

WebSphere Application Server Support for OSGi Applications





Agenda

- Standards and Open Source
- WAS v7 OSGi Feature Pack
- WAS v8 Beta OSGi Applications Support
- WAS v8 Beta Demo



Standards and Open Source



Standards and Open Source

- OSGi Enterprise Specification
 - OSGi Enterprise Expert Group (EEG)
 - First release 22 March 2010
 - Brings Enterprise technologies and OSGi together
 - Adds Spring-derived *Blueprint* component model and DI container



Standards and Open Source

- OSGi Enterprise Specification
 - OSGi Enterprise Expert Group (EEG)
 - First release 22 March 2010
 - Brings Enterprise technologies and OSGi together
 - Adds Spring-derived *Blueprint* component model and DI container

- Apache Aries
 - http://aries.apache.org/
 - Provide enterprise OSGi spec implementations
 - Collaborative environment to experiment and drive further standardisation
 - Integrated into numerous projects and products





WebSphere Application Server V7 OSGi Applications Feature Pack



WAS v7 OSGi & JPA 2.0 Feature Pack

WebSphere Application Server V7 Feature Pack for OSGi Applications and Java Persistence API (JPA) 2.0

http://www-01.ibm.com/software/webservers/appserv/was/featurepacks/

- Early Access November 2009
- Most rapidly downloaded v7 FEP Beta release
- General Availability May 2010



Isolation and Sharing





Duane Appleby - applebyd@uk.ibm.com © 2011 IBM Corporation



WAS v7 OSGi : Getting Started WARs to WABs

- No java code change : war modules → bundles
- Common bundles factored out and used at specific versions



WAS v7 OSGi : Enterprise Bundle Archive ("OSGi Applications")

blog.eba

- An isolated, cohesive application consisting of a collection of bundles, is deployed as a logical unit in a ".eba" archive
- Constituent bundles may be contained ("by-value") or referenced from a bundle repository.
- Services provided by the application are isolated to the application unless explicitly exposed through EBA-level application manifest
- Config by exception absence of APPLICATION.MF means: application content is the set of bundles contained by-value plus any repository-hosted dependencies identified during deployment.



Bundle Repository

json4j.jar

WAS v7 OSGi : Internal Bundle Repository

Integrated Solutions Console Welcome		Help Logout	
View: All tasks	Cell=localhostCell01, Profile=dmgr		Close page
Welcome			
Guided Activities Guided Activi	Internal bundle repository		Field help For field help information.
Servers	The internal bundle repository can store bundles that are	referenced by OSGi applications running in WebSphere Application	select a field label or list
	contents of the asset, the contents of the internal bundle	repository, and the contents of any available external bundle	cursor is displayed.
E Services	Preferences		Page help
Resources			More information about this page
Security	New Delete		
Environment			
 Virtual hosts Undate clobal Web server plugs in configuration 	Select Bundle symbolic name 🛟	Bundle version 💲	
 Opdate global web server plug in comparation WebSphere variables 	You can administer the following resources:		
Shared libraries	com.ibm.json.java	1.0.0	
Replication domains			
	Total 1		
Naming OSGi bundle repositories			
E Caternal hundle repositories			
External bundle repositories Internal bundle repository			

🗄 UDDI

System administration
 Users and Groups
 Monitoring and Tuning
 Troubleshooting
 Service integration

WAS v7 OSGi : External Bundle Repository

Integrated Solutions Console Welcome		lluull⊗
View: All tasks	Cell=localhostCell01, Profile=dmgr Close pa	ge
Welcome	External bundle repositories ? - Help	
Guided Activities Guided Activi	External bundle repositories > New Field help	
	The configuration settings for the selected external bundle repository.	1
	Configuration marker when the help cursor is displayed.	
	Page help	
	More information about	
	General Properties	
Environment	* Bundle repository name	
 Virtual hosts Update global Web server plug-in configuration WebSphere variables Shared libraries Replication domains URI Groups Naming OSGi bundle repositories External bundle repositories Internal bundle repository 	Bundle repository description Bundle repository URL	
	Apply OK Reset Cancel	
Monitoring and Tuning		
Troubleshooting		
E UDDI		



WAS v7 OSGi : Deploying an OSGi Application (EBA)

Integrated Solutions Console Welcome			Help Logout	
View: All tasks	Cell=local	hostCell01, Profile=dmgr		Close page
Welcome	Assets		?	– Help –
Guided Activities Guided Activi	Assets			Field help
	Use thi	s page to manage assets in the asset repository.	Assets represent physical binaries. Examples of	For field help information, select a field label or list
⊟ Applications	Service	Component Architecture (SCA) composite JAX files	n (EJB) Java(TM) archive (JAR) files, EAR files, s, mediation JAR files, shared library JAR files,	marker when the help cursor is displayed.
New Application Application Types	and no	n-Java EE contents such as PHP applications. erences		Page help More information about
WebSphere enterprise applications Business-level applications Accere		ort Delete Update Export		this page Command Assistance
	પા પ			View administrative scripting command for last
	Select	Name 🗘	Description 🗘	action
H Resources	You ci	an administer the following resources:		
Security		com.ibm.ws.eba.example.blabber.app.eba	Blabber	
Environment		com.ibm.ws.eba.example.blog.eba	Aries Blog	
	Total	2		
Honitoring and Tuning				
Troubleshooting				
Service integration				
UDDI				

1) Import the asset (eba)



WAS v7 OSGi : Application-centric Bundle Management



WAS v7 OSGi : Deploying an OSGi Application (EBA)

Integrated Solutions Console Welcome					Help Logo	out	
View: All tasks	Cell=localhostCell01, Profile=dmgr	r					Close page
Welcome	Business-level applications					? -	Help –
Guided Activities Guided Activities Guided Activities Subscript Activities Guided Activities Subscript Activities Subscrite Subscript	Business-level applications > bl	og					Field help
± Servers	Use this page to manage the co	omposition units in t	he business-level	application.			For field help information, select a field label or list
Applications							marker when the help cursor is displayed.
 New Application Application Types WebSphere enterprise applications Business-level applications Assets 	General Properties Name blog Description					-	Page help More information about this page Command Assistance View administrative
Services							scripting command for last action
Resources							
Environment							
	Deployed assets						
Users and Groups	Add • Delete						
Monitoring and Tuning							
Troubleshooting Troubleshooting							
Service integration	Select Name	Description	Туре	Status	<u>ů</u>		
■ UDDI	None						
	Add Delete Add Delete Select Name None	Description	St	atus 👲			

- 1) Import the asset (eba)
- 2) Create Business-level Application (BLA)
- 3) Add Deployed asset

WAS v7 OSGi : Deploying an OSGi Application (EBA)

