Configuration Listener

Final

12 Pages

Abstract

10 point Arial Centered.

This RFC specifies a new Configuration Listener for Configuration Admin to allow bundles to be notified of configuration database changes.

0 Document Information

0.1 Table of Contents

All Page Within This Box

0.2 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 [1].

`Source code is shown in this typeface.`

0.3 Revision History

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

Revision	Date	Comments
Draft 1	23 November 2004	Initial draft based upon a design sketch reviewed by Peter Kriens. BJ Hargrave, hargrave@us.ibm.com
Draft 2	26 November 2004	Second draft incorporating feedback from the CPEG mail list. 1. Added ConfigurationEvent rather than having specific methods in ConfigurationListener. This will more easily allow extension in the future without breaking code implementing ConfigurationListener. BJ Hargrave, hargrave@us.ibm.com
Proposed Final Draft	13 December 2004	Accepted all changes. Made clear that event is sent asynchronously. BJ Hargrave, hargrave@us.ibm.com
Final	27 May 2005	No Changes BJ Hargrave, hargrave@us.ibm.com

1 Introduction

The Configuration Admin (CA) service is a key element in the configuration of bundles and services. However external bundles need a means to be informed of changes to the configuration database. The current

Configuration Admin service does not provide a mechanism to do this. This RFC proposed a simple extension to the CA service to allow other bundles to be notified of changes to the configuration database.

2 Application Domain

The Configuration Admin (CA) service provides a means for an administrative entity to set and store configuration information and also for bundles to receive that configuration information. Bundles which expect to receive configuration information can register ManagedService (or ManagedServiceFactory) services to receive the configuration information and be notified of changes to the configuration information. Other bundles can register as ConfigurationPlugins which can enable them to inspect and/or alter configuration information as it is being delivered to the receiving bundle.

3 Problem Description

The current CA service design does not provide any means for a 3rd party to be notified of changes to the configuration database (set of configurations). It provides a means of setting, updating, removing and introspecting configurations for administrative bundles. It provides a means for configuration targets (ManagedService or ManagedService Factory services registered by bundles) to receive configurations and updates to configurations. It also provides a means for 3rd party bundles to inspect and/or alter configurations as they are being delivered to configuration targets. But there is no specified way for a 3rd party bundle to be notified of a change to the configuration database.

Using a ConfigurationPlugin does not address this problem, since ConfigurationPlugins are only called during the delivery of a configuration to a configuration target. If no configuration target is registered to receive a configuration, then no ConfigurationPlugin will be called if the configuration is created, updated or deleted.

The Service Component Runtime (SCR) from RFC 80 needs a mechanism to watch configuration changes and will then use the ConfigurationAdmin API to obtain the configurations. SCR needs this to properly support component configurations.

4 Requirements

The CA service must provide a mechanism for a 3rd party bundle to be aware of changes to the configuration database. This mechanism must not require the 3rd party bundle to registering a ManagedService or ManagedServiceFactory with the configuration pids in which the 3rd party bundle may be interested.

5 Technical Solution

A new whiteboard listener and event class are added to the Configuration Admin specification. Specific event types are defined for updated and deleted configurations.

5.1 org.osgi.service.cm Interface ConfigurationListener

public interface **ConfigurationListener**

Listener for Configuration Events.

`ConfigurationListener` objects are registered with the Framework service registry and are notified with a `ConfigurationEvent` object when an event is broadcast.

`ConfigurationListener` objects can inspect the received `ConfigurationEvent` object to determine its type, the pid of the `Configuration` object with which it is associated, and the Configuration Admin service that broadcasted the event.

Security Considerations. Bundles wishing to monitor configuration events will require `ServicePermission[ConfigurationListener,REGISTER]` to register a `ConfigurationListener` service.

Method Summary	
`void`	`configurationEvent(ConfigurationEvent event)` Receives notification of a broadcast `ConfigurationEvent` object.

Method Detail

5.1.1 configurationEvent

`public void configurationEvent(ConfigurationEvent event)`

Receives notification of a broadcast `ConfigurationEvent` object.

Parameters:

`event` - The broadcasted `ConfigurationEvent` object.

5.2 org.osgi.service.cm
Class ConfigurationEvent

```
java.lang.Object
  └org.osgi.service.cm.ConfigurationEvent
```

public class **ConfigurationEvent**
extends java.lang.Object

A Configuration Event.

`ConfigurationEvent` objects are delivered to all registered `ConfigurationListener` service objects. ConfigurationEvents must be asynchronously delivered in chronological order with respect to each listener.

A type code is used to identify the type of event. The following event types are defined:

- `CM_UPDATED`
- `CM_DELETED`

Additional event types may be defined in the future.

Security Considerations. `ConfigurationEvent` objects do not provide `Configuration` objects, so no sensitive configuration information is available from the event. If the listener wants to locate the `Configuration` object for the specified pid, it must use `ConfigurationAdmin`.

See Also:

`ConfigurationListener`

Field Summary

`static int`	**`CM_DELETED`** A `Configuration` has been deleted.
`static int`	**`CM_UPDATED`** A `Configuration` has been updated.

Constructor Summary

`ConfigurationEvent``(org.osgi.framework.ServiceReference reference, int type,`

`java.lang.String factoryPid, java.lang.String pid)` Constructs a `ConfigurationEvent` object from the given `ServiceReference` object, event type, and pids.

Method Summary

`java.lang.String`	`getFactoryPid()` Returns the factory pid of the associated configuration.
`java.lang.String`	`getPid()` Returns the pid of the associated configuration.
`org.osgi.framework.ServiceReference`	`getReference()` Return the `ServiceReference` object of the Configuration Admin service that created this event.
`int`	`getType()` Return the type of this event.

