6.

* Servlet就是sun公司开发动态web的一门技术
* Sun在这些APi中提供一个接口叫做：Servlet，如果你想开发一个Servlet程序，只需要完成两个小步骤：
  + 编写一个类，实现Servlet接口
  + 把开发好java类部署到web服务器中。

**把实现了Servlet接口的Java程序叫做，Servlet**

6.1构建结构

父类删除src, xml里添加两个dependencies



子类创建再创建模块，maven统一，src->main->一个java一个resources,

这样的好处在于，父项目的dependencies子项目可以直接使用

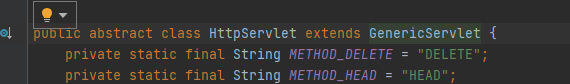
6.2

编写一个Servlet程序

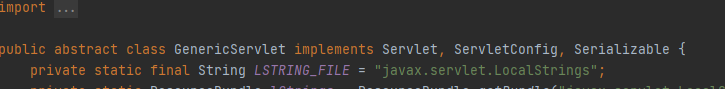
两步：

6.2.1 编写一个java类，实现Servlet接口（implements servlet interface）：

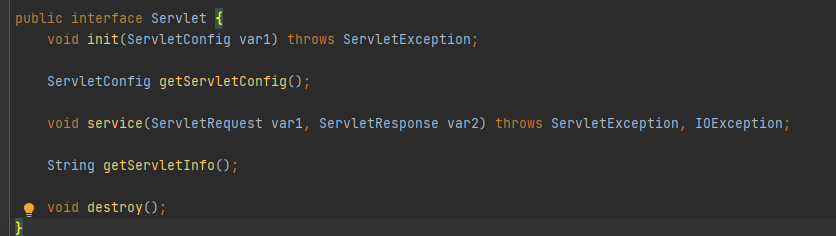
servlet的两个默认的实现类：HttpServlet，GenericServled



这个实现类HttpServlet是abstract class



它的父类GenericServlet也是abstract class,同时implements了Servlet接口

接口所有方法是不能直接描写的

别的都好理解，主要是service，输入req,response,然后我们可以得到对应的service

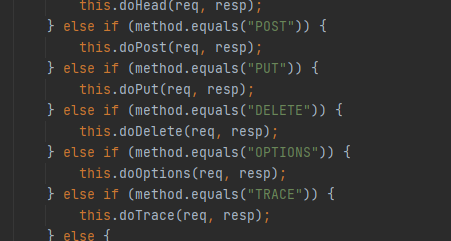
genericServlet



单纯继承接口，抽象方法，不变



HttpServlet的service有了变化，他从request里提取出method，根据method是啥，调用不同的方法，doGet,doPost,doPut…



我们要做的不是改service,而是重写对应的方法，doGet,doPost,doPut



6.5 映射

我们写的是JAVA程序，但是要通过浏览器访问·，而浏览器需要一个对应的web服务器(tomcat),所以我们需要在web服务中心**注册**我们写的Servlet程序，还需要给他映射一个路径 //请求路径/映射路径

