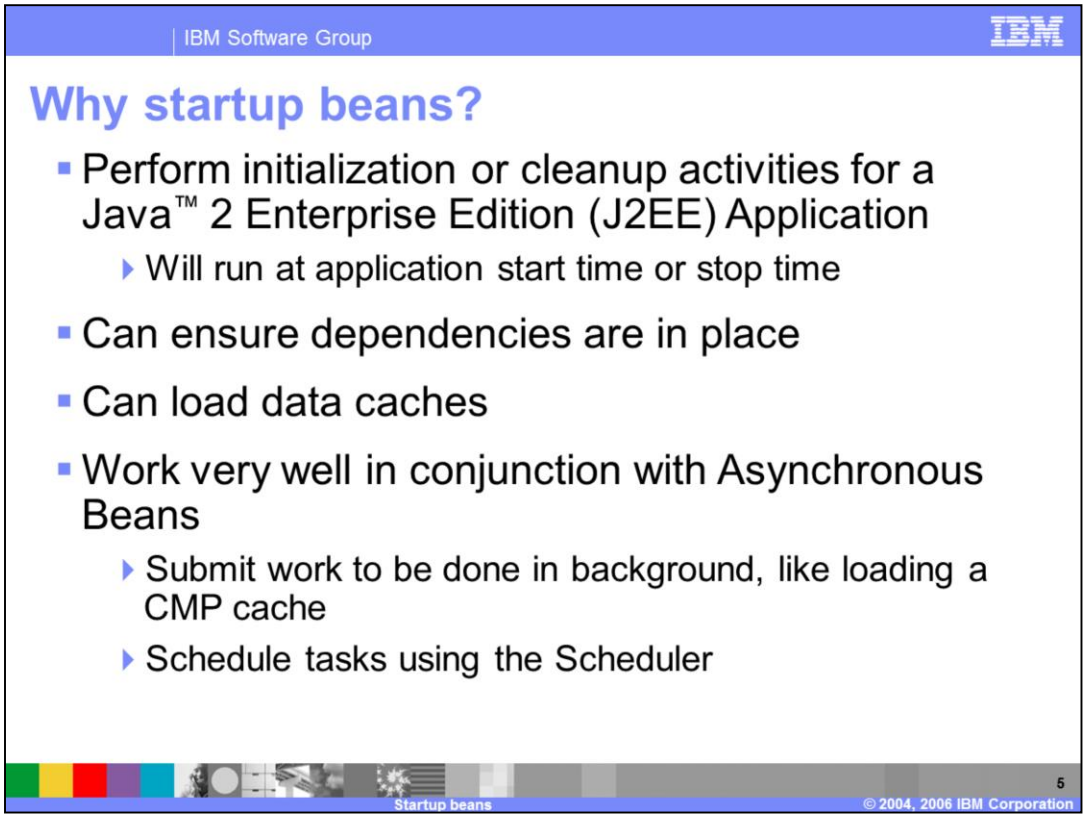
The goals for this presentation are to understand Startup Beans functionality; understand the implications of security, transaction support, and clustering of Startup beans; and learn about the programming model.



