Module Activator

Overview

A module activator provides means for module developers to register their modules with the core API so that they get notified during application startup, shutdown and when the spring applicationContext gets refreshed. This is useful for things like caching of some module specific resources at application startup or cleanup of special resources before the application shuts down.

A module Activator class MUST implement the interface 'org.openmrs.module.ModuleActivator'. There is an abstract class 'org.openmrs.module. BaseModuleActivator' that is optionally available as a default implementation and "buffer" against changes. We recommend extending the abstract class in your module for "forward compatibility" sake. These are available in version 1.7.0 and later.

If you need your module to support earlier versions up to 1.6.x, your activator class should implement the deprecated 'org.openmrs.module.Activator' which has only 2 methods. ModuleActivator gives more control to the developer with its 6 methods and so is preferred over the older and simpler Activator interface.

Below is the break down of the methods that are provided by the new ModuleActivator interface and abstract class. Please see the Basic Module for a code example of how to register and write module activators for your modules.

Methods in 'org.openmrs.module.ModuleActivator'

(Available with version 1.7.x and later)

- willRefreshContext()
  Called for each module just before spring's application context is refreshed. This method could be called multiple times in a module lifecycle i.e. whenever a new module gets started, at application startup or a developer chooses to refresh the context programatically.

- contextRefreshed()
  Called for each module after spring's application context is refreshed , this method is also called multiple times i.e. whenever a new module gets started and at application startup.

- willStart()
  Called after a module has been loaded but before the application context is refreshed, at this point the module's service methods aren't yet available so they can't be called, so calls like 'Context.getService(MyService.class)' will return null.
  This is called *once* per module per time it is started.
  This method will be authenticated as the Daemon user and have all privileges.

- started()
  Called after a module is started, the application context has been refreshed and the module's service methods are ready to be called. This is called *once* per module per time it is started.

- willStop()
  Called just before a module is stopped. This is called *once* per module per time it is started.

- stopped()
  Called after a module is stopped. This is called *once* per module per time it is started.

Methods available in older 'org.openmrs.module.Activator'

(Deprecated and available with version 1.6.x or earlier)

- startup()
  Called after a module has been loaded but before the application context is refreshed, at this point the module's service methods aren't yet available so they can't be called, so calls like 'Context.getMyService()' will return null.
  As of 1.7.0, this method will be authenticated as the Daemon user and have all privileges.

- shutdown()
  Called after a module is stopped.