JavaWEB

# JavaWEB基础知识

## 基础概念

* WEB，在英语中web即表示网页的意思，它用于表示Internet主机上供外界访问的资源。
* Internet上供外界访问的Web资源分为：
  + 静态web资源（如html 页面）：指web页面中供人们浏览的数据始终是不变。
  + 动态web资源：指web页面中供人们浏览的数据是由程序产生的，不同时间点访问web页面看到的内容各不相同。
* 静态web资源开发技术
  + Html
* 常用动态web资源开发技术：
  + JSP/Servlet、ASP、PHP等
  + 在Java中，动态web资源开发技术统称为Javaweb，我们课程的重点也是教大家如何使用Java技术开发动态的web资源，即动态web页面。

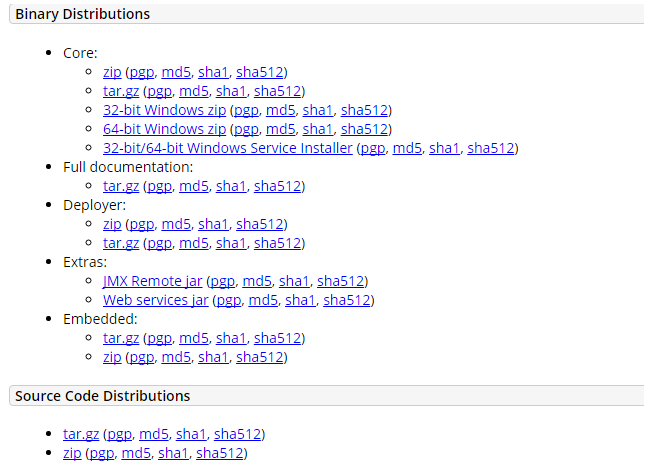
## 常见的web服务器

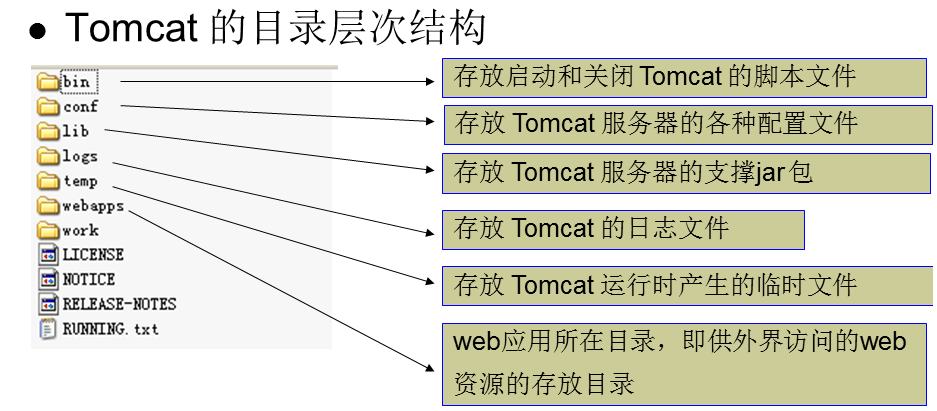
* WebLogic是BEA公司的产品，是目前应用最广泛的Web服务器，支持J2EE规范，而且不断的完善以适应新的开发要求，启动界面如图
* 另一个常用的Web服务器是IBM公司的WebSphere，支持J2EE规范，启动界面如图
* **在小型的应用系统或者有特殊需要的系统中，可以使用一个 免费的Web服务器：Tomcat，该服务器支持全部JSP以及Servlet规范，启动界面如图**

## 下载和配置Tomcat

**Tomcat官方站点**：https://tomcat.apache.org/download-80.cgi

* **获取Tomcat安装程序包**
  + tar.gz文件是Linux操作系统下的安装版本
  + exe文件是Windows系统下的安装版本
  + zip文件是Windows系统下的压缩版本
* **安装Tomcat**



