Template version: proc-RFC_template-1_01_011016.dot

RFC 0045 - Framework Launch & Shutdown Enhancements

Confidential, Draft
rfc0045-framework_launch_shutdown

10 Pages

Abstract

The current framework (1.1) specification does not provide any control over the order of bundle starting during the framework launch or bundle stopping during framework shutdown. This RFC describes enhancements to the framework specification to provide a flexible mechanism for the user (operator) to control this.

0 Document Information

0.1 Table of Contents

0.2 Status

This document specifies enhancements to the Framework 1.1 specification for the Open Services Gateway Initiative, and requests discussion and suggestions for improvements. Distribution of this document is unlimited within OSGi.

0.3 Acknowledgement

This document is based upon the contributions of CPEG members.

All Page within this Box

All Page Within This Box

0.4 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.
```

0.5 Revision History

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

Revision	Date	Comments
Initial draft	March 06 2002	The initial draft is based upon the verbal discussion that took place during the CPEG Las Vegas meeting(s) Jan, 2002. BJ Hargrave, IBM, hargrave@us.ibm.com
2nd draft	March 12 2002	Incorporate comments from Peter Kriens. BJ Hargrave, IBM, hargrave@us.ibm.com
3rd draft	March 13 2002	Incorporate comments from Niclas Nilsson. Added new method to query persistent state of bundle. BJ Hargrave, IBM, hargrave@us.ibm.com
4th draft	May 7, 2002	Updates from April 2002 Virginia meeting. Renamed Runlevel to StartLevel to avoid confusion with UNIX-style runlevels. Improved description of setCurrentStartLevel. BJ Hargrave, IBM, hargrave@us.ibm.com

1 Introduction

The current OSGi Framework (1.1) specification[2] provides no way to influence the order in which bundles are started during the launch of the framework or to influence the order in which bundles are stopped during the shutdown of the framework. The only control currently available is to have a bundle started when the framework is launched. There is no way to influence the order of bundle startup or whether a bundle should even be started on this specific launch of the framework.

This RFC will describe enhancements to the framework specific to enable this level of control. This document attempts to address the requirements specified in section 3.1 of RFP 34 [3].

2 Motivation and Rationale

A solution to this problem is necessary for several reasons.

The current specification only provides for starting all bundles (marked to be started) during framework launch. There is no way to control the start order. Thus a random start order among bundles is implied and framework vendors may chose to implement different orderings. This can lead to inconsistent bundle start orderings, some of which may lead to reliability problems (due to programming choices/errors by bundle programmers). The operator of the framework would desire to have some predictability in the startup and shutdown sequence of the framework to minimize problems of this type.

It is also desirable for the operator to prescribe a start ordering among the installed bundles. The may be necessary to ensure that operator installed management bundles are always started before other application bundles. It may also be desirable for the operator to start a “first screen” (or “splash screen”) application before other background applications to improve the user’s perception of the boot time of the device.

From a reliability standpoint, it may also be desirable for the operator to enable a “safe mode” for the device. For example, when an error situation is detected requiring the framework to be restarted, the framework can be restarted into “safe mode” consistent of known good bundles and other bundles can be selectively started in a controlled manner in an attempt to isolate a defective or malicious bundle.

None of these requirements can currently be met within the current framework specification. Vendors can implement similar behavior through proprietary extensions, but it is in OSGi’s best interest to standardize important function like this.

3 Technical Discussion

The enhancements to the framework specification include the ability to control the initial start level of the framework, to modify the current start level of the framework and to assign a bundle to a specific start level.

A start level is defined to be a state of execution in which the framework exists. Start levels are defined as non-negative integers with 0 (zero) being the state where the framework has not been launched (or has completed shutdown – these are the same thing). In this state, no bundles can be running. Progressively higher integral values represent progressively higher start levels -- 2 is a higher start level than 1. The framework will support a very large number of start level values (all positive int values). A bundle can be individually assigned any (positive) start level value.

Here are some scenarios on how start levels can be used. In the scenarios below, the Framework will start a bundle during launch only if the bundle is persistently marked to be started.

- All bundle are assigned a start level between 1 and N to influence their start order. Let N be the highest start level used by the bundles. The framework is started and launched with an argument specifying the initial start level of N. The framework launches and progresses through start levels starting bundles. Start level N is reached, all bundles have been started, and the FrameworkEvent.STARTED event is published.

- All bundle are assigned a start level between 1 and N to influence their start order. Let M be a start level such that M < N and all trusted (including management) bundles use a start level less than or equal to M. The framework is started and launched with an argument specifying the initial start level of M. The framework launches and progresses through start levels starting bundles. Start level M is reached, all trusted bundles have been started, and the FrameworkEvent.STARTED event is published. A management bundle will now continue to start bundles with start levels > M at its discretion.