//在webApp的webXml里注册



servlet负责连接java文件

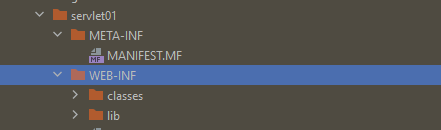
servlet-mapping负责请求一个能够访问的web路径/hello

两个name相同保证连接

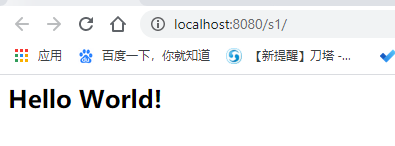
6.6 tomcat

右上角run旁边编辑tomcat

run完以后会生成一个target



web-inf里class存的是我们java文件，lib里是jar包，index.jsp是对应页面

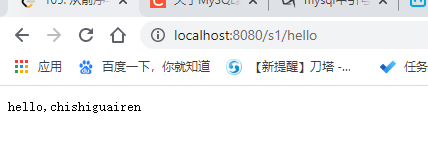


S1是我们tomcat给整个项目的地址

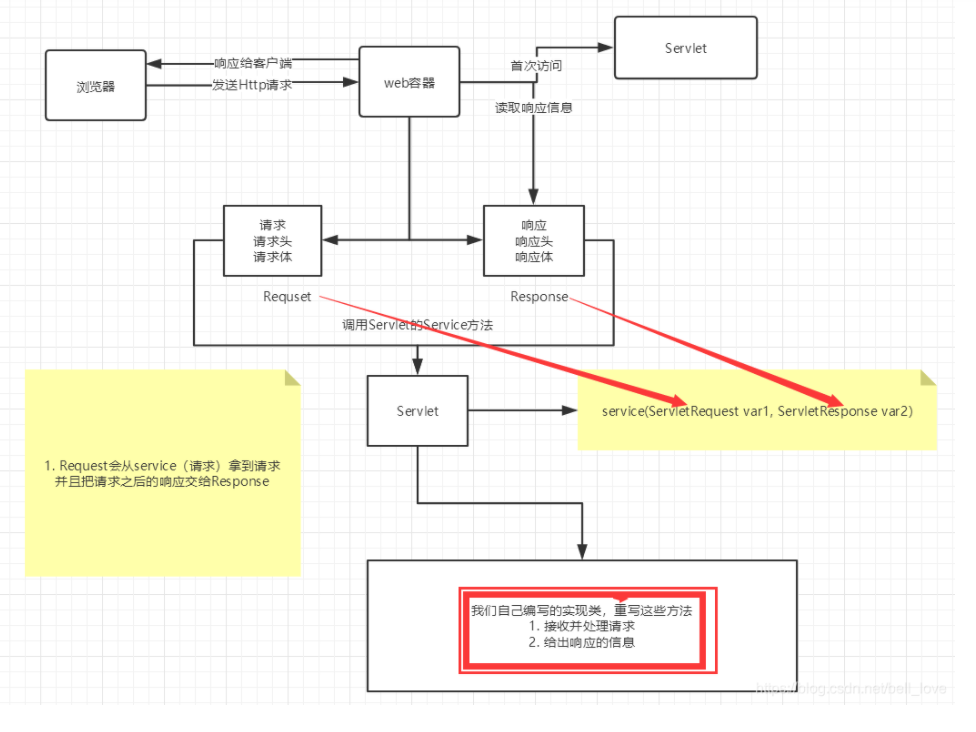
我们自己写的模块是/hello



对应java文件HelloServlet



你好！吃屎怪人！



web容器就是web服务器tomcat,

浏览器输入地址并按下enter的一瞬间会发送http请求

请求分为请求头和请求体，以及一个等待接受的response object//因为是参数传递，所以object也会被改变

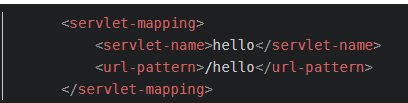
Servlet接口中的Service读取到这两个var，

我们自己编写的实体类会根据请求进行对应的操作，并且把相应交给response

web服务器再次读取我们的响应，返回

Mapping问题

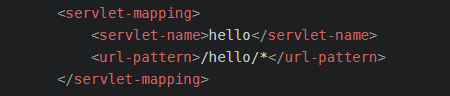
一个Servlet程序可以指定一个映射路径



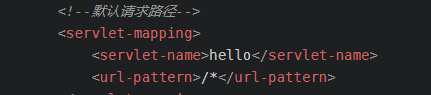
1. 一个servlet可以指定多个映射路径



**3.** 一个servlet可以指定通用映射路径

****

1. 默认请求路径

****

1. 指定一些后缀或者前缀等等…

<!--可以自定义后缀实现请求映射

注意点，\*前面不能加项目映射的路径

hello/sajdlkajda.qinjiang

-->

<servlet-mapping>

<servlet-name>hello</servlet-name>

<url-pattern>\*.qinjiang</url-pattern>

</servlet-mapping>

1. 优先级问题  
   指定了固有的映射路径优先级最高，如果找不到就会走默认的处理请求；

<!--404-->

<servlet>

<servlet-name>error</servlet-name>

