OSGi运行时环境技术选型研究

目录

[1. 目的 2](#_Toc357417561)

[2. OSGi介绍 2](#_Toc357417562)

[3. 研究内容 3](#_Toc357417563)

[3.1. Eclipse Equinox介绍 3](#_Toc357417564)

[3.2. Apache Felix介绍 4](#_Toc357417565)

[3.3. Apache Felix开发简介 5](#_Toc357417566)

[3.4. Apache Felix Karaf 7](#_Toc357417567)

[3.5. Apache Aries 8](#_Toc357417568)

[3.6. 选型比较 8](#_Toc357417569)

[4. 总结 9](#_Toc357417570)

1. 目的

OSGi规范有不同厂家的各种实现，比较知名的有Eclipse Equinox、Apache Felix，都是基于OSGi的标准实现，单纯基于OSGi的规范实现做选择，只要稳定可靠，有丰富的应用案例，差别不是很大，但是不同的产品还是有自己的特色的，提供的扩展特性也是不一样的，因此长远考虑，还是要选择一款更合适的，能够深入掌握和使用。

经过前期初步预研分析，考虑选择Apache Felix作为物流平台OSGi运行时环境，本文档对预研分析的过程及相关开源产品做简要介绍，方面大家理解，也作为正式技术决策分析的基础。

1. OSGi介绍
2. **OSGi概念**

当前OSGi的含义已经和其原始名称有了很大的差别，所以大家就把OSGi当作一个技术名词就可以了。OSGi是一个规范，它致力于提供给Java项目一个模块化的底层环境，以及一系列通用的服务。和普通的JVM程序相比，OSGi的程序天生拥有动态模块的特点，不同的模块(OSGi里称之为Bundle)有着独立的生命周期，可以独立进行安装、启动、停止、卸载的操作，模块间的依赖性管理也由OSGi提供。OSGi非常适合需要进行插件化管理的项目。OSGi还规范了一系列我们常见的操作，日志、配置文件、事件队列、Web开发、JPA&JDBC等等，大部分部署OSGi标准的框架都提供了这些服务，这样一方面规范了我们代码的结构，一方面节约了我们开发的时间。

1. **OSGi R4~R5发布时间**
2. **为什么需要OSGi**

模块化的好处是显而易见的，最直接的好处应该是能够降低复杂产品的开发和维护成本，缩短研发周期；实现这些好处需要合理设计低耦合、高内聚的系统模块，支持逻辑和物理模块化的合二为一，实现模块的并行开发和独立生命周期管理等。OSGi规范不仅对这些特性提供了支持，而且随着新版本的不断发布，为企业应用开发提供更多丰富的支持，成为实现模块化企业应用开发一个一个很好的选择。

1. **OSGi与JAR包相比模块化的优势**

通常，Java 应用程序模块化为 JAR 包。但是使用 JAR 文件有局限性：

* JAR 包通过类路径环境变量解析，而这并未提供管理 JAR 依赖关系的可靠框架。
* JAR无法进行版本控制，因此无法跟踪所创建或修改的 JAR 包的历史。
* 没有用于在运行时出现代码更改的情况下动态更新 JAR 文件的框架。

为了处理上述问题，可以使用 OSGi 框架，因为其中对 Java 的模块化系统进行了重新定义。相对于传统的 JAR 模块而言，基于OSGi的系统具有以下优势：

* OSGi 提供了可靠的集成环境，包可以在其中作为服务发布并导出供其他包使用。
* OSGi为每个新组件提供了包版本控制功能。因此可以跟踪包创建和更改的历史。
* 通过OSGi，可以在运行时出现更改的情况下随时动态更新包。