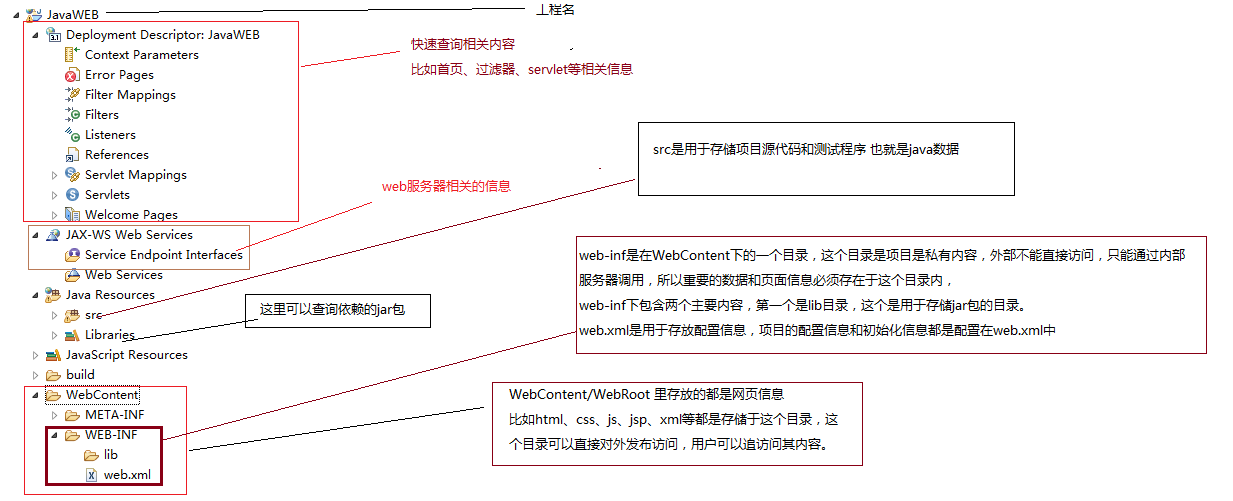


Eclipseee中配置tomcat

# Servlet开发

* Servlet是sun公司提供的一门用于开发动态web资源的技术。
* Sun公司在其API中提供了一个servlet接口，用户若想用发一个动态web资源(即开发一个Java程序向浏览器输出数据)，需要完成以下2个步骤：
  + 编写一个Java类，实现servlet接口。
  + 把开发好的Java类部署到web服务器中。
* 快速入门，用servlet向浏览器输出“hello servlet”。
  + 阅读Servlet API，解决两个问题：
  + 输出hello servlet的java代码应该写在servlet的哪个方法内？
  + 如何向IE浏览器输出数据？

## Web的目录结构



## Servlet入门基础

**方式一： 使用配置文件的方式配置servlet**

**package** com.guigu.base;

**import** java.io.IOException;

**import** javax.servlet.ServletException;

**import** javax.servlet.http.HttpServlet;

**import** javax.servlet.http.HttpServletRequest;

**import** javax.servlet.http.HttpServletResponse;

**public** **class** HelloServlet **extends** HttpServlet {

@Override

**protected** **void** doPost(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {

doGet(req, resp);

}

@Override

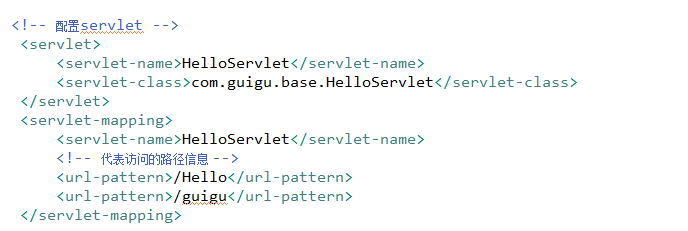
**protected** **void** doGet(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {

resp.getOutputStream().write("hello servlet".getBytes());

}

}

在web.xml中进行相关的配置



**方式二： 使用注解的方式配置servlet**

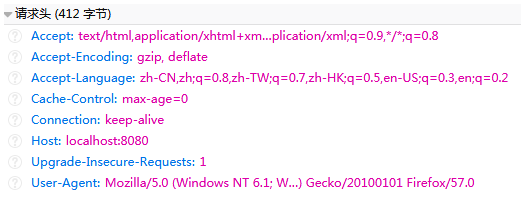


## Servlet的运行流程

Servlet

WEB服务器

http://localhost:8080/JavaWEB/helloguigu



2.发送http请求

1.连接web服务器

IE浏览器

3.解析客户端需要访问的主机

http://localhost:8080

4.解析需要访问的web应用

http://localhost:8080/JavaWEB

http://localhost:8080/JavaWEB/helloguigu

5.接续客户端需要访问的web资源

6.如果是第一次访问servlet 服务器就是加载servlet对象，并且创建servlet对象

11.web服务解析response对象然后根据解析的结果

返回给浏览器

10.把response对象返回给web服务器

9.service方法执行完毕会把数据写入到response对象中

调用service方法

response.getWriter().append("Served at: guigu\_hello").append(request.getContextPath());

8.调用service方法，service方法会根据请求

分别调用doGet和doPost方法

调用init方法

7.如果是第一次访问调用servlet的init方法进行初始化

## Servlet的生命周期

笔面试题 请描述servlet的生命周期

**Servlet的生命周期和工作原理**

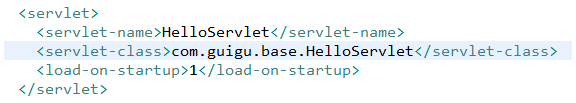
Servlet的的声明周期分为三个阶段，1.初始化阶段。2响应客户阶段。3.终止阶段

