



OSGiTM
Alliance

RFC 195 Service Scopes

Draft

69 Pages

Abstract

Add prototype service scope to OSGi Service Layer.

0 Document Information

0.1 License

DISTRIBUTION AND FEEDBACK LICENSE, Version 2.0

The OSGi Alliance hereby grants you a limited copyright license to copy and display this document (the “Distribution”) in any medium without fee or royalty. This Distribution license is exclusively for the purpose of reviewing and providing feedback to the OSGi Alliance. You agree not to modify the Distribution in any way and further agree to not participate in any way in the making of derivative works thereof, other than as a necessary result of reviewing and providing feedback to the Distribution. You also agree to cause this notice, along with the accompanying consent, to be included on all copies (or portions thereof) of the Distribution. The OSGi Alliance also grants you a perpetual, non-exclusive, worldwide, fully paid-up, royalty free, limited license (without the right to sublicense) under any applicable copyrights, to create and/or distribute an implementation of the Distribution that: (i) fully implements the Distribution including all its required interfaces and functionality; (ii) does not modify, subset, superset or otherwise extend the OSGi Name Space, or include any public or protected packages, classes, Java interfaces, fields or methods within the OSGi Name Space other than those required and authorized by the Distribution. An implementation that does not satisfy limitations (i)-(ii) is not considered an implementation of the Distribution, does not receive the benefits of this license, and must not be described as an implementation of the Distribution. “OSGi Name Space” shall mean the public class or interface declarations whose names begin with “org.osgi” or any recognized successors or replacements thereof. The OSGi Alliance expressly reserves all rights not granted pursuant to these limited copyright licenses including termination of the license at will at any time.

EXCEPT FOR THE LIMITED COPYRIGHT LICENSES GRANTED ABOVE, THE OSGi ALLIANCE DOES NOT GRANT, EITHER EXPRESSLY OR IMPLIEDLY, A LICENSE TO ANY INTELLECTUAL PROPERTY IT, OR ANY THIRD PARTIES, OWN OR CONTROL. Title to the copyright in the Distribution will at all times remain with the OSGi Alliance. The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted therein are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

THE DISTRIBUTION IS PROVIDED “AS IS,” AND THE OSGi ALLIANCE (INCLUDING ANY THIRD PARTIES THAT HAVE CONTRIBUTED TO THE DISTRIBUTION) MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NON-INFRINGEMENT, OR TITLE; THAT THE CONTENTS OF THE DISTRIBUTION ARE SUITABLE FOR ANY PURPOSE; NOR THAT THE IMPLEMENTATION OF SUCH CONTENTS WILL NOT INFRINGE ANY THIRD PARTY PATENTS, COPYRIGHTS, TRADEMARKS OR OTHER RIGHTS.

NEITHER THE OSGi ALLIANCE NOR ANY THIRD PARTY WILL BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF OR RELATING TO ANY USE OR DISTRIBUTION OF THE DISTRIBUTION.

Implementation of certain elements of this Distribution may be subject to third party intellectual property rights, including without limitation, patent rights (such a third party may or may not be a member of the OSGi Alliance). The OSGi Alliance is not responsible and shall not be held responsible in any manner for identifying or failing to identify any or all such third party intellectual property rights.

The Distribution is a draft. As a result, the final product may change substantially by the time of final publication, and you are cautioned against relying on the content of this Distribution. You are encouraged to update any implementation of the Distribution if and when such Distribution becomes a final specification.

The OSGi Alliance is willing to receive input, suggestions and other feedback (“Feedback”) on the Distribution. By providing such Feedback to the OSGi Alliance, you grant to the OSGi Alliance and all its Members a non-exclusive, non-transferable,

worldwide, perpetual, irrevocable, royalty-free copyright license to copy, publish, license, modify, sublicense or otherwise distribute and exploit your Feedback for any purpose. Likewise, if incorporation of your Feedback would cause an implementation of the Distribution, including as it may be modified, amended, or published at any point in the future ("Future Specification"), to necessarily infringe a patent or patent application that you own or control, you hereby commit to grant to all implementers of such Distribution or Future Specification an irrevocable, worldwide, sublicenseable, royalty free license under such patent or patent application to make, have made, use, sell, offer for sale, import and export products or services that implement such Distribution or Future Specification. You warrant that (a) to the best of your knowledge you have the right to provide this Feedback, and if you are providing Feedback on behalf of a company, you have the rights to provide Feedback on behalf of your company; (b) the Feedback is not confidential to you and does not violate the copyright or trade secret interests of another; and (c) to the best of your knowledge, use of the Feedback would not cause an implementation of the Distribution or a Future Specification to necessarily infringe any third-party patent or patent application known to you. You also acknowledge that the OSGi Alliance is not required to incorporate your Feedback into any version of the Distribution or a Future Specification.

I HEREBY ACKNOWLEDGE AND AGREE TO THE TERMS AND CONDITIONS DELINEATED ABOVE.

0.2 Trademarks

OSGi™ is a trademark, registered trademark, or service mark of the OSGi Alliance in the US and other countries. Java is a trademark, registered trademark, or service mark of Oracle Corporation in the US and other countries. All other trademarks, registered trademarks, or service marks used in this document are the property of their respective owners and are hereby recognized.

0.3 Feedback

This document can be downloaded from the OSGi Alliance design repository at <https://github.com/osgi/design> The public can provide feedback about this document by opening a bug at <https://www.osgi.org/bugzilla/>.

0.4 Table of Contents

0 Document Information.....	2
0.1 License.....	2
0.2 Trademarks.....	3
0.3 Feedback.....	3
0.4 Table of Contents.....	3
0.5 Terminology and Document Conventions.....	4
0.6 Revision History.....	4
1 Introduction.....	4
2 Application Domain.....	5
3 Problem Description.....	5
4 Requirements.....	6
5 Technical Solution.....	6
5.1 Scope.....	6
5.2 Service API changes.....	7
5.3 Declarative Services.....	8
5.3.1 Providing Services.....	8
5.3.2 Consuming Services.....	8
5.3.3 Annotations.....	8
5.3.4 Schema.....	8
5.4 Blueprint.....	8

6 Data Transfer Objects.....	9
7 Javadoc.....	9
8 Considered Alternatives.....	68
8.1 Parameterization.....	68
9 Security Considerations.....	69
10 Document Support.....	69
10.1 References.....	69
10.2 Author's Address.....	69
10.3 Acronyms and Abbreviations.....	69
10.4 End of Document.....	69

0.5 Terminology and Document Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY" and "OPTIONAL" in this document are to be interpreted as described in 10.1.

Source code is shown in this typeface.

0.6 Revision History

The last named individual in this history is currently responsible for this document.

Revision	Date	Comments
Initial	14 Nov 2012	Initial draft. Started from a discussion at the Orlando F2F.
2 nd draft	17 Jan 2013	Updated after CPEG call. Added new DS bind/updated/unbind method signature. Added DS annotation changes.

1 Introduction

The OSGi Service Layer has been part of the OSGi Core spec since Release 1. It provides a service broker model where bundles can publish, find and bind services. The service layer has always allowed a service provider to provide either a singleton service object shared by all consumers or to provide a unique service object per

bundle consuming the services. This RFC will introduce the concept of a service scope and define a new scope type to allow there to be many service objects for a consuming bundle.

2 Application Domain

The OSGi service layer allows bundles to provide services by publishing them in the service registry. It also allows bundles to find services by performing a lookup in the service registry and by listening to service lifecycle events. A bundle can consume a service by binding to the service thus obtaining a service object that can be called.

The provider has 2 ways to provide a service object to which a consumer can bind. The provider can directly register a service object. This one service objects is then available for use by all consumers. Alternatively, the service provider can register an object implementing the ServiceFactory interface. This ServiceFactory object can then be called by the OSGi framework implementation each time a consuming bundle binds to the service. This allows the providing bundle to create a unique service object for each bundle consuming the service. The framework ensures that the ServiceFactory is only called once for each consuming bundle and that the consuming bundle is only bound to a single object. When a consuming bundle releases the bound service, the framework will again call the ServiceFactory object to release the unique service object created for that consuming bundle.

3 Problem Description

Sometimes it is necessary to for a consuming bundle to have access for more than a single instance of a service. This may be necessary if the service is stateful and different parts of the bundle need services having different state.

The discussion of how to support stateful EJBs in RFC 194 lead to the ideas which spawned this RFC. Since the EJB support will need to inject EJBs into client code, to support all inter-bundle “communication” being done via OSGi services, there needs to be a way to obtain multiple instances of a service for a single consuming bundle. This is currently not possible with the OSGi service layer API.

The EJB implementation could use a bland factory-type (e.g. EJBFactory) but this would side step the type-safety support in the OSGi service registry by obscuring the actual types from the service registry as they would need to be in some agreed service property.

Multiple instance support can also be important for Remote Services Admin implementations. This will allow an implementation bundle to obtain multiple instances of a local service to support multiple remote consumers.

4 Requirements

Some of the following requirements are derived from the requirements in RFC 158 [3].

- S0001 – The Framework must provide a mechanism that allows a provider bundle to register a service that enables a consumer bundle to obtain multiple instances of the service.
- S0002 – The instance creation method should use a different API from the normal `getService` to minimize confusion
- S0003 – The mechanism must be implemented within the existing concepts of service lifecycle, `ServiceReference`, `ServiceRegistration`, `ServiceListener` and service hooks.
- S0004 – Clients must be able to release any instances when the provider unregisters the service. Services must conform to normal service lifecycle rules. Therefore, instances must follow the same life cycle as the service registration.
- S0005 – The existing way of providing and consumer services must remain possible.
- S0006 – The scope type of the service must be introspectable. That is, a potential consumer of a service must be able to tell if it can bind to multiple instances of a service.
- S0007 – Existing consumers must be able to consume services, using expected semantics, provided by providers supporting multiple instances.
- S0008 – New consumers must be able to consume service provided by existing providers.

5 Technical Solution

5.1 Scope

We introduce a new term to the specification: *service scope*. Without explicitly using this term, the current service layer allows for two scopes: *singleton* and *bundle*. This RFC also introduces a third scope: *prototype*. (The prototype name is “inherited” from the Blueprint specification. See 121.5.5.)

When a bundle registers a plain object as the service object, we now call this singleton scope. All consumers of the service use the same service object.

When a bundle registers a `ServiceFactory` object as the service object, we now call this bundle scope. Each consuming bundle uses a unique service object. But there is only a single service object per consuming bundles.

We introduce a new subtype of `ServiceFactory` called `PrototypeServiceFactory` such that when a bundle registers a `PrototypeServiceFactory` object as the service object, we now call this prototype scope. A consuming bundle which is aware of the newly introduced `BundleContext.getServiceObjects` method, can now obtain multiple instances of the service.

Finally, we also introduce a new service property called `service.scope`. Like `service.id`, this property is automatically added by the framework to all service registrations and set to the scope of the service. This allows new consumers to locate prototype scope services and properly interact with them. This property will be especially important for component models like Declarative Services and Blueprint since they will need to know they can properly obtain multiple instances for the declared components.

5.2 Service API changes

A new `ServiceFactory` subtype is introduced called *PrototypeServiceFactory*. Implementing this interface and registering it as the service object tell the framework, that the service provider is capable of creating multiple instances of the service for a single consuming bundle.

It is necessary to define a new “factory” type rather than simply calling existing `ServiceFactory` implementations to create multiple instances of a service for a given bundle. Since the `ServiceFactory` contract states that it will only be called to create a single instance (at a time) for a given bundle, `ServiceFactory` implementations may reply upon this. For example, the consuming bundle could be a key in a map of bundle to service instance. Calling this `ServiceFactory` multiple times for a given bundle would break the implementation.

So implementing the new `PrototypeServiceFactory` type indicates that the providing bundle is aware that the factory can be called multiple times per consuming bundle.

On the consumer side, we also need a means for the consumer to consume multiple instances. The current `BundleContext` methods `getService` and `ungetService` must retain their current behavior. Using these methods, a consuming bundle will only ever be exposed to a single service instance (at a time). We introduce a new method to `BundleContext` `getServiceObjects` which returns the newly introduced *ServiceObjects*<S> type rather than a service S.

```
<S> ServiceObjects<S> getServiceObjects(ServiceReference<S> reference)
```

The *ServiceObjects* type contains the simple `S getService()` and `void ungetService(S service)` methods. If the service scope is singleton or bundle, calling these methods can only return the single (at a time) service object the provider is able to provide for the bundle. However, if the scope is prototype, then each call to `getService` will return a new service object.

The lifecycle of service objects of prototype scope is the same as the other scopes. When the consuming bundle is stopped, then all service objects obtained by the bundle must be released. If the provider bundle unregisters the service, then all service objects obtained by any bundle must be released. This means the framework must track all consumed service objects so they may be released when necessary.

5.3 Declarative Services

Declarative Services is currently being updated by RFC 190. The introduction of prototype scope services means we also need to update DS to support this new service feature.

5.3.1 Providing Services

The servicefactory attribute on the service element is deprecated and replaced by a scope attribute supporting the values: singleton (default), bundle and prototype. servicefactory=false maps to scope=singleton and servicefactory=true maps to scope=bundle.

This allows SCR to support components being prototype scope services. Since DS never registers the actual component object (that is, even for scope=singleton, DS always registers a ServiceFactory to delay component creation and activation), components will never be visible in the service registry with service.scope=singleton.

5.3.2 Consuming Services

A scope attribute is added to the reference element. The scope attribute supports the values: bundle(default) and prototype. When using scope=bundle, all references to the service by components in the same bundle will share the same service object. That is, SCR must use BundleContext.getService to obtain the service object. When using scope=prototype, each instance of the component will use a difference instance of the service. That is, SCR must use BundleContext.getServiceObjects to obtain the service object and the referenced service must have service.scope=prototype. A service without service.scope=prototype cannot be used as a bound service for a scope=prototype reference since it cannot fulfill the requirement to create multiple service instances for the bundle.

The valid signatures for bind, updated and unbind will be extended to allow ServiceObjects to be injected.

```
void <method-name>(ServiceObjects);
```

This method signature can only be used when the reference is scope=prototype.

5.3.3 Annotations

The DS Annotations will also be updated to support these new features.

Enum ServiceScope is added with values SINGLETON, BUNDLE and PROTOTYPE.

ServiceScope Component.scope() is added. Component.servicefactory() is deprecated and ignored when Component.scope() is specified.

Enum ReferenceScope is added with values BUNDLE and PROTOTYPE.

ReferenceScope Reference.scope() is added.

5.3.4 Schema

The DS XML Schema is updated to v1.3.0 and a scope attribute is added to the service and the reference elements. The servicefactory attribute of the service element is removed since it is replaced by the new scope attribute.

5.4 Blueprint

Blueprint is currently being updated by RFC 184. The introduction of prototype scope services means we also need to update Blueprint to support this new service feature.

Similar changes to those proposed for DS are needed. Design TBD.

6 Data Transfer Objects

No DTOs changes are required. The `ServiceReferenceDTO` proposed by RFC 185 will be sufficient for this design.

7 Javadoc

A subset of the `org.osgi.framework` javadoc is included which contains the main API changes for this RFC.

OSGi Javadoc
1/18/13 11:55 AM

Package Summary		Page
org.osgi.framework	Framework Package Version 1.8.	Error: Reference source not found
org.osgi.service.component.annotations	Service Component Annotations Package Version 1.3.	Error: Reference source not found

Package org.osgi.framework

Framework Package Version 1.8.

See:

[Description](#)

Interface Summary		Page
BundleContext	A bundle's execution context within the Framework.	Error: Reference source not found
Constants	Defines standard names for the OSGi environment system properties, service properties, and Manifest header attribute keys.	Error: Reference source not found
PrototypeServiceFactory	A factory for prototype_scope services.	Error: Reference source not found
ServiceFactory	A factory for bundle_scope services.	Error: Reference source not found
ServiceObjects	Allows multiple service objects for a service to be obtained.	Error: Reference source not found

Package org.osgi.framework Description
Framework Package Version 1.8.

Bundles wishing to use this package must list the package in the Import-Package header of the bundle's manifest.

Example import for consumers using the API in this package:

```
Import-Package: org.osgi.framework; version="[1.8,2.0) "
```

Interface BundleContext

[org.osgi.framework](#)

All Superinterfaces:

[org.osgi.framework.BundleReference](#)

```
public interface BundleContext
    extends org.osgi.framework.BundleReference
```

A bundle's execution context within the Framework. The context is used to grant access to other methods so that this bundle can interact with the Framework.

BundleContext methods allow a bundle to:

- Subscribe to events published by the Framework.
- Register service objects with the Framework service registry.
- Retrieve [ServiceReferences](#) from the Framework service registry.
- Get and release service objects for a referenced service.
- Install new bundles in the Framework.
- Get the list of bundles installed in the Framework.
- Get the [org.osgi.framework.Bundle](#) object for a bundle.
- Create [File](#) objects for files in a persistent storage area provided for the bundle by the Framework.

A [BundleContext](#) object will be created for a bundle when the bundle is started. The [Bundle](#) object associated with a [BundleContext](#) object is called the *context bundle*.

The [BundleContext](#) object will be passed to the [org.osgi.framework.BundleActivator.start\(BundleContext\)](#) method during activation of the context bundle. The same [BundleContext](#) object will be passed to the [org.osgi.framework.BundleActivator.stop\(BundleContext\)](#) method when the context bundle is stopped. A [BundleContext](#) object is generally for the private use of its associated bundle and is not meant to be shared with other bundles in the OSGi environment.

The [BundleContext](#) object is only valid during the execution of its context bundle; that is, during the period from when the context bundle is in the [STARTING](#), [STOPPING](#), and [ACTIVE](#) bundle states. If the [BundleContext](#) object is used subsequently, an [IllegalStateException](#) must be thrown. The [BundleContext](#) object must never be reused after its context bundle is stopped.

Two [BundleContext](#) objects are equal if they both refer to the same execution context of a bundle. The Framework is the only entity that can create [BundleContext](#) objects and they are only valid within the Framework that created them.

