新一代轻量级应用服务器 — WebSphere Liberty Profile Server 介绍

## 什么是 WebSphere Liberty Profile Server ？

Liberty 是一款全新的轻量级应用服务器，它将用户的良好开发体验作为最主要的出发点。其主要特点和内容包括：

* 高模块化——该功能允许用户根据自己应用程序的需求启用或者禁用相关的 feature（所谓 feature，在这里指的是运行应用程序所需要的各种资源的支持。比如，应用程序用到了 JSP，我们就需要启动 JSP 这个 feature，如果不在需要此 feature，就可以将其禁用。通过这种模块化的控制，我们可以按需启动不同的 feature，包括 JSP, Servlet, JPA 等等，这种控制是实现轻量级应用服务器的基础）。
* 轻量级—— Liberty 采用了多种技术进行瘦身和优化（主要是 OSGi 的应用），与传统的 WAS 相比，极大的减少了运行时的内存消耗。
* 高动态性——由于采用了 OSGi 的架构，程序员在开发的时候，可以动态修改配置文件，应用程序以及服务器的运行时信息，这些修改都是实时生效的，不需要重启服务器。
* 快速——由于采用了高模块化的设计，Liberty 应用服务器会“按需启动”，自身消耗的系统资源很少，对执行任务的反应时间也非常短，例如，一般来说 Liberty 的启动时间小于 5 秒。
* 配置简单——在整个开发过程中，用户只需要和一个配置文件（server.xml）打交道，而且配置的使用也非常灵活。

尽管 Liberty 与传统 WAS 相比只是一个“小块头”，但是在 Web 和 OSGi 应用程序的开发上，Liberty 与传统 WAS 完全保持一致，在 Liberty 上开发的程序，可以直接移植到传统 WAS 上，不需要做任何的改变。相比传统 WAS 的庞大，Liberty 在为用户提供良好的开发体验上遥遥领先。

作为应用服务器，Liberty 支持与多种开发工具的结合：

* 支持在 Rational Application Developer 中使用 Liberty，推荐在企业级应用开发中使用这种方式，因为这种方式支持最大范围的编程模型以及 Cloud。
* 支持在 Eclipse 3.6(Helios) 或者 3.7(Indigo) 中使用 Liberty, 需要从 Eclipse Marketplace 安装 WebSphere Application Server V8.5 Developer Tools for Eclipse 或者 WebSphere Application Server V8.5 Liberty Profile Developer Tools for Eclipse。在 Eclipse 中使用 Liberty 是完全免费的，当然，如果你需要得到 IBM 工程师的支持，则需要支付额外的费用。

同时，Liberty 服务器也支持多种主流的操作系统平台，包括 Windows、Linux、Unix、z/OS 以及 Mac OS 等。Liberty 既可以用在开发环境中，也可以用于产品环境（Mac OS 除外）。Liberty 同时支持 Oracle 和 IBM JDK，支持的最低版本分别是：Oracle Java™ 6 update 26 和 IBM Java 6.0 (J9 2.6) SR 1。

## 初探 WebSphere Liberty Profile Server

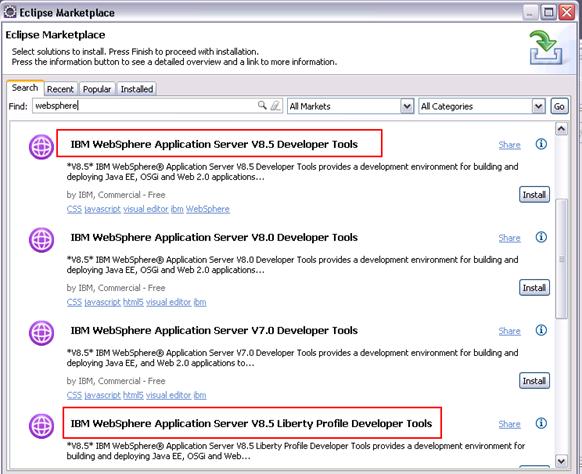
### 快速安装

在本文中我们主要介绍 Eclipse 与 Liberty 的结合使用，在开始搭建 Liberty 的开发环境前，读者需要自行下载和安装 Oracle 或者 IBM JDK，并配置好环境变量，这是 Liberty 能正常运行的必要条件。如果是以开发为目的，那么使用 Liberty 以及 Liberty 工具是完全免费的，并且没有时间限制。

