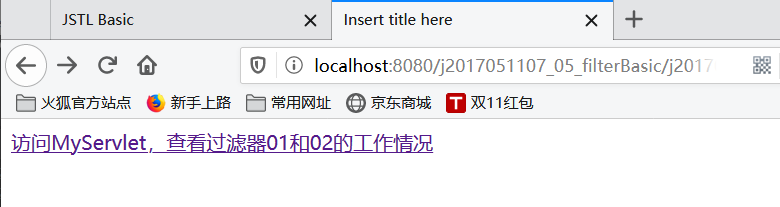
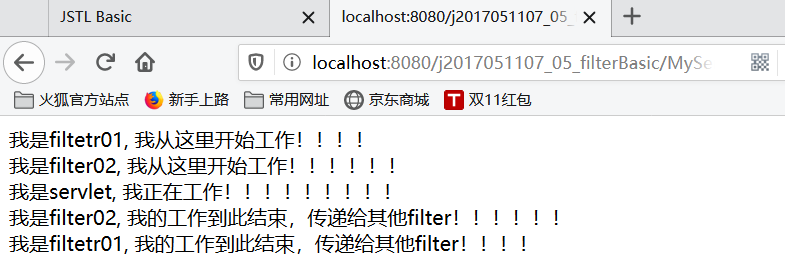
JAVA EE WEEK02

1. 项目描述
2. j2017051107\_05\_filterBasic：该项目主要实验过滤器的工作过程
3. 项目导航界面



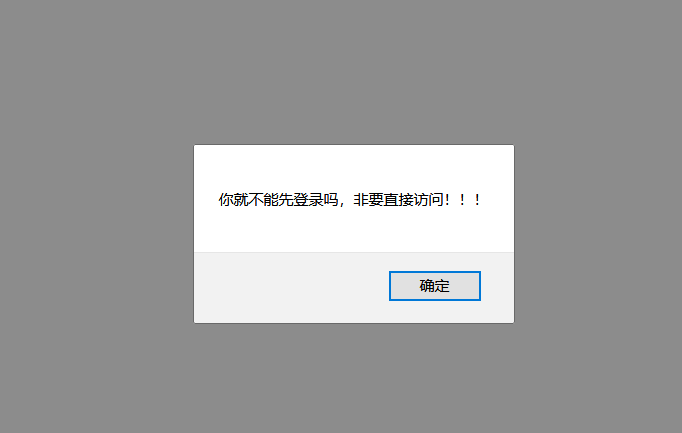
1. 项目演示

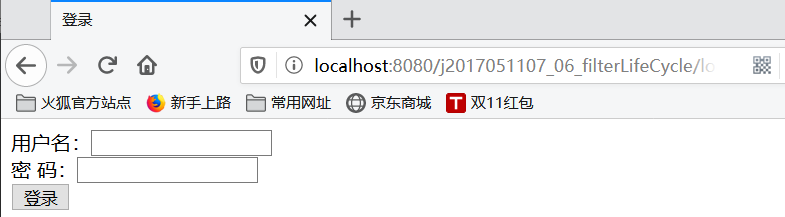


1. j2017051107\_06\_filterLifeCycle：项目主要将过滤器和servlet相配合，实现用户登录的功能、在线人数统计的功能、重复登录检测的功能以及过滤器工作顺序的实验
   1. 项目导航界面