这三个阶段分别对应init方法 service方法 和destory方法 这三个方法本别在不同的阶段进行调用。

**1）Servlet的初始化阶段**

在以下时刻servlet进行初始化

1.servlet容器启动后，自动加载某些配置信息。需要配置<load-on-startup></load-on-startup>



2.在servlet容器启动后，客户端第一次向servlet发起请求

3.servlet的文件被更改后 重新加载servlet

**注意： init方法只会在第一次初始化servlet的时候执行。**

**2）servlet响应客户端阶段**

Servlet当接受到客户端请求后，servlet会根据这个请求创建两个对象HttpRequest对象和HttpResponse对象，这个对象分别代表接受请求和响应请求，然后servlet调用service方法。Service方法会根据HttpRequest对象从请求中获取相关的数据，然后处理请求，最后通过HttpResponse对象把数据写出到客户端。

**3）servlet终止阶段**

当web应用被终止，或者servlet容器终止运行，或servlet容器被重新加载

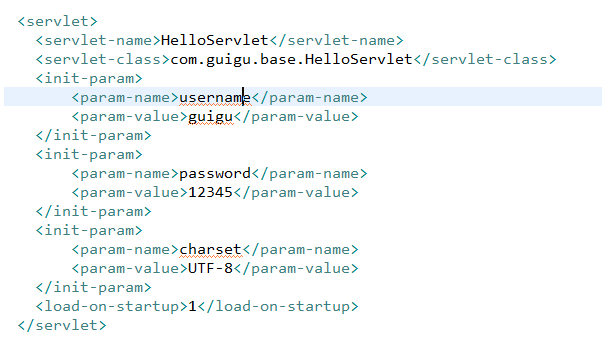
当前终止的时候回调用destroy方法进行终止。

**连接源码**

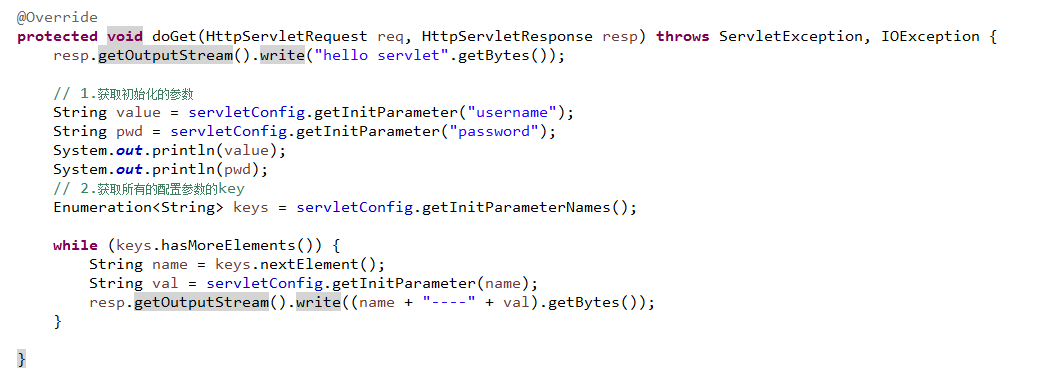
## Servlet相关的对象

### ServletConfig对象

* 在Servlet的配置文件中，可以使用一个或多个<init-param>标签为servlet配置一些初始化参数。
* 当servlet配置了初始化参数后，web容器在创建servlet实例对象时，会自动将这些初始化参数封装到ServletConfig对象中，并在调用servlet的init方法时，将ServletConfig对象传递给servlet。进而，程序员通过ServletConfig对象就可以得到当前servlet的初始化参数信息。
* 阅读ServletConfig API，并举例说明该对象的作用：
  + 获得字符集编码
  + 获得数据库连接信息
  + 获得配置文件，查看struts案例的web.xml文件