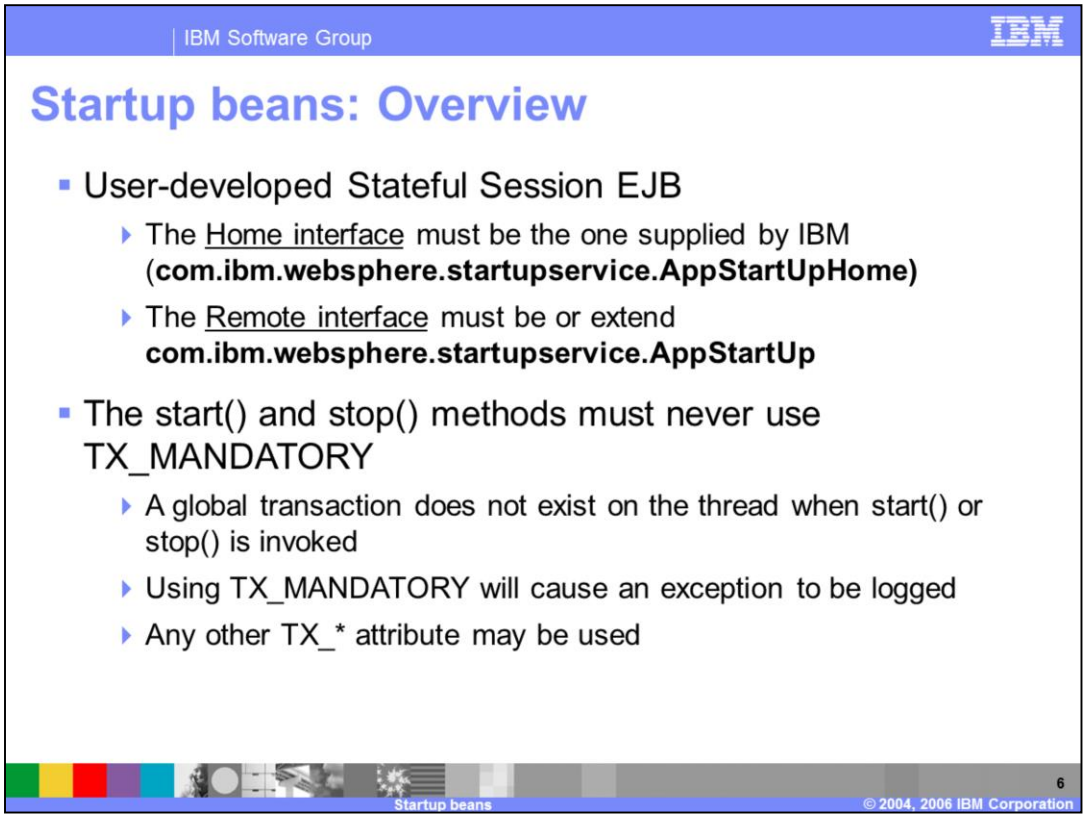
The agenda for this presentation includes an overview of the Startup beans functionality and how to enable the service, the programming model, and a summary.



The next section will discuss an overview of the Startup Beans functionality.