当然还有其他更多的优势，不再一一罗列。

1. 研究内容

OSGi规范的开源实现有很多，Eclipse Equinox、Apache Felix无论出身、人气等各方面均有明显的优势，因此本文档仅重点介绍这两款产品。

* 1. Eclipse Equinox介绍

[Equinox](http://www.eclipse.org/equinox/)是Eclipse旗下的OSGi框架，本身也被Eclipse采用，是Eclipse注明的PDE开发环境的底层。Equinox本身也是相当的全面的框架，提供的功能不比Felix少多少。但是它功能的分类就稍显混乱，文档和Sample也组织的不是很好。事实上相当Equinox还是被当做开发Eclipse Plugin的应用较多，如果你要开发一个Web程序，你就会感到它本身的功能和文档不够全面。Equinox最大的优势在于它和Eclipse结合紧密，只要你安装了PDE(Plug-in Develop Environment)，你就已经有了Equinox，可以方便的在Eclipse里设置你开发的Bundle，启动、部署等操作也异常简单，而且有专门的Debug界面。Equinox 3.7中已经实现了OSGi 4.3运行时。

* 1. Apache Felix介绍

[Apache Felix](http://felix.apache.org/)是Apache旗下的一个OSGi框架，项目本身非常成熟，已经被用到了很多其他的项目中，例如Apache Service mix。它本身提供的服务也是最全的，几乎涵盖了全部的OSGi 4.2的标准。除此之外还提供了一些非标准的功能，例如iPOJO。框架本身非常紧凑，你只需要3个包加一个shell就可以运行了，无论是开发还是Debug都非常简便。除了Felix，还有两个项目是和OSGi相关的。一个是Apache Felix Karaf，它本身是Felix的一个子项目，但他其实是封装了Felix提供更高一层的Runtime，例如提供了JAAS。另一个是[Apache Aries](http://incubator.apache.org/aries" \t "_blank)，目前还处于起步阶段，它作为Felix的补充，提供OSGi企业级规范，包括JPA、JDBC、JTA、JNDI等等。

总的来说，Apache Felix是我个人推荐的最佳OSGi框架，它简单的结构也更适合出学OSGi的开发人员。下面表格中是Felix包含的子项目，参考了解下。

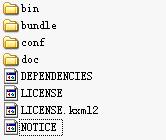
|  |  |  |  |
| --- | --- | --- | --- |
| Artifact | Version | Artifact | Version |
| AutoConf Resource Processor | 0.1.0 | Shell Text UI | 1.4.1 |
| Bundle Repository | 1.6.6 | Shell | 1.4.3 |
| Configuration Admin | 1.6.0 | UPnP Base Driver (JDK 1.3) | 0.8.0 |
| Dependency Manager Annotation | 3.1.0 | UPnP Base Driver (JDK 1.4+) | 0.8.0 |
| Dependency Manager Compat | 3.0.1 | UPnP Extra | 0.4.0 |
| Dependency Manager Runtime | 3.1.0 | UPnP Tester | 0.4.0 |
| Dependency Manager Shell | 3.0.1 | User Admin File-based store | 1.0.2 |
| Dependency Manager | 3.1.0 | User Admin MongoDB-based store | 1.0.1 |