**在线安装 Liberty 开发工具**

打开 Eclipse，依次点击 Help > Eclipse MarketPlace，在出现的页面搜索栏里输入 websphere，并点击搜索。

##### 图 1. 安装 Liberty 开发工具



在搜索结果中会看到如上红色方框标注的工具，其中 WebSphere Application Server V8.5 Liberty Profile Developer Tools 是 WebSphere Application Server V8.5 Developer Tools 的子集，对于搭建 Liberty 开发环境，直接安装前者就可以，如果读者还需要将 Eclipse 与传统的 WAS 结合使用，那么就需要安装后一个工具。在本文中，我们直接安装 WebSphere Application Server V8.5 Liberty Profile Developer Tools。在安装过程结束后，需要重启 Eclipse。待 Eclipse 重启后，选择 Servers 视图，右击新建一个服务器，出现如下图所示，则表明 Liberty 开发工具已经成功安装

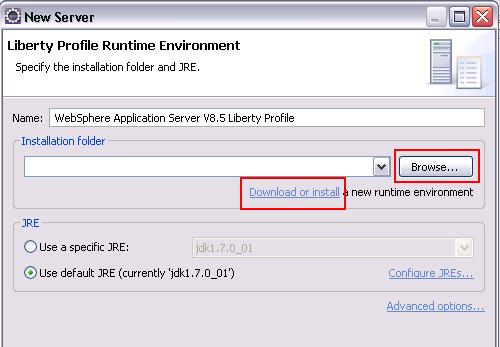
##### 图 2. 验证 Liberty 开发工具安装成功



**创建 Liberty 服务器**

在安装好 Liberty 开发工具后，我们需要在 Eclipse 中创建 Liberty 服务器，接着图 2 所示，选择 WebSphere Application Server V8.5 Liberty Profile，点击 Next

##### 图 3. 选择 Liberty 的安装位置



在上图中，我们需要指定 Liberty 服务器的安装位置，对于如何下载 Liberty 服务器，有两种选择：1. 访问 wasdev.net 站点，下载 Liberty 的安装文件，Liberty 安装文件只有不到 50M，下载成功后只需要解压该文件到磁盘的任何一个位置即可完成 Liberty 的安装，“解压即安装”，安装 Liberty 就是如此简单。然后，点击 Browser 选择 Liberty 的安装位置，即可继续创建 Liberty 服务器；2. 直击点击上图的 Download or install，安装提示进行操作，只需要几分钟就可以在线下载并安装一个全新的 Liberty 服务器。在这里我们需要指出的是，您也可以选择直接使用 Liberty 服务器，而不是将其与 Eclipse 搭配起来使用，直接解压下载的 Liberty 到任意目录，就可以使用了。点击 Next，因为 Liberty 默认是没有创建服务器的，所以可以得到下图

##### 图 4. 空白的 Liberty 服务器



点击 New，创建你的第一个 Liberty 服务器

##### 图 5. 创建 Liberty 服务器



为你的 Liberty 任意指定一个名字，点击 Finish 完成创建。在 Servers 视图中，我们将看到创建好的 DemoServer。

##### 图 6. Liberty 服务器概览



从上图中我们可以看到 DemoServer 是停止状态，右键 DemoServer，选择 Start 启动 Liberty 服务器，在 Console 视图中检查 log，可以看到 Liberty 已经正常启动。

##### 清单 1. Liberty 启动日志

|  |  |
| --- | --- |
| 1  2  3  4  5 | Launching DemoServer (wlp-1.0.0.20120428-1251/  websphere-kernel\_1.0.0)  on Java HotSpot(TM) Client VM, version 1.7.0\_01-b08 (en\_US)  [AUDIT   ] CWWKE0001I: The server DemoServer has been launched.  [AUDIT   ] CWWKZ0058I: Monitoring dropins for applications.  [AUDIT   ] CWWKF0011I: The server DemoServer  is ready to run a smarter planet. |