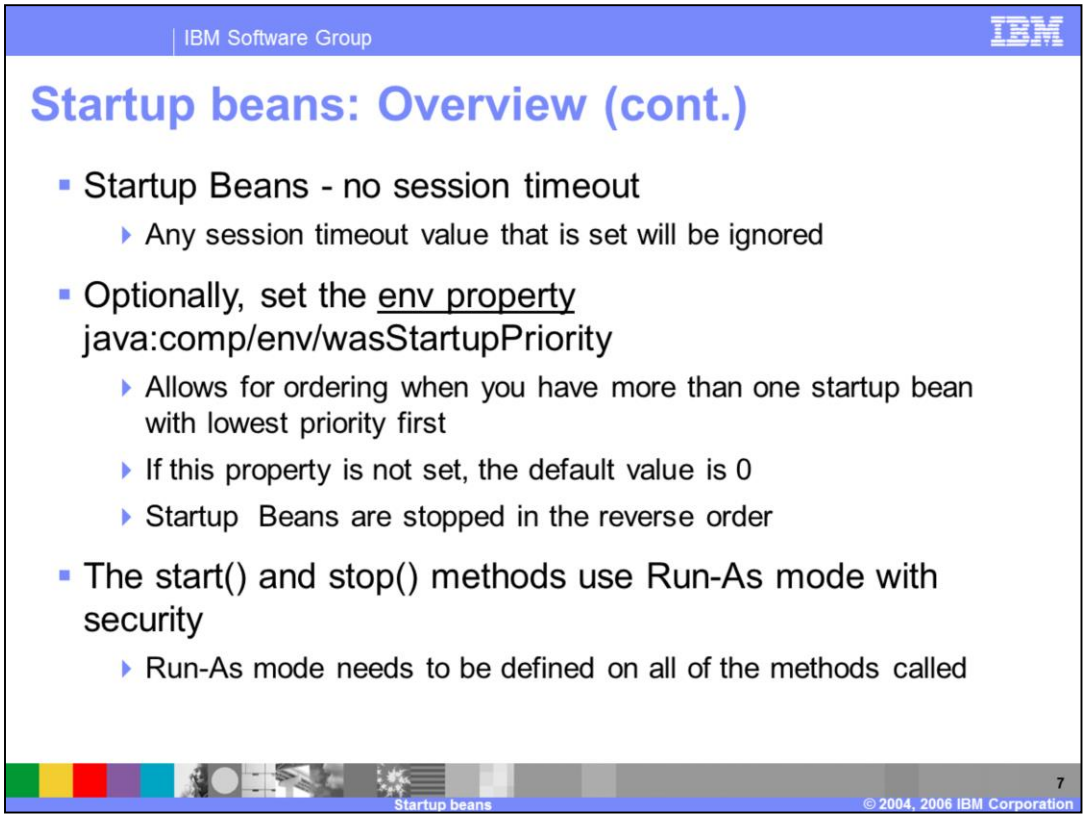


Startup Beans allow you the capability to run specific business logic at the time of application start time or stop time. Usage examples of Startup Beans include checking that application dependencies are in place and enforced, loading of a data cache, initiating work using Asynchronous Beans, or scheduling tasks with the Scheduler function.



Startup beans are user-defined, session Enterprise JavaBeans that use the IBM-supplied home interface of com.ibm.websphere.startupservice.AppStartUpHome and the IBM-supplied remote interface of com.ibm.websphere.startupservice.AppStartUp. Using these interfaces will cause the container to recognize the EJBs as startup beans.

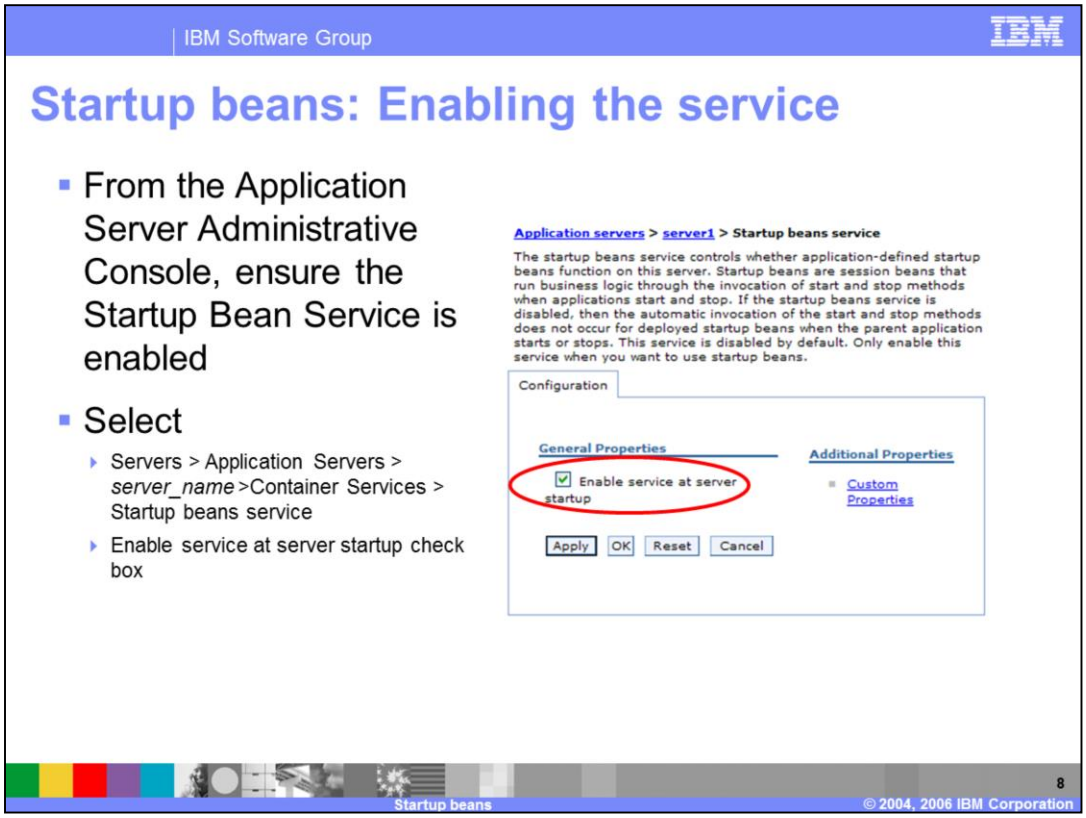
The remote interface exposes two methods that the container will invoke. The start() method will be invoked as the enterprise application containing the startup bean starts. The stop() method will be invoked as the enterprise application containing the startup bean stops. These two methods must not use the TX_MANDATORY transaction attribute, since a global transaction does not exist on the thread when either method is invoked. Any other transaction attribute may be used. The startup beans can initiate their own transactions or call other EJBs that start their own transactions.