| Deployment Admin | 0.9.0 | User Admin | 1.0.3 |
| Event Admin | 1.3.2 | Utils | 1.2.0 |
| File Install | 3.2.6 | Web Console DS Plugin | 1.0.0 |
| Framework Security | 2.2.0 | Web Console Event Admin Plugin | 1.0.2 |
| Framework | 4.2.1 | Web Console Memory Usage Plugin | 1.0.4 |
| Gogo Command | 0.12.0 | Web Console OBR Plugin | 1.0.0 |
| Gogo Runtime | 0.10.0 | Web Console PackageAdmin Plugin | 1.0.0 |
| Gogo Shell | 0.10.0 | Web Console ServiceDiagnostics Plugin | 0.1.2 |
| HTTP Service API | 2.2.0 | Web Console UPNP Plugin | 1.0.2 |
| HTTP Service Base | 2.2.0 | Web Console | 4.0.0 |
| HTTP Service Bridge | 2.2.0 | iPOJO API | 1.6.0 |
| HTTP Service Bundle | 2.2.0 | iPOJO Annotations | 1.8.6 |
| HTTP Service Jetty | 2.2.0 | iPOJO Architecture Command for Gogo | 1.0.1 |
| HTTP Service Proxy | 2.2.0 | iPOJO Architecture Command for the Felix Shell | 1.6.0 |
| HTTP Service Whiteboard | 2.2.0 |
| Lightweight HTTP Service Complete | 0.1.4 | iPOJO BND Plugin | 1.8.6 |
| Lightweight HTTP Service Core | 0.1.4 | iPOJO Composite | 1.8.6 |
| Log | 1.0.1 | iPOJO Event Admin handler | 1.8.0 |
| Main | 4.2.1 | iPOJO Extender pattern handler | 1.4.0 |
| Metatype | 1.0.6 | iPOJO JMX handler | 1.4.0 |
| OSGi OBR service API | 1.0.2 | iPOJO Manipulator | 1.8.6 |
| Preferences | 1.0.4 | iPOJO Temporal Dependency handler | 1.6.0 |
| Remote Shell | 1.1.2 | iPOJO WebConsole Plugin | 1.6.0 |
| Resolver | 1.0.0 | iPOJO Whiteboard pattern handler | 1.6.0 |
| SCR (Declarative Services) | 1.6.2 | iPOJO | 1.8.6 |
| SCR Annotations | 1.7.0 | junit4osgi - shell command | 1.0.0 |
| SCR DS Annotations | 1.2.2 | junit4osgi - swing gui | 1.0.0 |
| SCR Generator | 1.4.0 | junit4osgi | 1.0.0 |