<servlet-class>com.kuang.servlet.ErrorServlet</servlet-class>

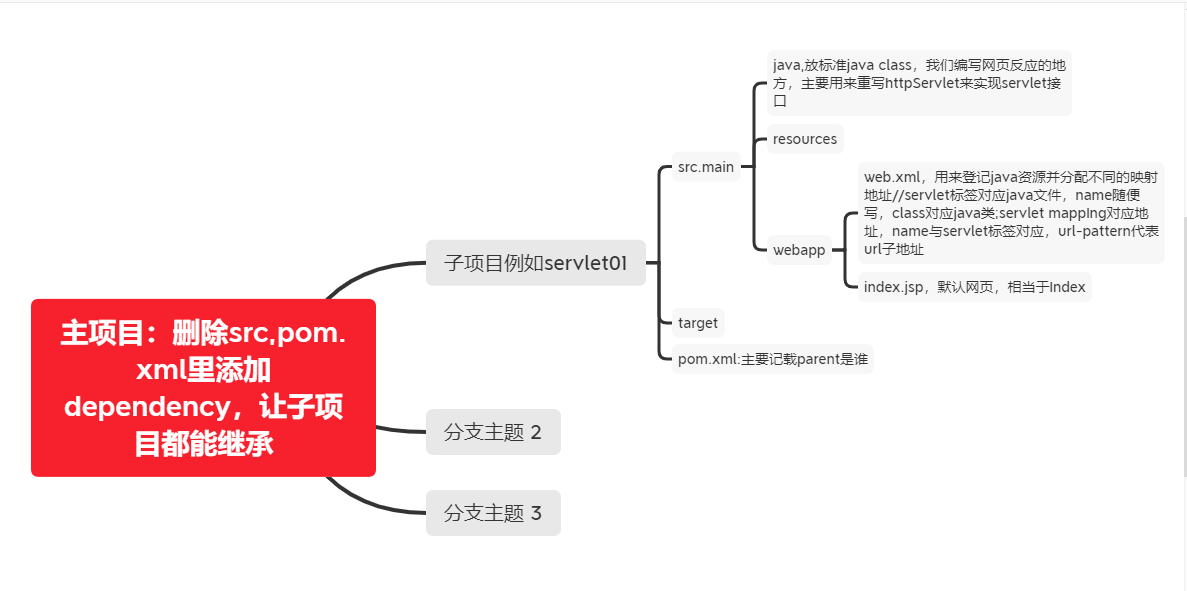
</servlet>

<servlet-mapping>

<servlet-name>error</servlet-name>

<url-pattern>/\*</url-pattern>

</servlet-mapping>

****

**Servlet Context**

1.不同Servlet间交换数据

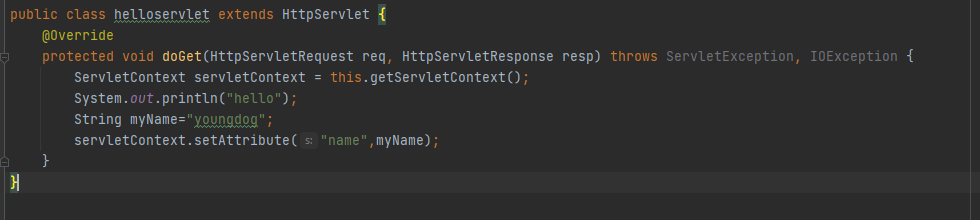
怎么样在不同的Servlet 页面间交换数据

例如我在主页输入了一个想要查询的邮件编号，跳转的副业也应该有这个编号

为了中转数据，我们使用了Servlet Context，

一个Web应用应该只有一个Servlet Context,包含多个Servlet

发送页面：



用this.getServletContext()得到公共ServletContext

使用servlectContext.setAttribute输入数据，第一个参数是这个数据的attribute Name用来定位，第二个参数是要输入的东西本身，类型是Object

接收页面：



可以通过getAttribute并且通过attribute Name直接得到Object,然后转换成我们要用的

然后用response输出，也就是resp的Writer

如果是中文可能需要设置response Header的参数不然会乱码

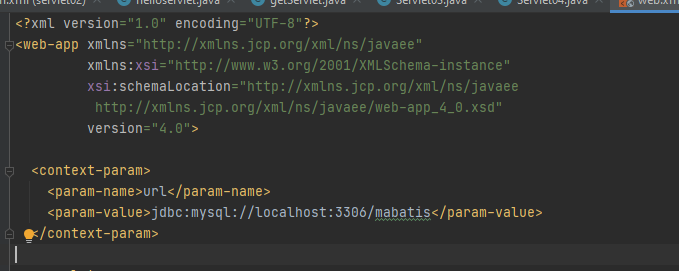
.



注意要先进入发送页面，再进入接收页面，

如果不进入发送页面，那么发送页面是不会默认运转的

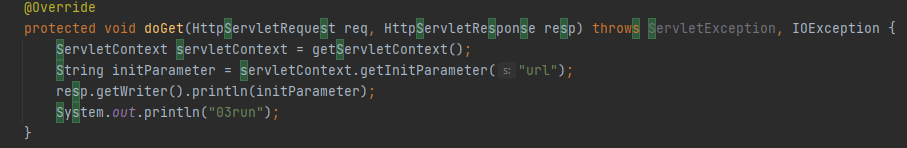
2.获取参数



context-param是context自己的参数parameter,记录在web.xml里

servletContext.setAttribute()/getAttribute()是 其他Servlet希望contest传递的参数attribute,

/03



通过getInitParameter得到了一开始手写的context-param

3.requestDispatcher

b是04，C是03，这里04使用forward，把他的req和resp转发给了C，然后C会通过得到的req update resp。

/04



可以看到requestDispatcher是servletContext的一个组件，所以requestDispatcher应该也只有一个，他的参数是要转向的目的地