A [org.osgi.framework.Bundle](#) can be adapted to its [BundleContext](#). In order for this to succeed, the caller must have the appropriate [AdminPermission\[bundle,CONTEXT\]](#) if the Java Runtime Environment supports permissions. Version:

\$Id: d780fe8b65e03499b1a95afe60347f2a224de7d5 \$

ThreadSafe

Method Summary		Page
void addBundleListener (org.osgi.framework.BundleListener listener)	Adds the specified BundleListener object to the context bundle's list of listeners if not already present.	Error: Reference source not found
void addFrameworkListener (org.osgi.framework.FrameworkListener listener)	Adds the specified FrameworkListener object to the context bundle's list of listeners if not already present.	Error: Reference source not found
void addServiceListener (org.osgi.framework.ServiceListener listener)	Adds the specified ServiceListener object to the context bundle's list of listeners.	Error: Reference source not found
void addServiceListener (org.osgi.framework.ServiceListener listener, String filter)	Adds the specified ServiceListener object with the specified filter to the context bundle's list of listeners.	Error: Reference source not found

Interface BundleContext

org.osgi.framework.Filter	createFilter (String filter) Creates a Filter object.	Error: Reference source not found
org.osgi.framework.ServiceReference<?>[]	getAllServiceReferences (String clazz, String filter) Returns an array of ServiceReference objects.	Error: Reference source not found
org.osgi.framework.Bundle	getBundle () Returns the Bundle object associated with this BundleContext.	Error: Reference source not found
org.osgi.framework.Bundle	getBundle (String location) Returns the bundle with the specified location.	Error: Reference source not found
org.osgi.framework.Bundle	getBundle (long id) Returns the bundle with the specified identifier.	Error: Reference source not found
org.osgi.framework.Bundle[]	getBundles () Returns a list of all installed bundles.	Error: Reference source not found
File	getDataFile (String filename) Creates a File object for a file in the persistent storage area provided for the bundle by the Framework.	Error: Reference source not found
String	getProperty (String key) Returns the value of the specified property.	Error: Reference source not found
ServiceReference<S>	getService (org.osgi.framework.ServiceReference<S> reference) Returns the service object for the service referenced by the specified ServiceReference object.	Error: Reference source not found
ServiceObjects<S>	getServiceObjects (org.osgi.framework.ServiceReference<S> reference) Returns the ServiceObjects object for the service referenced by the specified ServiceReference object.	Error: Reference source not found
org.osgi.framework.ServiceReference<S>	getServiceReference (Class<S> clazz) Returns a ServiceReference object for a service that implements and was registered under the name of the specified class.	Error: Reference source not found

org.osgi.framework.ServiceReference<?>	getServiceReference (String clazz) Returns a ServiceReference object for a service that implements and was registered under the specified class.	Error: Reference source not found
Collection<org.osgi.framework.ServiceReference<S>>	getServiceReferences (Class<S> clazz, String filter) Returns a collection of ServiceReference objects.	Error: Reference source not found
org.osgi.framework.ServiceReference<?>[]	getServiceReferences (String clazz, String filter) Returns an array of ServiceReference objects.	Error: Reference source not found
org.osgi.framework.Bundle	installBundle (String location) Installs a bundle from the specified location identifier.	Error: Reference source not found
org.osgi.framework.Bundle	installBundle (String location, InputStream input) Installs a bundle from the specified InputStream object.	Error: Reference source not found
org.osgi.framework.ServiceRegistration<S>	registerService (Class<S> clazz, S service, Dictionary<String,?> properties) Registers the specified service object with the specified properties under the name of the specified class with the Framework.	Error: Reference source not found
org.osgi.framework.ServiceRegistration<?>	registerService (String clazz, Object service, Dictionary<String,?> properties) Registers the specified service object with the specified properties under the specified class name with the Framework.	Error: Reference source not found
org.osgi.framework.ServiceRegistration<?>	registerService (String[] clazzes, Object service, Dictionary<String,?> properties) Registers the specified service object with the specified properties under the specified class names into the Framework.	Error: Reference source not found
void	removeBundleListener (org.osgi.framework.BundleListener listener) Removes the specified BundleListener object from the context bundle's list of listeners.	Error: Reference source not found
void	removeFrameworkListener (org.osgi.framework.FrameworkListener listener) Removes the specified FrameworkListener object from the context bundle's list of listeners.	Error: Reference source not found
void	removeServiceListener (org.osgi.framework.ServiceListener listener) Removes the specified ServiceListener object from the context bundle's list of listeners.	Error: Reference source not found

<code>boolean</code>	<code>ungetService</code> (<code>org.osgi.framework.ServiceReference<?> reference</code>) Releases the service object for the service referenced by the specified <code>ServiceReference</code> object.	Error: Reference source not found
----------------------	--	-----------------------------------

Method Detail

getProperty

`String getProperty(String key)`
Returns the value of the specified property. If the key is not found in the Framework properties, the system properties are then searched. The method returns `null` if the property is not found.
All bundles must have permission to read properties whose names start with "org.osgi."
Parameters:
 `key` - The name of the requested property.
Returns:
 The value of the requested property, or `null` if the property is undefined.
Throws:
 `SecurityException` - If the caller does not have the appropriate `PropertyPermission` to read the property, and the Java Runtime Environment supports permissions.

`org.osgi.framework.Bundle getBundle()`
Returns the `Bundle` object associated with this `BundleContext`. This bundle is called the context bundle.
Specified by:
 `getBundle` in interface `org.osgi.framework.BundleReference`
Returns:
 The `Bundle` object associated with this `BundleContext`.
Throws:
 `IllegalStateException` - If this `BundleContext` is no longer valid.

`org.osgi.framework.Bundle installBundle(String location, InputStream input)`
throws `org.osgi.framework.BundleException`
Installs a bundle from the specified `InputStream` object.
If the specified `InputStream` is `null`, the Framework must create the `InputStream` from which to read the bundle by interpreting, in an implementation dependent manner, the specified `location`.
The specified `location` identifier will be used as the identity of the bundle. Every installed bundle is uniquely identified by its location identifier which is typically in the form of a URL.
The following steps are required to install a bundle:
 1. If a bundle containing the same location identifier is already installed, the `Bundle` object for that bundle is returned.
 2. The bundle's content is read from the input stream. If this fails, a `org.osgi.framework.BundleException` is thrown.
 3. The bundle's associated resources are allocated. The associated resources minimally consist of a unique identifier and a persistent storage area if the platform has file system support. If this step fails, a `BundleException` is thrown.
 4. The bundle's state is set to `INSTALLED`.
 5. A bundle event of type `org.osgi.framework.BundleEvent.INSTALLED` is fired.
 6. The `Bundle` object for the newly or previously installed bundle is returned.

Postconditions, no exceptions thrown
 `getState()` in { `INSTALLED`, `RESOLVED` }.
 Bundle has a unique ID.
Postconditions, when an exception is thrown
 Bundle is not installed. If there was an existing bundle for the specified location, then that bundle must still be in the state it was prior to calling this method.
Parameters:
 `location` - The location identifier of the bundle to install.
 `input` - The `InputStream` object from which this bundle will be read or `null` to indicate the Framework must create the input stream from the specified location identifier. The input stream must always be closed when this method completes, even if an exception is thrown.
Returns:

The `Bundle` object of the installed bundle.

Throws:

`org.osgi.framework.BundleException` - If the installation failed. `BundleException` types thrown by this method include: `org.osgi.framework.BundleException.READ_ERROR`, `org.osgi.framework.BundleException.DUPLICATE_BUNDLE_ERROR`, `org.osgi.framework.BundleException.MANIFEST_ERROR`, and `org.osgi.framework.BundleException.REJECTED_BY_HOOK`.

`SecurityException` - If the caller does not have the appropriate `AdminPermission[installed bundle,LIFECYCLE]`, and the Java Runtime Environment supports permissions.

`IllegalStateException` - If this `BundleContext` is no longer valid.

installBundle

```
org.osgi.framework.Bundle installBundle(String location)
    throws org.osgi.framework.BundleException
```

Installs a bundle from the specified location identifier.

This method performs the same function as calling [installBundle\(String,InputStream\)](#) with the specified location identifier and a null `InputStream`.

Parameters:

location - The location identifier of the bundle to install.

Returns:

The `Bundle` object of the installed bundle.

Throws:

`org.osgi.framework.BundleException` - If the installation failed. `BundleException` types thrown by this method include: `org.osgi.framework.BundleException.READ_ERROR`, `org.osgi.framework.BundleException.DUPLICATE_BUNDLE_ERROR`, `org.osgi.framework.BundleException.MANIFEST_ERROR`, and `org.osgi.framework.BundleException.REJECTED_BY_HOOK`.

`SecurityException` - If the caller does not have the appropriate `AdminPermission[installed bundle,LIFECYCLE]`, and the Java Runtime Environment supports permissions.

`IllegalStateException` - If this `BundleContext` is no longer valid.

See Also:

[getBundle](#) [installBundle\(String,InputStream\)](#)

```
org.osgi.framework.Bundle getBundle(long id)
```

Returns the bundle with the specified identifier.

Parameters:

id - The identifier of the bundle to retrieve.

Returns:

A `Bundle` object or `null` if the identifier does not match any installed bundle.

getBundles

```
org.osgi.framework.Bundle[] getBundles()
```

Returns a list of all installed bundles.

This method returns a list of all bundles installed in the OSGi environment at the time of the call to this method. However, since the Framework is a very dynamic environment, bundles can be installed or uninstalled at anytime.

Returns:

An array of `Bundle` objects, one object per installed bundle.

addServiceListener

```
void addServiceListener(org.osgi.framework.ServiceListener listener,
    String filter)
    throws org.osgi.framework.InvalidSyntaxException
```

Adds the specified `ServiceListener` object with the specified filter to the context bundle's list of listeners. See `org.osgi.framework.Filter` for a description of the filter syntax. `ServiceListener` objects are notified when a service has a lifecycle state change.

If the context bundle's list of listeners already contains a listener `l` such that (`l==listener`), then this method replaces that listener's filter (which may be `null`) with the specified one (which may be `null`).

The listener is called if the filter criteria is met. To filter based upon the class of the service, the filter should reference the [Constants.OBJECTCLASS](#) property. If filter is `null`, all services are considered to match the filter.

When using a filter, it is possible that the `ServiceEvent` `s` for the complete lifecycle of a service will not be delivered to the listener. For example, if the filter only matches when the property `x` has the value `1`, the listener will not be called if the service is registered with the property `x` not set to the value `1`. Subsequently, when the service is modified setting property `x` to the value `1`, the filter will match and the listener will be called with a `ServiceEvent` of type `MODIFIED`. Thus, the listener will not be called with a `ServiceEvent` of type `REGISTERED`.

If the Java Runtime Environment supports permissions, the `ServiceListener` object will be notified of a service event only if the bundle that is registering it has the `ServicePermission` to get the service using at least one of the named classes the service was registered under.

Parameters:

`listener` - The `ServiceListener` object to be added.

`filter` - The filter criteria.

Throws:

`org.osgi.framework.InvalidSyntaxException` - If `filter` contains an invalid filter string that cannot be parsed.

`IllegalStateException` - If this `BundleContext` is no longer valid.

See Also:

`org.osgi.framework.ServiceEvent`, `org.osgi.framework.ServiceListener`,

`org.osgi.framework.ServicePermission`

~~addServiceListener~~

`void addServiceListener(org.osgi.framework.ServiceListener listener)`

Adds the specified `ServiceListener` object to the context bundle's list of listeners.

This method is the same as calling `BundleContext.addServiceListener(ServiceListener listener, String filter)` with `filter` set to `null`.

Parameters:

`listener` - The `ServiceListener` object to be added.

Throws:

`IllegalStateException` - If this `BundleContext` is no longer valid.

See Also:

[`addServiceListener\(ServiceListener, String\)`](#)

~~removeServiceListener~~

`void removeServiceListener(org.osgi.framework.ServiceListener listener)`

Removes the specified `ServiceListener` object from the context bundle's list of listeners.

If `listener` is not contained in this context bundle's list of listeners, this method does nothing.

Parameters:

`listener` - The `ServiceListener` to be removed.

Throws:

`IllegalStateException` - If this `BundleContext` is no longer valid.

~~addBundleListener~~

`void addBundleListener(org.osgi.framework.BundleListener listener)`

Adds the specified `BundleListener` object to the context bundle's list of listeners if not already present.

`BundleListener` objects are notified when a bundle has a lifecycle state change.

If the context bundle's list of listeners already contains a listener `l` such that `(l==listener)`, this method does nothing.

Parameters:

`listener` - The `BundleListener` to be added.

Throws:

`IllegalStateException` - If this `BundleContext` is no longer valid.

`SecurityException` - If `listener` is a `SynchronousBundleListener` and the caller does not have the appropriate `AdminPermission[context bundle, LISTENER]`, and the Java Runtime Environment supports permissions.

See Also:

`org.osgi.framework.BundleEvent`, `org.osgi.framework.BundleListener`

~~removeBundleListener~~

`void removeBundleListener(org.osgi.framework.BundleListener listener)`

Removes the specified `BundleListener` object from the context bundle's list of listeners.

If `listener` is not contained in the context bundle's list of listeners, this method does nothing.

Parameters:

`listener` - The `BundleListener` object to be removed.

Throws:

`IllegalStateException` - If this `BundleContext` is no longer valid.

`SecurityException` - If `listener` is a `SynchronousBundleListener` and the caller does not have the appropriate `AdminPermission[context bundle, LISTENER]`, and the Java Runtime Environment supports permissions.

~~addFrameworkListener~~

`void addFrameworkListener(org.osgi.framework.FrameworkListener listener)`

Adds the specified `FrameworkListener` object to the context bundle's list of listeners if not already present.

`FrameworkListeners` are notified of general Framework events.

If the context bundle's list of listeners already contains a listener `l` such that `(l==listener)`, this method does nothing.

Parameters:

`listener` - The `FrameworkListener` object to be added.

Throws:

`IllegalStateException` - If this `BundleContext` is no longer valid.

See Also:

`org.osgi.framework.FrameworkEvent`, `org.osgi.framework.FrameworkListener`
`removeFrameworkListener(org.osgi.framework.FrameworkListener listener)`

Removes the specified `FrameworkListener` object from the context bundle's list of listeners.

If `listener` is not contained in the context bundle's list of listeners, this method does nothing.

Parameters:

`listener` - The `FrameworkListener` object to be removed.

Throws:

`IllegalStateException` - If this `BundleContext` is no longer valid.

`org.osgi.framework.ServiceRegistration<?> registerService(String[] clazzes,
Object service,
Dictionary<String,?> properties)`

Registers the specified service object with the specified properties under the specified class names into the Framework. A `ServiceRegistration` object is returned. The `ServiceRegistration` object is for the private use of the bundle registering the service and should not be shared with other bundles. The registering bundle is defined to be the context bundle. Other bundles can locate the service by using one of the `getServiceReferences(Class, String)`, `getServiceReferences(String, String)`, `getServiceReference(Class)` or `getServiceReference(String)` methods.

A bundle can register a service object that implements the `ServiceFactory` interface to have more flexibility in providing service objects to other bundles.

The following steps are required to register a service:

1. If `service` does not implement `ServiceFactory`, an `IllegalArgumentException` is thrown if `service` is not an instance of all the specified class names.
2. The Framework adds the following service properties to the service properties from the specified `Dictionary` (which may be null):
 - A property named `Constants.SERVICE_ID` identifying the registration number of the service
 - A property named `Constants.OBJECTCLASS` containing all the specified classes.
 - A property named `Constants.SERVICE_SCOPE` identifying the scope of the service.Properties with these names in the specified `Dictionary` will be ignored.
3. The service is added to the Framework service registry and may now be used by other bundles.
4. A service event of type `org.osgi.framework.ServiceEvent.REGISTERED` is fired.
5. A `ServiceRegistration` object for this registration is returned.

Parameters:

`clazzes` - The class names under which the service can be located. The class names in this array will be stored in the service's properties under the key `Constants.OBJECTCLASS`.

`service` - The service object or an object implementing `ServiceFactory`.

`properties` - The properties for this service. The keys in the properties object must all be `String` objects. See `Constants` for a list of standard service property keys. Changes should not be made to this object after calling this method. To update the service's properties the `org.osgi.framework.ServiceRegistration.setProperties(Dictionary)` method must be called. The set of properties may be null if the service has no properties.

Returns:

A `ServiceRegistration` object for use by the bundle registering the service to update the service's properties or to unregister the service.

Throws:

`IllegalArgumentException` - If one of the following is true:

`service` is null.

`service` does not implement `ServiceFactory` and is not an instance of all the specified classes.

`properties` contains case variants of the same key name.

`SecurityException` - If the caller does not have the `ServicePermission` to register the service for all the named classes and the Java Runtime Environment supports permissions.

`IllegalStateException` - If this `BundleContext` is no longer valid.

See Also:

`org.osgi.framework.ServiceRegistration`, `PrototypeServiceFactory`, `ServiceFactory`
`org.osgi.framework.ServiceRegistration<?> registerService(String clazz,
Object service,
Dictionary<String,?> properties)`

Registers the specified service object with the specified properties under the specified class name with the Framework.

This method is otherwise identical to [registerService\(String\[\], Object, Dictionary\)](#) and is provided as a convenience when `service` will only be registered under a single class name. Note that even in this case the value of the service's [Constants.OBJECTCLASS](#) property will be an array of string, rather than just a single string.

Parameters:

- `clazz` - The class name under which the service can be located.
- `service` - The service object or an object implementing `ServiceFactory`.
- `properties` - The properties for this service.

Returns:

A `ServiceRegistration` object for use by the bundle registering the service to update the service's properties or to unregister the service.

Throws:

- `IllegalStateException` - If this `BundleContext` is no longer valid.

See Also:

[registerService\(String\[\], Object, Dictionary\)](#)

```
org.osgi.framework.ServiceRegistration<S> registerService(Class<S> clazz,  
                                                         S service,  
                                                         Dictionary<String,?> properties)
```

Registers the specified service object with the specified properties under the name of the specified class with the Framework.

This method is otherwise identical to [registerService\(String, Object, Dictionary\)](#) and is provided to return a type safe `ServiceRegistration`.

Type Parameters:

`S` - Type of Service.

Parameters:

- `clazz` - The class under whose name the service can be located.
- `service` - The service object or an object implementing `ServiceFactory`.
- `properties` - The properties for this service.