* 1. Apache Felix开发简介

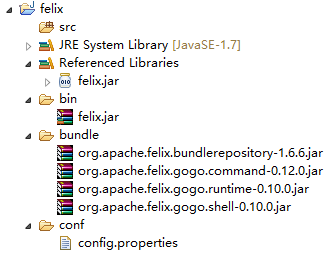
1. 下载Felix：<http://felix.apache.org/downloads.cgi>



下载后解压后的路径：

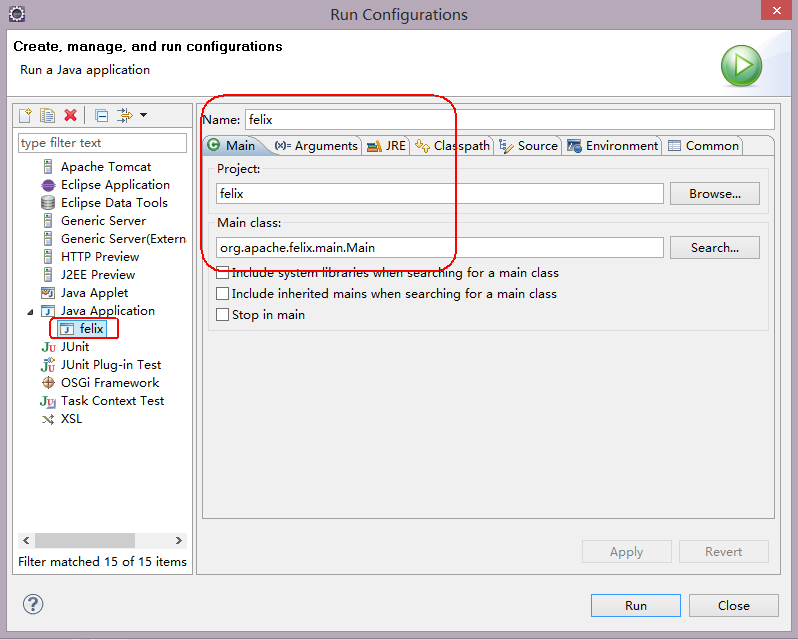


1. 在Eclipse创建Java项目：

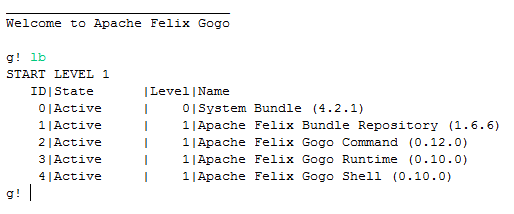


注意添加felix.jar的引用。

1. 添加运行参数配置：



1. 启动“Run”运行后，输入“lb”命令回车后控制台信息：



输入“help”命令回车，可以查看所有可用的控制台命令。

1. 基于Eclipse开发向导开发一个OSGi实例，有两种方式可以部署：

一种方式可以通过“Install”命令进行安装部署，然后使用“start”命令启动。启动的时候，start命令后接着那个bundle的启动ID就可以启动bundle了。

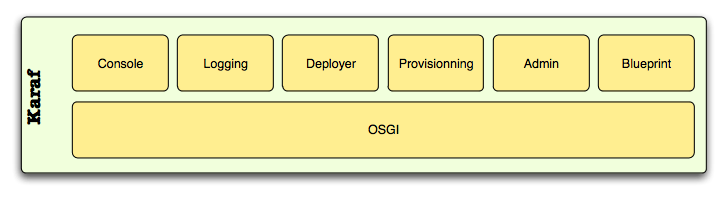
另外一种方式是通过修改conf/config.properties配置文件来加载bundle。

说明：因为本文档主要目的不是提供操作指南，所以具体开发细节就不再详细描述，会有其它专门的文档提供。

* 1. Apache Felix Karaf

Apache Karaf起源自Apache Servicemix4的kernel， 一个基于OSGi core的项目，因此后来Apache Servicemix4 kernel成为了Apache Felix的一个子项目， 并且改名为Karaf，经过一段时间的发展，社区认为Karaf独立成为一个顶级项目会更有利于该项目的发展，因此经过相关的讨论和投票表决，于2010年6月份开始Karaf成为Apache的一个顶级项目（TLP, Top Level Project）。

Karaf是个运行时包，包含了一个OSGi框架（Equinox或Felix）、一个命令shell（Felix Gogo）及默认情况下内置的大量实用工具。参考下图所示了解其它构成。



虽然Equinox与Felix可以单独使用，但Karaf旨在结合这两个框架出色的OSGi功能，并且保证其开箱即用。比如说，它包含了一个可配置的日志系统（基于Log4J，但针对众多通用的日志系统进行了包装）、通过SSH实现的远程访问、通过ConfigAdmin（源代码位于etc目录中）的配置以及内建的JAAS支持。不仅如此，Karaf还安装了Pax URL的MVN协议，这样就可以从Maven中央仓库（在必要的情况下会自动将其包装为bundle）安装bundle了。