- A single start level, 1, is used and all bundles are assigned start level 1. The framework is started and launched with an argument specifying the initial start level of 1. The framework launches and starts bundles. Start level 1 is reached, all bundles have been started, and the FrameworkEvent.STARTED event is published. This may be considered a compatibility mode with prior version Frameworks in that all bundles are started and there is no control over the start order.

The changes to the framework spec to enable this include a new service registered by the framework via the system bundle and a change to FrameworkEvent for start level change notification.

3.1 org.osgi.service.startlevel.StartLevel

public interface **StartLevel**

The StartLevel service allows operators to manage the start level assigned assigned to each bundle and the current start level of the Framework. There is at most one StartLevel service present in the Framework.

Access to the StartLevel service is protected by corresponding `ServicePermission`. In addition the `AdminPermission` is required to actually modify start level information.

StartLevel support in the Framework includes the ability to control the initial start level of the Framework, to modify the current start level of the Framework and to assign a specific start level to a bundle. This service can be used to modify the current start level of the Framework and to a specific start level to a bundle. How the initial start level of a Framework is specified is implementation dependent. It may be a command line argument when invoking the Framework implementation.

A start level is defined to be a state of execution in which the Framework exists. StartLevel values are defined as non-negative integers with 0 (zero) being the state where the Framework is not launched. Progressively higher integral values represent progressively higher start levels. e.g. 2 is a higher start level than 1.

When the Framework is first started it is at start level 0. In this state, no bundles are running. This is the initial state of the Framework before it is launched. When the Framework is launched, the Framework will enter start level 1 and all bundles which are assigned to start level 1 and are persistently marked to be started are started as described in the `Bundle.start` method. Within a start level, bundles are started

in ascending order by `Bundle.getBundleId`. The Framework will continue to increase the start level, starting bundles at each start level, until the Framework has reached a specified initial start level. At this point the Framework has completed starting bundles and will then broadcast a Framework event of type `FrameworkEvent.STARTED` to announce it has completed its launch.

After launching, the Framework must register the StartLevel service. The StartLevel service can be used by management bundles to alter the current start level of the framework. Prior to performing a shutdown, the Framework must unregister the StartLevel service.

Method Summary

int	**getBundleStartLevel**(org.osgi.framework.Bundle bundle) Return the assigned start level value for the specified Bundle.
int	**getCurrentStartLevel**() Return the current start level value for the Framework.
int	**getInitialBundleStartLevel**() Return the initial start level value that is assigned to a Bundle when it is first installed.
boolean	**isBundlePersistentlyStarted**(org.osgi.framework.Bundle bundle) Return the persistent state of the specified bundle.
void	**setBundleStartLevel**(org.osgi.framework.Bundle bundle, int startlevel) Assign a start level value to the specified Bundle.
void	**setCurrentStartLevel**(int startlevel) Modify the current start level of the Framework.
void	**setInitialBundleStartLevel**(int startlevel) Set the initial start level value that is assigned to a Bundle when it is first installed.

Method Detail

3.1.1 getCurrentStartLevel

`public int` **`getCurrentStartLevel`**`()`

Return the current start level value for the Framework.

Returns:

The current start level value of the Framework.

3.1.2 setCurrentStartLevel

`public void` **`setCurrentStartLevel`**`(int startlevel)`

Modify the current start level of the Framework.

The Framework will move to the new start level. This method will return immediately to the caller and the start level change will occur asynchronously on another thread.

All Page within this Box

All Page Within This Box

If the specified start level is higher than the previous start level, the Framework will continue to increase the start level until the Framework has reached the specified start level starting bundles at each start level which are persistently marked to be started as described in the `Bundle.start` method. At each intermediate start level value on the way to and including the target start level, the framework must:

1. Change the current start level to the intermediate start level value.
2. Start bundles at the intermediate start level in ascending order by `Bundle.getBundleId`.

When this process completes after the target start level is reached, the Framework will broadcast a Framework event of type `FrameworkEvent.STARTLEVEL_CHANGED` to announce it has moved to the specified start level.

If the specified start level is lower than the previous start level, the Framework will continue to decrease the start level until the Framework has reached a specified start level stopping bundles at each start level as described in the `Bundle.stop` method except that their persistently recorded state indicates that they must be restarted in the future. At each intermediate start level value on the way to and including the target start level, the framework must:

1. Stop bundles at the intermediate start level in descending order by `Bundle.getBundleId`.
2. Change the current start level to the intermediate start level value.

When this process completes after the target start level is reached, the Framework will broadcast a Framework event of type `FrameworkEvent.STARTLEVEL_CHANGED` to announce it has moved to the specified start level.

If the specified start level is equal than the previous start level, then no action is taken.

Parameters:
`startlevel` - The new start level for the Framework.
Throws:
`java.lang.IllegalArgumentException` - If the specified start level is less than or equal to zero.
`java.lang.SecurityException` - If the caller does not have the `AdminPermission` and the Java runtime environment supports permissions.

3.1.3 getBundleStartLevel