Returns:

A `ServiceRegistration` object for use by the bundle registering the service to update the service's properties or to unregister the service.

Throws:

- `IllegalStateException` - If this `BundleContext` is no longer valid.

Since:

1.6

See Also:

[registerService\(String, Object, Dictionary\)](#)

```
org.osgi.framework.ServiceReference<?>[] getServiceReferences(String clazz,  
                                                            String filter)  
throws org.osgi.framework.InvalidSyntaxException
```

Exception

Returns an array of `ServiceReference` objects. The returned array of `ServiceReference` objects contains services that were registered under the specified class, match the specified filter expression, and the packages for the class names under which the services were registered match the context bundle's packages as defined in `org.osgi.framework.ServiceReference.isAssignableTo(Bundle, String)`.

The list is valid at the time of the call to this method. However since the Framework is a very dynamic environment, services can be modified or unregistered at any time.

The specified `filter` expression is used to select the registered services whose service properties contain keys and values which satisfy the filter expression. See `org.osgi.framework.Filter` for a description of the filter syntax. If the specified `filter` is `null`, all registered services are considered to match the filter. If the specified `filter` expression cannot be parsed, an `org.osgi.framework.InvalidSyntaxException` will be thrown with a human readable message where the filter became unparseable.

The result is an array of `ServiceReference` objects for all services that meet all of the following conditions:

- If the specified class name, `clazz`, is not `null`, the service must have been registered with the specified class name. The complete list of class names with which a service was registered is available from the service's [objectClass](#) property.

- If the specified `filter` is not `null`, the filter expression must match the service.

- If the Java Runtime Environment supports permissions, the caller must have `ServicePermission` with the `GET` action for at least one of the class names under which the service was registered.

- For each class name with which the service was registered, calling `org.osgi.framework.ServiceReference.isAssignableTo(Bundle, String)` with the context bundle and the class name on the service's `ServiceReference` object must return `true`

Parameters:

`clazz` - The class name with which the service was registered or `null` for all services.

`filter` - The filter expression or `null` for all services.

Returns:

An array of `ServiceReference` objects or `null` if no services are registered which satisfy the search.

Throws:

`org.osgi.framework.InvalidSyntaxException` - If the specified `filter` contains an invalid filter expression that cannot be parsed.

`IllegalStateException` - If this `BundleContext` is no longer valid.

getAllServiceReferences

```
org.osgi.framework.ServiceReference<?>[] getAllServiceReferences(String clazz,  
                                                                String filter)  
throws org.osgi.framework.InvalidSyntaxException
```

taxException

Returns an array of `ServiceReference` objects. The returned array of `ServiceReference` objects contains services that were registered under the specified class and match the specified filter expression.

The list is valid at the time of the call to this method. However since the Framework is a very dynamic environment, services can be modified or unregistered at any time.

The specified `filter` expression is used to select the registered services whose service properties contain keys and values which satisfy the filter expression. See `org.osgi.framework.Filter` for a description of the filter syntax. If the specified `filter` is `null`, all registered services are considered to match the filter. If the specified `filter` expression cannot be parsed, an `org.osgi.framework.InvalidSyntaxException` will be thrown with a human readable message where the filter became unparseable.

The result is an array of `ServiceReference` objects for all services that meet all of the following conditions:

If the specified class name, `clazz`, is not `null`, the service must have been registered with the specified class name. The complete list of class names with which a service was registered is available from the service's `objectClass` property.

If the specified `filter` is not `null`, the filter expression must match the service.

If the Java Runtime Environment supports permissions, the caller must have `ServicePermission` with the `GET` action for at least one of the class names under which the service was registered.

Parameters:

`clazz` - The class name with which the service was registered or `null` for all services.

`filter` - The filter expression or `null` for all services.

Returns:

An array of `ServiceReference` objects or `null` if no services are registered which satisfy the search.

Throws:

`org.osgi.framework.InvalidSyntaxException` - If the specified `filter` contains an invalid filter expression that cannot be parsed.

`IllegalStateException` - If this `BundleContext` is no longer valid.

Since:

getServiceReference

```
org.osgi.framework.ServiceReference<?> getServiceReference(String clazz)
```

Returns a `ServiceReference` object for a service that implements and was registered under the specified class.

The returned `ServiceReference` object is valid at the time of the call to this method. However as the Framework is a very dynamic environment, services can be modified or unregistered at any time.

This method is the same as calling `getServiceReferences(String, String)` with a `null` filter expression and then finding the reference with the highest priority. It is provided as a convenience for when the caller is interested in any service that implements the specified class.

If multiple such services exist, the service with the highest priority is selected. This priority is defined as the service reference with the highest ranking (as specified in its `Constants.SERVICE_RANKING` property) is returned.

If there is a tie in ranking, the service with the lowest service ID (as specified in its `Constants.SERVICE_ID` property); that is, the service that was registered first is returned.

Parameters:

`clazz` - The class name with which the service was registered.

Returns:

A `ServiceReference` object, or `null` if no services are registered which implement the named class.

Throws:

`IllegalStateException` - If this `BundleContext` is no longer valid.

See Also:

[getServiceReferences\(String, String\)](#)

getServiceReference

```
org.osgi.framework.ServiceReference<S> getServiceReference(Class<S> clazz)
```

Returns a `ServiceReference` object for a service that implements and was registered under the name of the specified class.

The returned `ServiceReference` object is valid at the time of the call to this method. However as the Framework is a very dynamic environment, services can be modified or unregistered at any time.

This method is the same as calling [getServiceReferences\(Class, String\)](#) with a `null` filter expression. It is provided as a convenience for when the caller is interested in any service that implements the specified class.

If multiple such services exist, the service with the highest ranking (as specified in its [Constants.SERVICE_RANKING](#) property) is returned.

If there is a tie in ranking, the service with the lowest service ID (as specified in its [Constants.SERVICE_ID](#) property); that is, the service that was registered first is returned.

Type Parameters:

S - Type of Service.

Parameters:

clazz - The class under whose name the service was registered. Must not be `null`.

Returns:

A `ServiceReference` object, or `null` if no services are registered which implement the specified class.

Throws:

`IllegalStateException` - If this `BundleContext` is no longer valid.

Since:

1.6

See Also:

[getServiceReferences\(Class, String\)](#)

getServiceReferences

```
Collection<org.osgi.framework.ServiceReference<S>> getServiceReferences(Class<S> clazz,
                                                                    String filter)
                                                                    throws org.osgi.framework.InvalidSyntaxException
```

Returns a collection of `ServiceReference` objects. The returned collection of `ServiceReference` objects contains services that were registered under the name of the specified class, match the specified filter expression, and the packages for the class names under which the services were registered match the context bundle's packages as defined in `org.osgi.framework.ServiceReference.isAssignableTo(Bundle, String)`.

The collection is valid at the time of the call to this method. However since the Framework is a very dynamic environment, services can be modified or unregistered at any time.

The specified `filter` expression is used to select the registered services whose service properties contain keys and values which satisfy the filter expression. See `org.osgi.framework.Filter` for a description of the filter syntax. If the specified `filter` is `null`, all registered services are considered to match the filter. If the specified `filter` expression cannot be parsed, an `org.osgi.framework.InvalidSyntaxException` will be thrown with a human readable message where the filter became unparseable.

The result is a collection of `ServiceReference` objects for all services that meet all of the following conditions:

The service must have been registered with the name of the specified class. The complete list of class names with which a service was registered is available from the service's [objectClass](#) property.

If the specified `filter` is not `null`, the filter expression must match the service.

If the Java Runtime Environment supports permissions, the caller must have `ServicePermission` with the `GET` action for at least one of the class names under which the service was registered.

For each class name with which the service was registered, calling `org.osgi.framework.ServiceReference.isAssignableTo(Bundle, String)` with the context bundle and the class name on the service's `ServiceReference` object must return `true`

Type Parameters:

S - Type of Service

Parameters:

clazz - The class under whose name the service was registered. Must not be `null`.

filter - The filter expression or `null` for all services.

Returns:

A collection of `ServiceReference` objects. May be empty if no services are registered which satisfy the search.

Throws:

`org.osgi.framework.InvalidSyntaxException` - If the specified `filter` contains an invalid filter expression that cannot be parsed.

`IllegalStateException` - If this `BundleContext` is no longer valid.

Since:

1.6

`getService`

```
S getService(org.osgi.framework.ServiceReference<S> reference)
```

Returns the service object for the service referenced by the specified `ServiceReference` object.

A bundle's use of a service object obtained from this method is tracked by the bundle's use count of that service object. Each time the service object is returned by [getService\(ServiceReference\)](#) the context bundle's use count for the service object is incremented by one. Each time the service object is released by [ungetService\(ServiceReference\)](#) the context bundle's use count for the service object is decremented by one.

When a bundle's use count for the service object drops to zero, the bundle should no longer use the service object.

This method will always return `null` when the service associated with the specified `reference` has been unregistered.

The following steps are required to get the service object:

1. If the service has been unregistered, `null` is returned.
2. If the context bundle's use count for the service object is currently zero and the service was registered with an object implementing the `ServiceFactory` interface, the [ServiceFactory.getService\(Bundle, ServiceRegistration\)](#) method is called to create the service object for the context bundle. If the service object returned by the `ServiceFactory` object is `null`, not an instance of all the classes named when the service was registered or the `ServiceFactory` object throws an exception or will be recursively called for the context bundle, `null` is returned and a Framework event of type `org.osgi.framework.FrameworkEvent.ERROR` containing a `org.osgi.framework.ServiceException` describing the error is fired. This service object is cached by the Framework. While the context bundle's use count for the service object is greater than zero, subsequent calls to get the service's service object for the context bundle will return the cached service object.
3. The context bundle's use count for the service object is incremented by one.
4. The service object for the service is returned.

Type Parameters:

`S` - Type of Service.

Parameters:

`reference` - A reference to the service.

Returns:

A service object for the service associated with `reference` or `null` if the service is not registered, the service object returned by a `ServiceFactory` does not implement the classes under which it was registered or the `ServiceFactory` threw an exception.

Throws:

`SecurityException` - If the caller does not have the `ServicePermission` to get the service using at least one of the named classes the service was registered under and the Java Runtime Environment supports permissions.

`IllegalStateException` - If this `BundleContext` is no longer valid.

`IllegalArgumentException` - If the specified `ServiceReference` was not created by the same framework instance as this `BundleContext`.

See Also:

`ungetService` [ungetService\(ServiceReference\)](#), [ServiceFactory](#)

```
boolean ungetService(org.osgi.framework.ServiceReference<?> reference)
```

Releases the service object for the service referenced by the specified `ServiceReference` object. If the context bundle's use count for the service object is zero, this method returns `false`. Otherwise, the context bundle's use count for the service object is decremented by one.

The service object must no longer be used and all references to it should be destroyed when a bundle's use count for the service object drops to zero.

The following steps are required to unget the service object:

1. If the context bundle's use count for the service object is zero or the service has been unregistered, `false` is returned.
2. The context bundle's use count for the service object is decremented by one.
3. If the context bundle's use count for the service object is currently zero and the service was registered with a `ServiceFactory` object, the [ServiceFactory.ungetService\(Bundle, ServiceRegistration, Object\)](#) method is called to release the service object for the context bundle.
4. `true` is returned.

Parameters:

reference - A reference to the service to be released.

Returns:

false if the context bundle's use count for the service object is zero or if the service has been unregistered; true otherwise.

Throws:

IllegalStateException - If this BundleContext is no longer valid.

IllegalArgumentException - If the specified ServiceReference was not created by the same framework instance as this BundleContext.

See Also:

[getServiceObjects](#), [getService\(ServiceReference\)](#), [ServiceFactory](#)
[ServiceObjects](#)

`ServiceObjects<S> getServiceObjects(org.osgi.framework.ServiceReference<S> reference)`

Returns the [ServiceObjects](#) object for the service referenced by the specified `ServiceReference` object.

The [ServiceObjects](#) object can be used to obtain multiple service objects for services with [prototype](#) scope. For services with [singleton](#) or [bundle](#) scope, the [ServiceObjects.getService\(\)](#) method behaves the same as the [getService\(ServiceReference\)](#) method and the [ServiceObjects.ungetService\(Object\)](#) method behaves the same as the [ungetService\(ServiceReference\)](#) method. That is, only one, use-counted service object is available from the [ServiceObjects](#) object.

This method will always return null when the service associated with the specified `reference` has been unregistered.

Type Parameters:

S - Type of Service.

Parameters:

reference - A reference to the service.

Returns:

A [ServiceObjects](#) object for the service associated with the specified `reference` or null if the service is not registered.

Throws:

SecurityException - If the caller does not have the `ServicePermission` to get the service using at least one of the named classes the service was registered under and the Java Runtime Environment supports permissions.

IllegalStateException - If this BundleContext is no longer valid.

IllegalArgumentException - If the specified ServiceReference was not created by the same framework instance as this BundleContext.

Since:

1.8

See Also:

[getDataFile](#), [PrototypeServiceFactory](#)

`File getDataFile(String filename)`

Creates a `File` object for a file in the persistent storage area provided for the bundle by the Framework. This method will return null if the platform does not have file system support.

A `File` object for the base directory of the persistent storage area provided for the context bundle by the Framework can be obtained by calling this method with an empty string as `filename`.

If the Java Runtime Environment supports permissions, the Framework will ensure that the bundle has the `java.io.FilePermission` with actions `read,write,delete` for all files (recursively) in the persistent storage area provided for the context bundle.

Parameters:

filename - A relative name to the file to be accessed.

Returns:

A `File` object that represents the requested file or null if the platform does not have file system support.

Throws:

IllegalStateException - If this BundleContext is no longer valid.

`org.osgi.framework.Filter createFilter(String filter)`

throws `org.osgi.framework.InvalidSyntaxException`

Creates a `Filter` object. This `Filter` object may be used to match a `ServiceReference` object or a `Dictionary` object.

If the filter cannot be parsed, an `org.osgi.framework.InvalidSyntaxException` will be thrown with a human readable message where the filter became unparseable.

Parameters:

filter - The filter string.

Returns:

A `Filter` object encapsulating the filter string.

Throws:

`org.osgi.framework.InvalidSyntaxException` - If filter contains an invalid filter string that cannot be parsed.

`NullPointerException` - If filter is null.

`IllegalStateException` - If this `BundleContext` is no longer valid.

Since:

1.1

See Also:

"Framework specification for a description of the filter string syntax.",

`getBundle` `org.osgi.framework.FrameworkUtil.createFilter(String)`

`org.osgi.framework.Bundle getBundle(String location)`

Returns the bundle with the specified location.

Parameters:

`location` - The location of the bundle to retrieve.

Returns:

A `Bundle` object or `null` if the location does not match any installed bundle.

Since:

1.6

Interface Constants

[org.osgi.framework](#)

public interface Constants

Defines standard names for the OSGi environment system properties, service properties, and Manifest header attribute keys.

The values associated with these keys are of type `String`, unless otherwise indicated.

Since:

1.1

Version:

\$Id: 85d13ecd84fd9fe7604e152e046ba63864083139 \$

Field Summary		Page
String	ACTIVATION_LAZY Bundle activation policy declaring the bundle must be activated when the first class load is made from the bundle.	Error: Reference source not found
String	BUNDLE_ACTIVATIONPOLICY Manifest header identifying the bundle's activation policy.	Error: Reference source not found
String	BUNDLE_ACTIVATOR Manifest header attribute identifying the bundle's activator class.	Error: Reference source not found
String	BUNDLE_CATEGORY Manifest header identifying the bundle's category.	Error: Reference source not found
String	BUNDLE_CLASSPATH Manifest header identifying a list of directories and embedded JAR files, which are bundle resources used to extend the bundle's classpath.	Error: Reference source not found
String	BUNDLE_CONTACTADDRESS Manifest header identifying the contact address where problems with the bundle may be reported; for example, an email address.	Error: Reference source not found
String	BUNDLE_COPYRIGHT Manifest header identifying the bundle's copyright information.	Error: Reference source not found
String	BUNDLE_DESCRIPTION Manifest header containing a brief description of the bundle's functionality.	Error: Reference source not found
String	BUNDLE_DOCURL Manifest header identifying the bundle's documentation URL, from which further information about the bundle may be obtained.	Error: Reference source not found

String	BUNDLE_LOCALIZATION Manifest header identifying the base name of the bundle's localization entries.	Error: Reference source not found
String	BUNDLE_LOCALIZATION_DEFAULT_BASENAME Default value for the Bundle-Localization manifest header.	Error: Reference source not found
String	BUNDLE_MANIFESTVERSION Manifest header identifying the bundle manifest version.	Error: Reference source not found
String	BUNDLE_NAME Manifest header identifying the bundle's name.	Error: Reference source not found
String	BUNDLE_NATIVECODE Manifest header identifying a number of hardware environments and the native language code libraries that the bundle is carrying for each of these environments.	Error: Reference source not found
String	BUNDLE_NATIVECODE_LANGUAGE Manifest header attribute identifying the language in which the native bundle code is written specified in the Bundle-NativeCode manifest header.	Error: Reference source not found
String	BUNDLE_NATIVECODE_OSNAME Manifest header attribute identifying the operating system required to run native bundle code specified in the Bundle-NativeCode manifest header).	Error: Reference source not found
String	BUNDLE_NATIVECODE_OSVERSION Manifest header attribute identifying the operating system version required to run native bundle code specified in the Bundle-NativeCode manifest header).	Error: Reference source not found
String	BUNDLE_NATIVECODE_PROCESSOR Manifest header attribute identifying the processor required to run native bundle code specified in the Bundle-NativeCode manifest header).	Error: Reference source not found
String	BUNDLE_REQUIREDEXECUTIONENVIRONMENT Deprecated. As of 1.6.	Error: Reference source not found
String	BUNDLE_SYMBOLICNAME Manifest header identifying the bundle's symbolic name.	Error: Reference source not found