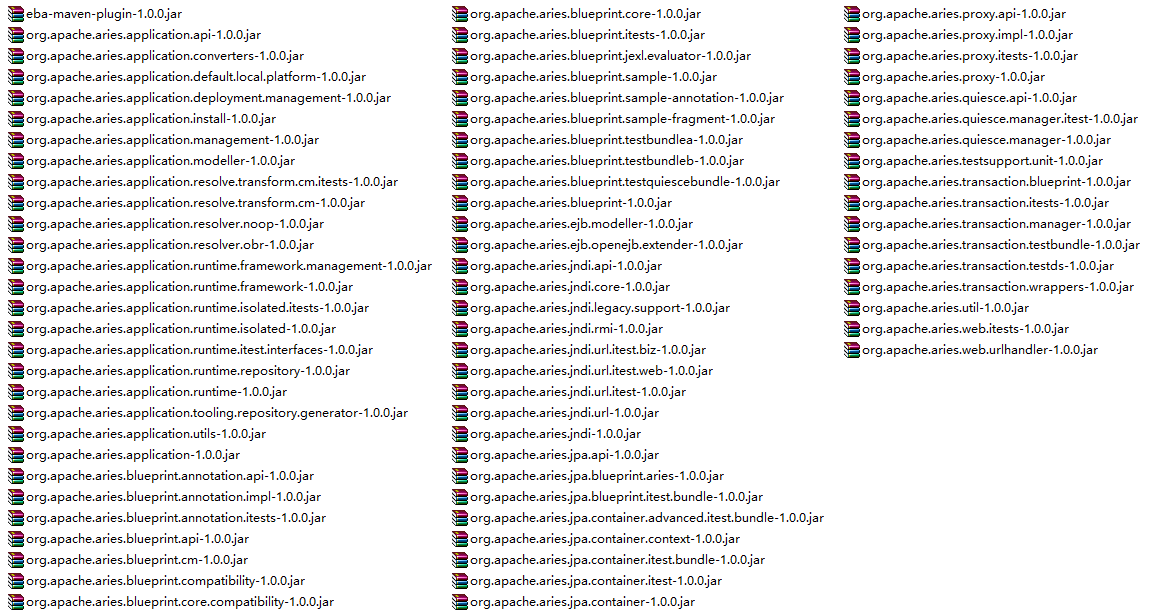
此外，Karaf还提出了特性的概念，所谓特性就是bundle的集合，能以组的形式安装到运行着的OSGi运行时当中。特性包含了对obr(Apache Felix OSGi Bundle Repository)、jetty以及spring的支持，做到了开箱即用。这样，如果需要安装多个bundle，但这些bundle之间并没有严格的运行期依赖，那么这种支持就可以大大简化这种情况。

在迁移到Apache Felix项目中前Karaf是ServiceMix Kernel，并且最终成为了Apache的顶级项目。Karaf还加入到了其他框架当中，如Eclipse Virgo和EclipseRT packages，提供了预先配置的框架与好用的OSGi bundle，这样在上手使用OSGi运行时时就会比以往更加简单。

同时Karaf已经被诸多Apache项目作为基础容器，例如:Apache Geronimo、Apache ServiceMix、Fuse ESB，由此可见Karaf在性能、功能和稳定性上都是个不错的选择。

* 1. [Apache Aries](http://incubator.apache.org/aries)

Aries 项目提供了一组可嵌入式的 Java 组件，用于企业 OSGi 应用编程模型，该项目还包含了 OSGI 联盟企业专家组的规范实现和扩展。可以从下图中大致了解下Aries提供的支持。



* 1. 选型比较

|  |  |  |
| --- | --- | --- |
|  | Eclipse Equinox | Apache Felix |
| 开源协议 | Eclipse Public License Version 1.0 | Apache License Version 2.0 |
| 最新版本 | 3.8.2 | 4.2.1 |
| OSGi版本  兼容性 | Equinox 3.7中已经实现了OSGi 4.3运行时 | Felix V4.0.0已经通过OSGi R4.3兼容性测试 |
| 资料数量 | ★★★ | ★★ |
| 配套项目 | ★★★ | ★★★★★ |
| 易学习程度 | ★★★ | ★★★★ |

1. 总结

单纯从OSGi容器的角度比较Eclipse Equinox和Apache Felix是没有太多实际意义的，毕竟OSGi规范本身就是对标准化、兼容性的一个强制约束。两款产品都有强大的开源组织和社区做后盾，成熟性、稳定性、版本更新的及时性都是有优势的。我们选择一款产品的主要目的还是期望基于此产品的学习曲线会比较低，资料能够比较丰富，源代码可读性比较强，同步配套的项目比较丰富等。综合考虑，个人认为由于Eclipse Equinox由于自身源自于Eclipse项目，可能更多会倾向于Eclipse集成开发环境的支持和基于Eclipse的工具的开发；而Apache Felix配合Apache Aires等项目对Web应用开发提供了更加广泛的支持。因此建议将Apache Felix作为运行时环境的首选。