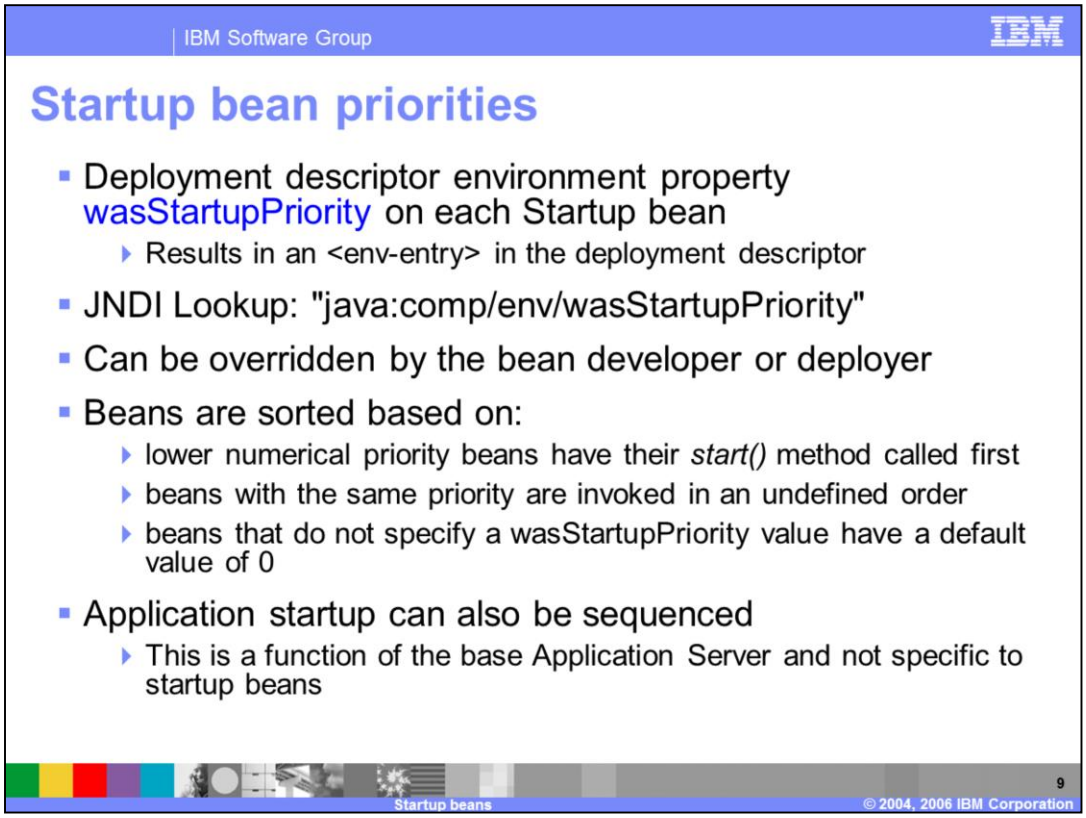
It is important to ensure that the container does not passivate and subsequently destroy the startup bean due to inactivity. This would cause a failure when the stop() method is invoked. Therefore, any session timeout parameter set by the user will be ignored, and the startup beans will never experience a session timeout.