在上图中，双击 Server Configuration，可以看到如下配置文件（server.xml）。

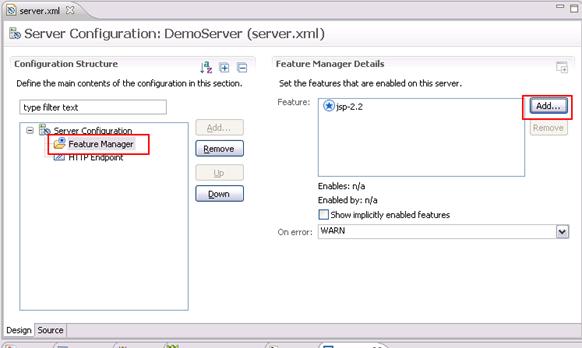
##### 清单 2. 初始的 server.xml 文件

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | <server description="new server">       <!-- Enable features -->     <featureManager>         <feature>jsp-2.2</feature>     </featureManager>       <httpEndpoint id="defaultHttpEndpoint"                   host="localhost"                   httpPort="9080"                   httpsPort="9443" />    </server> |

这个 server.xml 就是 Liberty 服务器中唯一的配置文件，我们在开发过程中对应用程序或者 Liberty 运行时的配置都是在这个文件中完成。在默认情况下，Liberty 已经启用了 JSP 功能，我们可以动态的根据应用的需求为 Liberty 增加各种功能支持，方法如下：

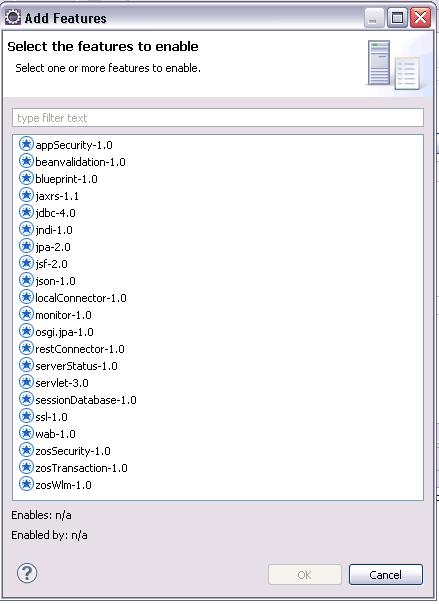
在 server.xml 编辑窗口下方，选择 Design 模式，

##### 图 7. 图形化编辑配置文件



选择左侧 Feature Manager，然后选择右侧 Add，出现下图列表，根据需要添加相应的功能支持。

##### 图 8. Features 列表



当然，读者也可以直接手工编辑 server.xml，手动添加需要的 Feature，但是要注意遵循 Liberty 配置文件的编写规范。到此，我们已经成功的安装了 Liberty 服务器，并搭建好了 Liberty 服务器的开发环境。开始我们的例子程序之前，下文首先向读者更深层次的介绍 Liberty 的功能特性。

### 简化服务器配置

在 Liberty 中，和用户打交道的只有一个配置文件 server.xml。关于应用程序或者服务器运行时的所有配置都是在这一个文件中完成的，简化了服务器的配置，也减轻了程序员的负担。我们可以在 Eclipse 中直接编辑该文件，修改内容会实时生效。这个配置文件是可以被导出，也可以被不同的服务器共享。

### 灵活的服务器配置

##### 图 9. 灵活配置

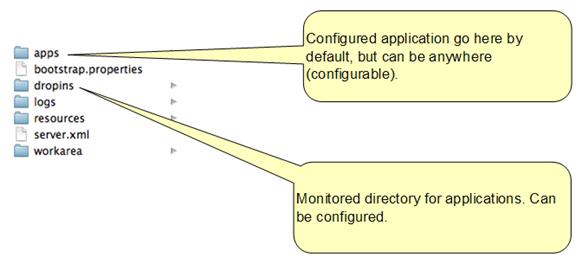


对于配置文件的使用是非常灵活的，例如：如果有很多个 Liberty 都需要用到同一个配置文件，那么可以将该文件单独导出，存放在一个共享的位置，单个的 Liberty 服务器可以利用 <incluse /> 属性导入共享的配置文件；又比如配置文件的内容非常多而且繁杂，我们就可以将其中的内容分别写在不同的配置文件中（称为不同的配置文件片段），同样利用 <include /> 将这些片段文件导入到主配置文件中，运行时会负责去实时的解析这些片段文件。

### 应用部署

在 Liberty 中部署应用也是非常简单的事情，打开你的 Liberty 安装目录 <Liberty\_Install\_dir>\usr\servers\server\_name ，可以看到如下目录结构 :