View: All tasks Close page * Welcome Close page © Guided Activities Close page © Guided Activities Close page © Applications Step 21 Map © Asplications Step 31 Map context Name Composition unit upon distribution Resources Step 51 Summary Step 51 Summary Start composition unit upon distribution @ System administration DEFAULT @ Sancie inservation Next @ Sancie inservation Next	Integrated Solutions Console Welcome		Help Logout	
■ Welcome ? Idepleted ■ Guided Activities Idepleted Field help B Servers Use this page to specify options for the composition unit to be added to the business-level application. Field help ■ Applications Step 1: Set options Set options Set options ■ Application Step 2: Map Composition unit settings. Backing 1D ■ Applications Step 3: Map context. Backing 1D More information, about this page ■ Assets Step 4: Map modules to virtual hosts Step 4: Map modules to virtual hosts Step 5: Summary B Services Step 5: Summary Start composition unit upon distribution Extend for unit get part B Services Step 5: Summary Start composition unit upon distribution Start composition unit upon distribution B Services modinistration Users and Groups Start composition unit upon distribution Start composition unit upon distribution B Services insorting Next Cancel Next Cancel	View: All tasks	Cell=localhostCell01, Profile;	=dmgr	Close page
Image: Guided Activities Image: Guided Activities Field help For field help information. Image: Guided Activities Applications Step 1: Set options for the composition unit to be added to the business-level application. Field help For field help information. Image: Guided Activities New Application Step 1: Set options Set options Composition unit settings. Backing 1D Guided Activities Field help For field help information. Image: Guided Activities Step 2: Map composition unit to a target Composition unit settings. Backing 1D Guided Activities More information about this page to specify options for the composition unit settings. Backing 1D More information about this page to specify options for the composition unit settings. Backing 1D More information about this page to specify option a target More information about this page to specify option at the page to specify option at target Step 3: Map context not the page to specify option at the page to specify option at target Step 3: Map context not target More information about this page to specify option at target Step 3: Map context not target<	Welcome	Set options settings	2 -	_ Help _
B Servers Use this page to specify options for the composition unit to be added to the business-level application. For field help information, select a field label or list marker when the help cursor is displayed. Application S Step 1: Set options Set options Composition unit settings. B Application Types Step 2: Map composition unit to a target Composition unit settings. Backing ID WebSphere enterprise applications Step 3: Map context roots Name com.ibm.ws.eba.example.blog_0001.eba Mare information about this page B Services Step 1: Summary Starting weight Starting weight Start composition unit upon distribution B Services and Groups Monitoring and Tuning Next Cancel Next Cancel	Guided Activities Guided Activi			- Field help
 Applications Application Application Types WebSphere enterprise applications Business-level applications Assets Services Security Security Step 3: Map context nocts Step 4: Map modules to virtual hosts Step 5: Summary Start ing weight Start composition unit upon distribution Start composition unit upon distribution Start composition unit upon distribution Resources Start composition unit upon distribution Restart behavior on update DEFAULT ▼ 		Use this page to specify o	pptions for the composition unit to be added to the business-level application.	For field help information,
 New Application Application Types WebSphere enterprise applications Business-level applications Assets Step 3: Map context roots Step 4: Map modules to virtual hosts Step 5: Summary Step 5: Summary Starting weight Start composition unit upon distribution Resources Step 5: Summary Start composition unit upon distribution Restart behavior on update DEFAULT Next Cancel 	Applications	→ Step 1: Set options	Set options	marker when the help
Step 4: Map modules to virtual hosts Security Step 5: Summary Step 5: Summary Start composition unit upon distribution Start composition unit upon distribution Woitoring and Tuning Troubleshooting Image: Service integration Next Cancel Concion.ibm.ws.eba.example.blog_0001.eba View administrative scription action View administrative Description Step 5: Summary Start composition unit upon distribution Restart behavior on update DEFAULT	 New Application Application Types WebSphere enterprise applications Business-level applications Assets 	Step 2: Map composition unit to a target Step 3: Map context roots	Composition unit settings. Backing ID WebSphere:assetname=com.ibm.ws.eba.example.blog.eba Name	Page help More information about this page Command Assistance
 		Step 4: Map	com.ibm.ws.eba.example.blog_0001.eba	View administrative scripting command for last
 ➡ Security ➡ Environment ➡ System administration ➡ Users and Groups ➡ Monitoring and Tuning ➡ Troubleshooting ➡ Service integration 	Resources	hosts		action
 Environment Image: System administration Image: System administration	E Security Securit	Step 5: Summary	Starting weight	
System administration System administration Start composition unit upon distribution Users and Groups Restart behavior on update Monitoring and Tuning Troubleshooting Next Cancel	Environment			
			Start composition unit upon distribution	
E Monitoring and Tuning DEFAULT DEFAULT DEFAULT DEFAULT			Restart behavior on update	
Troubleshooting Next Cancel	Monitoring and Tuning		DEFAULT	
	Troubleshooting	Next Cancel		
	E Service integration			
⊕ UDDI	1 UDDI	L		

1) Import the asset (eba)

- 2) Create Business-level Application (BLA)
- 3) Add Deployed asset
- 4) Configure in wizard



WAS v7 OSGi : Controlling OSGi and Enterprise Applications

Integrated Solutions Console Welcome				Help Logout	
View: All tasks	Cell=loca	lhostCell01, Profile=dmgr			Close page
Welcome	Business	-level applications		?	= Help =
Guided Activities Guided Activities Guided Activities Subscript Activities Guided Activities Subscript Activities Subscrite Subscript	Busine	ess-level applications			Field help
E Servers	Use th	is page to manage business-leve	el applications. A business-level a	application is a configuration that	select a field label or list
Applications	Enterp	rise Edition (Java EE) applications	or modules, shared libraries, da	ta files, or other business-level	marker when the help cursor is displayed.
New Application Application Types	applica	ferences			Page help More information about
WebSphere enterprise applications Business-level applications	Sta	rt Stop New Delete			this page Command Assistance
= Assets			,		View administrative
Services	Select	Name 🛟	Description 🗘	Status 🙆	action
	You d	an administer the following resou	irces:		
Security		an enterprise app ear		*	
Environment		blabber		8	
				8	
		blog		8	
	Total	3			
Troubleshooting					
Service integration					
UDDI UDDI					

WAS v7 OSGi : Application-centric Bundle Management

Integrated Solutions Console Welcome	Help Log	put The second s	
View: All tasks	Cell=localhostCell01, Profile=dmgr	Close page	
Welcome	Assets	? _	
Guided Activities Guided Activi	Assets > com.ibm.ws.eba.example.blog.eba		
1 Servers	Use this page to manage assets in the asset repository. Assets represent physical binaries. Ex	amples of assets include compressed	
□ Applications	(zip) files, Enterprise JavaBean (EJB) Java(TM) archive (JAR) files, EAR files, Service Componer files, mediation JAR files, shared library JAR files, and non-Java EE contents such as PHP applic	at Architecture (SCA) composite JAR	
New Application			
Application Types	General Properties	Additional Properties	
WebSphere enterprise applications	Asset name	Export the	
Business-level applications	com.ibm.ws.eba.example.blog.eba	deployment	
Assets	Asset description	this application	
Services	Aries Blog	Import a	
Resources		deployment manifest into this	
Security		application	
Environment	Asset binaries destination URL	<u>Update bundle</u> <u>versions in this</u>	
	\${USER_INSTALL_ROOT}/installedEBAs/com.ibπ	application	
	Asset type aspects		
Monitoring and Tuning	EBA,version=1.0		
Troubleshooting			
Service integration	File permissions		
	Allow all files to be read but not written to		
E 9991	Allow HTML and image files to be read by everyone		
	.*\.dll=755#.*\.so=755#.*\.a=755#.*\.sl=755		
		Internal bundle repository	? =
	Asset relationships	Internal bundle repository	
	Current asset relationships	The internal bundle repository can store bundles the	at are referenced by OSGi applications running in
	none	WebSphere Application Server. When an OSGi appli attempts to satisfy all its dependencies by using th	cation is imported as an asset, the provisioner e contents of the asset, the contents of the internal
	Manage Relationships	bundle repository, and the contents of any available	e external bundle repositories.
		Naw Dalata	
	Validate asset		
	EBA Dependencies	Select Bundle symbolic name 💸	Bundle version 🗘
	Bundle downloads are complete.	com.ibm.ison.java	1.0.0
			110
			1.1.0
	OK Cancel	Total 2	

WAS v7 OSGi : Application-centric Bundle Management

Integrated Solutions Console Welcome

Help | Logout



Close page

*

Applications	^	Cell=irobinsNode01Cell, Pr	rofile=AppSrv01						Close
 New Application Application Types WebSphere enterprise applications Business-level applications Assets 		Assets Assets > com.ibm.ws.e Update the versions of Application bundle cont	ba.example.blo the bundles that tent	q <u>.eba</u> > (compris	J pdat e this	t e bundle application	versions	in t	his application
E Services Resources		Symbolic name		Cor	ntent ≥	Sharing	Deploye version	ed I	New version
		com.ibm.ws.eba.exam	ple.blog	Bur	dle	Isolated	1.0.0		No preference 💌
Environment		com.ibm.ws.eba.exam	ple.blog.api	Bur	dle	Isolated	1.0.0		
 Virtual hosts Update global Web server plug-in 		com.ibm.ws.eba.exam	ple.blog.persiste	nce Bur	dle	Isolated	1.0.0		1.1.0
WebSphere variables		com.ibm.ws.eba.exam	ple.blog.web	Bur	dle	Isolated	1.0.0		No preference 1.0.0 1.1.0
 Shared libraries Replication domains 		Use bundle content							
Naming		Symbolic name	Content type	Sharing	Dep	ployed ver	sion	New	version
 OSGi bundle repositories External bundle repositories Internal bundle repository 		com.ibm.json.java	Bundle	Shared	1.0	.0		No	o preference 💌
		Preview Cancel							