String	<u>BUNDLE_SYMBOLICNAME_ATTRIBUTE</u> Manifest header attribute identifying the symbolic name of a bundle that exports a package specified in the Import-Package manifest header.	Error: Reference source not found
String	<u>BUNDLE_UPDATELOCATION</u> Manifest header identifying the location from which a new bundle version is obtained during a bundle update operation.	Error: Reference source not found
String	<u>BUNDLE_VENDOR</u> Manifest header identifying the bundle's vendor.	Error: Reference source not found
String	<u>BUNDLE_VERSION</u> Manifest header identifying the bundle's version.	Error: Reference source not found
String	<u>BUNDLE_VERSION_ATTRIBUTE</u> Manifest header attribute identifying a range of versions for a bundle specified in the Require-Bundle or Fragment-Host manifest headers.	Error: Reference source not found
String	<u>DYNAMICIMPORT_PACKAGE</u> Manifest header identifying the packages that the bundle may dynamically import during execution.	Error: Reference source not found
String	<u>EFFECTIVE_ACTIVE</u> Manifest header directive value identifying a capability that is effective at active time.	Error: Reference source not found
String	<u>EFFECTIVE_DIRECTIVE</u> Manifest header directive identifying the effective time of the provided capability.	Error: Reference source not found
String	<u>EFFECTIVE_RESOLVE</u> Manifest header directive value identifying a capability that is effective at resolve time.	Error: Reference source not found
String	<u>EXCLUDE_DIRECTIVE</u> Manifest header directive identifying a list of classes to exclude in the exported package..	Error: Reference source not found
String	<u>EXPORT_PACKAGE</u> Manifest header identifying the packages that the bundle offers to the Framework for export.	Error: Reference source not found

String	<u>EXPORT_SERVICE</u> Deprecated. As of 1.2.	Error: Reference source not found
String	<u>EXTENSION_BOOTCLASSPATH</u> Manifest header directive value identifying the type of extension fragment.	Error: Reference source not found
String	<u>EXTENSION_DIRECTIVE</u> Manifest header directive identifying the type of the extension fragment.	Error: Reference source not found
String	<u>EXTENSION_FRAMEWORK</u> Manifest header directive value identifying the type of extension fragment.	Error: Reference source not found
String	<u>FILTER_DIRECTIVE</u> Manifest header directive identifying the capability filter specified in the Require-Capability manifest header.	Error: Reference source not found
String	<u>FRAGMENT_ATTACHMENT_ALWAYS</u> Manifest header directive value identifying a fragment attachment type of always.	Error: Reference source not found
String	<u>FRAGMENT_ATTACHMENT_DIRECTIVE</u> Manifest header directive identifying if and when a fragment may attach to a host bundle.	Error: Reference source not found
String	<u>FRAGMENT_ATTACHMENT_NEVER</u> Manifest header directive value identifying a fragment attachment type of never.	Error: Reference source not found
String	<u>FRAGMENT_ATTACHMENT_RESOLVETIME</u> Manifest header directive value identifying a fragment attachment type of resolve-time.	Error: Reference source not found
String	<u>FRAGMENT_HOST</u> Manifest header identifying the symbolic name of another bundle for which that the bundle is a fragment.	Error: Reference source not found
String	<u>FRAMEWORK_BEGINNING_STARTLEVEL</u> Framework launching property specifying the beginning start level of the framework.	Error: Reference source not found

Interface Constants

String	FRAMEWORK_BOOTDELEGATION Framework launching property identifying packages for which the Framework must delegate class loading to the parent class loader of the bundle.	Error: Reference source not found
String	FRAMEWORK_BSNVERSION Framework launching property specifying whether multiple bundles having the same symbolic name and version may be installed.	Error: Reference source not found
String	FRAMEWORK_BSNVERSION_MANAGED Specifies the framework must consult the <code>bundle collision hook</code> services to determine if it will be an error to install a bundle or update a bundle to have the same symbolic name and version as another installed bundle.	Error: Reference source not found
String	FRAMEWORK_BSNVERSION_MULTIPLE Specifies the framework will allow multiple bundles to be installed having the same symbolic name and version.	Error: Reference source not found
String	FRAMEWORK_BSNVERSION_SINGLE Specifies the framework will only allow a single bundle to be installed for a given symbolic name and version.	Error: Reference source not found
String	FRAMEWORK_BUNDLE_PARENT Framework launching property specifying the parent class loader type for all bundle class loaders.	Error: Reference source not found
String	FRAMEWORK_BUNDLE_PARENT_APP Specifies to use the application class loader as the parent class loader for all bundle class loaders.	Error: Reference source not found
String	FRAMEWORK_BUNDLE_PARENT_BOOT Specifies to use of the boot class loader as the parent class loader for all bundle class loaders.	Error: Reference source not found
String	FRAMEWORK_BUNDLE_PARENT_EXT Specifies to use the extension class loader as the parent class loader for all bundle class loaders.	Error: Reference source not found
String	FRAMEWORK_BUNDLE_PARENT_FRAMEWORK Specifies to use the framework class loader as the parent class loader for all bundle class loaders.	Error: Reference source not found
String	FRAMEWORK_COMMAND_ABSPATH Specified the substitution string for the absolute path of a file.	Error: Reference source not found

String	<u>FRAMEWORK_EXECPERMISSION</u> Framework launching property specifying an optional OS specific command to set file permissions on extracted native code.	Error: Reference source not found
String	<u>FRAMEWORK_EXECUTIONENVIRONMENT</u> Deprecated. As of 1.6.	Error: Reference source not found
String	<u>FRAMEWORK_LANGUAGE</u> Framework launching property identifying the Framework implementation language (see ISO 639 for possible values).	Error: Reference source not found
String	<u>FRAMEWORK_LIBRARY_EXTENSIONS</u> Framework launching property specifying a comma separated list of additional library file extensions that must be used when a bundle's class loader is searching for native libraries.	Error: Reference source not found
String	<u>FRAMEWORK_OS_NAME</u> Framework launching property identifying the Framework host-computer's operating system.	Error: Reference source not found
String	<u>FRAMEWORK_OS_VERSION</u> Framework launching property identifying the Framework host-computer's operating system version number.	Error: Reference source not found
String	<u>FRAMEWORK_PROCESSOR</u> Framework launching property identifying the Framework host-computer's processor name.	Error: Reference source not found
String	<u>FRAMEWORK_SECURITY</u> Framework launching property specifying the type of security manager the framework must use.	Error: Reference source not found
String	<u>FRAMEWORK_SECURITY_OSGI</u> Specifies that a security manager that supports all security aspects of the OSGi core specification including postponed conditions must be installed.	Error: Reference source not found
String	<u>FRAMEWORK_STORAGE</u> Framework launching property specifying the persistent storage area used by the framework.	Error: Reference source not found
String	<u>FRAMEWORK_STORAGE_CLEAN</u> Framework launching property specifying if and when the persistent storage area for the framework should be cleaned.	Error: Reference source not found

Interface Constants

String	FRAMEWORK_STORAGE_CLEAN_ONFIRSTINIT Specifies that the framework storage area must be cleaned before the framework is initialized for the first time.	Error: Reference source not found
String	FRAMEWORK_SYSTEMCAPABILITIES Framework launching property identifying capabilities which the system bundle must provide.	Error: Reference source not found
String	FRAMEWORK_SYSTEMCAPABILITIES_EXTRA Framework launching property identifying extra capabilities which the system bundle must additionally provide.	Error: Reference source not found
String	FRAMEWORK_SYSTEMPACKAGES Framework launching property identifying packages which the system bundle must export.	Error: Reference source not found
String	FRAMEWORK_SYSTEMPACKAGES_EXTRA Framework launching property identifying extra packages which the system bundle must export from the current execution environment.	Error: Reference source not found
String	FRAMEWORK_TRUST_REPOSITORIES Framework launching property specifying the trust repositories used by the framework.	Error: Reference source not found
String	FRAMEWORK_UUID Framework environment property identifying the Framework's universally unique identifier (UUID).	Error: Reference source not found
String	FRAMEWORK_VENDOR Framework environment property identifying the Framework implementation vendor.	Error: Reference source not found
String	FRAMEWORK_VERSION Framework environment property identifying the Framework version.	Error: Reference source not found
String	FRAMEWORK_WINDOWSYSTEM Framework launching property specifying the current windowing system.	Error: Reference source not found
String	IMPORT_PACKAGE Manifest header identifying the packages on which the bundle depends.	Error: Reference source not found

String	<u>IMPORT_SERVICE</u> Deprecated. As of 1.2.	Error: Reference source not found
String	<u>INCLUDE_DIRECTIVE</u> Manifest header directive identifying a list of classes to include in the exported package.	Error: Reference source not found
String	<u>MANDATORY_DIRECTIVE</u> Manifest header directive identifying names of matching attributes which must be specified by matching Import-Package statements in the Export-Package manifest header.	Error: Reference source not found
String	<u>OBJECTCLASS</u> Service property identifying all of the class names under which a service was registered in the Framework.	Error: Reference source not found
String	<u>PACKAGE_SPECIFICATION_VERSION</u> Deprecated. As of 1.3.	Error: Reference source not found
String	<u>PROVIDE_CAPABILITY</u> Manifest header identifying the capabilities that the bundle offers to provide to other bundles.	Error: Reference source not found
String	<u>REMOTE_CONFIGS_SUPPORTED</u> Service property identifying the configuration types supported by a distribution provider.	Error: Reference source not found
String	<u>REMOTE_INTENTS_SUPPORTED</u> Service property identifying the intents supported by a distribution provider.	Error: Reference source not found
String	<u>REQUIRE_BUNDLE</u> Manifest header identifying the symbolic names of other bundles required by the bundle.	Error: Reference source not found
String	<u>REQUIRE_CAPABILITY</u> Manifest header identifying the capabilities on which the bundle depends.	Error: Reference source not found
String	<u>RESOLUTION_DIRECTIVE</u> Manifest header directive identifying the resolution type in the Import-Package, Require-Bundle or Require-Capability manifest header.	Error: Reference source not found

Interface Constants

String	<u>RESOLUTION_MANDATORY</u> Manifest header directive value identifying a mandatory resolution type.	Error: Reference source not found
String	<u>RESOLUTION_OPTIONAL</u> Manifest header directive value identifying an optional resolution type.	Error: Reference source not found
String	<u>SCOPE_BUNDLE</u> Service scope is bundle.	Error: Reference source not found
String	<u>SCOPE_PROTOTYPE</u> Service scope is prototype.	Error: Reference source not found
String	<u>SCOPE_SINGLETON</u> Service scope is singleton.	Error: Reference source not found
String	<u>SELECTION_FILTER_ATTRIBUTE</u> Manifest header attribute is used for selection by filtering based upon system properties.	Error: Reference source not found
String	<u>SERVICE_DESCRIPTION</u> Service property identifying a service's description.	Error: Reference source not found
String	<u>SERVICE_EXPORTED_CONFIGS</u> Service property identifying the configuration types that should be used to export the service.	Error: Reference source not found
String	<u>SERVICE_EXPORTED_INTENTS</u> Service property identifying the intents that the distribution provider must implement to distribute the service.	Error: Reference source not found
String	<u>SERVICE_EXPORTED_INTENTS_EXTRA</u> Service property identifying the extra intents that the distribution provider must implement to distribute the service.	Error: Reference source not found
String	<u>SERVICE_EXPORTED_INTERFACES</u> Service property marking the service for export.	Error: Reference source not found

Interface Constants

String	<u>SERVICE_ID</u> Service property identifying a service's registration number.	Error: Reference source not found
String	<u>SERVICE_IMPORTED</u> Service property identifying the service as imported.	Error: Reference source not found
String	<u>SERVICE_IMPORTED_CONFIGS</u> Service property identifying the configuration types used to import the service.	Error: Reference source not found
String	<u>SERVICE_INTENTS</u> Service property identifying the intents that this service implement.	Error: Reference source not found
String	<u>SERVICE_PID</u> Service property identifying a service's persistent identifier.	Error: Reference source not found
String	<u>SERVICE_RANKING</u> Service property identifying a service's ranking number.	Error: Reference source not found
String	<u>SERVICE_SCOPE</u> Service property identifying a service's scope.	Error: Reference source not found
String	<u>SERVICE_VENDOR</u> Service property identifying a service's vendor.	Error: Reference source not found
String	<u>SINGLETON_DIRECTIVE</u> Manifest header directive identifying whether a bundle is a singleton.	Error: Reference source not found
String	<u>SUPPORTS_BOOTCLASSPATH_EXTENSION</u> Framework environment property identifying whether the Framework supports bootclasspath extension bundles.	Error: Reference source not found
String	<u>SUPPORTS_FRAMEWORK_EXTENSION</u> Framework environment property identifying whether the Framework supports framework extension bundles.	Error: Reference source not found

Interface Constants

String	SUPPORTS_FRAMEWORK_FRAGMENT Framework environment property identifying whether the Framework supports fragment bundles.	Error: Reference source not found
String	SUPPORTS_FRAMEWORK_REQUIREBUNDLE Framework environment property identifying whether the Framework supports the Require-Bundle manifest header.	Error: Reference source not found
long	SYSTEM_BUNDLE_ID Identifier of the OSGi <i>system bundle</i> , which is defined to be 0.	Error: Reference source not found
String	SYSTEM_BUNDLE_LOCATION Location identifier of the OSGi <i>system bundle</i> , which is defined to be "System Bundle".	Error: Reference source not found
String	SYSTEM_BUNDLE_SYMBOLICNAME Alias for the symbolic name of the OSGi <i>system bundle</i> .	Error: Reference source not found
String	USES_DIRECTIVE Manifest header directive identifying a list of packages that an exported package or provided capability uses.	Error: Reference source not found
String	VERSION_ATTRIBUTE Manifest header attribute identifying the version of a package specified in the Export-Package or Import-Package manifest header.	Error: Reference source not found
String	VISIBILITY_DIRECTIVE Manifest header directive identifying the visibility of a required bundle in the Require-Bundle manifest header.	Error: Reference source not found
String	VISIBILITY_PRIVATE Manifest header directive value identifying a private visibility type.	Error: Reference source not found
String	VISIBILITY_REEXPORT Manifest header directive value identifying a reexport visibility type.	Error: Reference source not found

Field Detail

SYSTEM_BUNDLE_LOCATION

public static final String SYSTEM_BUNDLE_LOCATION = "System Bundle"

Location identifier of the OSGi *system bundle*, which is defined to be "System Bundle".

~~SYSTEM_BUNDLE_SYMBOLICNAME~~

public static final String SYSTEM_BUNDLE_SYMBOLICNAME = "system.bundle"

Alias for the symbolic name of the OSGi *system bundle*. It is defined to be "system.bundle".

Since:

~~SYSTEM_BUNDLE_ID~~

public static final long SYSTEM_BUNDLE_ID = 0L

Identifier of the OSGi *system bundle*, which is defined to be 0.

Since:

~~BUNDLE_CATEGORY~~

public static final String BUNDLE_CATEGORY = "Bundle-Category"

Manifest header identifying the bundle's category.

The header value may be retrieved from the Dictionary object returned by the Bundle.getHeaders

~~BUNDLE_CLASSPATH~~

public static final String BUNDLE_CLASSPATH = "Bundle-ClassPath"

Manifest header identifying a list of directories and embedded JAR files, which are bundle resources used to extend the bundle's classpath.

The header value may be retrieved from the Dictionary object returned by the Bundle.getHeaders

~~BUNDLE_COPYRIGHT~~

public static final String BUNDLE_COPYRIGHT = "Bundle-Copyright"

Manifest header identifying the bundle's copyright information.

The header value may be retrieved from the Dictionary object returned by the Bundle.getHeaders

~~BUNDLE_DESCRIPTION~~

public static final String BUNDLE_DESCRIPTION = "Bundle-Description"

Manifest header containing a brief description of the bundle's functionality.

The header value may be retrieved from the Dictionary object returned by the Bundle.getHeaders

~~BUNDLE_NAME~~

public static final String BUNDLE_NAME = "Bundle-Name"

Manifest header identifying the bundle's name.

The header value may be retrieved from the Dictionary object returned by the Bundle.getHeaders

~~BUNDLE_NATIVECODE~~

public static final String BUNDLE_NATIVECODE = "Bundle-NativeCode"

Manifest header identifying a number of hardware environments and the native language code libraries that the bundle is carrying for each of these environments.

The header value may be retrieved from the Dictionary object returned by the Bundle.getHeaders

~~EXPORT_PACKAGE~~

public static final String EXPORT_PACKAGE = "Export-Package"

Manifest header identifying the packages that the bundle offers to the Framework for export.

The header value may be retrieved from the Dictionary object returned by the Bundle.getHeaders

~~EXPORT_SERVICE~~

public static final String EXPORT_SERVICE = "Export-Service"

Deprecated.

Manifest header identifying the fully qualified class names of the services that the bundle may register (used for informational purposes only).

The header value may be retrieved from the Dictionary object returned by the Bundle.getHeaders

~~IMPORT_PACKAGE~~

public static final String IMPORT_PACKAGE = "Import-Package"

Manifest header identifying the packages on which the bundle depends.

The header value may be retrieved from the Dictionary object returned by the Bundle.getHeaders

~~DYNAMIC_IMPORT_PACKAGE~~

public static final String DYNAMICIMPORT_PACKAGE = "DynamicImport-Package"

Manifest header identifying the packages that the bundle may dynamically import during execution.

The header value may be retrieved from the Dictionary object returned by the Bundle.getHeaders method.

Since:

~~IMPORT_SERVICE~~

public static final String IMPORT_SERVICE = "Import-Service"

Deprecated.

Manifest header identifying the fully qualified class names of the services that the bundle requires (used for informational purposes only).

The header value may be retrieved from the `Dictionary` object returned by the `Bundle.getHeaders` method.

BUNDLE_VENDOR

```
public static final String BUNDLE_VENDOR = "Bundle-Vendor"
```

Manifest header identifying the bundle's vendor.

The header value may be retrieved from the `Dictionary` object returned by the `Bundle.getHeaders` method.

BUNDLE_VERSION

```
public static final String BUNDLE_VERSION = "Bundle-Version"
```

Manifest header identifying the bundle's version.

The header value may be retrieved from the `Dictionary` object returned by the `Bundle.getHeaders` method.

BUNDLE_DOCURL

```
public static final String BUNDLE_DOCURL = "Bundle-DocURL"
```

Manifest header identifying the bundle's documentation URL, from which further information about the bundle may be obtained.