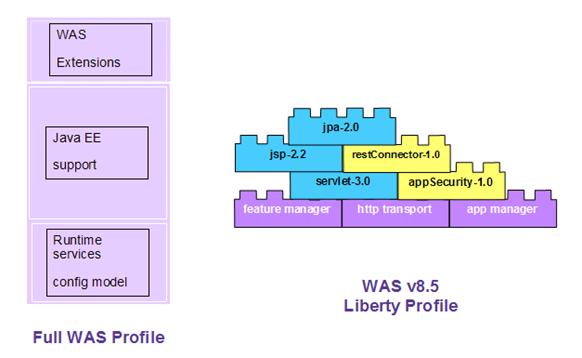
##### 图 10. 应用部署目录



在上图中 apps 和 dropins 是两个存放安装的应用的文件夹，我们可以通过 Eclipse 部署应用到 Liberty，应用文件将被放在 apps 下面（这个是应用部署的默认目录，也可以通过配置，将应用文件部署到其他任何位置）。我们也可以直接拷贝应用程序（例如 war 文件）到 dropins 目录，当 Liberty 启动的时候，就会自动扫描上面的两个文件夹，然后启动文件夹内的应用程序。当然，启动应用前，要确保已经配置好 server.xml 里的相关信息。

### 动态加载模块

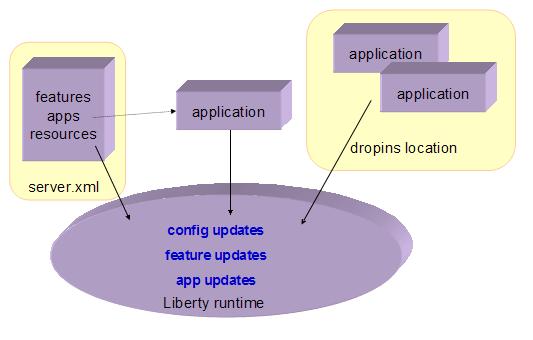
##### 图 11. 动态化加载



与传统 WAS 不同的是，Liberty 并不是一个静态的服务器，它是动态的，可以根据应用程序的需要动态加载相应的模块。这些模块是以“feature”来定义的，比如 JSP, Servlet 等，都是不同的 feature。可以将 feature 看作是一个独立的模块，它由一到多个 OSGi bundle 组成，共同来完成某一个特定的任务。这些 feature 可是随时被添加和移除，而不需要重启服务器，正是得益于这样的动态化设计，Liberty 无论在启动时间还是反应时间上都更胜一筹，为程序员带来了快速的开发体验。

### 运行时动态更新

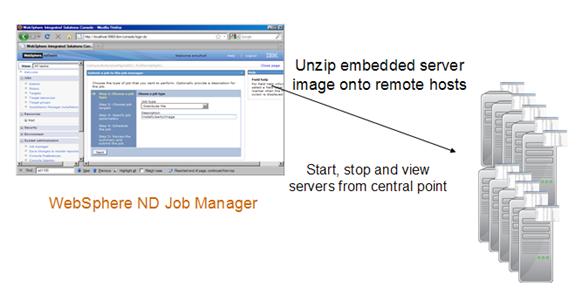
##### 图 12. 动态更新



Liberty 为我们提供了一个可以动态实时更新的运行时，无论你是修改了应用程序，还是修改配置文件都会实时的被 Liberty 检测到，并立即更新运行时到最新状态，无需重启 Liberty 服务器。新的 feature 可以被随时的添加到 server.xml 中，运行时为负责实时启动相应的功能模块，这一切都与程序员来说都是透明的。例如：我们首先启动 Liberty 服务器，然后将一个 JSP 应用放到 dropins 目录里，如果你没有在 server.xml 中启动 JSP 功能，那么该应用就不会被启动。你将会得到一个警告信息，提示你应该首先启用 JSP 功能，然后该应用才会被实时的启动。同样，如果你将 JSP 功能从 server.xml 中移除，该应用也会被立刻停止。