- Preview changes
- Commit changes
- Restart BLA

WAS v7 OSGi : Application-centric Bundle Management

com.ibm.ws.eba.example.blog.eba

Integrated Solutions Console Welcome Help Logout Close page Cell=localhostCell01, Profile=dmgr View: All tasks -**Business-level applications** Help Welcome Guided Activities Field help Business-level applications > blog > com.ibm.ws.eba.example.blog_0001.eba Servers For field help Use this page to manage the composition unit. A composition unit is backed by an asset and contains information, select a New server configuration metadata. It contains customized configuration for such service definitions, references field label or list Server Types and other relevant configuration data. It also contains a list of deployment targets or runtime marker when the help WebSphere application servers environments along with the runtime environment specific configuration where the composition unit is cursor is displayed. WebSphere proxy servers expected to run. Generic servers Page help Version 5 JMS servers **General Properties** WebSphere MQ servers More information Additional Properties about this page Web servers Name Relationship Clusters com.ibm.ws.eba.example.blog_0001.eba options Command + DataPower Assistance + Core Groups Description View administrative Applications scripting command for last action New Application Application Types WebSphere enterprise applications Business-level applications Assets Backing ID Services WebSphere:assetname=com.ibm.ws.eba.example.blog.eba + Resources ± Security * Starting weight Indicates **BLA** + Environment 1 System administration Start on distribution Cell restart required to Save changes to master repository Recycle behavior on update Deployment manager DEFAULT 🔻 move to latest Nodes Node agents Node groups Target mapping asset deployment Centralized Installation Manager Console Preferences Modify Targets... Console Identity + Users and Groups Current targets Monitoring and Tuning WebSphere:node=localhostNode01,server=server1 Troubleshooting Service integration **Composition unit status** + UDDI Composition unit status Backing asset

New version available.



WAS v7 OSGi : Blueprint Components and Services



The Blueprint Dependency Injection container is a part of the server runtime (compared to the Spring container which is part of the application.)

- Standardised established Spring conventions
- Configuration and dependencies declared in XML 'module blueprint' (standardisation of Spring 'application context' XML)
 - Extended for OSGi: publishes and consumes components as OSGi services
- Simplifies unit test outside either Java EE or OSGi runtime



WAS v7 OSGi : Blueprint

Components and Services

- injected service references
- services can change over time can be temporarily absent without the bundle caring
- managed by Blueprint container



- "prototype" scope indicates a new instance is created by the container for each use.
- "singleton" scope is the default.



Persistence and Transactions

- OpenJPA is default persistence provider in WebSphere
- Container managed JPA support integrated into Blueprint container:
 - @PersistenceUnit or @PersistenceContext (managed)
 - or <jpa:unit>, <jpa:context> bean property injection
 - Familiar development experience for JPA developers
 - Load-time enhancement of Entity classes
- Same container managed transaction attributes as EJBs:
 - Required, RequiresNew, Mandatory, NotSupported, Supports, Never



WAS v7 OSGi : OSGi Service Registry and JNDI

- OSGi services are published to and looked up from OSGi service registry.
 - Declarations in Blueprint XML
- Interactions with the OSGi service registry:
 - Services also available in JNDI via the osgi:service URL scheme (additionally as aries:services)
 - Resources bound to JNDI published as services in the OSGi the Service Registry. Published as a service property called "osgi.jndi.service.name"



WAS v7 OSGi : OSGi Service Registry and JNDI



```
<persistence-unit name="blogExample" transaction-type="JTA">
......
<provider>
    org.apache.openjpa.persistence.PersistenceProviderImpl
  </provider>
    (jta-data-source>
        aries:services/javax.sql.DataSource/(osgi.jndi.service.name=jdbc/blogdb)
  </jta-data-source>
        aries:services/javax.sql.DataSource/(osgi.jndi.service.name=jdbc/blogdbnojta)
        </non-jta-data-source>
        ......
</persistence-unit>
```



WAS v7 OSGi : Sample Application Architecture







WAS v7 OSGi : Composite Bundle Archive (CBA)

- Composite Bundle Manifest
- All bundles placed into internal bundle repository
- Preference to resolve against CBAs

Manifest-Version: 1.0 CompositeBundle-ManifestVersion: 1 Bundle-Name: Blog Application Bundle-SymbolicName: com.ibm.ws.osgi.example.Blog Bundle-Version: 1.0 CompositeBundle-Content: com.ibm.ws.osgi.example.blog;version="[1.0,1.0]", com.ibm.ws.osgi.example.blog.persistence;version="[1.0,1.0]" Import-Package: com.ibm.ws.other.pkge;version=1.0.0 Export-Package: com.ibm.ws.osgi.example.blog;version=1.0.0 CompositeBundle-ExportService: com.ibm.ws.osgi.example.blog.BloggingService;filter="(blog.type=community)" CompositeBundle-ImportService: com.ibm.ws.osgi.example.auth.UserAuthService



WAS v7 OSGi Summary

- Isolation and Sharing
- Classpath Control
- Versioning Control
- Integration of established technologies JPA/JNDI/JTA
- Blueprint dependency injection publish and consume services
- EBA
- Bundle Repositories
- CBA



WebSphere Application Server V8 Beta



Disclaimer

IBM's statements regarding its plans, directions, and intent are subject to change or withdrawal without notice at IBM's sole discretion. Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision. The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract. The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.



WAS v8 Beta

WebSphere Application Server V8 Beta

https://www14.software.ibm.com/iwm/web/cc/earlyprograms/websphere/wsasoa/index.shtml

• Latest Beta refresh 11th March 2011

WAS v8 Beta : Post Install Configuration Changes

WebSphere. software		Welcome	Help Logout
View: All tasks	Cell=localhostCell01, Profile=dmgr		Close page
Welcome	Business-level applications	2 -	Help –
± Guided Activities			Field heln
± Jobs	Business-level applications > Config Changes > test_0001.eba		For field help
± Servers	Use this page to manage the composition unit. A composition unit is backet configuration metadata. It contains customized configuration for such ser	ed by an asset and contains	information, select a
Applications	other relevant configuration data. It also contains a list of deployment tar	rgets or runtime environments	field label or list marker when the help
 New Application Application Types WebSphere enterprise applications Business-level applications Assets Global deployment settings 	along with the runtime environment specific configuration where the comp General Properties Name test_0001.eba	Additional Properties	cursor is displayed. Page help More information about this page
Services	Description	options	Command
± Resources		Session	View administrative
+ Security		management	scripting command for
± Environment		Context roots	last action
System administration		Security role to user/group	
🗄 Users and Groups		mapping	
Monitoring and Tuning	Backing ID	Map RunAs roles	
Troubleshooting	WebSphere:assetname=test.eba	to users	
Ervice integration	* Starting weight	Virtual hosts	
Estrvice integration UDDI	 * Starting weight Start on distribution Recycle behavior on update DEFAULT Target mapping Modify Targets Current targets WebSphere:node=localhostNode01,server=server1 OSGi application deployment status Using latest OSGi application deployment. Update to latest deployment 	 Manage extensions for this composition unit 	

WAS v7 OSGi : Screen shot

Integrated Solutions Console Welcome Help | Logout Cell=localhostCell01, Profile=dmgr Close page View: All tasks -Business-level applications Help Welcome Guided Activities Field help Business-level applications > blog > com.ibm.ws.eba.example.blog_0001.eba Servers For field help Use this page to manage the composition unit. A composition unit is backed by an asset and contains information, select a New server configuration metadata. It contains customized configuration for such service definitions, references field label or list Server Types and other relevant configuration data. It also contains a list of deployment targets or runtime marker when the help WebSphere application servers environments along with the runtime environment specific configuration where the composition unit is cursor is displayed. WebSphere proxy servers expected to run. Generic servers Page help Version 5 JMS servers **General Properties** More information WebSphere MQ servers Additional Properties about this page Web servers Name Relationship Clusters com.ibm.ws.eba.example.blog_0001.eba Command options DataPower Assistance ± Core Groups Description View administrative Applications scripting command for last action New Application Application Types WebSphere enterprise applications Business-level applications Assets Backing ID Services WebSphere:assetname=com.ibm.ws.eba.example.blog.eba + Resources ± Security * Starting weight Environment 1 System administration Start on distribution Cell V7 FEP lacked any post Save changes to master repository Recycle behavior on update Deployment manager DEFAULT -Nodes config changes Node agents Node groups Target mapping 🗄 Centralized Installation Manager Console Preferences Console Identity Modify Targets... + Users and Groups Current targets Monitoring and Tuning WebSphere:node=localhostNode01,server=server1 + Troubleshooting Service integration Composition unit status ± UDDI Backing asset Composition unit status com.ibm.ws.eba.example.blog.eba New version available.