Multiple startup beans can exist in the same EJB jar file. They can be prioritized by assigning a value to the optional wasStartupPriority environment variable. If no priority value is specified, the priority will default to zero. Startup beans set with the same priority will be called in an undefined order. The container will call the start() method of the startup beans sorted in order with the lowest numerical priority first and will call the stop() method in the reverse order.

The start and stop methods on the remote interface use Run-As mode. Run-As mode specifies the credential information to be used by the security service to determine the permissions that a principal has on various resources. If security is enabled, the Run-As mode needs to be defined on all of the methods called. The identity of the bean without this setting is undefined.



To use the Startup Beans Service on the WebSphere Application Server, use the administrative console to ensure the service is enabled. This service is disabled by default. Start the administrative console, select servers > application servers > server_name > Container Services > Startup beans service, then select the “Enable service at server startup” box.

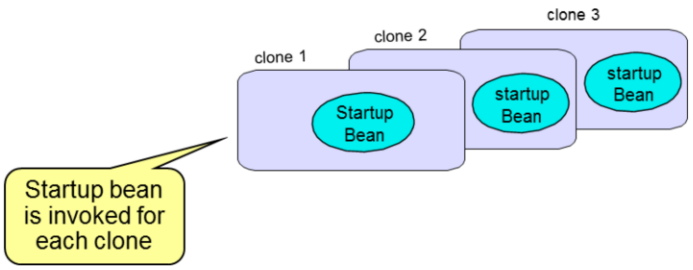


As previously mentioned, startup beans can be sequenced by assigning a priority value. The priority is set by assigning a value to the environment property, wasStartupPriority, in the deployment descriptor. The priority value must be an integer. The start method of the bean with the lowest numerical priority is called first. The stop method of the bean is invoked in the opposite order.

Startup beans are invoked on an application-by-application basis. The overall sequence of starting enterprise applications in a WebSphere Application Server can also be configured as a function of the base Application Server and is not specific to startup beans.