### 新型的软件部署方式

##### 图 13. 集中管理 Liberty 服务器



当我们使用 Liberty 开发完应用程序后，Liberty 为我们提供了一个全新的软件部署方式，我们可以将 Liberty 服务器和应用程序一起打包为 ZIP 文件，然后将这个 ZIP 传输到需要部署软件的服务器上直接解压即可。为了支持适合云端和大规模应用的部署，WebSpere（传统 WAS）还提供了集中管理部署的功能（需要安装 WAS ND 版本），使用该功能可以将我们事先打好的 ZIP 包远程的传输到多个服务器上，在传送完成后，WAS 还会负责解压和安装。另外，WAS 的集中管理功能还提供了远程启动、停止和卸载 Liberty 服务器的功能。

### Liberty 之于开发人员

对于开发人员来说，Liberty 的出现是一个十分振奋人心的消息。首先，Liberty 作为一个轻量级的应用服务器，和 Eclipse 可以完美的结合，其最重要的特性就是关注程序员的开发体验，包括快速的下载、安装和环境搭建，在开发过程中，实时的应用更新而无需重启服务器，这些都会极大的提高程序员的开发效率；其次，在开发，测试和部署应用的过程中，Liberty 都极其友好和方便，其与传统 WAS 的完全兼容，也为应用程序的迁移提供了有力的保障。

## 与主流 WEB 容器的对比

从某种程度上来讲，目前，Liberty 与 Tomcat 处在同一个产品层次上，我们从功能特性、使用体验以及标准支持上对这两个服务器进行了对比，从中，我们可以看出 Liberty 在轻量级服务器中所具有的独特优势：

##### 表 1. Liberty VS Tomcat

| **对比项** | **Tomcat** | **Liberty** | | |
| --- | --- | --- | --- | --- |
| 大小 | <30M | <50M | | |
| 启动速度 | <5s | <5s | | |
| Java EE 标准支持 | JSP 2.2, Servlet 3.0, JSTL 1.2 用户需要添加相关的 jar 包来实现对其它标准的支持 | JSP 2.2, Servlet 3.0, JSF 2.0, JSTL 1.2,  JPA 2.0, JDBC 4.0, JAX-RS 1.0,  JNDI 1.0, Bean Validation 1.0,  SSL 1.0, Security 1.0,  Web Security 1.0,  JMX 1.0  可动态加载或卸载对应的 feature | | |
| OSGi 编程模型 | 否 | 是，支持 Blue Print 1.0, WAB 1.0 | | |
| 开发工具 | 支持 Eclipse | 支持 eclipse,  IBM Rational Application Developer | | |
| 支持的操作系统 | Linux, Windows | Linux, Windows, AIX,  Mac, HP-UX, Solaris, iSeries, zSeries | | |
| 性能 | 一般 | 与传统 WAS 具有相同的核心代码，  性能好 | | |
| 应用程序热部署 | 支持 | 支持 | | |
| 配置文件 | 多个配置文件，更新后需重启服务器 | 一个配置文件，更新后及时生效，  无需重启 | | |
| 可移植性 | 您可以利用 IBM Application Migration Toolkit 将 Tomcat 上开发的应用快速移植到 WebSphere 应用服务器。反之，则不然。 | Liberty 和传统 WAS 对编程模型  和标准的支持一致。  在 Liberty 上开发的应用可以  直接运行在传统 WAS 之上 | | |
| 文档 | 官方文档涵盖内容有限，但网络上  相关内容（包括：blog，技术文章等）广泛。 | | http://wasdev.net 上提  供下载地址、文档、视频、  音频、样本引用等。  还有 WAS 的信息中心  也有大量文档可供参考阅读。 | |
| 社区 | http://tomcat.apache.org 有 dev  和 user mailing list，没有官方支持，  有时回复不够及时。 | | http://wasdev.net，可以与  开发人员直接交流 |
| 长期战略优势 | 一般用于开发环境，生产环产能  用的比较少。支持的并发度不高，  可扩展性不强。 | | Liberty 与传统 WAS  共享同一代码，天生具  有其高效稳定的特点。  开发 - 测试 –  生产切换零代价。  可扩展性高，  支持多个 server 间的  负载均衡和故障恢复。 | |