The header value may be retrieved from the `Dictionary` object returned by the `Bundle.getHeaders` method.

BUNDLE_CONTACTADDRESS

```
public static final String BUNDLE_CONTACTADDRESS = "Bundle-ContactAddress"
```

Manifest header identifying the contact address where problems with the bundle may be reported; for example, an email address.

The header value may be retrieved from the `Dictionary` object returned by the `Bundle.getHeaders` method.

BUNDLE_ACTIVATOR

```
public static final String BUNDLE_ACTIVATOR = "Bundle-Activator"
```

Manifest header attribute identifying the bundle's activator class.

If present, this header specifies the name of the bundle resource class that implements the `BundleActivator` interface and whose `start` and `stop` methods are called by the Framework when the bundle is started and stopped, respectively.

The header value may be retrieved from the `Dictionary` object returned by the `Bundle.getHeaders` method.

BUNDLE_UPDATELOCATION

```
public static final String BUNDLE_UPDATELOCATION = "Bundle-UpdateLocation"
```

Manifest header identifying the location from which a new bundle version is obtained during a bundle update operation.

The header value may be retrieved from the `Dictionary` object returned by the `Bundle.getHeaders` method.

PACKAGE_SPECIFICATION_VERSION

```
public static final String PACKAGE_SPECIFICATION_VERSION = "specification-version"
```

Deprecated.

Manifest header attribute identifying the version of a package specified in the Export-Package or Import-Package manifest header.

BUNDLE_NATIVECODE_PROCESSOR

```
public static final String BUNDLE_NATIVECODE_PROCESSOR = "processor"
```

Manifest header attribute identifying the processor required to run native bundle code specified in the Bundle-NativeCode manifest header).

The attribute value is encoded in the Bundle-NativeCode manifest header like:

```
Bundle-NativeCode: http.so ; processor=x86 ...
```

See Also:

BUNDLE_NATIVECODE_OSNAME

```
public static final String BUNDLE_NATIVECODE_OSNAME = "osname"
```

Manifest header attribute identifying the operating system required to run native bundle code specified in the Bundle-NativeCode manifest header).

The attribute value is encoded in the Bundle-NativeCode manifest header like:

```
Bundle-NativeCode: http.so ; osname=Linux ...
```

See Also:

BUNDLE_NATIVECODE_OSVERSION

```
public static final String BUNDLE_NATIVECODE_OSVERSION = "osversion"
```

Manifest header attribute identifying the operating system version required to run native bundle code specified in the Bundle-NativeCode manifest header).

The attribute value is encoded in the Bundle-NativeCode manifest header like:

```
Bundle-NativeCode: http.so ; osversion="2.34" ...
```

See Also:

BUNDLE_NATIVECODE_LANGUAGE

```
public static final String BUNDLE_NATIVECODE_LANGUAGE = "language"
```

Manifest header attribute identifying the language in which the native bundle code is written specified in the Bundle-NativeCode manifest header. See ISO 639 for possible values.

The attribute value is encoded in the Bundle-NativeCode manifest header like:

Bundle-NativeCode: http.so ; language=nl_be ...

See Also:

[BUNDLE_NATIVECODE](#) **BUNDLE_REQUIRED_EXECUTION_ENVIRONMENT**

```
public static final String BUNDLE_REQUIRED_EXECUTION_ENVIRONMENT = "Bundle-RequiredExecutionEnvironment"
```

Deprecated.

Manifest header identifying the required execution environment for the bundle. The service platform may run this bundle if any of the execution environments named in this header matches one of the execution environments it implements.

The header value may be retrieved from the `Dictionary` object returned by the `Bundle.getHeaders` method.

Since:

[BUNDLE_SYMBOLICNAME](#) **BUNDLE_SYMBOLICNAME**

```
public static final String BUNDLE_SYMBOLICNAME = "Bundle-SymbolicName"
```

Manifest header identifying the bundle's symbolic name.

The header value may be retrieved from the `Dictionary` object returned by the `Bundle.getHeaders` method.

Since:

[SINGLETON_DIRECTIVE](#) **SINGLETON_DIRECTIVE**

```
public static final String SINGLETON_DIRECTIVE = "singleton"
```

Manifest header directive identifying whether a bundle is a singleton. The default value is `false`.

The directive value is encoded in the Bundle-SymbolicName manifest header like:

Bundle-SymbolicName: com.acme.module.test; singleton:=true

Since:

1.3

See Also:

[BUNDLE_SYMBOLICNAME](#) **FRAGMENT_ATTACHMENT_DIRECTIVE**

```
public static final String FRAGMENT_ATTACHMENT_DIRECTIVE = "fragment-attachment"
```

Manifest header directive identifying if and when a fragment may attach to a host bundle. The default value is `always`.

The directive value is encoded in the Bundle-SymbolicName manifest header like:

Bundle-SymbolicName: com.acme.module.test; fragment-attachment:="never"

Since:

1.3

See Also:

[BUNDLE_SYMBOLICNAME](#), [FRAGMENT_ATTACHMENT_ALWAYS](#), [FRAGMENT_ATTACHMENT_RESOLVETIME](#), [FRAGMENT_ATTACHMENT_NEVER](#)

FRAGMENT_ATTACHMENT_ALWAYS

```
public static final String FRAGMENT_ATTACHMENT_ALWAYS = "always"
```

Manifest header directive value identifying a fragment attachment type of always. A fragment attachment type of always indicates that fragments are allowed to attach to the host bundle at any time (while the host is resolved or during the process of resolving the host bundle).

The directive value is encoded in the Bundle-SymbolicName manifest header like:

Bundle-SymbolicName: com.acme.module.test; fragment-attachment:="always"

Since:

1.3

See Also:

[FRAGMENT_ATTACHMENT_DIRECTIVE](#) **FRAGMENT_ATTACHMENT_RESOLVETIME**

```
public static final String FRAGMENT_ATTACHMENT_RESOLVETIME = "resolve-time"
```

Manifest header directive value identifying a fragment attachment type of resolve-time. A fragment attachment type of resolve-time indicates that fragments are allowed to attach to the host bundle only during the process of resolving the host bundle.

The directive value is encoded in the Bundle-SymbolicName manifest header like:

Bundle-SymbolicName: com.acme.module.test;
fragment-attachment:="resolve-time"

Since:

1.3

See Also:

[FRAGMENT_ATTACHMENT_DIRECTIVE](#) **FRAGMENT_ATTACHMENT_NEVER**

```
public static final String FRAGMENT_ATTACHMENT_NEVER = "never"
```

Manifest header directive value identifying a fragment attachment type of never. A fragment attachment type of never indicates that no fragments are allowed to attach to the host bundle at any time.

The directive value is encoded in the Bundle-SymbolicName manifest header like:

Bundle-SymbolicName: com.acme.module.test; fragment-attachment:="never"

Since:

1.3

See Also:

BUNDLE_LOCALIZATION [FRAGMENT_ATTACHMENT_DIRECTIVE](#)

public static final String BUNDLE_LOCALIZATION = "Bundle-Localization"

Manifest header identifying the base name of the bundle's localization entries.

The header value may be retrieved from the `Dictionary` object returned by the `Bundle.getHeaders` method.

Since:

1.3

See Also:

BUNDLE_LOCALIZATION_DEFAULT_BASENAME [BUNDLE_LOCALIZATION_DEFAULT_BASENAME](#)

public static final String BUNDLE_LOCALIZATION_DEFAULT_BASENAME = "OSGI-INF/l10n/bundle"

Default value for the `Bundle-Localization` manifest header.

Since:

1.3

See Also:

REQUIRE_BUNDLE [BUNDLE_LOCALIZATION](#)

public static final String REQUIRE_BUNDLE = "Require-Bundle"

Manifest header identifying the symbolic names of other bundles required by the bundle.

The header value may be retrieved from the `Dictionary` object returned by the `Bundle.getHeaders` method.

Since:

1.3

BUNDLE_VERSION_ATTRIBUTE

public static final String BUNDLE_VERSION_ATTRIBUTE = "bundle-version"

Manifest header attribute identifying a range of versions for a bundle specified in the `Require-Bundle` or `Fragment-Host` manifest headers. The default value is `0.0.0`.

The attribute value is encoded in the `Require-Bundle` manifest header like:

`Require-Bundle: com.acme.module.test; bundle-version="1.1"`

`Require-Bundle: com.acme.module.test; bundle-version="[1.0,2.0)"`

The `bundle-version` attribute value uses a mathematical interval notation to specify a range of bundle versions. A `bundle-version` attribute value specified as a single version means a version range that includes any bundle version greater than or equal to the specified version.

Since:

1.3

See Also:

FRAGMENT_HOST [REQUIRE_BUNDLE](#)

public static final String FRAGMENT_HOST = "Fragment-Host"

Manifest header identifying the symbolic name of another bundle for which that the bundle is a fragment.

The header value may be retrieved from the `Dictionary` object returned by the `Bundle.getHeaders` method.

Since:

1.3

SELECTION_FILTER_ATTRIBUTE

public static final String SELECTION_FILTER_ATTRIBUTE = "selection-filter"

Manifest header attribute is used for selection by filtering based upon system properties.

The attribute value is encoded in manifest headers like:

`Bundle-NativeCode: libgtk.so; selection-filter="(ws=gtk)"; ...`

Since:

1.3

See Also:

BUNDLE_MANIFESTVERSION [BUNDLE_NATIVECODE](#)

public static final String BUNDLE_MANIFESTVERSION = "Bundle-ManifestVersion"

Manifest header identifying the bundle manifest version. A bundle manifest may express the version of the syntax in which it is written by specifying a bundle manifest version. Bundles exploiting OSGi Release 4, or later, syntax must specify a bundle manifest version.

The bundle manifest version defined by OSGi Release 4 or, more specifically, by version 1.3 of the OSGi Core Specification is "2".

The header value may be retrieved from the `Dictionary` object returned by the `Bundle.getHeaders` method.

Since:

1.3

VERSION_ATTRIBUTE

public static final String VERSION_ATTRIBUTE = "version"

Manifest header attribute identifying the version of a package specified in the Export-Package or Import-Package manifest header.

The attribute value is encoded in the Export-Package or Import-Package manifest header like:

```
Export-Package: org.osgi.framework; version="1.1"
```

Since:

1.3

See Also:

[BUNDLE_SYMBOLICNAME_ATTRIBUTE](#)

```
public static final String BUNDLE_SYMBOLICNAME_ATTRIBUTE = "bundle-symbolic-name"
```

Manifest header attribute identifying the symbolic name of a bundle that exports a package specified in the Import-Package manifest header.

The attribute value is encoded in the Import-Package manifest header like:

```
Import-Package: org.osgi.framework;  
bundle-symbolic-name="com.acme.module.test"
```

Since:

1.3

See Also:

[RESOLUTION_DIRECTIVE](#)

```
public static final String RESOLUTION_DIRECTIVE = "resolution"
```

Manifest header directive identifying the resolution type in the Import-Package, Require-Bundle or Require-Capability manifest header. The default value is [mandatory](#).

The directive value is encoded in the Import-Package, Require-Bundle or Require-Capability manifest header like:

```
Import-Package: org.osgi.framework; resolution="optional"  
Require-Bundle: com.acme.module.test; resolution="optional"  
Require-Capability: com.acme.capability; resolution="optional"
```

Since:

1.3

See Also:

[IMPORT_PACKAGE](#), [REQUIRE_BUNDLE](#), [REQUIRE_CAPABILITY](#), [RESOLUTION_MANDATORY](#),

[RESOLUTION_MANDATORY](#)

```
public static final String RESOLUTION_MANDATORY = "mandatory"
```

Manifest header directive value identifying a mandatory resolution type. A mandatory resolution type indicates that the import package, require bundle or require capability must be resolved when the bundle is resolved. If such an import, require bundle or require capability cannot be resolved, the module fails to resolve.

The directive value is encoded in the Import-Package, Require-Bundle or Require-Capability manifest header like:

```
Import-Package: org.osgi.framework; resolution="mandatory"  
Require-Bundle: com.acme.module.test; resolution="mandatory"  
Require-Capability: com.acme.capability; resolution="mandatory"
```

Since:

1.3

See Also:

[RESOLUTION_OPTIONAL](#)

```
public static final String RESOLUTION_OPTIONAL = "optional"
```

Manifest header directive value identifying an optional resolution type. An optional resolution type indicates that the import, require bundle or require capability is optional and the bundle may be resolved without the import, require bundle or require capability being resolved. If the import, require bundle or require capability is not resolved when the bundle is resolved, the import, require bundle or require capability may not be resolved until the bundle is refreshed.

The directive value is encoded in the Import-Package, Require-Bundle or Require-Capability manifest header like:

```
Import-Package: org.osgi.framework; resolution="optional"  
Require-Bundle: com.acme.module.test; resolution="optional"  
Require-Capability: com.acme.capability; resolution="optional"
```

Since:

1.3

See Also:

[USES_DIRECTIVE](#)

```
public static final String USES_DIRECTIVE = "uses"
```

Manifest header directive identifying a list of packages that an exported package or provided capability uses.

The directive value is encoded in the Export-Package or Provide-Capability manifest header like:


```
Export-Package: org.osgi.util.tracker; uses:="org.osgi.framework"  
Provide-Capability: com.acme.capability; uses:="com.acme.service"
```

Since:

1.3

See Also:

[INCLUDE_DIRECTIVE](#), [EXPORT_PACKAGE](#), [PROVIDE_CAPABILITY](#)

```
public static final String INCLUDE_DIRECTIVE = "include"
```

Manifest header directive identifying a list of classes to include in the exported package.

This directive is used by the Export-Package manifest header to identify a list of classes of the specified package which must be allowed to be exported. The directive value is encoded in the Export-Package manifest header like:

```
Export-Package: org.osgi.framework; include:="MyClass*"
```

This directive is also used by the Bundle-ActivationPolicy manifest header to identify the packages from which class loads will trigger lazy activation. The directive value is encoded in the Bundle-ActivationPolicy manifest header like:

```
Bundle-ActivationPolicy: lazy; include:="org.osgi.framework"
```

Since:

1.3

See Also:

[EXCLUDE_DIRECTIVE](#), [EXPORT_PACKAGE](#), [BUNDLE_ACTIVATIONPOLICY](#)

```
public static final String EXCLUDE_DIRECTIVE = "exclude"
```

Manifest header directive identifying a list of classes to exclude in the exported package..

This directive is used by the Export-Package manifest header to identify a list of classes of the specified package which must not be allowed to be exported. The directive value is encoded in the Export-Package manifest header like:

```
Export-Package: org.osgi.framework; exclude:="*Impl"
```

This directive is also used by the Bundle-ActivationPolicy manifest header to identify the packages from which class loads will not trigger lazy activation. The directive value is encoded in the Bundle-ActivationPolicy manifest header like:

```
Bundle-ActivationPolicy: lazy; exclude:="org.osgi.framework"
```

Since:

1.3

See Also:

[MANDATORY_DIRECTIVE](#), [EXPORT_PACKAGE](#), [BUNDLE_ACTIVATIONPOLICY](#)

```
public static final String MANDATORY_DIRECTIVE = "mandatory"
```

Manifest header directive identifying names of matching attributes which must be specified by matching Import-Package statements in the Export-Package manifest header.

The directive value is encoded in the Export-Package manifest header like:

```
Export-Package: org.osgi.framework; mandatory:="bundle-symbolic-name"
```

Since:

1.3

See Also:

[VISIBILITY_DIRECTIVE](#), [EXPORT_PACKAGE](#)

```
public static final String VISIBILITY_DIRECTIVE = "visibility"
```

Manifest header directive identifying the visibility of a required bundle in the Require-Bundle manifest header. The default value is [private](#).

The directive value is encoded in the Require-Bundle manifest header like:

```
Require-Bundle: com.acme.module.test; visibility:="reexport"
```

Since:

1.3

See Also:

[VISIBILITY_PRIVATE](#), [REQUIRE_BUNDLE](#), [VISIBILITY_PRIVATE](#), [VISIBILITY_REEXPORT](#)

```
public static final String VISIBILITY_PRIVATE = "private"
```

Manifest header directive value identifying a private visibility type. A private visibility type indicates that any packages that are exported by the required bundle are not made visible on the export signature of the requiring bundle.

The directive value is encoded in the Require-Bundle manifest header like:

```
Require-Bundle: com.acme.module.test; visibility:="private"
```

Since:

1.3

See Also:

[VISIBILITY_REEXPORT](#), [VISIBILITY_DIRECTIVE](#)

```
public static final String VISIBILITY_REEXPORT = "reexport"
```

Manifest header directive value identifying a reexport visibility type. A reexport visibility type indicates any packages that are exported by the required bundle are re-exported by the requiring bundle. Any arbitrary attributes with which they were exported by the required bundle are deleted.

The directive value is encoded in the Require-Bundle manifest header like:

```
Require-Bundle: com.acme.module.test; visibility="reexport"
```

Since:

1.3

See Also:

[EXTENSION_DIRECTIVE](#)

```
public static final String EXTENSION_DIRECTIVE = "extension"
```

Manifest header directive identifying the type of the extension fragment.

The directive value is encoded in the Fragment-Host manifest header like:

```
Fragment-Host: system.bundle; extension="framework"
```

Since:

1.3

See Also:

[EXTENSION_FRAMEWORK](#), [EXTENSION_BOOTCLASSPATH](#)

```
public static final String EXTENSION_FRAMEWORK = "framework"
```

Manifest header directive value identifying the type of extension fragment. An extension fragment type of framework indicates that the extension fragment is to be loaded by the framework's class loader.

The directive value is encoded in the Fragment-Host manifest header like:

```
Fragment-Host: system.bundle; extension="framework"
```

Since:

1.3

See Also:

[EXTENSION_BOOTCLASSPATH](#)

```
public static final String EXTENSION_BOOTCLASSPATH = "bootclasspath"
```

Manifest header directive value identifying the type of extension fragment. An extension fragment type of bootclasspath indicates that the extension fragment is to be loaded by the boot class loader.