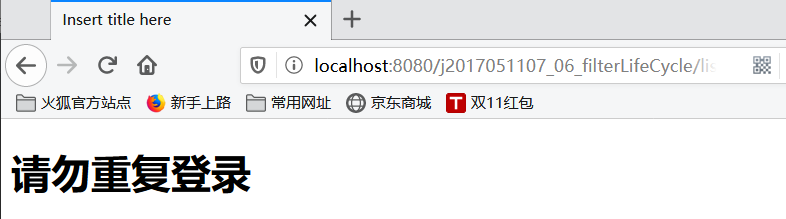
* 1. 第一个链接：



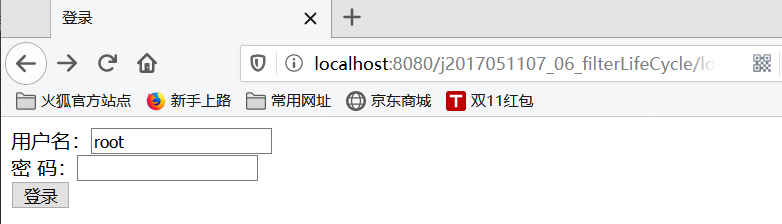


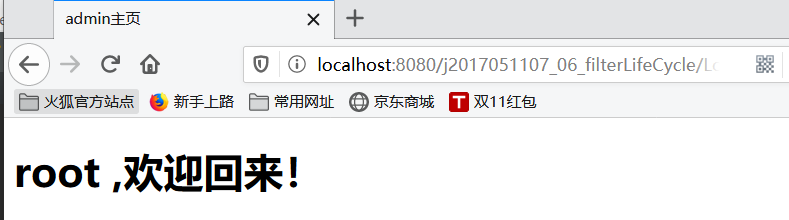
* 1. 第二个链接



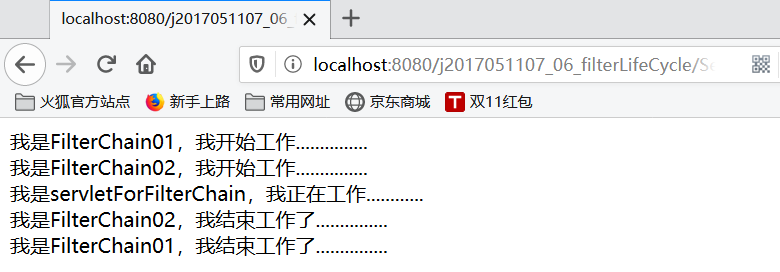


* 1. 第三个链接

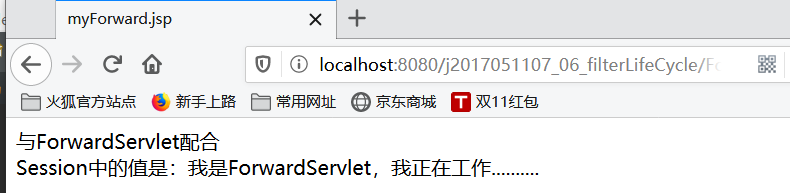




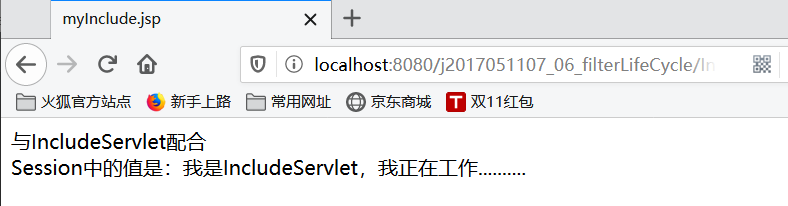
* 1. 第四个链接



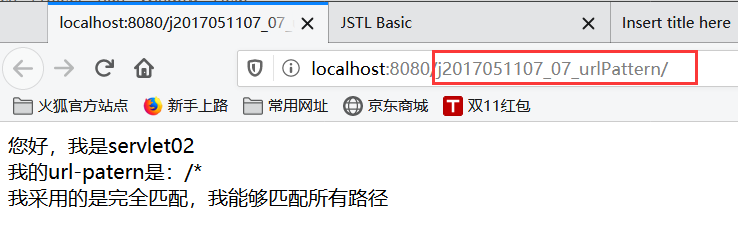
* 1. 第五个链接

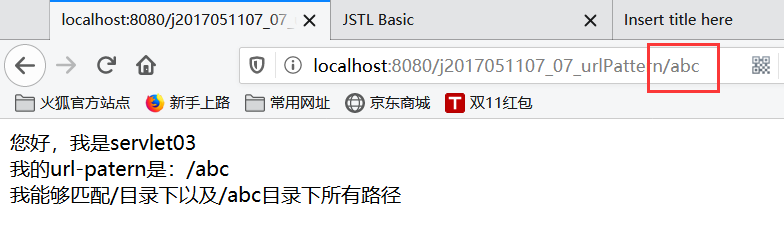