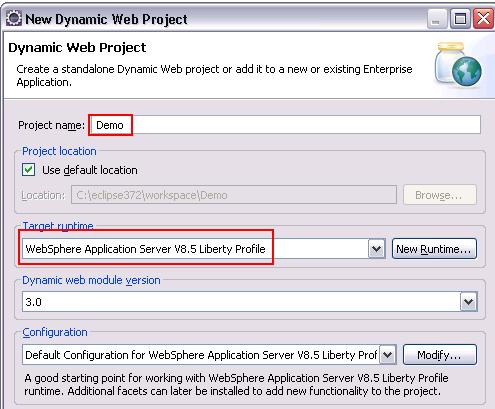
## Demo：使用 Eclipse 和 Liberty 开发 Web 应用

通过上面的介绍，相信大家对 Liberty 已经有了一个大概的认识，接下来我们通过一个 Demo 来向大家展示如何使用 Liberty 和 Eclipse 开发 Web 应用。开始 Demo 之前，请大家根据上文的介绍，搭建自己的 Liberty 开发环境，并安装好 JDK 并且配置好环境变量。

Demo 程序很简单，包含一个 Servlet，实现打印当前时间的功能。步骤如下：

1. 打开 Eclipse，选择 File > New > Web>Dynamic Web Project
2. 设置如下属性值：
   1. Project name: Demo
   2. Target runtime: WebSphere Application Server V8.5 Liberty Profile

##### 图 14. 创建动态 Web 工程

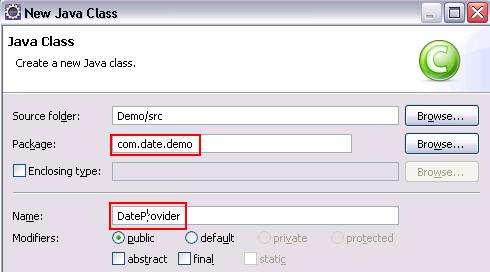


1. 点击 Next 两次，然后点击 Finish，完成工程的创建。

此时，你可以在左侧的资源导航页面看到刚刚创建完成的 Demo 工程，展开工程就可以看到一些默认创建的资源。

1. 右击 Src 目录，选择 New > Class
2. 设置如下属性值：
   1. Package: com.date.demo
   2. Name: DateProvider

##### 图 15. 创建 java 文件



1. 点击 Finish
2. 编辑如下代码到 DateProvider.java

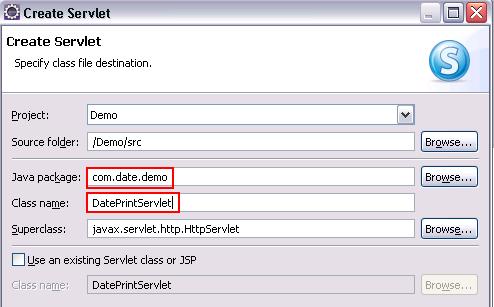
##### 清单 3. DateProvider.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | package com.date.demo;    import java.util.Date;    public class DateProvider {    public String getDate() {  // TODO Auto-generated method stub  return new Date().toString();  }  } |

该文件只包含一个方法，getDate() 用来获得当前时间。

1. 右击 Src，选择 New > Servlet
2. 设置如下属性值
   1. Java package: com.date.demo
   2. Class name: DatePrintServlet

##### 图 16. 创建 Servlet 文件



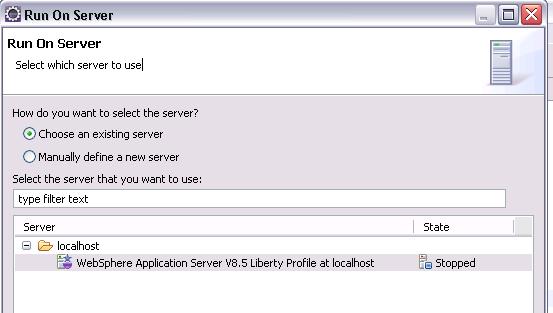
1. 点击 Finish，完成创建。
2. 编辑如下内容到 DatePrintServlet.java