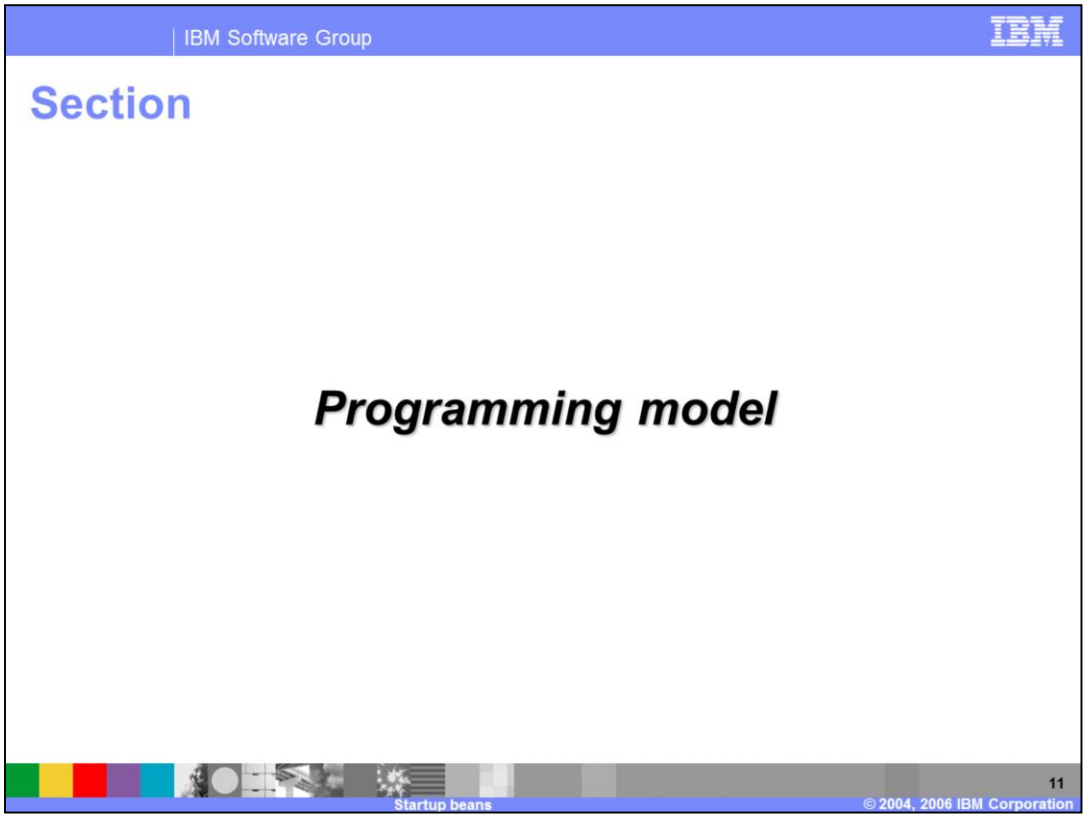
Clustering support

- A normal start of an application corresponds to a new Startup bean instance
- Server runtime caches a handle to the bean in memory to access it during shutdown
- In a cluster, the Startup bean is invoked for each cluster member
  - Important for developer awareness to avoid interference and inconsistent data
  - Critical when startup beans update persistent information