WAS v8 Beta : In-place Update

WebSphere. software	Welcome	Help Logout 🔢
View: All tasks	Cell=localhostCell01, Profile=dmgr	Close page
Welcome	Business-level applications 2 -	Help
± Guided Activities		Field help
± Jobs	<u>Business-level applications</u> > <u>Config Changes</u> > test_0001.eba	For field help
Servers	Use this page to manage the composition unit. A composition unit is backed by an asset and contains	information, select a
Applications	other relevant configuration data. It also contains a list of deployment targets or runtime environments	field label or list
New Application Application Types WebSphere enterprise applications Reviewed polications	along with the runtime environment specific configuration where the composition unit is expected to run. General Properties Additional Properties	cursor is displayed. Page help
Business-level applications Assets	Name View domain	about this page
Global deployment settings	test_0001.eba	
	Description	Command
± Resources	Session	View administrative
Security	management	scripting command for
Environment	Context roots	last action
System administration	Security role to user/group	
🛨 Users and Groups	mapping	
Monitoring and Tuning	Backing ID Map RunAs roles	
+ Troubleshooting	WebSphere:assetname=test.eba	
+ Service integration	Starting weight	
± UDDI	Manage 1 extensions for this composition unit	
	Start on distribution	
	Recycle behavior on update DEFAULT	
	Target mapping	
	Modify Targets	
	WebSphere:node=localhostNode01,server=server1	
	OSGi application deployment status	
	Using latest OSGi application deployment.	
	Update to latest deployment	



WAS v8 Beta : Application Extensions

ew: All tasks	Cell=localhostCell01, Profile=dmgr		-1
			Close page
Velcome	Business-level applications	? -	Help -
Guided Activities			Field help
Jobs	Business-level applications > Config Changes > test_0001.eba		For field help
Servers	Use this page to manage the composition unit. A composition unit is backet	ed by an asset and contains	information, select a
Applications	other relevant configuration data. It contains customized configuration for such set	rgets or runtime environments	field label or list
New Application	along with the runtime environment specific configuration where the comp	osition unit is expected to run.	cursor is displayed.
Application Types			
WebSphere enterprise applications	General Properties	Additional Properties	Page help
Business-level applications	Name	- Minur damain	More information
Assets Global deployment settings	test 0001.eba	View domain	about this page
- Clobal deployment settings		Relationship options	Command
Services	Description		Assistance
Resources		Session management	View administrative
Security			scripting command for
Environment			last action
System administration		Security role to user/group	
Jsers and Groups		mapping	
Aonitoring and Tuning	Backing ID	Map RunAs roles	
Troubleshooting	WebSphere:assetname=test.eba	to users	
Service integration	Ctarting weight	Virtual hosts	
UDDI	* Starting Weight	Manage	<
		extensions for this	
	Start on distribution		
	Recycle behavior on update		
	DEFAULT		
	Target mapping		
	Modify Targets		
	WebSphere:node=localhostNode01,server=server1		
	OSGi application deployment status		
	soor appreation acprogrammere status		
	Using latest OSGi application deployment.		
	Update to latest deployment		



WAS v8 Beta : Composite Bundle Archive (changes)

- Content Hiding
- Can be specified on Application-Content
- Can contain WABs
 - Configurable during 'addAsset'
- Exported assets now contain their CBA content
- Support for bulk bundle uploads via zip archives



WAS v8 Beta : Map RunAs roles to users

WebSphere. software		Welcome	Help Logout IBN
View: All tasks	Cell=localhostCell01, Profile=dmgr		Close page
Welcome	Business-level applications	2 -	Help –
Guided Activities	Business level and institute > Config Changes > test 0001 also		Field help
+ Jobs	Business-level applications > Config Changes > test_0001.eba		For field help
Servers	Use this page to manage the composition unit. A composition unit is line configuration metadata. It contains customized configuration for such	backed by an asset and contains	information, select a
Applications	other relevant configuration data. It also contains a list of deployment	nt targets or runtime environments	field label or list
New Application	along with the runtime environment specific configuration where the	composition unit is expected to run.	cursor is displayed.
Application Types			
WebSphere enterprise applications	General Properties	Additional Properties	Page help
Business-level applications Assets	Name	View domain	about this page
Global deployment settings	test_0001.eba	Relationship	about the page
+ Services	Description	options	Command
+ Resources		Session	Assistance
Security		management	scripting command for
E Environment		Context roots	last action
Environment System administration		Security role to	
		user/group mapping	
Users and Groups		Map RupAs roles	
+ Monitoring and Luning	Backing ID	to users	>
± Troubleshooting	WebSphere:assetname=test.eba	Virtual hosts	
± Service integration	* Starting weight	Manage	
± UDDI	1	extensions for this	
	Start on distribution	<u>composition unit</u>	
	Begyale behavior on undate		
	Target mapping		
	Modify Targets		
	Current targets		
	WebSphere:node=localhostNode01,server=server1		
	OSGi application deployment status		
	Using latest OSGi application deployment.		
	Update to latest deployment		


WAS v8 Beta : Session Replication

WebSphere. software		Welcome	Help Logout 🔢
View: All tasks	Cell=localhostCell01, Profile=dmgr		Close page
Welcome	Business-level applications	? -	Help
Guided Activities			Field help
Jobs	<u>Business-level applications</u> > <u>Config Changes</u> > test_0001.6	eba	For field help
Servers	Use this page to manage the composition unit. A composition unit	t is backed by an asset and contains	information, select a
Applications	other relevant configuration data. It also contains a list of deploy	yment targets or runtime environments	field label or list
 New Application Application Types WebSphere enterprise applications 	along with the runtime environment specific configuration where General Properties	the composition unit is expected to run.	cursor is displayed.
Business-level applications	Name	Additional Properties	More information
Assets	test 0001 eba	View domain	about this page
Global deployment settings		<u>Relationship</u> options	Command
Services	Description		Assistance
(esources		management	<u>View administrative</u> scripting command for
security		Context roots	last action
Invironment		Security role to	
ystem administration		user/group	
Jsers and Groups		mapping	
Aonitoring and Tuning	Backing ID	Map RunAs roles to usors	
Froubleshooting	WebSphere:assetname=test.eba		
Service integration	* Starting weight		
UDDI	1	extensions for this	
	Ctast on distribution	composition unit	
	DEFAULT		
	Target mapping		
	Modify Targets		
	Current targets WebSphere:node=localhostNode01,server=server1		
	OSGi application deployment status		
	Using latest OSGi application deployment.		
	Update to latest deployment		



WAS v8 Beta : Bundle Cache

View: All tasks 🔻	Cell=loca	hostCell01, Profile=dmgr				Close pa
Welcome	Bundle	cache			? -	Help
∃ Guided Activities						Field help
∃ Jobs	Bund	e cache				For field help
- Servers	The b	undle cache manager allows you to interact with bund	les that are in t	he bundle cache. The	bundle	information, select a
Applications	cache	is a local directory that contains bundles that are ref downloaded from both internal and external repositorie	erenced by OSG	i applications, and the the Bundle Cache Mar	at have	field label or list
New Application	det a	up-to-date list of the bundles in the bundle cache, t	o check if all bu	indles have been succ	essfully	marker when the help
Application Types	down	oaded, and to request that one or more bundles be do	wnloaded again		,	cursor is displayed.
 WebSphere enterprise applications 	🗄 Pre	ferences				Page help
Business-level applications	De	uplead Bundle Again Defrech List				More information
Assets						about this page
Global deployment settings		∃ ÷÷ ÷€				
E Services	Calar	Bundla Gumbalia Mana and Marian A	Charles A	Developed Obstate A	ciae A	
E Resources	Selec	Bundle Symbolic Name and Version V	State v	Download Status V	Size ~	
E Security	You	can administer the following resources:				
Environment		com.ibm.samples.websphere.osgi.logging.api 1.0.0.0	Downloaded	100%	1.5 KB	
 Virtual hosts Update global Web server plug-in configuration WebSphere variables 		com.ibm.samples.websphere.osgi.logging.impl 1.0.0.	0 Downloaded	100%	3.2 KB	
 Shared libraries SIP application routers 	Tota	12				
Replication domains						
URI Groups						
H Naming						
OSGi bundle repositories						
External bundle repositories						
Internal bundle repository						