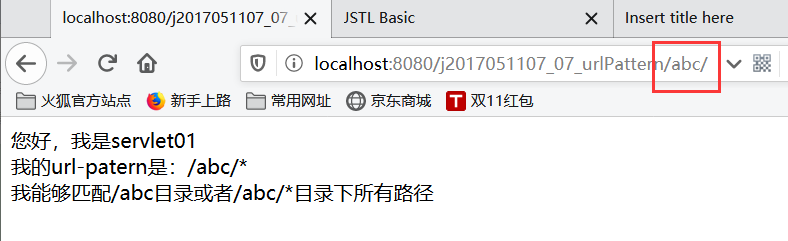
* 1. 第六个链接

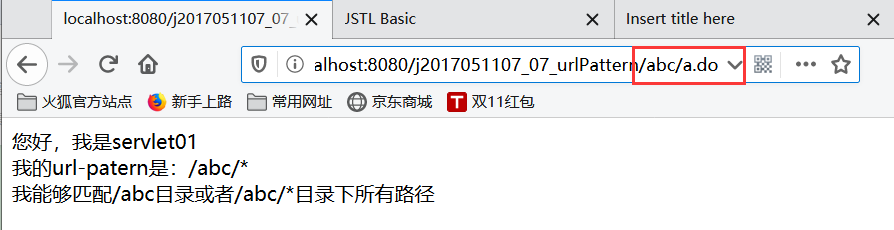


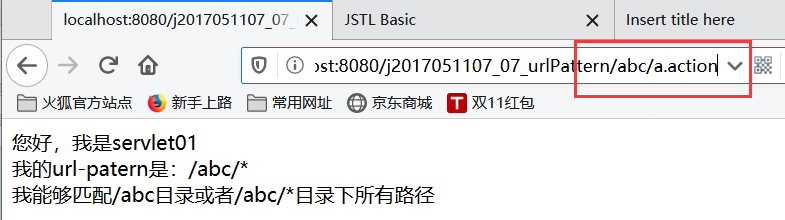
1. j2017051107\_07\_urlPattern：该项目主要实验URL的多重映射和统配符映射。用户通过访问不同的URL，来映射到不同的servlet，演示如下：





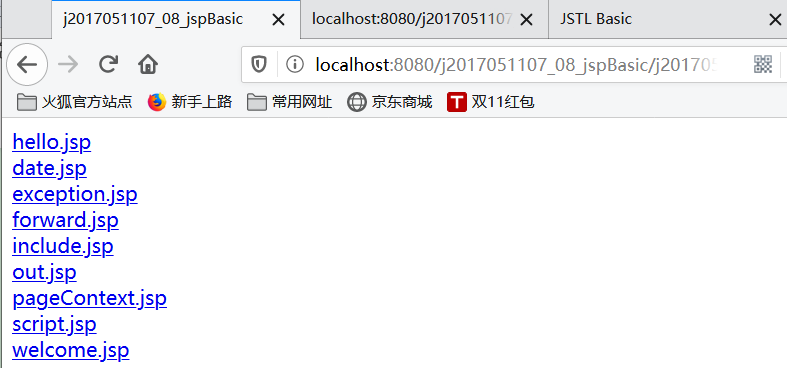




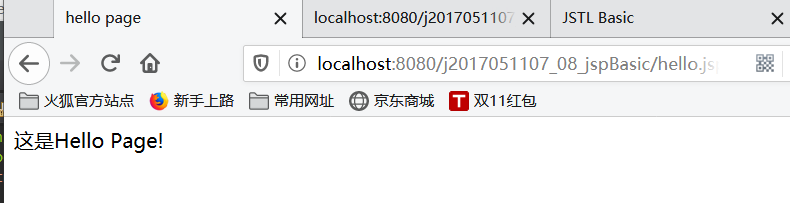


1. j2017051107\_08\_jspBasic：该项目主要进行JSP基础的实验，包括简单的JSP页面的创建、在JSP页面中嵌入JAVA代码块进行日期显示、JSP的page指令、forward、include动作指令的实验