**HelloServlet**



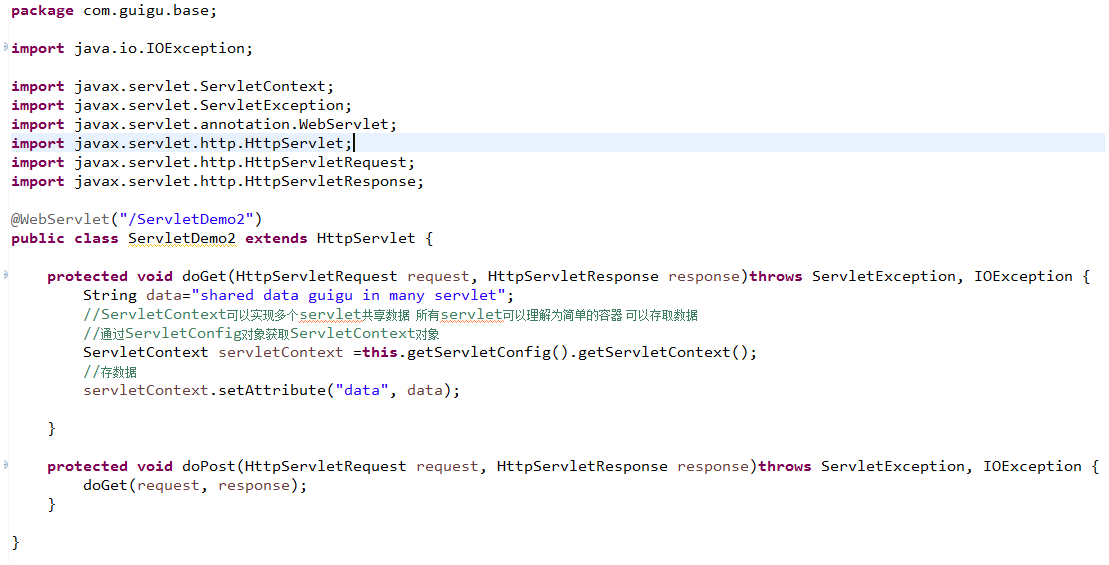
方式二在注解开发中使用配置参数



### ServletContext

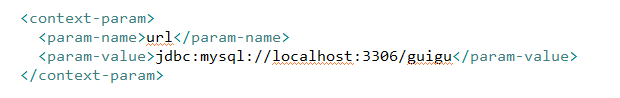
* WEB容器在启动时，它会为每个WEB应用程序都创建一个对应的ServletContext对象，它代表当前web应用。
* ServletConfig对象中维护了ServletContext对象的引用，开发人员在编写servlet时，可以通过ServletConfig.getServletContext方法获得ServletContext对象。
* 由于一个WEB应用中的所有Servlet共享同一个ServletContext对象，因此Servlet对象之间可以通过ServletContext对象来实现通讯。ServletContext对象通常也被称之为**context域对象**。
* 查看ServletContext API文档，了解ServletContext对象的功能。

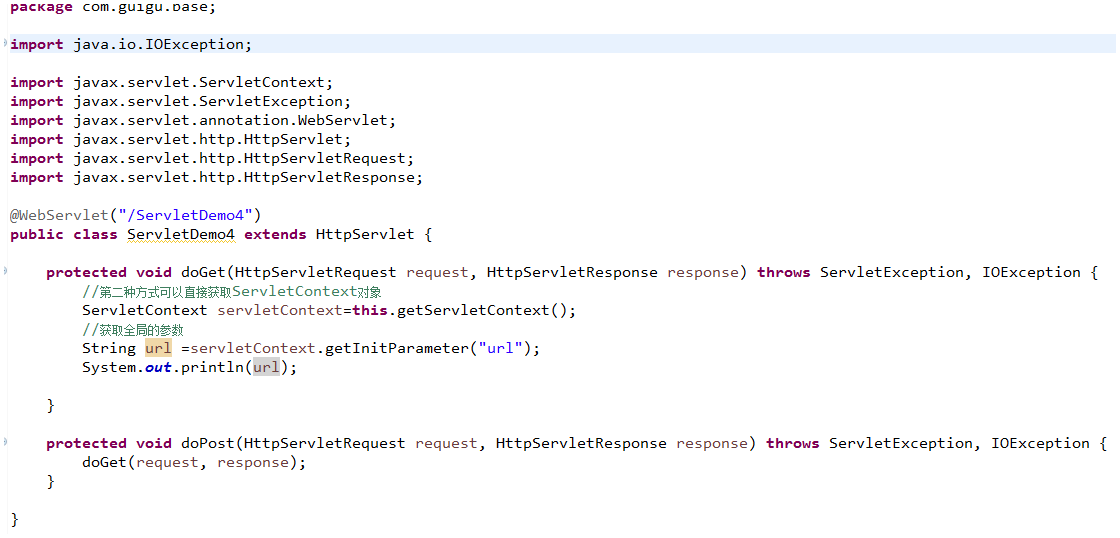
**第一个案例 多个servlet共享数据**



**第二个案例 获取全局的web初始化参数**

在web.xml中配置的信息





**第三个案例 实现servlet之间的转发**



**第四个案例 读取资源文件**