- Garbage collection of unused bundles from bundle cache
- New panel showing download status previously MBean only

Users and Groups
 Monitoring and Tuning

 Troubleshooting

 Service integration

 UDDI

WAS v8 Beta : Performance Monitoring Infrastructure (PMI)

WebSphere. software		Welcome Help	Logout IB]
View: All tasks	Cell=localhostNode01Cell, Profile=AppSrv01		Close pa
Welcome	Performance Monitoring Infrastructure (PMI)		1
Guided Activities Guided Activities Second Secon			
Servers	<u>Performance Monitoring Infrastructure (PMI)</u> > <u>server1</u> > Custom monitoring level		
	Use this page to configure Performance Monitoring Infrastructure (PMI)		
Services	Runtime Configuration		
Resources			
Security	Enable Disable		^
Environment	ExtensionRegistryStats name		
E System administration	Select Counter A		Ctatus ^
Users and Groups	Type V Description V	finne stings of this housedly.	
Monitoring and Tuning	Bundle method invocations CountStatistic The number of method	of invocations of this bundle	Disabled
 Performance Monitoring Infrastruc (PMI) 	E com.ibm.samples.webs Bundle method response TimeStatistic The average	response time of this bundle	Displad
Request metrics	time time method		Disabled
Performance Viewer	CountStatistic The number of Service invocations	of invocations of this service	Enabled
Troubleshooting Service integration	E Dynamic Caching Dynamic Caching Dyna	of invocations of this service	Disabled
± UDDI	E <u>JDBC Connection Pools</u> E Service method response time TimeStatistic The average method	response time of this service	Disabled
	JMR Runtime ⊕ Object Pool ⊕ Object Pool ⊕ Service response time ⊤ The average	response time of this service	Enabled
	Total 6		
	pm//webServiceModule Servlet Session Manager System Data Transaction Manager Web Applications DefaultApplication#DefaultV		



WAS v8 Beta : Performance Monitoring Infrastructure (PMI)

WebSphere. software					Welcome	Help Logout
View: All tasks	Tivon Performance viewer					
Welcome	Tivoli Performance Viewer > serve	er1				
+ Guided Activities	Use this page to view and refresh pe	rformance d	lata for the selected server.	change user and log	settings, and view summ	nary reports and
+ Servers	information on specific performance	nodules.	,		, ,	
+ Applications		More inform	ation about this page			
+ Services	Refresh View Module(s)					
+ Resources		Start Lo	gging			
+ Security	⊢ server1	100				
+ Environment	T Settings	100				
System administration	Summary Reports					
+ Users and Groups	Ė- Performance Modules	80	<u></u>			
Monitoring and Tuning	ExtensionRegistryStats.name		\backslash			
 Performance Monitoring Infrastructur (PMI) Request metrics Performance Viewer Current activity View logs 	Osci Applications Disci Applications	Values 40				
Troubleshooting	HAManager			• •	•	
Service integration	JVM Runtime	20				
± UDDI	Object Pool ORB pmiWebServiceModule Servlet Session Manager	0	49 15:36:19 1	15:36:49 15:37 Time	7:19 15:37:49	15:38:19
	System Data	RR		1 mc		
	Thread Pools		Reset To Zero Clear Buff	fer View Table	Show Legend	
		Select Ma	rker Name	Value	Scale Update	Scaled Value
	Desident All	com.ibm.sa	mples.websphere.osgi.blog.a	app_1.0.0		
	Deselect All		Service invocations ⑦	18.0	1.0	18.0
			Service response time	? 278.6111	0.1	27.861113
		If you see appropriate	more or fewer available stati ly. <u>Performance Monitoring I</u>	stics than expected,	, check that your PMI lev <u>JS</u>	el settings are set



Summary – WAS v8 Beta additional OSGi support

- Post deployment configuration changes
- In-place application update
- In-place application extension
- Composite Bundle Archive

- Run As roles
- Session replication management
- Bundle cache administration
- PMI
- Servlet 3.0



Additional Info v7 FEP & v8 Beta



Additional OSGi Integration (SCA)

- Service Component Architecture (SCA)
 - Assembly into heterogeneous composites of OSGi and non-OSGi components
 - Remoting of OSGi application services through SCA services with a variety of bindings including JMS, SOAP/HTTP, IIOP and JSON-RPC.



SCA Composite assembled from heterogeneous components including an **OSGi Application** component, and integrated through SCA services with configurable bindings (JMS, web services...).



OSGi Bundles assembled in an **OSGi Application** and integrated through services in the OSGi service registry



POJOs assembled using a Blueprint context and scoped by an **OSGi Bundle**.



Additional OSGi Integration (SCA)

• SCA Integration : implementation.osgiapp

```
Manifest-Version: 1.0
Application-ManifestVersion: 1.0
Application-Name: Aries Blog
Application-SymbolicName: com.ibm.ws.eba.example.blog.app
Application-Version: 1.0
Application-Content:
com.ibm.ws.eba.example.blog.api;version="1.0.0",
com.ibm.ws.eba.example.blog.persistence;version="1.0.0",
com.ibm.ws.eba.example.blog.web;version="1.0.0",
com.ibm.ws.eba.example.blog;version="1.0.0"
Use-Bundle: com.ibm.json.java;version="[1.0.0,2.0.0)"
Application-ExportService:
com.ibm.ws.eba.example.blog.Blog
Application-ImportService:
com.ibm.ws.eba.example.blog.UserAuthorization
```





Additional OSGi Integration (Tooling)

- Free Eclipse Plugin for
 - Includes features that increase developer productivity
 - Creates OSGi Applications for any Aries-based server runtime.
 - Eclipse WTP 3.6 (Helios) M6 or later required

- RAD Tooling (v8.0.2)
 - Integrated with Web Tools, JEE productivity tools, and other capabilities in RAD
 - Supports deployment to WAS v8 Beta
 - Supports deployment to WAS v7 OSGi FeP
 - Enhanced validation



Additional OSGi Integration (RAD)

O Ja	va EE - com.ibm.ws.eba.example.blog.app/META-INF/APPL	ICATION.MF - Ra	tional® Application Developer for	WebSphere® Software
File	Edit Navigate Search Project Run Window Help			
1 2	• 🛛 🖻 🗄 💀 • 🐼 • 🚱 • 🍕 • 🕴 🐯 • 🌮	· 😕 🔗 • 🛛	◎ 🖾 🖧 🖢 - 🖗 - 🍄 🗢	🔹 😅 👘 📴 😰 Java EE
98 B	nterprise Explorer 🙁 😤 Services 📄 🕏 🧐 🏹 🗖 🗍	🧔 com.ibm.ws.eba	.example.blog.app 🛛	
•	⇒ com.ibm.ws.eba.example.blog ⇒ com.ibm.ws.eba.example.blog.api	🔂 OSGi App	lication Manifest	?
	💿 New 📃 🗆 🔀	General Informa	ation:	Imports:
	Select a wizard	Specify the fields	below for this OSGi application.	Specifies a list of services to be imported.
	Create an OSG Application project	Name:	Aries Blog	Add
		Symbolic name:	com.ibm.ws.eba.example.blog.app	Remove
	<u>Wi</u> zards:	Version:	1.0	Properties
	type filter text	Manifest Version:	1.0	
	OSGi SGi SGi Application Project OSGi Bundle Project	Contained Bund Specifies the list o application.	les: f OSGi bundles to be included in this OSGi	Down
	Plug-in Development Profiling and Logging Pro- Service Component Architecture	 com.ibm.vis.e com.ibm.vis.e com.ibm.vis.e com.ibm.vis.e 	eba.example.blog.persist eba.example.blog.web 1. eba.example.blog 1.0.0	Specifies a list of services to be exported from this OSGi application.
	Show All Wizards.	<	New Up Down	Properties
	(<u>Back</u> <u>Next</u> > <u>Einish</u> Cancel	Overview APPLICA	TION.MF	Down
Q	type filter text	🚼 Markers 🔲 Pro	operties 👫 Servers 🛛 🙀 Data Sour	rce Explorer 📔 Snippets 🗔 Annotations 📃 🗖
: •	0 items selected			1



Further Information

Resource Hub (articles, tutorials, redbooks, forums)

http://www.ibm.com/software/websphere/osgi

<u>Team Blogs/Twitter</u> www.ianrobinson.blogspot.com www.devangelist.blogspot.com @sjmaple @notatibm @TimothyWard

<u>Email</u>

applebyd@uk.ibm.com



WAS v8 Beta Demo





- Standards and Open Source

Covers some historical aspects about the use of OSGi in WebSphere and the various standards and open source projects that have emerged and encouraged the exposing of OSGi applications support.