The directive value is encoded in the Fragment-Host manifest header like:

```
Fragment-Host: system.bundle; extension="bootclasspath"
```

Since:

1.3

See Also:

[BUNDLE_ACTIVATIONPOLICY](#)

```
public static final String BUNDLE_ACTIVATIONPOLICY = "Bundle-ActivationPolicy"
```

Manifest header identifying the bundle's activation policy.

The header value may be retrieved from the `Dictionary` object returned by the `Bundle.getHeaders` method.

Since:

1.4

See Also:

[ACTIVATION_LAZY](#), [INCLUDE_DIRECTIVE](#), [EXCLUDE_DIRECTIVE](#)

```
public static final String ACTIVATION_LAZY = "lazy"
```

Bundle activation policy declaring the bundle must be activated when the first class load is made from the bundle.

A bundle with the lazy activation policy that is started with the `START_ACTIVATION_POLICY` option will wait in the `STARTING` state until the first class load from the bundle occurs. The bundle will then be activated before the class is returned to the requester.

The activation policy value is specified as in the Bundle-ActivationPolicy manifest header like:

```
Bundle-ActivationPolicy: lazy
```

Since:

1.4

See Also:

[FRAMEWORK_VERSION](#), [BUNDLE_ACTIVATIONPOLICY](#), [org.osgi.framework.Bundle.start\(int\), org.osgi.framework.Bundle.START_ACTIVATION_POLICY](#)

```
public static final String FRAMEWORK_VERSION = "org.osgi.framework.version"
```

Framework environment property identifying the Framework version.

The value of this property may be retrieved by calling the `BundleContext.getProperty` method.

```
public static final String FRAMEWORK_VENDOR = "org.osgi.framework.vendor"
```

Framework environment property identifying the Framework implementation vendor.

The value of this property may be retrieved by calling the `BundleContext.getProperty` method.

```
public static final String FRAMEWORK_LANGUAGE = "org.osgi.framework.language"
```

Framework launching property identifying the Framework implementation language (see ISO 639 for possible values).

The value of this property may be retrieved by calling the `BundleContext.getProperty` method.

```
public static final String FRAMEWORK_OS_NAME = "org.osgi.framework.os.name"
```

Framework launching property identifying the Framework host-computer's operating system.

The value of this property may be retrieved by calling the `BundleContext.getProperty` method.

```
public static final String FRAMEWORK_OS_VERSION = "org.osgi.framework.os.version"
```

Framework launching property identifying the Framework host-computer's operating system version number.

The value of this property may be retrieved by calling the `BundleContext.getProperty` method.

```
public static final String FRAMEWORK_PROCESSOR = "org.osgi.framework.processor"
```

Framework launching property identifying the Framework host-computer's processor name.

The value of this property may be retrieved by calling the `BundleContext.getProperty` method.

```
public static final String FRAMEWORK_EXECUTIONENVIRONMENT = "org.osgi.framework.executionenvironment"
```

Deprecated.

Framework launching property identifying execution environments provided by the Framework.

The value of this property may be retrieved by calling the `BundleContext.getProperty` method.

Since:

FRAMEWORK_1.2 BOOTDELEGATION

```
public static final String FRAMEWORK_BOOTDELEGATION = "org.osgi.framework.bootdelegation"
```

Framework launching property identifying packages for which the Framework must delegate class loading to the parent class loader of the bundle.

The value of this property may be retrieved by calling the `BundleContext.getProperty` method.

Since:

1.3

See Also:

FRAMEWORK_1.2 SYSTEMPACKAGES

```
public static final String FRAMEWORK_SYSTEMPACKAGES = "org.osgi.framework.system.packages"
```

Framework launching property identifying packages which the system bundle must export.

If this property is not specified then the framework must calculate a reasonable default value for the current execution environment.

The value of this property may be retrieved by calling the `BundleContext.getProperty` method.

Since:

FRAMEWORK_1.3 SYSTEMPACKAGES_EXTRA

```
public static final String FRAMEWORK_SYSTEMPACKAGES_EXTRA = "org.osgi.framework.system.packages.extra"
```

Framework launching property identifying extra packages which the system bundle must export from the current execution environment.

This property is useful for configuring extra system packages in addition to the system packages calculated by the framework.

The value of this property may be retrieved by calling the `BundleContext.getProperty` method.

Since:

1.5

See Also:

SUPPORTS_1.3 FRAMEWORK_EXTENSION

```
public static final String SUPPORTS_FRAMEWORK_EXTENSION = "org.osgi.supports.framework.extension"
```

Framework environment property identifying whether the Framework supports framework extension bundles.

As of version 1.4, the value of this property must be `true`. The Framework must support framework extension bundles.

The value of this property may be retrieved by calling the `BundleContext.getProperty` method.

Since:

SUPPORTS_1.3 BOOTCLASSPATH_EXTENSION

```
public static final String SUPPORTS_BOOTCLASSPATH_EXTENSION = "org.osgi.supports.bootclasspath.extension"
```

Framework environment property identifying whether the Framework supports bootclasspath extension bundles.

If the value of this property is `true`, then the Framework supports bootclasspath extension bundles. The default value is `false`.

The value of this property may be retrieved by calling the `BundleContext.getProperty` method.

Since:

SUPPORTS_FRAMEWORK_FRAGMENT

```
public static final String SUPPORTS_FRAMEWORK_FRAGMENT =  
"org.osgi.supports.framework.fragment"
```

Framework environment property identifying whether the Framework supports fragment bundles.

As of version 1.4, the value of this property must be `true`. The Framework must support fragment bundles.

The value of this property may be retrieved by calling the `BundleContext.getProperty` method.

Since:

SUPPORTS_FRAMEWORK_REQUIREBUNDLE

```
public static final String SUPPORTS_FRAMEWORK_REQUIREBUNDLE =  
"org.osgi.supports.framework.requirebundle"
```

Framework environment property identifying whether the Framework supports the [Require-Bundle](#) manifest header.

As of version 1.4, the value of this property must be `true`. The Framework must support the `Require-Bundle` manifest header.

The value of this property may be retrieved by calling the `BundleContext.getProperty` method.

Since:

FRAMEWORK_SECURITY

```
public static final String FRAMEWORK_SECURITY = "org.osgi.framework.security"
```

Framework launching property specifying the type of security manager the framework must use. If not specified then the framework will not set the VM security manager.

Since:

1.5

See Also:

FRAMEWORK_SECURITY_OSGI

```
public static final String FRAMEWORK_SECURITY_OSGI = "osgi"
```

Specifies that a security manager that supports all security aspects of the OSGi core specification including postponed conditions must be installed.

If this value is specified and there is a security manager already installed, then a `SecurityException` must be thrown when the Framework is initialized.

Since:

1.5

See Also:

FRAMEWORK_STORAGE_SECURITY

```
public static final String FRAMEWORK_STORAGE = "org.osgi.framework.storage"
```

Framework launching property specifying the persistent storage area used by the framework. The value of this property must be a valid file path in the file system to a directory. If the specified directory does not exist then the framework will create the directory. If the specified path exists but is not a directory or if the framework fails to create the storage directory, then framework initialization must fail. The framework is free to use this directory as it sees fit. This area can not be shared with anything else.

If this property is not set, the framework should use a reasonable platform default for the persistent storage area.

Since:

FRAMEWORK_STORAGE_CLEAN

```
public static final String FRAMEWORK_STORAGE_CLEAN = "org.osgi.framework.storage.clean"
```

Framework launching property specifying if and when the persistent storage area for the framework should be cleaned. If this property is not set, then the framework storage area must not be cleaned.

Since:

1.5

See Also:

FRAMEWORK_STORAGE_CLEAN_ONFIRSTINIT

```
public static final String FRAMEWORK_STORAGE_CLEAN_ONFIRSTINIT = "onFirstInit"
```

Specifies that the framework storage area must be cleaned before the framework is initialized for the first time. Subsequent inits, starts or updates of the framework will not result in cleaning the framework storage area.

Since:

FRAMEWORK_LIBRARY_EXTENSIONS

```
public static final String FRAMEWORK_LIBRARY_EXTENSIONS =  
"org.osgi.framework.library.extensions"
```

Framework launching property specifying a comma separated list of additional library file extensions that must be used when a bundle's class loader is searching for native libraries. If this property is not set, then only the library name returned by `System.mapLibraryName(String)` will be used to search. This is needed for certain operating systems which allow more than one extension for a library. For example, AIX

allows library extensions of `.a` and `.so`, but `System.mapLibraryName(String)` will only return names with the `.a` extension.

Since:

FRAMEWORK_1.5EXECPERMISSION

```
public static final String FRAMEWORK_EXECPERMISSION =  
"org.osgi.framework.command.execpermission"
```

Framework launching property specifying an optional OS specific command to set file permissions on extracted native code. On some operating systems, it is required that native libraries be set to executable. This optional property allows you to specify the command. For example, on a UNIX style OS, this property could have the following value.

```
chmod +rx ${abspath}
```

The `${abspath}` is used by the framework to substitute the actual absolute file path.

Since:

FRAMEWORK_1.5COMMAND_ABSPATH

```
public static final String FRAMEWORK_COMMAND_ABSPATH = "abspath"
```

Specified the substitution string for the absolute path of a file.

Since:

1.6

See Also:

FRAMEWORK_1.5TRUST_REPOSITORIES

```
public static final String FRAMEWORK_TRUST_REPOSITORIES =  
"org.osgi.framework.trust.repositories"
```

Framework launching property specifying the trust repositories used by the framework. The value is a `java.io.File.pathSeparator` separated list of valid file paths to files that contain key stores. Key stores of type `JKS` must be supported and other key store types may be supported. The framework will use the key stores as trust repositories to authenticate certificates of trusted signers. The key stores are only used as read-only trust repositories to access public keys. No passwords are required to access the key stores' public keys.

Note that framework implementations are allowed to use other trust repositories in addition to the trust repositories specified by this property. How these other trust repositories are configured and populated is implementation specific.

Since:

FRAMEWORK_1.5WINDOWSYSTEM

```
public static final String FRAMEWORK_WINDOWSYSTEM = "org.osgi.framework.windowssystem"
```

Framework launching property specifying the current windowing system. The framework should provide a reasonable default if this is not set.

Since:

FRAMEWORK_1.5BEGINNING_STARTLEVEL

```
public static final String FRAMEWORK_BEGINNING_STARTLEVEL =  
"org.osgi.framework.startlevel.beginning"
```

Framework launching property specifying the beginning start level of the framework.

Since:

1.5

See Also:

FRAMEWORK_1.5BUNDLE_PARENT

```
public static final String FRAMEWORK_BUNDLE_PARENT = "org.osgi.framework.bundle.parent"
```

Framework launching property specifying the parent class loader type for all bundle class loaders. Default value is [boot](#).

Since:

1.5

See Also:

[FRAMEWORK_BUNDLE_PARENT_BOOT](#),

[FRAMEWORK_BUNDLE_PARENT_EXT](#),

FRAMEWORK_1.5BUNDLE_PARENT_BOOT

```
public static final String FRAMEWORK_BUNDLE_PARENT_BOOT = "boot"
```

Specifies to use of the boot class loader as the parent class loader for all bundle class loaders.

Since:

1.5

See Also:

FRAMEWORK_1.5BUNDLE_PARENT_EXT

```
public static final String FRAMEWORK_BUNDLE_PARENT_EXT = "ext"
```

Specifies to use the extension class loader as the parent class loader for all bundle class loaders.

Since:

1.5

See Also:

FRAMEWORK_BUNDLE_PARENT_APP

```
public static final String FRAMEWORK_BUNDLE_PARENT_APP = "app"
```

Specifies to use the application class loader as the parent class loader for all bundle class loaders. Depending on how the framework is launched, this may refer to the same class loader as [FRAMEWORK_BUNDLE_PARENT_FRAMEWORK](#).

Since:

1.5

See Also:

FRAMEWORK_BUNDLE_PARENT_FRAMEWORK

```
public static final String FRAMEWORK_BUNDLE_PARENT_FRAMEWORK = "framework"
```

Specifies to use the framework class loader as the parent class loader for all bundle class loaders. The framework class loader is the class loader used to load the framework implementation. Depending on how the framework is launched, this may refer to the same class loader as [FRAMEWORK_BUNDLE_PARENT_APP](#).

Since:

1.5

See Also:

OBJECTCLASS

```
public static final String OBJECTCLASS = "objectClass"
```

Service property identifying all of the class names under which a service was registered in the Framework. The value of this property must be of type `String[]`.

This property is set by the Framework when a service is registered.

SERVICE_ID

```
public static final String SERVICE_ID = "service.id"
```

Service property identifying a service's registration number. The value of this property must be of type

`Long`.

The value of this property is assigned by the Framework when a service is registered. The Framework assigns a unique value that is larger than all previously assigned values since the Framework was started.

These values are NOT persistent across restarts of the Framework.

SERVICE_PID

```
public static final String SERVICE_PID = "service.pid"
```

Service property identifying a service's persistent identifier.

This property may be supplied in the `propertiesDictionary` object passed to the `BundleContext.registerService` method. The value of this property must be of type `String`, `String[]`, or `Collection of String`.

A service's persistent identifier uniquely identifies the service and persists across multiple Framework invocations.

By convention, every bundle has its own unique namespace, starting with the bundle's identifier (see `org.osgi.framework.Bundle.getId()`) and followed by a dot (`.`). A bundle may use this as the prefix of the persistent identifiers for the services it registers.

SERVICE_RANKING

```
public static final String SERVICE_RANKING = "service.ranking"
```

Service property identifying a service's ranking number.

This property may be supplied in the `propertiesDictionary` object passed to the `BundleContext.registerService` method. The value of this property must be of type `Integer`.

The service ranking is used by the Framework to determine the *natural order* of services, see `org.osgi.framework.ServiceReference.compareTo(Object)`, and the *default* service to be returned from a call to the [BundleContext.getServiceReference\(Class\)](#) or [BundleContext.getServiceReference\(String\)](#) method.

The default ranking is zero (0). A service with a ranking of `Integer.MAX_VALUE` is very likely to be returned as the default service, whereas a service with a ranking of `Integer.MIN_VALUE` is very unlikely to be returned.

If the supplied property value is not of type `Integer`, it is deemed to have a ranking value of zero.

SERVICE_VENDOR

```
public static final String SERVICE_VENDOR = "service.vendor"
```

Service property identifying a service's vendor.

This property may be supplied in the `propertiesDictionary` object passed to the

SERVICE_DESCRIPTION

```
public static final String SERVICE_DESCRIPTION = "service.description"
```

Service property identifying a service's description.

This property may be supplied in the `propertiesDictionary` object passed to the

SERVICE_SCOPE

```
public static final String SERVICE_SCOPE = "service.scope"
```

Service property identifying a service's scope.

This property is set by the Framework when a service is registered. If the registered object implements [PrototypeServiceFactory](#), then the value of this service property will be [SCOPE_PROTOTYPE](#). Otherwise, if

the registered object implements [ServiceFactory](#), then the value of this service property will be [SCOPE_BUNDLE](#). Otherwise, the value of this service property will be [SCOPE_SINGLETON](#).

Since:

1.8

See Also:

[SCOPE_SINGLETON](#), [SCOPE_BUNDLE](#), [SCOPE_PROTOTYPE](#)

`public static final String SCOPE_SINGLETON = "singleton"`

Service scope is singleton. All bundles using the service receive the same service object.

Since:

1.8

See Also:

[SCOPE_BUNDLE](#), [SERVICE_SCOPE](#)

`public static final String SCOPE_BUNDLE = "bundle"`

Service scope is bundle. Each bundle using the service receives a distinct service object.

Since:

1.8

See Also:

[SCOPE_PROTOTYPE](#), [SERVICE_SCOPE](#)

`public static final String SCOPE_PROTOTYPE = "prototype"`

Service scope is prototype. Each bundle using the service receives either a distinct service object or can request multiple distinct service objects via [ServiceObjects](#).

Since:

1.8

See Also:

[FRAMEWORK_UUID](#), [SERVICE_SCOPE](#)

`public static final String FRAMEWORK_UUID = "org.osgi.framework.uuid"`

Framework environment property identifying the Framework's universally unique identifier (UUID). A UUID represents a 128-bit value. A new UUID is generated by the `org.osgi.framework.launch.Framework.init()` method each time a framework is initialized. The value of this property must conform to the UUID string representation specified in [RFC 4122](#).

The value of this property may be retrieved by calling the `BundleContext.getProperty` method.

Since:

1.6

[REMOTE_CONFIGS_SUPPORTED](#)

`public static final String REMOTE_CONFIGS_SUPPORTED = "remote.configs.supported"`

Service property identifying the configuration types supported by a distribution provider. Registered by the distribution provider on one of its services to indicate the supported configuration types.

The value of this property must be of type `String`, `String[]`, or `Collection of String`.

Since:

1.6

See Also:

["Remote Services Specification"](#)

`public static final String REMOTE_INTENTS_SUPPORTED = "remote.intents.supported"`

Service property identifying the intents supported by a distribution provider. Registered by the distribution provider on one of its services to indicate the vocabulary of implemented intents.

The value of this property must be of type `String`, `String[]`, or `Collection of String`.

Since:

1.6

See Also:

["Remote Services Specification"](#)

`public static final String SERVICE_EXPORTED_CONFIGS = "service.exported.configs"`

Service property identifying the configuration types that should be used to export the service. Each configuration type represents the configuration parameters for an endpoint. A distribution provider should create an endpoint for each configuration type that it supports.

This property may be supplied in the `propertiesDictionary` object passed to the `BundleContext.registerService` method. The value of this property must be of type `String`, `String[]`, or `Collection of String`.

Since:

1.6

See Also:

["Remote Services Specification"](#)

`public static final String SERVICE_EXPORTED_INTENTS = "service.exported.intents"`

Service property identifying the intents that the distribution provider must implement to distribute the service. Intents listed in this property are reserved for intents that are critical for the code to function correctly, for example, ordering of messages. These intents should not be configurable.

This property may be supplied in the `propertiesDictionary` object passed to the `BundleContext.registerService` method. The value of this property must be of type `String`, `String[]`, or `Collection of String`.