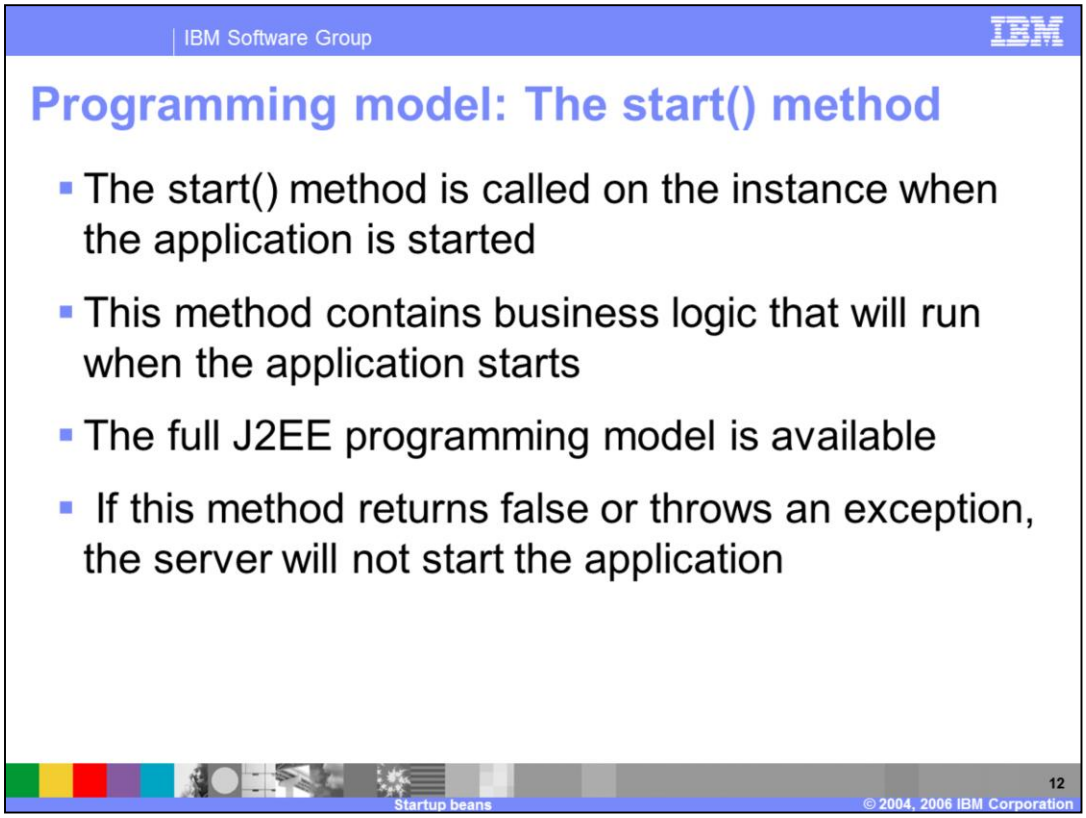


© 2004, 2006 IBM Corporation

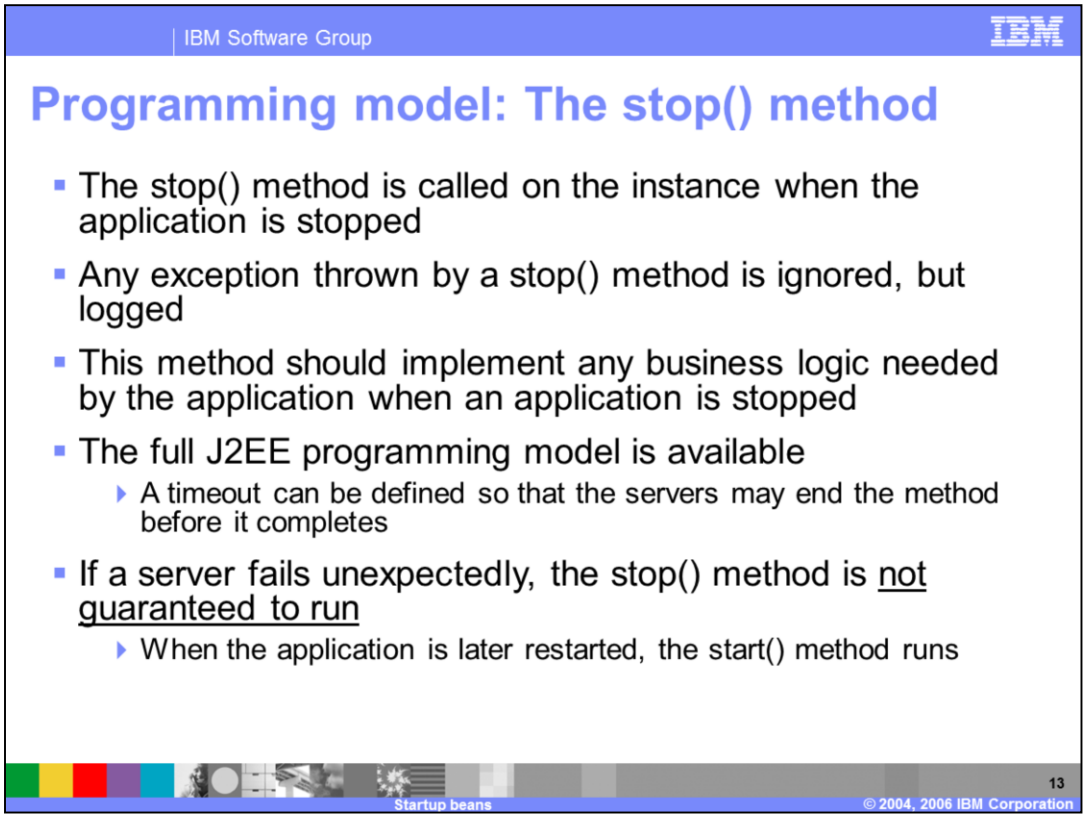
Startup beans are clustered with applications. However, cluster members do not have a way to recognize that other cluster members may be running the start or stop method on the same startup beans. Therefore, it is important for the developer to make sure that multiple instances of startup beans do not interfere with each other. This issue is particularly sensitive when startup beans update persistent information on a common persistent store; for instance, a startup bean may try to schedule a task whose definition is stored in a database.



The next section will discuss the programming model of Startup beans.



The start method is invoked when the application is started. You will need to provide an implementation of the start method for each startup bean. The full J2EE programming model is available. If the start method returns false or throws an exception, this is an indication that the application may not run successfully; therefore, the complete application will not be started.