- WAS v7 OSGi Feature Pack

Looks at the functionality provided and supported for OSGi applications and the administrative processes to install, run and monitor these applications.

- WAS v8 Beta

Looks at the advancements in functionality, the new administrative processes available and the benefits they provide.



OSGi has been with us for many years and in 2006 WAS v6.1 introduced the use of OSGi internally, however, there have been many issues surrounding how enterprise java applications make use of the benefits of OSGi in commercial enterprise Java runtimes like WebSphere, especially when we have many years of investment in Java EE where we have tools, runtimes and administrative processes.

These issues have been the primary concern of the OSGi Alliance Enterprise Expert Group and March 2010 saw the release of the first OSGi Enterprise Specification (v4.2)

The resulting specification describes how Java SE/EE technologies like JTA, JPA, JNDI, JMX, WebApps and so on run in an OSGi environment. There is no significant invention of new programming models, only the adaptation of what is already familiar into a more modular and dynamic runtime environment.

The one extension beyond pure Java EE is the specification of the Blueprint component model and dependency injection container, an evolution of the Spring framework as an OSGi <u>standard</u>

Java EE provides the core enterprise application programming model

Deploying modules as OSGi bundles simplifies reuse between applications, provides versioning, encourages (and enforces) modular design and enables dynamic module updates.



- There are a number of open source projects with a focus on Enterprise OSGi, the most complete is the Apache Aries project, formed in September 2009.

- The objectives of Apace Aries:

- To provide free, open source implementation of the enterprise OSGi technologies
- To provide an environment to collaborate and experiment with new technologies to inform EEG standardization, in particular around those technologies that affect the application programming model such as the Blueprint container.
- To establish a broad and open community with an interest in enterprise OSGi to encourage implementation and adoption of OSGi in enterprise applications.

- The Apache Aries project has grown to almost 50 contributors from companies including IBM, Progress, SAP, Redhat, Ericsson, LinkedIn, and others as well as individual contributors.

- Aries does not intend to provide a server runtime environment for enterprise OSGi but rather components that can be used in such an environment. The Apache Aries project provides implementations of

- Blueprint container
- JPA integration
- JTA integration
- IMX integration
- JNDI integration
- Application assembly and deployment

These have been integrated into Apache Geronimo and WebSphere Application Server as well as a number of other projects and products including Apache Felix Karaf and JBossOSGi.





Points to note about feature pack releases :-

- Same level of support as the base product
- Released to bring new technologies to customers at the earliest available opportunity and to give capabilities for the current release, not just to see if there is a market for the technology.
- The WAS commitment is that all feature packs must be available on future release (subject to any deprecation notices) as either:
 - a feature pack installable or,
 - integrated as part of the base product



In Java EE, modules are isolated within an application and applications are isolated from one another which makes sharing modules difficult

In OSGi 4.2 all bundles have shared visibility to the externals of all others bundles within an OSGi framework (JVM) which makes isolating applications difficult though eliminates the shortcomings of plain Java classloading:

- 1. Only declared exports are visible outside the bundle
- 2. Dependencies are resolved to specific versions and multiple versions of packages can be available concurrently for different client bundles

3. Dependencies are explicit so that bundles will not start if all dependencies cannot be resolved

Application Isolation is an important consideration. At one extreme, Java EE provides no portable notion of module sharing between enterprise applications – everything is isolated and sharing libraries is difficult.

At the other extreme, OSGi bundles have shared visibility to the externals of all other bundles within an OSGi framework, which typically means within a JVM. This makes isolating applications difficult.

The current version of Equinox, which is used inside WAS and is the reference implementation of OSGi 4.2, supports the notion of nested frameworks whereby multiple peer frameworks are isolated from one another but may share a common parent framework. WAS OSGi Feature Pack exploits this by deploying the isolated content of each OSGi Application into its own isolated child framework. Whilst any bundles intended to be shared between applications can be deployed into the single parent framework.



Getting started with OSGi application support in WAS is made very simple with no need to make any changes to web application implementation.

WAR modules can be deployed to WAS as web application bundles with no change of runtime behaviour. On its own this is not all that interesting but it becomes interesting when you have multiple applications that use common libraries. What we can do now is place versioned, common libraries in an OSGi bundle repository so that each application using these libraries delivers only their unique modules.

Remember - a bundle is just a jar with additional OSGi metadata and a classloader which respects that metadata. Throughout this presentation I use the bundle symbol to indicate a jar that is actually a bundle. In the illustration here we can take 3 web application archives which include using a number of common libraries and refactor these as 3 web application bundles containing only the unique content with the common libraries installed once into an OSGi bundle repository.



The WAS OSGi Feature Pack makes use of many concepts and reference implementations from the Apache Aries open source project

One such concept is that of 'OSGi applications' which are packaged in a new type of archive called an "enterprise bundle archive" or "EBA" for short. This is similar to an EAR but its modules are deployed as bundles to the desired target servers. The EBA archive represents a single isolated OSGi application consisting of one of more modules and is the unit of deployment for an enterprise OSGi application.

Like an EAR file, an EBA archive may contain all the constituent modules/bundles that make up the application but it may just contain the metadata required to locate those bundles from a configured bundle repository.

The metadata is in the form of an EBA-level "APPLICATION MANIFEST" file which describes the content of the application and whether the application exposes any external services and references. Just like a bundle manifest describes the modularity characteristics of a bundle, the application manifest describes the modularity characteristics of the application as well as the deployable content of the application.

The example here shows 2 OSGi Applications packaged as EBAs which contain various application bundles and an APPLICATION MANIFEST which refers to an additional "json4j.jar" bundle. When the application is deployed the json4j.jar is obtained from the configured bundle repository and doesn't have to be packaged inside the EBA.

NOTE:

Configuration is by exception – the absence of an application manifest indicates that application content is contained within the archive and the app exposes no services or references externally.



The WAS OSGi Feature Pack introduces new administrative support for using OSGi bundle repositories to simplify deploying common libraries as OSGi bundles.

It is possible to configure an externally managed repository or, more conveniently, a WAS specific internal bundle repository. Whilst external repositories have to supply there own tools for population and maintenance, the contents of the internal repository can be managed through the admin console or through scripting.

By placing common bundles into a configured repository we are able to reduce disk usage and memory footprint.

	and the second se	
•		
• •		
1		
- 99 - C C C C C C C.		
- T. C.		
-		

The external bundle repository configuration panel. External repositories contents must be managed by external tooling to WAS.

A second se
 A set of the set of
REALITY OF THE PARTY OF THE AND
End and a second
Elizabet
Constant deficience C
••

To install OSGi applications we have to follow the same routine as for other enterprise applications, although this may look slightly unfamiliar if you have only ever used the 'WebSphere enterprise applications' shortcut which is provided as a convenience.

When using this WebSphere enterprise application install shortcut, you will find WAS;

- imports the application as an asset
- creates a business level application (BLA)
- deploys the asset to the BLA

and these are precisely the steps we follow when installing OSGi applications. We need to note however, that any bundles that are provisioning content to our application from the shard bundle space and are **not** contained within the EBA, **must** already be present in a bundle repository or our asset import will not resolve and a warning about the missing dependencies will be shown.

This resolution ensures we never receive a ClassNotFoundException from the application at runtime if our manifests have been correctly constructed.



Since other applications, deployed at a later date to the same servers, could contribute shared bundles whose Java packages influence the dependency resolution that occurs when the application is started, WebSphere generates a "deployment manifest" that "freeze dries" the deployment of an application.

This means that each time the application is restarted, the resolution process always calculates the same result regardless of other applications. Unlike the authored application manifest, the generated deployment manifest does contain the transitively closed content – the result of the deploy-time resolution.

The deployment manifest is the description of the deployed application. This can be exported from one deployment to another to ensure that an application moving from a Test to a Production system continues to resolve in exactly the same way it did during testing.



After importing our EBA as an asset, we can add it to either a new or an existing BLA by selecting the "Add Asset" option from the drop down in the Deployed Asset section.