Since:

1.6

See Also:

["Remote Services Specification"](#)

```
public static final String SERVICE_EXPORTED_INTENTS_EXTRA = "service.exported.intents.extra"
```

Service property identifying the extra intents that the distribution provider must implement to distribute the service. This property is merged with the `service.exported.intents` property before the distribution provider interprets the listed intents; it has therefore the same semantics but the property should be configurable so the administrator can choose the intents based on the topology. Bundles should therefore make this property configurable, for example through the Configuration Admin service.

This property may be supplied in the `propertiesDictionary` object passed to the `BundleContext.registerService` method. The value of this property must be of type `String`, `String[]`, or `Collection of String`.

Since:

1.6

See Also:

["Remote Services Specification"](#)

```
public static final String SERVICE_EXPORTED_INTERFACES = "service.exported.interfaces"
```

Service property marking the service for export. It defines the interfaces under which this service can be exported. This list must be a subset of the types under which the service was registered. The single value of an asterisk (`'*' \u002A`) indicates all the interface types under which the service was registered excluding the non-interface types. It is strongly recommended to only export interface types and not concrete classes due to the complexity of creating proxies for some type of concrete classes.

This property may be supplied in the `propertiesDictionary` object passed to the `BundleContext.registerService` method. The value of this property must be of type `String`, `String[]`, or `Collection of String`.

Since:

1.6

See Also:

["Remote Services Specification"](#)

```
public static final String SERVICE_IMPORTED = "service.imported"
```

Service property identifying the service as imported. This service property must be set by a distribution provider to any value when it registers the endpoint proxy as an imported service. A bundle can use this property to filter out imported services.

The value of this property may be of any type.

Since:

1.6

See Also:

["Remote Services Specification"](#)

```
public static final String SERVICE_IMPORTED_CONFIGS = "service.imported.configs"
```

Service property identifying the configuration types used to import the service. Any associated properties for this configuration types must be properly mapped to the importing system. For example, a URL in these properties must point to a valid resource when used in the importing framework. If multiple configuration types are listed in this property, then they must be synonyms for exactly the same remote endpoint that is used to export this service.

The value of this property must be of type `String`, `String[]`, or `Collection of String`.

Since:

1.6

See Also:

["Remote Services Specification"](#), [SERVICE_EXPORTED_CONFIGS](#)

```
public static final String SERVICE_INTENTS = "service.intents"
```

Service property identifying the intents that this service implement. This property has a dual purpose:

A bundle can use this service property to notify the distribution provider that these intents are already implemented by the exported service object.

A distribution provider must use this property to convey the combined intents of: the exporting service, the intents that the exporting distribution provider adds, and the intents that the importing distribution provider adds.

To export a service, a distribution provider must expand any qualified intents. Both the exporting and importing distribution providers must recognize all intents before a service can be distributed.

The value of this property must be of type `String`, `String[]`, or `Collection of String`.

Since:

1.6

See Also:

["Remote Services Specification"](#)

PROVIDE_CAPABILITY

```
public static final String PROVIDE_CAPABILITY = "Provide-Capability"
```

Manifest header identifying the capabilities that the bundle offers to provide to other bundles.

The header value may be retrieved from the `Dictionary` object returned by the `Bundle.getHeaders` method.

Since:

1.6

REQUIRE_CAPABILITY

```
public static final String REQUIRE_CAPABILITY = "Require-Capability"
```

Manifest header identifying the capabilities on which the bundle depends.

The header value may be retrieved from the `Dictionary` object returned by the `Bundle.getHeaders` method.

Since:

1.6

EFFECTIVE_DIRECTIVE

```
public static final String EFFECTIVE_DIRECTIVE = "effective"
```

Manifest header directive identifying the effective time of the provided capability. The default value is [resolve](#).

The directive value is encoded in the Provide-Capability manifest header like:

```
Provide-Capability: com.acme.capability; effective:="resolve"
```

Since:

1.6

See Also:

[PROVIDE_CAPABILITY](#), [EFFECTIVE_RESOLVE](#), [EFFECTIVE_ACTIVE](#)

EFFECTIVE_RESOLVE

```
public static final String EFFECTIVE_RESOLVE = "resolve"
```

Manifest header directive value identifying a capability that is effective at resolve time. Capabilities with an effective time of resolve are the only capabilities which are processed by the resolver.

The directive value is encoded in the Provide-Capability manifest header like:

```
Provide-Capability: com.acme.capability; effective:="resolve"
```

Since:

1.6

See Also:

[EFFECTIVE_DIRECTIVE](#)

EFFECTIVE_ACTIVE

```
public static final String EFFECTIVE_ACTIVE = "active"
```

Manifest header directive value identifying a capability that is effective at active time. Capabilities with an effective time of active are ignored by the resolver.

The directive value is encoded in the Provide-Capability manifest header like:

```
Provide-Capability: com.acme.capability; effective:="active"
```

Since:

1.6

See Also:

[EFFECTIVE_DIRECTIVE](#)

FILTER_DIRECTIVE

```
public static final String FILTER_DIRECTIVE = "filter"
```

Manifest header directive identifying the capability filter specified in the Require-Capability manifest header. The directive value is encoded in the Require-Capability manifest header like:

```
Require-Capability: com.acme.capability; filter:="(someattr=somevalue)"
```

Since:

1.6

See Also:

FRAMEWORK_SYSTEMCAPABILITIES

```
public static final String FRAMEWORK_SYSTEMCAPABILITIES =  
"org.osgi.framework.system.capabilities"
```

Framework launching property identifying capabilities which the system bundle must provide.

If this property is not specified then the framework must calculate a reasonable default value for the current execution environment.

The value of this property may be retrieved by calling the `BundleContext.getProperty` method.

Since:

1.6

~~FRAMEWORK_SYSTEMCAPABILITIES_EXTRA~~

```
public static final String FRAMEWORK_SYSTEMCAPABILITIES_EXTRA =  
"org.osgi.framework.system.capabilities.extra"
```

Framework launching property identifying extra capabilities which the system bundle must additionally provide.

This property is useful for configuring extra system capabilities in addition to the system capabilities calculated by the framework.

The value of this property may be retrieved by calling the `BundleContext.getProperty` method.

Since:

1.6

See Also:

~~FRAMEWORK_BSNVERSION~~ [FRAMEWORK_SYSTEMCAPABILITIES](#)

```
public static final String FRAMEWORK_BSNVERSION = "org.osgi.framework.bsnversion"
```

Framework launching property specifying whether multiple bundles having the same [symbolic name](#) and [version](#) may be installed.

Default value is [managed](#) in this release of the specification. This default may change in a future specification release. Therefore, code must not assume the default behavior is `managed` and should interrogate the value of this property to determine the behavior.

The value of this property may be retrieved by calling the `BundleContext.getProperty` method.

Since:

1.6

See Also:

[FRAMEWORK_BSNVERSION_MULTIPLE](#),

[FRAMEWORK_BSNVERSION_SINGLE](#),

~~FRAMEWORK_BSNVERSION_MULTIPLE~~ [FRAMEWORK_BSNVERSION_MANAGED](#)

```
public static final String FRAMEWORK_BSNVERSION_MULTIPLE = "multiple"
```

Specifies the framework will allow multiple bundles to be installed having the same symbolic name and version.

Since:

1.6

See Also:

~~FRAMEWORK_BSNVERSION_SINGLE~~ [FRAMEWORK_BSNVERSION](#)

```
public static final String FRAMEWORK_BSNVERSION_SINGLE = "single"
```

Specifies the framework will only allow a single bundle to be installed for a given symbolic name and version. It will be an error to install a bundle or update a bundle to have the same symbolic name and version as another installed bundle.

Since:

1.6

See Also:

~~FRAMEWORK_BSNVERSION_MANAGED~~ [FRAMEWORK_BSNVERSION](#), `org.osgi.framework.BundleException.DUPLICATE_BUNDLE_ERROR`

```
public static final String FRAMEWORK_BSNVERSION_MANAGED = "managed"
```

Specifies the framework must consult the bundle collision hook services to determine if it will be an error to install a bundle or update a bundle to have the same symbolic name and version as another installed bundle. If no bundle collision hook services are registered, then it will be an error to install a bundle or update a bundle to have the same symbolic name and version as another installed bundle.

Since:

1.7

See Also:

[FRAMEWORK_BSNVERSION](#), `org.osgi.framework.BundleException.DUPLICATE_BUNDLE_ERROR`

Interface *PrototypeServiceFactory*
[org.osgi.framework](#)
Type Parameters:
S - Type of Service
All Superinterfaces:
[ServiceFactory](#)<S>

public interface *PrototypeServiceFactory*
extends [ServiceFactory](#)<S>

A factory for [prototype_scope](#) services. The factory can provide multiple, unique service objects in the OSGi environment.

When registering a service, a *PrototypeServiceFactory* object can be used instead of a service object, so that the bundle developer can create a unique service object for each caller that is using the service.

When a caller uses a [ServiceObjects](#) to [request](#) a service instance, the framework calls the [getService](#) method to return a service object specifically for the requesting caller. The caller can [release](#) the returned service object and the framework will call the [ungetService](#) method with the service object.

When a bundle uses the [BundleContext.getService\(ServiceReference\)](#) method to obtain a service object, the framework must act as if the service has [bundle_scope](#). That is, the framework will call the [getService](#) method to obtain a bundle-scoped instance which will be cached and have a use count. See [ServiceFactory](#).

A bundle can use both [ServiceObjects](#) and [BundleContext.getService\(ServiceReference\)](#) to obtain a service object for a service. [ServiceObjects.getService\(\)](#) will always return an instance provided by a call to [getService\(Bundle, ServiceRegistration\)](#) and [BundleContext.getService\(ServiceReference\)](#) will always return the bundle-scoped instance.

PrototypeServiceFactory objects are only used by the Framework and are not made available to other bundles in the OSGi environment. The Framework may concurrently call a *PrototypeServiceFactory*.

Since:

1.8

Version:

\$Id: 361182d1086ef443cc901a72f3a8d6995bc47cd7 \$

See Also:

[BundleContext.getServiceObjects\(ServiceReference\)](#), [ServiceObjects](#)

ThreadSafe

Method Summary		Page
getService (org.osgi.framework.Bundle bundle, org.osgi.framework.ServiceRegistration< S > registration)	Returns a service object for a caller.	rror: Reference source not found
void ungetService (org.osgi.framework.Bundle bundle, org.osgi.framework.ServiceRegistration< S > registration, S service)	Releases a service object created for a caller.	Error: Reference source not found

Method Detail

getService

[S](#) getService(org.osgi.framework.Bundle bundle, org.osgi.framework.ServiceRegistration<[S](#)> registration)

Returns a service object for a caller.

The Framework invokes this method for each caller requesting a service object using [ServiceObjects.getService\(\)](#). The factory can then return a specific service object for the caller.

The Framework must check that the returned service object is valid. If the returned service object is null or is not an instanceof all the classes named when the service was registered, a framework event of type org.osgi.framework.FrameworkEvent.ERROR is fired containing a service exception of type org.osgi.framework.ServiceException.FACTORY_ERROR and null is returned to the caller. If this method throws an exception, a framework event of type org.osgi.framework.FrameworkEvent.ERROR is fired containing a service exception of type

`org.osgi.framework.ServiceException.FACTORY_EXCEPTION` with the thrown exception as the cause and `null` is returned to the caller.

Specified by:

[getService](#) in interface [ServiceFactory](#)

Parameters:

`bundle` - The bundle requesting the service.

`registration` - The `ServiceRegistration` object for the requested service.

Returns:

A service object that must be an instance of all the classes named when the service was registered.

See Also:

[ungetService](#) [ServiceObjects.getService\(\)](#)

```
void ungetService(org.osgi.framework.Bundle bundle,  
                 org.osgi.framework.ServiceRegistration<S> registration,  
                 S service)
```

Releases a service object created for a caller.

The Framework invokes this method when a service has been released by a bundle such as by calling [ServiceObjects.ungetService\(Object\)](#). The service object may then be destroyed.

If this method throws an exception, a framework event of type `org.osgi.framework.FrameworkEvent.ERROR` is fired containing a service exception of type `org.osgi.framework.ServiceException.FACTORY_EXCEPTION` with the thrown exception as the cause.

Specified by:

[ungetService](#) in interface [ServiceFactory](#)

Parameters:

`bundle` - The bundle releasing the service.

`registration` - The `ServiceRegistration` object for the service being released.

`service` - The service object returned by a previous call to the [getService](#) method.

See Also:

[ServiceObjects.ungetService\(Object\)](#)

Interface ServiceFactory

[org.osgi.framework](#)

Type Parameters:

S - Type of Service

All Known Subinterfaces:

[PrototypeServiceFactory](#)

public interface ServiceFactory

A factory for [bundle scope](#) services. The factory can provide service objects unique to each bundle in the OSGi environment.

When registering a service, a `ServiceFactory` object can be used instead of a service object, so that the bundle developer can create a unique service object for each bundle that is using the service.

When a bundle [requests](#) the service object, the framework calls the [getService](#) method to create a service object specifically for the requesting bundle. The returned service object is cached by the Framework for subsequent calls to [BundleContext.getService\(ServiceReference\)](#) until the bundle releases its use of the service.

When the bundle's use count for the service is [decremented](#) to zero (including the bundle stopping or the service being unregistered), the framework will call the [ungetService](#) method.

`ServiceFactory` objects are only used by the Framework and are not made available to other bundles in the OSGi environment. The Framework may concurrently call a `ServiceFactory`.

Version:

\$Id: 71fe1dc6db667308ab24b793f07717e71de600ba \$

See Also:

[BundleContext.getService\(ServiceReference\)](#)

ThreadSafe

Method Summary		Page
getService (org.osgi.framework.Bundle bundle, org.osgi.framework.ServiceRegistration<S> registration)	Returns a service object for a bundle.	rror: Refer ence sourc e not found
ungetService (org.osgi.framework.Bundle bundle, org.osgi.framework.ServiceRegistration<S> registration, S service)	Releases a service object created for a bundle.	Error: Refer ence sourc e not found

Method Detail

getService

[S](#) `getService`([org.osgi.framework.Bundle](#) bundle, [org.osgi.framework.ServiceRegistration<S>](#) registration)

Returns a service object for a bundle.

The Framework invokes this method the first time the specified `bundle` requests a service object using the [BundleContext.getService\(ServiceReference\)](#) method. The factory can then return a specific service object for each bundle.

The Framework must check that the returned service object is valid. If the returned service object is `null` or is not an instanceof all the classes named when the service was registered, a framework event of type `org.osgi.framework.FrameworkEvent.ERROR` is fired containing a service exception of type `org.osgi.framework.ServiceException.FACTORY_ERROR` and `null` is returned to the bundle. If this method throws an exception, a framework event of type `org.osgi.framework.FrameworkEvent.ERROR` is fired containing a service exception of type `org.osgi.framework.ServiceException.FACTORY_EXCEPTION` with the thrown exception as the cause and `null` is returned to the bundle. If this method is recursively called for the specified bundle, a framework event of type `org.osgi.framework.FrameworkEvent.ERROR` is fired containing a service exception of type `org.osgi.framework.ServiceException.FACTORY_RECURSION` and `null` is returned to the bundle.

The Framework caches the valid service object and will return the same service object on any future call to [BundleContext.getService\(ServiceReference\)](#) for the specified bundle. This means the Framework must not allow this method to be concurrently called for the specified bundle.

Parameters:

bundle - The bundle requesting the service.

registration - The `ServiceRegistration` object for the requested service.

Returns:

A service object that must be an instance of all the classes named when the service was registered.

See Also:

`ungetService` [BundleContext.getService\(ServiceReference\)](#)

```
void ungetService(org.osgi.framework.Bundle bundle,  
                 org.osgi.framework.ServiceRegistration<S> registration,  
                 S service)
```

Releases a service object created for a bundle.

The Framework invokes this method when a service has been released by a bundle. The service object may then be destroyed.

If this method throws an exception, a framework event of type `org.osgi.framework.FrameworkEvent.ERROR` is fired containing a service exception of type `org.osgi.framework.ServiceException.FACTORY_EXCEPTION` with the thrown exception as the cause.

Parameters:

bundle - The bundle releasing the service.

registration - The `ServiceRegistration` object for the service being released.

service - The service object returned by a previous call to the [getService](#) method.

See Also:

[BundleContext.ungetService\(ServiceReference\)](#)

Interface ServiceObjects

[org.osgi.framework](#)

Type Parameters:

S - Type of Service

```
public interface ServiceObjects
```

Allows multiple service objects for a service to be obtained.

For services with [prototype](#) scope, multiple service objects for the service can be obtained. For services with [singleton](#) or [bundle](#) scope, only one, use-counted service object is available.

Any unreleased service objects obtained from this `ServiceObjects` object are automatically released by the framework when the bundle associated with the `BundleContext` used to create this `ServiceObjects` object is stopped.

Since:

1.8

Version:

\$Id: ce86deab59b5a3006a59eec06e9e2e1545eaafa2 \$

See Also:

[BundleContext.getServiceObjects\(ServiceReference\)](#), [PrototypeServiceFactory](#)

ThreadSafe

Method Summary		Page
	getService() Returns a service object for the referenced service.	Error: Reference source not found
<code>org.osgi.framework.ServiceReference</code>	getServiceReference() Returns the <code>org.osgi.framework.ServiceReference</code> for the service associated with this <code>ServiceObjects</code> object.	Error: Reference source not found
<code>void</code>	ungetService(S service) Releases a service object for the referenced service.	Error: Reference source not found

Method Detail

getService

[S](#) `getService()`

Returns a service object for the [referenced](#) service.

This `ServiceObjects` object can be used to obtain multiple service objects for the referenced service if the service has [prototype](#) scope. If the referenced service has [singleton](#) or [bundle](#) scope, this method behaves the same as calling the [BundleContext.getService\(ServiceReference\)](#) method for the referenced service. That is, only one, use-counted service object is available from this [ServiceObjects](#) object.

This method will always return `null` when the referenced service has been unregistered.

For a prototype scope service, the following steps are required to get the service object:

1. If the referenced service has been unregistered, `null` is returned.
2. The [PrototypeServiceFactory.getService\(Bundle, ServiceRegistration\)](#) method is called to create a service object for the caller.
3. If the service object returned by the `PrototypeServiceFactory` object is `null`, not an instance of all the classes named when the service was registered or the `PrototypeServiceFactory` object throws an exception, `null` is returned and a Framework event of type `org.osgi.framework.FrameworkEvent.ERROR` containing a `org.osgi.framework.ServiceException` describing the error is fired.
4. The service object is returned.