```
public int getBundleStartLevel(org.osgi.framework.Bundle bundle)
```

Return the assigned start level value for the specified Bundle.
Parameters:
`bundle` - The target bundle.
Returns:
The startlevel value of the specified Bundle.
Throws:
`java.lang.IllegalArgumentException` - If the specified bundle has been uninstalled.

3.1.4 setBundleStartLevel

```
public void setBundleStartLevel(org.osgi.framework.Bundle bundle,
                                int startlevel)
```

Assign a start level value to the specified Bundle.

All Page within this Box

All Page Within This Box

The specified bundle will be assigned the specified start level. The start level value assigned to the bundle will be persistently recorded by the Framework. If the new start level for the bundle is lower than or equal to the current start level of the Framework, the Framework will start the specified bundle as described in the `Bundle.start` method if the bundle is persistently marked to be started. If the new start level for the bundle is higher than the current start level of the Framework, the Framework will stop the specified bundle as described in the `Bundle.stop` method except that the persistently recorded state for the bundle indicates that the bundle must be restarted in the future.

Parameters:
`bundle` - The target bundle.
`start` - level The new start level for the specified Bundle.
Throws:
`java.lang.IllegalArgumentException` - If the specified bundle has been uninstalled or if the specified start level is less than or equal to zero.
`java.lang.SecurityException` - if the caller does not have the `AdminPermission` and the Java runtime environment supports permissions.

3.1.5 getInitialBundleStartLevel

`public int` **`getInitialBundleStartLevel`**`()`

Return the initial start level value that is assigned to a Bundle when it is first installed.

When a Bundle is installed via `BundleContext.installBundle`, it is assigned the start level value returned by this method.

The default initial Bundle start level value is 1 unless `setInitialBundleStartLevel(int)` has been called to assign a different initial Bundle start level value.

Returns:
The initial start level value for Bundles.

3.1.6 setInitialBundleStartLevel

`public void` **`setInitialBundleStartLevel`**`(int startlevel)`

Set the initial start level value that is assigned to a Bundle when it is first installed.
Parameters:
`startlevel` - The initial start level for newly installed Bundles.
Throws:
`java.lang.IllegalArgumentException` - If start level is less than or equal to zero.
`java.lang.SecurityException` - if the caller does not have the `AdminPermission` and the Java runtime environment supports permissions.

3.1.7 isBundlePersistentlyStarted

`public boolean` **`isBundlePersistentlyStarted`**`(org.osgi.framework.Bundle bundle)`

Return the persistent state of the specified bundle.

This method returns the persistent state of a bundle. The persistent state of a bundle indicates whether a bundle is persistently marked to be started when it's start level is reached.

Returns:
`true` if the bundle is persistently marked to be started, `false` if the bundle is not persistently marked to be started.
Throws:
`java.lang.IllegalArgumentException` - If the specified bundle has been uninstalled.

3.2 org.osgi.framework.FrameworkEvent changes

3.2.1 STARTLEVEL_CHANGED

`public static final int` **`STARTLEVEL_CHANGED`**
A StartLevel.setCurrentStartLevel operation has completed.

This event is broadcast when the Framework has completed changing the current start level initiated by a call to the StartLevel.setCurrentStartLevel method.

The value of `STARTLEVEL_CHANGED` is 0x00000008.

See Also:
`org.osgi.service.startlevel.StartLevel`
Since:
1.2

3.3 Alternatives considered

It was considered to simply mark bundles as transient, i.e. their started state would not be persistent stored. Thus the bundle would not automatically be started when the framework restarted. Another management bundle, which was always started, would be responsible for starting the transient bundles at its discretion. This proposal was determined to be a subset of the start level solution.

4 Security Considerations

The APIs in this RFC are accessible by callers with ServicePermission for the new StartLevel service. Furthermore, the caller must also have AdminPermission to modify any of the start level information. This will allow some bundles to interrogate the information and other bundles to modify the information.

5 Document Support

5.1 References

[1]. Bradner, S., Key words for use in RFCs to Indicate Requirement Levels, RFC2119, March 1997.

[2]. OSGi Service Platform Release 2 specification, October 2001

[3]. RFP 34 Fault Management Foundation

5.2 Author's Address

Name	BJ Hargrave
Company	IBM Pervasive Computing Division
Address	11400 Burnet Road, Austin, TX 78750 USA
Voice	+1 512 838 9938
e-mail	mailto:hargrave@us.ibm.com

5.3 Acronyms and Abbreviations

5.4 End of Document

All Page Within This Box