  
可以看见地址还是04，因为浏览器并不和03直接接触，但是内容却是03

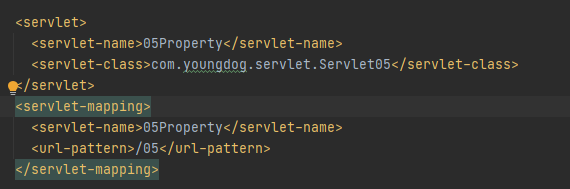


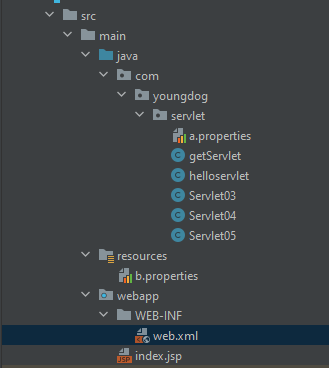
03 先run ,04后run

4.

得到参数

在mapping虚拟地址与对应java文件时，默认的根目录是java下面的文件

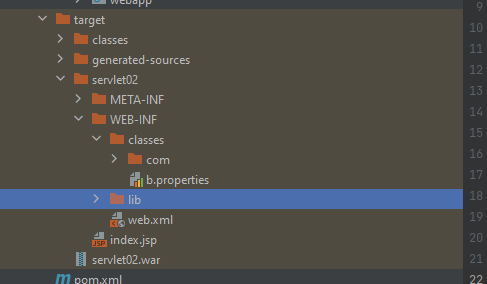




而在取Properties时



要用一个inputStream得到他，换句话说，**得到prperties需要一个文件流**，地址的根目录是基于生成的target/对应的项目名字



Http Response

Response下载

下载文件

1. **要获取下载文件的路径 //用来设置4的输入流**
2. 下载的文件名是啥？ //3的固定格式需要文件名而不是整个路径
3. 设置想办法让浏览器能够支持下载我们需要的东西**//设置固定代码告诉我们要下载**
4. 获取下载文件的输入流 //FileInputStream,永远是这个
5. **创建缓冲区**
6. 获取OutputStream对象
7. 将FileOutputStream流写入到buffer缓冲区 //把文件写入buffer
8. 使用OutputStream将缓冲区中的数据输出到客户端！

可以看见filePath是给FileInputStream使用的

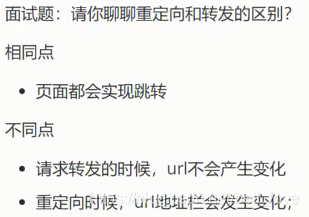
fileName是给Header使用的，header需要告诉他下载

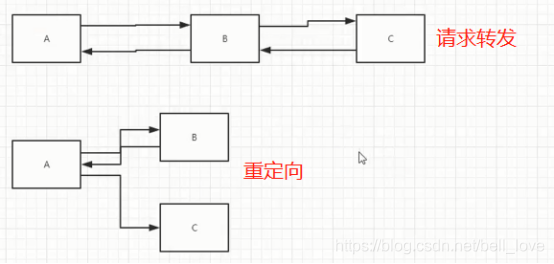
buffer就是一个byte字符串

fileInputStream.read(buffer)是读取并存储到buffer里

重定向

重定向和转发的区别





A找B要，B告诉A，找C要，A再找C要

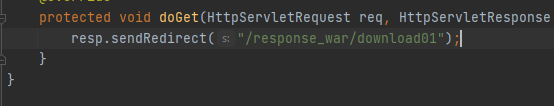
重定向参数问题



重定向默认的起始位置是localhost:8080/



然而实际上我们是在localhost:8080/response\_war这个根目录下的



会转到localhost:8080/response\_war/download01,才能下载

如果只有download01找不到

重定向的两次请求：

Request

HttpServletRequest代表客户端的请求,用户通过Http协议访问服务器, HTTP请求中的所有信息会被封装到HttpServletRequest,通过这个HttpServletRequest的方法,**获得客户端的所有信息;**



req包含了客户端所有的信息，所以主要用get

resp是返回给客户端的信息，所以主要方法是set

public class RequestTest extends HttpServlet {

@Override

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

//处理方求

String username = req.getParameter( s: "username");

String password rea.getParameter( s: "password");

System.out.println(username+":"+password);

resp.sendRedirect(s: "/r/success.jsp");

直接用getParameter就行，参数等于 name