Returns:

A service object for the referenced service or `null` if the service is not registered, the service object returned by a `ServiceFactory` does not implement the classes under which it was registered or the `ServiceFactory` threw an exception.

Throws:

`IllegalStateException` - If the `BundleContext` used to create this `ServiceObjects` object is no longer valid.

See Also:

`ungetService` [ungetService\(Object\)](#)

`void ungetService(S service)`

Releases a service object for the [referenced](#) service.

This `ServiceObjects` object can be used to obtain multiple service objects for the referenced service if the service has [prototype](#) scope. If the referenced service has [singleton](#) or [bundle](#) scope, this method behaves the same as calling the [BundleContext.ungetService\(ServiceReference\)](#) method for the referenced service. That is, only one, use-counted service object is available from this [ServiceObjects](#) object.

For a prototype scope service, the following steps are required to release the service object:

1. If the referenced service has been unregistered, this method returns without doing anything.
2. The [PrototypeServiceFactory.ungetService\(Bundle, ServiceRegistration, Object\)](#) method is called to release the specified service object.

The specified service object must no longer be used and all references to it should be destroyed after calling this method.

Parameters:

`service` - A service object previously provided by this `ServiceObjects` object.

Throws:

`IllegalStateException` - If the `BundleContext` used to create this `ServiceObjects` object is no longer valid.

`IllegalArgumentException` - If the specified service was not provided by this `ServiceObjects` object.

See Also:

`getServiceReference` [getService\(\)](#)

`org.osgi.framework.ServiceReference<S> getServiceReference()`

Returns the `org.osgi.framework.ServiceReference` for the service associated with this `ServiceObjects` object.

Returns:

The `org.osgi.framework.ServiceReference` for the service associated with this `ServiceObjects` object.

Package *org.osgi.service.component.annotations*

Service Component Annotations Package Version 1.3.

See:

[Description](#)

Enum Summary		Page
ReferenceScope	Reference scope for the Reference annotation.	Error: Reference source not found
ServiceScope	Service scope for the Component annotation.	Error: Reference source not found

Annotation Types Summary		Page
Component	Identify the annotated class as a Service Component.	Error: Reference source not found
Reference	Identify the annotated method as a <code>bind</code> method of a Service Component.	Error: Reference source not found

Package *org.osgi.service.component.annotations* Description
Service Component Annotations Package Version 1.3.

This package is not used at runtime. Annotated classes are processed by tools to generate Component Descriptions which are used at runtime.

Annotation Type Component

[org.osgi.service.component.annotations](#)

@Retention(value=RetentionPolicy.CLASS)

@Target(value=ElementType.TYPE)

public @interface Component

Identify the annotated class as a Service Component.

The annotated class is the implementation class of the Component.

This annotation is not processed at runtime by a Service Component Runtime implementation. It must be processed by tools and used to add a Component Description to the bundle.

Version:

\$Id: 0a2fd42933e5072ebe91709a55ced77f65b4fda6 \$

See Also:

"The component element of a Component Description."

Required Element Summary		Page
String	configurationPid The configuration PID for the configuration of this Component.	Error: Reference source not found
org.osgi.service.component.annotations.ConfigurationPolicy	configurationPolicy The configuration policy of this Component.	Error: Reference source not found
boolean	enabled Declares whether this Component is enabled when the bundle containing it is started.	Error: Reference source not found
String	factory The factory identifier of this Component.	Error: Reference source not found
boolean	immediate Declares whether this Component must be immediately activated upon becoming satisfied or whether activation should be delayed.	Error: Reference source not found
String	name The name of this Component.	Error: Reference source not found
String[]	properties Property entries for this Component.	Error: Reference source not found
String[]	property Properties for this Component.	Error: Reference source not found

<code>ServiceScope</code>	<code>scope</code> The service scope for the service of this Component.	Error: Reference source not found
<code>Class<?>[]</code>	<code>service</code> The types under which to register this Component as a service.	Error: Reference source not found
<code>boolean</code>	<code>servicefactory</code> Deprecated. Since 1.3.	Error: Reference source not found
<code>String</code>	<code>xmlns</code> The XML name space of the Component Description for this Component.	Error: Reference source not found

Element Detail

name

`public abstract String name`
The name of this Component.
If not specified, the name of this Component is the fully qualified type name of the class being annotated.
Default:
""
See Also:
`service` "The name attribute of the component element of a Component Description."

`public abstract Class<?>[] service`
The types under which to register this Component as a service.
If no service should be registered, the empty value {} must be specified.
If not specified, the service types for this Component are all the *directly* implemented interfaces of the class being annotated.
Default:
{ }
See Also:
`factory` "The service element of a Component Description."

`public abstract String factory`
The factory identifier of this Component. Specifying a factory identifier makes this Component a Factory Component.
If not specified, the default is that this Component is not a Factory Component.
Default:
""
See Also:

`@Deprecated`
`public abstract boolean servicefactory`
Deprecated. *Declares whether this Component uses the OSGi ServiceFactory concept and each bundle using this Component's service will receive a different component instance. This element is ignored when the `scope()` element does not have the default value. If `true`, this Component uses `bundle` service scope. If `false` or not specified, this Component uses `singleton` service scope.*
Declares whether this Component uses the OSGi ServiceFactory concept and each bundle using this Component's service will receive a different component instance.

This element is ignored when the [scope\(\)](#) element does not have the default value. If `true`, this Component uses [bundle](#) service scope. If `false` or not specified, this Component uses [singleton](#) service scope.

Default:

`false`

See Also:

enabled "The servicefactory attribute of the service element of a Component Description."

`public abstract boolean enabled`

Declares whether this Component is enabled when the bundle containing it is started.

If `true`, this Component is enabled. If `false` or not specified, this Component is disabled.

Default:

`true`

See Also:

immediate "The enabled attribute of the component element of a Component Description."

`public abstract boolean immediate`

Declares whether this Component must be immediately activated upon becoming satisfied or whether activation should be delayed.

If `true`, this Component must be immediately activated upon becoming satisfied. If `false`, activation of this Component is delayed. If this property is specified, its value must be `false` if the [factory\(\)](#) property is also specified or must be `true` if the [service\(\)](#) property is specified with an empty value.

If not specified, the default is `false` if the [factory\(\)](#) property is specified or the [service\(\)](#) property is not specified or specified with a non-empty value and `true` otherwise.

Default:

`false`

See Also:

property "The immediate attribute of the component element of a Component Description."

`public abstract String[] property`

Properties for this Component.

Each property string is specified as `"key=value"`. The type of the property value can be specified in the key as `key:type=value`. The type must be one of the property types supported by the type attribute of the property element of a Component Description.

To specify a property with multiple values, use multiple key, value pairs. For example, `"foo=bar"`, `"foo=baz"`.

Default:

`{}`

See Also:

properties "The property element of a Component Description."

`public abstract String[] properties`

Property entries for this Component.

Specifies the name of an entry in the bundle whose contents conform to a standard Java Properties File. The entry is read and processed to obtain the properties and their values.

Default:

`{}`

See Also:

xmlns "The properties element of a Component Description."

`public abstract String xmlns`

The XML name space of the Component Description for this Component.

If not specified, the XML name space of the Component Description for this Component should be the lowest Declarative Services XML name space which supports all the specification features used by this Component.

Default:

`""`

See Also:

configurationPolicy "The XML name space specified for a Component Description."

`public abstract org.osgi.service.component.annotations.ConfigurationPolicy configurationPolicy`

The configuration policy of this Component.

Controls whether component configurations must be satisfied depending on the presence of a corresponding Configuration object in the OSGi Configuration Admin service. A corresponding configuration is a Configuration object where the PID equals the name of the component.

If not specified, the `OPTIONAL` configuration policy is used.

Default:

`org.osgi.service.component.annotations.ConfigurationPolicy.OPTIONAL`

Since:

1.1

See Also:

configurationPid "The configuration-policy attribute of the component element of a Component Description."

public abstract String configurationPid

The configuration PID for the configuration of this Component.

Allows the configuration PID for this Component to be different than the name of this Component.

If not specified, the name of this Component is used as the configuration PID of this Component.

Default:

""

Since:

1.2

See Also:

scope "The configuration-pid attribute of the component element of a Component Description."

public abstract [ServiceScope](#) scope

The service scope for the service of this Component.

If not specified and the deprecated [servicefactory\(\)](#) element is not specified, the [singleton](#) service scope is used.

Default:

[ServiceScope.DEFAULT](#)

Since:

1.3

See Also:

"The scope attribute of the service element of a Component Description."

Annotation Type Reference

[org.osgi.service.component.annotations](#)

@Retention(value=RetentionPolicy.CLASS)

@Target(value=ElementType.METHOD)

public @interface Reference

Identify the annotated method as a `bind` method of a Service Component.

The annotated method is a `bind` method of the Component.

This annotation is not processed at runtime by a Service Component Runtime implementation. It must be processed by tools and used to add a Component Description to the bundle.

In the generated Component Description for a component, the references must be ordered in ascending lexicographical order (using `String.compareTo`) of the reference [names](#).

Version:

\$Id: e7a5dced89714436efdb077e74a3a3f660824bfb \$

See Also:

"The reference element of a Component Description."

Required Element Summary		Page
<code>org.osgi.service.component.annotations.ReferenceCardinality</code>	cardinality The cardinality of the reference.	Error: Reference source not found
<code>String</code>	name The name of this reference.	Error: Reference source not found
<code>org.osgi.service.component.annotations.ReferencePolicy</code>	policy The policy for the reference.	Error: Reference source not found
<code>org.osgi.service.component.annotations.ReferencePolicyOption</code>	policyOption The policy option for the reference.	Error: Reference source not found
<code>ReferenceScope</code>	scope The requested service scope for this Reference.	Error: Reference source not found
<code>Class<?></code>	service The type of the service to bind to this reference.	Error: Reference source not found
<code>String</code>	target The target filter for the reference.	Error: Reference source not found
<code>String</code>	unbind The name of the unbind method which is associated with the annotated bind method.	Error: Reference source not found

String	updated The name of the updated method which is associated with the annotated bind method.	Error: Reference source not found
--------	---	-----------------------------------

Element Detail

name

```
public abstract String name
```

The name of this reference.

If not specified, the name of this reference is based upon the name of the method being annotated. If the method name begins with `bind`, `set` or `add`, that is removed.

Default:

```
""
```

See Also:

[service](#) "The name attribute of the reference element of a Component Description."

```
public abstract Class<?> service
```

The type of the service to bind to this reference.

If not specified, the type of the service to bind is based upon the type of the first argument of the method being annotated.

Default:

```
Object.class
```

See Also:

[cardinality](#) "The interface attribute of the reference element of a Component Description."

```
public abstract org.osgi.service.component.annotations.ReferenceCardinality cardinality
```

The cardinality of the reference.

If not specified, the reference has a `1..1` cardinality.

Default:

```
org.osgi.service.component.annotations.ReferenceCardinality.MANDATORY
```

See Also:

[policy](#) "The cardinality attribute of the reference element of a Component Description."

```
public abstract org.osgi.service.component.annotations.ReferencePolicy policy
```

The policy for the reference.

If not specified, the `STATIC` reference policy is used.

Default:

```
org.osgi.service.component.annotations.ReferencePolicy.STATIC
```

See Also:

[target](#) "The policy attribute of the reference element of a Component Description."

```
public abstract String target
```

The target filter for the reference.

Default:

```
""
```

See Also:

[unbind](#) "The target attribute of the reference element of a Component Description."

```
public abstract String unbind
```

The name of the unbind method which is associated with the annotated bind method.

To declare no unbind method, the value `"-"` must be used.

If not specified, the name of the unbind method is derived from the name of the annotated bind method. If the annotated method name begins with `bind`, `set` or `add`, that is replaced with `unbind`, `unset` or `remove`, respectively, to derive the unbind method name. Otherwise, `un` is prefixed to the annotated method name to derive the unbind method name. The unbind method is only set if the component type contains a method with the derived name.

Default:

```
""
```

See Also:

[policyOption](#) "The unbind attribute of the reference element of a Component Description."

```
public abstract org.osgi.service.component.annotations.ReferencePolicyOption policyOption
```

The policy option for the reference.

If not specified, the `RELUCTANT` reference policy option is used.

Default:

`org.osgi.service.component.annotations.ReferencePolicyOption.RELUCTANT`

Since:

1.2

See Also:

updated "The policy-option attribute of the reference element of a Component Description."

`public abstract String updated`

The name of the updated method which is associated with the annotated bind method.

To declare no updated method, the value `"-"` must be used.

If not specified, the name of the updated method is derived from the name of the annotated bind method. If the annotated method name begins with `bind`, `set` or `add`, that is replaced with `updated` to derive the updated method name. Otherwise, `updated` is prefixed to the annotated method name to derive the updated method name. The updated method is only set if the component type contains a method with the derived name.

Default:

`""`

Since:

1.2

See Also:

scope "The updated attribute of the reference element of a Component Description."

`public abstract ReferenceScope scope`

The requested service scope for this Reference.

If not specified, the [bundle](#) service scope is requested.

Default:

[ReferenceScope.BUNDLE](#)

Since:

1.3

See Also:

"The scope attribute of the reference element of a Component Description."


```
Enum ReferenceScope
org.osgi.service.component.annotations
java.lang.Object
└─ java.lang.Enum<ReferenceScope>
    └─ org.osgi.service.component.annotations.ReferenceScope
```

All Implemented Interfaces:
Comparable<ReferenceScope>, Serializable

```
public enum ReferenceScope
extends Enum<ReferenceScope>
Reference scope for the Reference annotation.
Since:
    1.3
Version:
    $Id$
```

Enum Constant Summary		Page
BUNDLE	A single service object is used for all references to the service in this bundle.	Error: Reference source not found
PROTOTYPE	If the referenced service has prototype service scope, then each instance of the component with this reference can receive a unique instance of the service.	Error: Reference source not found

Method Summary		Page
String	toString ()	Error: Reference source not found
static ReferenceScope	valueOf (String name)	Error: Reference source not found
static ReferenceScope []	values ()	Error: Reference source not found

Enum Constant Detail

BUNDLE

```
public static final ReferenceScope BUNDLE
A single service object is used for all references to the service in this bundle.
PROTOTYPE
public static final ReferenceScope PROTOTYPE
If the referenced service has prototype service scope, then each instance of the component with this reference can receive a unique instance of the service. If the referenced service does not have prototype service scope, then no service object will be received.
```

Method Detail

values

```
public static ReferenceScope[] values()  
public static ReferenceScope valueOf(String name)  
public String toString()
```

Overrides:
 toString in class Enum

Enum ServiceScope
[org.osgi.service.component.annotations](#)
java.lang.Object
└─ java.lang.Enum<[ServiceScope](#)>
 └─ org.osgi.service.component.annotations.ServiceScope

All Implemented Interfaces:
Comparable<[ServiceScope](#)>, Serializable

public enum ServiceScope
extends Enum<[ServiceScope](#)>
Service scope for the [Component](#) annotation.
Since: 1.3
Version: \$Id\$

Enum Constant Summary		Page
BUNDLE	When the component is registered as a service, it will be registered as a bundle scope service and an instance of the component will be created for each bundle using the service.	rror: Reference source not found
DEFAULT	Default element value for annotation.	Error: Reference source not found
PROTOTYPE	When the component is registered as a service, it will be registered as a prototype scope service.	Error: Reference source not found
SINGLETON	When the component is registered as a service, it will be registered as a bundle scope service but only a single instance of the component will be used for all bundles using the service.	Error: Reference source not found

Method Summary		Page
String toString ()		Error: Reference source not found
static ServiceScope valueOf (String name)		Error: Reference source not found
static ServiceScope [] values ()		Error: Reference source not found

Enum Constant Detail

SINGLETON

```
public static final ServiceScope SINGLETON
```

When the component is registered as a service, it will be registered as a bundle scope service but only a single instance of the component will be used for all bundles using the service.

```
public static final ServiceScope BUNDLE
```

When the component is registered as a service, it will be registered as a bundle scope service and an instance of the component will be created for each bundle using the service.

```
public static final ServiceScope PROTOTYPE
```

When the component is registered as a service, it will be registered as a prototype scope service.

```
public static final ServiceScope DEFAULT
```

Default element value for annotation. This is used to distinguish the default value for an element and should not otherwise be used.

Method Detail

values

```
public static ServiceScope[] values()
```

```
public static ServiceScope valueOf(String name)
```

```
public String toString()
```

Overrides:

toString in class Enum

Java API documentation generated with [DocFlex/Doclet](#) v1.5.6

DocFlex/Doclet is both a multi-format Javadoc doclet and a free edition of [DocFlex/Javadoc](#). If you need to customize your Javadoc without writing a full-blown doclet from scratch, DocFlex/Javadoc may be the only tool able to help you! Find out more at www.docflex.com

8 Considered Alternatives

8.1 Parameterization

RFC 158 Parameterized Services [3]. explored multiple service objects with the additional requirement to allow parameterization of the service instance creation. This created additional issues with ensuring type safety of the parameter types which may not appear in the service type package. This proved unworkable and ultimately led to the withdrawal of the RFC with the recommendation to simply define and use a factory-type service whose signature would reference the parameterization types and the service type (return type of the instance method). This RFC avoids this issue since the design does not support parameterization of the services instances.

9 Security Considerations

There are no additional security considerations for this design. Normal ServicePermission rules will address service security.

10 Document Support

10.1 References

- [1]. Bradner, S., Key words for use in RFCs to Indicate Requirement Levels, RFC2119, March 1997.
- [2]. Software Requirements & Specifications. Michael Jackson. ISBN 0-201-87712-0
- [3]. RFC 158 Parameterized Services. <https://www.osgi.org/members/svn/documents/trunk/rfcs/rfc0158/rfc-0158-ParameterizedServices.pdf>

10.2 Author's Address

Name	BJ Hargrave
Company	IBM Corporation
e-mail	hargrave@us.ibm.com

10.3 Acronyms and Abbreviations

DS – Declarative Services

SCR – The implementation of Declarative Services

10.4 End of Document