##### 清单 4. DatePrintServlet.java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41 | package com.date.demo;    import java.io.IOException;  import javax.servlet.ServletException;  import javax.servlet.annotation.WebServlet;  import javax.servlet.http.HttpServlet;  import javax.servlet.http.HttpServletRequest;  import javax.servlet.http.HttpServletResponse;    /\*\*  \* Servlet implementation class DatePrintServlet  \*/  @WebServlet("/DatePrintServlet")  public class DatePrintServlet extends HttpServlet {  private static final long serialVersionUID = 1L;     /\*\*      \* @see HttpServlet#HttpServlet()      \*/     public DatePrintServlet() {         super();         // TODO Auto-generated constructor stub     }  /\*\*  \* @see HttpServlet#doGet(HttpServletRequest request,  HttpServletResponse response)  \*/  protected void doGet(HttpServletRequest request,  HttpServletResponse response)   throws ServletException, IOException {  // TODO Auto-generated method stub  DateProvider dateProvider=new DateProvider();  response.getWriter().println(dateProvider.getDate());  }    /\*\*  \* @see HttpServlet#doPost(HttpServletRequest request,  HttpServletResponse response)  \*/  protected void doPost(HttpServletRequest request,  HttpServletResponse response)  throws ServletException, IOException {  // TODO Auto-generated method stub  }    } |

1. 在左侧导航栏选择 DatePrintServlet，点击右键 Run As > Run on Server , 如下图：

##### 图 17. 选择 Liberty 服务器



1. 点击 Finish，Liberty 开发工具将会为我们部署上面的 Demo 程序，并启动 Liberty 服务器。从 console 的日志里我们可以看到如下描述：

[AUDIT ] CWWKZ0001I: Application Demo started in 1.235 seconds.

表明 Demo 应用已经成功部署了，同时，在 Eclipse 自带的浏览器里可以看到如下页面信息：

##### 图 18. Demo 运行页面



让我们回过头来看看 server.xml 里面的内容：

##### 清单 5. server.xml

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | <server description="new server">       <!-- Enable features -->     <featureManager>         <feature>jsp-2.2</feature>         <feature>localConnector-1.0</feature>       </featureManager>       <httpEndpoint host="localhost" httpPort="9080"  httpsPort="9443"     id="defaultHttpEndpoint"/>       <applicationMonitor updateTrigger="mbean"/>       <application id="Demo" location="Demo.war"  name="Demo" type="war"/>  </server> |
|  |  |

其中：

<feature /> 包含了应用程序中所用到的功能包，如果你启用了 jsp-2.2，那么 Liberty 默认支持对 Servlet 3.0。

<httpEndpoint /> 指定了 HTTP 访问的端口，我们可以在这里改变默认的端口。

<applicationMonitor /> 用来配置对应用程序的监控模式，updateTrigger 属性是用来控制动态更新的，通过 Liberty 工具部署应用时 updateTrigger 的值设置为 mbean，当我们直接把应用程序拖进 dropins 时，需要将 updateTrigger 设为 polled，当然，如果我们不需要应用的动态更新功能，可以将这个值设为 disabled.

<application /> 用来指定和应用本身相关的属性，id 和 name 是应用的唯一标识，type 指定了应用的类型，location 指定应用所在的位置，默认是在 apps 目录里面，我们可以通过配置修改这个默认设置。

## 总结

通过上面的介绍，相信读者已经对 WAS V8.5 最重要的特性 Liberty 有了一个全面的认识，作为新一代轻量级应用服务器，Liberty 采用了和传统 WAS 相同的代码，继承了传统 WAS 的优点，无论在性能上还是在稳定性上都优越于其他同类产品。Liberty 无论是在开发环境中还是在产品环境中，都具有其他产品不可比拟的优势。我们有理由相信，在追求开发效率和软件质量的今天，每一个程序员都会很快的接受这个 WAS 家族的“小块头”。

#### 相关主题

* 访问 [WebSphere Application Server V8.5 信息中心](http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp) 了解更多关于 WAS 的信息。
* 访问 [wasdev.net 主页](http://wasdev.net/) 学习更多 Liberty 的知识，并直接和开发人员进行交流。
* 访问 [WebSphere Developer Tools V8.5 信息中心](http://publib.boulder.ibm.com/infocenter/radhelp/v8r5/index.jsp) 学习更多 WDT 的知识。
* [IBM developerWorks 中国 WebSphere 专区](http://www.ibm.com/developerworks/cn/websphere/)：为使用 WebSphere 产品的开发人员准备的技术信息和资料。这里提供产品下载、how-to 信息、支持资源以及免费技术库，包含 2000 多份技术文章、教程、最佳实践、IBM Redbook 和在线产品手册。
* [最受欢迎的 WebSphere 试用软件下载](http://www.ibm.com/developerworks/cn/websphere/downloads/)：下载关键 WebSphere 产品的免费试用版。