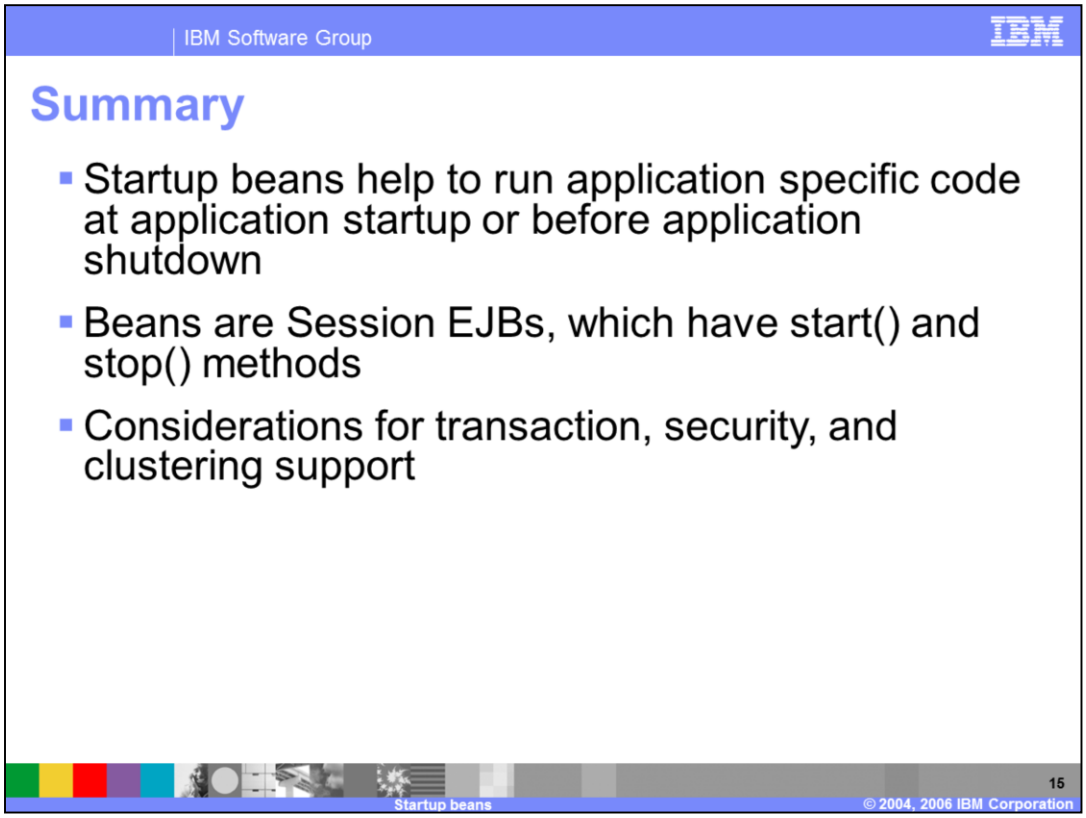
Similar considerations apply to the stop method as those that apply to the start method. The stop method is invoked when the application is stopped. You will need to provide an implementation of the stop method for each startup bean. The full J2EE programming model is available. If the stop method throws an exception, the exception is logged but otherwise ignored.

Typically, the stop method performs cleanup operations or provides notification to any other processes requiring notification of the fact that the application is stopping. If a server fails unexpectedly, the stop method may not be called. Once the server restarts and the application is started, the start method will be invoked.

Section

Summary

© 2004, 2006 IBM Corporation



In this presentation, you learned about Startup beans and how they are used to run specific business logic code at application startup or shutdown. Startup beans are session beans which have start and stop methods implemented by the developer. When using Startup beans, you need to consider transactional support, security settings, and clustering support.

Trademarks, copyrights, and disclaimers

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM	CICS	IMS	MQSeries	Tivoli
IBM(logo)	Cloudscape	Informix	OS/390	WebSphere
e(logo)business	DB2	iSeries	OS/400	xSeries
AIX	DB2 Universal Database	Lotus	pSeries	zSeries

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

Microsoft, Windows, Windows NT, and the Windows logo are registered trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel, ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a registered trademark of Linus Torvalds.

Other company, product and 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 and/or changes in the product(s) and/or program(s) 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.

Information is provided "AS IS" without warranty of any kind. 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 (e.g., 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.

© Copyright International Business Machines Corporation 2004, 2006. All rights reserved.

Note to U.S. Government Users - Documentation related to restricted rights-Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract and IBM Corp.