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Field Detail

5.2.1 CM_UPDATED

`public static final int` **`CM_UPDATED`**

A `Configuration` has been updated.

This `ConfigurationEvent` type that indicates that a `Configuration` object has been updated with new properties. An event is asynchronously broadcast when a call to `Configuration.update` successfully changed a configuration.

The value of `CM_UPDATED` is 1.

See Also:
Constant Field Values

5.2.2 CM_DELETED

`public static final int` **`CM_DELETED`**

A `Configuration` has been deleted.

This `ConfigurationEvent` type that indicates that a `Configuration` object has been deleted. An event is asynchronously broadcast when a call to `Configuration.delete` successfully deletes a configuration.

The value of `CM_DELETED` is 2.

See Also:
Constant Field Values

Constructor Detail

5.2.3 ConfigurationEvent

```
public ConfigurationEvent(org.osgi.framework.ServiceReference reference,
                          int type,
                          java.lang.String factoryPid,
                          java.lang.String pid)
```

Constructs a `ConfigurationEvent` object from the given `ServiceReference` object, event type, and pids.

Parameters:

`reference` - The `ServiceReference` object of the Configuration Admin service that created this event.
`type` - The event type. See `getType()`.
`factoryPid` - The factory pid of the associated configuration if the target of the configuration is a ManagedServiceFactory. Otherwise `null` if the target of the configuration is a ManagedService.
`pid` - The pid of the associated configuration.

Method Detail

5.2.4 getFactoryPid

```
public java.lang.String getFactoryPid()
```

Returns the factory pid of the associated configuration.
Returns:
Returns the factory pid of the associated configuration if the target of the configuration is a ManagedServiceFactory. Otherwise `null` if the target of the configuration is a ManagedService.

5.2.5 getPid

```
public java.lang.String getPid()
```

Returns the pid of the associated configuration.
Returns:
Returns the pid of the associated configuration.

5.2.6 getType

```
public int getType()
```

Return the type of this event.

The type values are:

- `CM_UPDATED`
- `CM_DELETED`

Returns:
The type of this event.

All Page Within This Box

5.2.7 getReference

`public org.osgi.framework.ServiceReference` **`getReference`**`()`

Return the `ServiceReference` object of the Configuration Admin service that created this event.

Returns:

The `ServiceReference` object for the Configuration Admin service that created this event.

6 Considered Alternatives

Removed having specific methods on the listener interface in favor of an event class. This will allow the event to be extended in the future.

6.1 org.osgi.service.cm Interface ConfigurationListener

public interface **ConfigurationListener**

Listener for Configuration changes.

`ConfigurationListener` objects are registered with the Framework service registry and are notified when a `Configuration` object is updated or deleted.

`ConfigurationListener` objects are passed the type of configuration change.

One of the change methods will be called with `CM_UPDATED` when `Configuration.update` is called or with `CM_DELETED` when `Configuration.delete` is called. Notification will be asynchronous to the update or delete method call. The design is very lightweight in that is does not pass `Configuration` objects, the listener is merely advised that the configuration information for a given pid has changed. If the listener wants to locate the `Configuration` object for the specified pid, it must use `ConfigurationAdmin`.

Security Considerations. Bundles wishing to monitor `Configuration` changes will require `ServicePermission[ConfigurationListener,REGISTER]` to register a `ConfigurationListener` service. Since `Configuration` objects are not passed to the listener, no sensitive configuration information is available to the listener.

Field Summary	
`static int`	**`CM_DELETED`** Change type that indicates that `Configuration.delete` was called.

All Page Within This Box

static int	**CM_UPDATED** Change type that indicates that `Configuration.update` was called.

Method Summary

void	**configurationChanged**(java.lang.String pid, int type) Receives notification a configuration has changed.
void	**factoryConfigurationChanged**(java.lang.String factoryPid, java.lang.String pid, int type) Receives notification a factory configuration has changed.

Field Detail

6.1.1 CM_UPDATED

```
public static final int CM_UPDATED
```

Change type that indicates that `Configuration.update` was called.

See Also:

Constant Field Values

6.1.2 CM_DELETED

```
public static final int CM_DELETED
```

Change type that indicates that `Configuration.delete` was called.

See Also:

Constant Field Values

Method Detail

6.1.3 configurationChanged

```
public void configurationChanged(java.lang.String pid,
                                 int type)
```

Receives notification a configuration has changed.

This method is only called if the target of the configuration is a ManagedService.

Parameters:

`pid` - The pid of the configuration which changed.
`type` - The type of the configuration change.

6.1.4 factoryConfigurationChanged

```
public void factoryConfigurationChanged(java.lang.String factoryPid,
                                        java.lang.String pid,
                                        int type)
```

Receives notification a factory configuration has changed.

This method is only called if the target of the configuration is a ManagedServiceFactory.

Parameters:
`factoryPid` - The factory pid for the changed configuration.
`pid` - The pid of the configuration which changed.
`type` - The type of the configuration change.

7 Security Considerations

Bundles wishing to monitor configuration events will require `ServicePermission[ConfigurationListener, REGISTER]` to register a `ConfigurationListener` service.

`ConfigurationEvent` objects do not provide `Configuration` objects, so no sensitive configuration information is available from the event. If the listener wants to locate the `Configuration` object for the specified pid, it must use `ConfigurationAdmin`.

8 Document Support

8.1 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

8.2 Author’s Address

Name	BJ Hargrave
Company	IBM
Address	11501 Burnet Rd, Austin, TX 78758 USA
Voice	+1 512 838 8838
e-mail	hargrave@us.ibm.com

8.3 Acronyms and Abbreviations

CA – Configuration Admin

SCR – Service Component Runtime

8.4 End of Document

All Page Within This Box