On selecting this option, we will be displayed with a list of assets from which to choose.



After having selected to add our EBA asset to a BLA we progress through a configuration wizard where we can modify/set various aspects of our application.

The set of wizard steps displayed will depend on the content of your application and any configuration files contained with that application and bundles, web.xml, bindings and extension files.

Once an OSGi application has been successfully deployed then all its constituent bundles are pushed out to the appropriate target servers and the application can be administratively started. Starting the application causes its constituent bundles to go through the OSGi lifecycle states "installed", "resolved" and "active".

<u>NOTE</u>

- On single server topologies, bundles are pushed out to the config tree on 'save'.

- On network deployed topologies, bundles are pushed out to the relevant nodes config trees on 'sync node' after having performed a 'save'.



You can start/stop and explore all you applications from the Business level applications panel, you will also see BLAs for standard WebSphere enterprise applications installed via the shortcut previously discussed, as well as the newly created BLAs for you OSGi applications.



Exporting/Importing of a deployment manifest as previously discussed is done from the asset detail panel.

One of the other key functions available from this panel is updating of bundle versions in use by an application. By placing new versions of bundles in the IBR we are able to use a configuration wizard to wire our application into using these new bundles on an application by application basis.

This is particularly useful for:

- Testing and using new application functionality
- Applying security updates/fixes to particular exposures in a given bundle



On selecting the update option in the asset details panel, we are able to configure a custom configuration of bundle versions from those available.

On previewing this configuration a re-resolve of the entire application takes place to test whether the configuration is valid. (Note that this **does not** affect the running application)

If the configuration is not valid, we are not permitted to commit the changes and are informed of resolution problems.

If the configuration is valid we commit the change to disk. This writes out a new frozen deployment manifest to the config tree. This new configuration becomes active after the restart of the BLA to which the asset is deployed.



The BLA detail panel for a given compositional unit indicates whether a new deployment is available and a BLA restart required.



One of the significant features of the WebSphere OSGi Application feature pack is its introduction of the Blueprint component model. This is the result of the standardization activity around the Spring framework in the OSGi Alliance.

Spring provides a convenient way for business logic to be encapsulated into POJO components, which have all their dependencies injected into them by the Spring container. The Spring framework is a container that is packaged as a library with the application. In a Java EE environment the Spring container delegates to the underlying Application server for the management of resources such as database connections and for the application of qualities of service such as transactions and security. Spring is essentially a proxy container to the native web container provided by the Application server and can add a significant amount of pathlength.

By standardizing the Spring XML configuration format in the OSGi Alliance and delivering the container as an OSGi bundle, it has become possible to pull the dependency injection container out of the application and into the middleware. The standards-based evolution of the DI container is called the Blueprint container and the WebSphere OSGi feature pack integrates this container as part of the Application sever.

The Blueprint XML configuration file has the same structure as the Spring XML configuration file but in an OSGi namespace. The Blueprint XML is a bean definition file for all the beans provided by a single bundle. In addition to the bean definitions that will be familiar to Spring developers, the Blueprint model adds new service and reference elements as part of the integration with the OSGi environment. Service elements direct the Blueprint container to expose a service interface for a component outside the bundle and a reference element directs the Blueprint container to locate a service that can be consumed from outside the bundle. The yellow arrows in the figure indicate OSGi services that are published to and discovered from the OSGi Service Registry by the blueprint container. The OSGi service registry is a standard part of OSGi and provides a mechanism akin to JNDI for the publication of OSGi services, although the underlying use of the service registry is abstracted from application developers by the Blueprint container.

Ultimately, the Blueprint container manages the lifecycle and dependencies of the POJO beans that contain the application logic as well as the services and references each bundle provides, and ensures references are wired to available services.

Unit test for Blueprint components is simplified by the dependency injection pattern, which allows one bean to access another bean without having to implement any code to create the bean instance. The Blueprint Container creates the required bean instance, using information contained in the Blueprint configuration file. This eliminates compiled dependencies on either the OSGi Framework or the application server runtime environment.



Blueprint Components and Services

Top left, is a simple example of the blueprint.xml for an eCommerce bundle which contains a "shop" bean which uses a BillingService provided by another bundle. The Blueprint container locates the Billing Service provider and injects it into the shop bean at runtime.

OSGi services are dynamic so if the service provider can be changed then the blueprint container is able to dynamically rewire the reference to a new service without impacting the shop bean instance.

On the provider side, the BillingServiceImpl is another POJO implementing a Java interface for which a service is registered by the Blueprint container when the Billing bundle is started.

By default beans are created with "singleton" scope which means only a single instance is created by the container. If the bean maintains any state then it can be declared with "prototype" scope so that the container creates a new instance each time it needs to inject the dependency into a client.

Blueprint Transactions and Persistence

The WebSphere Blueprint container does a lot more than the spring framework for managing JPA contexts. It understands standard JPA annotations in the blueprint components it manages and will inject an EntityManagerFactory or EntityManager into annotated components or components whose bean definition contains a "jpa" element as illustrated in the example.

For the managed JPA case, the Blueprint container will manage the association between the EntityManager and the transaction context so the application does not have to. The WebSphere Blueprint container also provides full declarative transaction support using the same container-managed transaction attributes as EJBs.

You don't *need* to use Blueprint components in your OSGi applications just like you don't *need* Spring in Java EE but the Blueprint container model provides significant ease of use benefits to developers, is based on an OSGi industry standard, is well-integrated with the server runtime



The OSGi Service Registry is a standard part of OSGi and is where services are registered by service providers for consumption by other bundles.

Existing web components are not aware of the OSGi service registry and use JNDI for service lookups.

To enable existing Java EE mechanisms to interact with OSGi services, and vice versa, the Enterprise OSGi integration of JNDI defines a mechanism for OSGi services to be made available through JNDI and vice versa.

In the WebSphere OSGi Application feature pack, services published in the SR are available to JNDI clients using the osgi:service URL scheme for the lookup. This is the primary method by which web applications discover Blueprint services. Equally, administered resources bound to JNDI are also published as services in the OSGi SR with the JNDI name contained in a service property called osgi.jndi.service.name.



Examples of JNDI lookups


The APPLICATION MANIFEST here shows how isolated content is defined by the Application-Content header and how shared content is defined by the Use-Bundle header.

The figure describes one of the sample applications shipped with the OSGi Application feature pack. It is a web application that provides a Blog. It consists of:

- A web bundle to provide the user interface via standard servlets and dojo.
- A blueprint bundle containing 3 beans which encapsulate the business logic. The entry point is a Blogging Service which is accessed through JNDI by the web app.
- A persistence bundle containing a standard persistence.xml and entities representing the persistent data
- A database where blog entries and author information are read from and written to through JPA.

The sample can be used to learn how OSGi applications are deployed as a set of bundles to WAS and also illustrates the isolated and shared frameworks and the placement of bundles in each.



WebSphere also introduces the notion of a 'Composite Bundle Archive' (CBA). A CBA groups shared bundles together into aggregates. It can either directly contain OSGi bundles, or reference bundles that are hosted in the internal bundle repository.

You create a CBA when you want to ensure consistent behavior from a set of shared bundles and you can use the CBA to wire that set of bundles to an application.

A CBA is a zip archive with a .cba file extension. It contains a composite manifest, which is located at META-INF/COMPOSITEBUNDLE.MF which defines the CBA and optionally some OSGi bundles with which to seed the repository. The bundles that a CBA contains or references are defined with exact versions, in contrast to an EBA, where bundle can be defined with version ranges.

You install a composite bundle in the internal bundle repository. If the CBA directly contains OSGi bundles, these bundles are installed into the repository as though they are individually uploaded, and the CBA is also added to the bundle repository. If the CBA references OSGi bundles, these bundles must be present in the internal bundle repository.

After a CBA is installed in the internal bundle repository, its bundles are available to all applications that want to use them when the application is resolved. If a required package or service is available at the same version from both a bundle and a CBA, the provisioning process selects the package or service from the CBA.

	IBM
WAS v7 OSGi Summary	
• Isolation and Sharing	
Classpath Control	
Versioning Control	
• Integration of established technologies JPA/JNDI/JTA	
• Blueprint dependency injection – publish and consume servi	ces
• EBA	
• Bundle Repositories	
• CBA	
	Duane Appleby - applebyd@uk ibm.com
27	© 2011 IBM Corporation

So to summarize, we have seen how with the WAS v7 OSGi feature pack we can;

- Create 'OSGi applications'
- Exploit both isolation and sharing
- Ability to share common bundles outside of applications
- Deploy and use multiple versions of packages within an application
- Upgrade deployed applications
- Provide a cohesive locked down composite bundle configuration using CBAs
- Use dependency injection from the blueprint container
- Consume and publish services via blueprint to the OSGi service registry
- Use JNDI to look-up resources and services from the OSGi service registry



We will now cover the way this initial functionality has been extended and is trialled in the WAS v8 beta download available from the ibm website.



One point I <u>MUST</u> make clear, is that everything I am about to talk about refers to a beta product and there is no guarantee that this beta will become a shipped product or contain all the functionality we discuss herein.

As previously mentioned however, WAS ensures all feature packs are available on future WAS releases as either feature pack installables or as part of the base product. So all we have seen up to this point would be available.





One noticeably large improvement in the beta compared to the feature pack is the ability to post configure a deployed application.

Post configuration panels are accessed on the composition unit detail panel and we are now able to modify the properties set during deployment time; virtual host mappings, context roots, security settings etc...

When you modify your configuration, these changes take effect;

- on save for a single server topology
- on sync (after a save) on network deployed (nd) topologies.



One noticeably large improvement in the beta compared to the feature pack is the ability to post configure a deployed application.

Post configuration panels are accessed on the composition unit detail panel and we are now able to modify the properties set during deployment time; virtual host mappings, context roots, resource references, security settings etc...

When you modify your configuration, these changes take effect;

- on save for a single server topology
- on sync (after a save) on network deployed (nd) topologies.

	•	
	Duan	e Appleby - applebyd@uk.ibm.com @ 2011 IBM Corrocation

In-place update is an extension to capabilities in the v7 Feature Pack. In the feature pack, after changing the versions of bundles that composed an applications deployment, the entire application required a restart.

In the v8 Beta, after changing bundle versions (via the wizard available through the asset's detail panel), we can now navigate through our BLA to our deployed asset/composition unit and rather than seeing a notice informing us of a new deployment, we have the opportunity to press a button to move us to the latest deployment.

Rather than restarting the entire application, this button recycles only those bundles that are affected by changes to the deployment.

Note however that major changes which include package and service imports and exports may well trigger a restart of the entire application.

Configuration wizards may be displayed at this point depending on the nature of the update and the ability to modify/specify new configuration information such as virtual hosts or security bindings.

JNDI support for the 'blueprint:comp' namespace is added in the v8 beta which allows injection of blueprint components by id.

If we have a reference injected, rather than just using an 'osgi:service' lookup, our web applications are able to handle services coming and going during in-place bundle updates, rather than throwing ServiceUnavailableExceptions as we are given a damped reference.

WAS vo Bela.	Application Extensions		
	THE DECK A	•	

Application extensions are functional extensions to a given application and are implemented through the use of CBAs.

When we select the "Manage extensions for this composition unit" option from the composition unit detail panel, we can choose a CBA to add as an extension our application.

Application extensions, are similar to in-pace update (bundle version) changes, in that they can become active in-place, without application restarts.

Application extensions may be useful in situations where you have deployed a core application where you know you may want to use and remove different service providers.

For example, the core application is written to use various payment providers, and each of these providers is deployed as their own CBA providing their payment service capability through the OSGi service registry. The core application is written to be able to use a number of providers (via a reference list in its blueprint xml for example) thus allowing us to add or remove providers as needed.

Another example would be where your web design/skin is provided via some service mechanism and you can add and remove those skins as required ie different Christmas, Easter/ summer etc...



Composite bundle archives have undergone a quite significant rework between the WAS v7 feature pack and this Was v8 Beta.

In the v8 Beta the first significant change for CBAs is the aspect of content hiding. Where previously CBAs had their content explode out into the internal bundle repository (IBR) where it was possible for other applications to access said content, this is not the v8 Beta behavior.

CBAs in the Beta act as a more cohesive unit – as a set of bundles with a united purpose or as a strictly regulated set of dependencies.

It is now possible to specify a CBA on the Application-Content stanza of the EBAs application manifest – where previously we could only Use-Bundle the CBA or Import-Package from a bundle in the CBA and depend on the resolution preference of CBAs to resolve correctly. By specifying the CBA on Application-Content we bring the CBA out of the shared framework and into our applications isolated framework.

In addition to these benefits, you can now also include WABs in CBAs, as you will be presented with configuration options during deployment of you asset for the WABs contained within the CBAs. This is also a useful benefit for application extensions as just mentioned.

Other points to note are that when exporting an EBA asset in the Beta, the binary product now also contains the requisite CBA content and that bulk bundle uploads to the IBR are still possible by providing a zip (.zip) file of bundles to the IBR upload dialog, or respective wsadmin command.

WAS v8 Beta :	Map RunAs roles to users	
and the second s	THAT IS NOT	
		>
		Duane Appleby - applebyd@uk.ibm.com

Support for run as roles is now included during deployment and as a post deployment configuration change option.

This enables you to specify OSGi application-specific privileges for individual users to run specific tasks using another user identity in precisely the same manner as for enterprise applications.



Again, bringing OSGi application options inline with enterprise applications, we can now override session management/replication options from the topology defaults. To do this we again navigate to the composition unit detail panel and select the "Session management" link.

From this link we can select to override set options with our own custom settings.

As we have previously discussed, artifacts in the internal bundle repository and downloaded into a bundle cache, a runtime store if you will, of bundles in use by installed assets (EBAs).

In the feature pack on WAS v7 these bundles, once downloaded, resided permanently in the bundle cache unless manually cleared using scripting and the bundle cache manager Mbean. In the WAS v8 Beta we provide an admin console interface for controlling the contents of the bundle cache, as well as the Mbean should you wish to use scripting.

Further to this new panel interface is the inclusion of a bundle cache garbage collection. When all assets (EBAs) that were using a bundle in the bundle cache and deleted, then the bundle cache garbage collector removes this bundle from the runtime cache. It will still exist in the internal bundle repository as this is a administratively configured store, not a runtime location.











The SCA feature pack for WAS v7 and subsequent integration into the WAS v8 Beta allows a further level of assembly for OSGi applications. With SCA we can assemble OSGi applications into an SCA composite to provide an SOA abstraction.

Within an SCA composite the OSGi Application is a component that can be wired to other components with different implementation types. For example, an SCA composite could contain an osgi-application component, a JEE component containing EJBs, a BPEL component and so on.

Each component within an SCA composite declares abstract services and references to which concrete bindings can be applied and it is through these services and references that the components of an SCA's composites are wired together.

The OSGi Application architecture was designed with this form of assembly in mind so that the services and references declared in a Blueprint XML configuration can be exposed through the Application manifest to be visible outside the application. Such exposed services and references can then be mapped to SCA services and references with the full range of available SCA bindings applied to them.

This enables OSGi applications to participate in two new scenarios:

1) Assembly into heterogeneous composites of OSGi and non-OSGi components

2) Remoting of OSGi application services through SCA services with a variety of bindings including JMS, SOAP/HTTP, IIOP and JSON-RPC.





Tooling support for OSGi Applications is available in RAD v8.0.2.

The new tooling is structured so that server-independent development and assembly tooling can be installed as a plugin into any Eclipse WTP 3.6 environment. While this is pre-integrated in RAD, the availability of the new tools in Eclipse WTP configurations other than RAD, better enables these common tools to be used to develop OSGi Applications for deployment to Geronimo and, in the future, other non-WAS servers that integrate the Apache Aries runtime components.

The common development tooling includes new project type for OSGi Applications, the ability to import and export .eba archives, form-based editors for bundle manifests, application manifests and Blueprint configuration files as well as tutorials and documentation.

Additionally integrated into the RAD are WebSphere deployment tools, a WAS v8 Beta test environment, enhanced validation tools as well as integration with Web and JEE productivity tools.



.

IBM

Further Information

Resource Hub (articles, tutorials, redbooks, forums) http://www.ibm.com/software/websphere/osgi

Team Blogs/Twitter

www.ianrobinson.blogspot.com

www.devangelist.blogspot.com

@sjmaple @notatibm @TimothyWard

Email applebyd@uk.ibm.com

47

Duane Appleby - applebyd@uk.ibm.com © 2011 IBM Corporation



NO FURTHER SLIDES