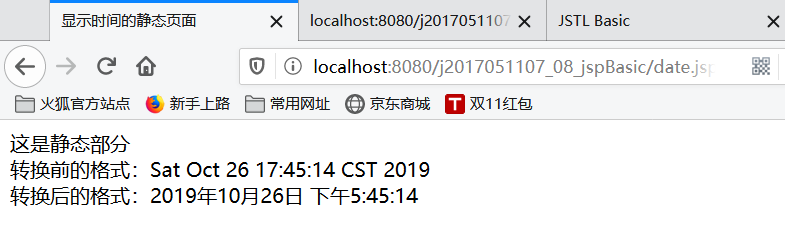
项目导航：



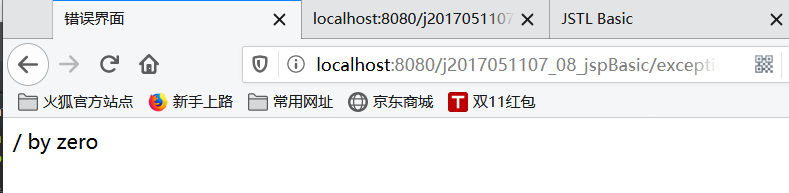
1. Hello.jsp：



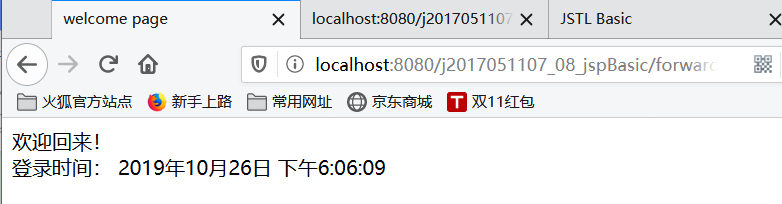
1. Date.jsp：



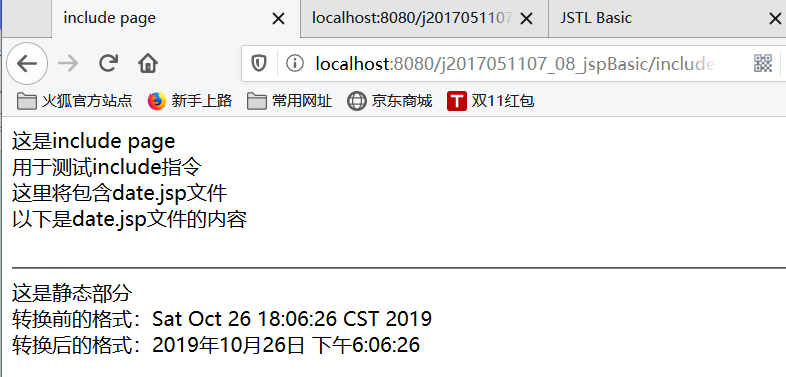
1. exception.jsp：



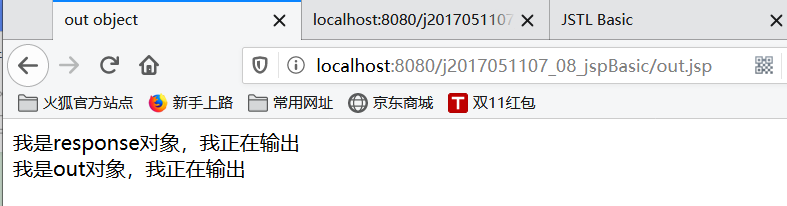
1. forward.jsp



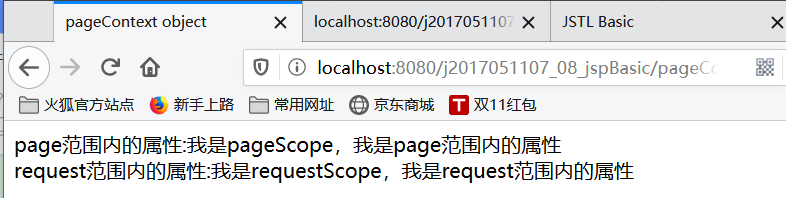
1. include.jsp



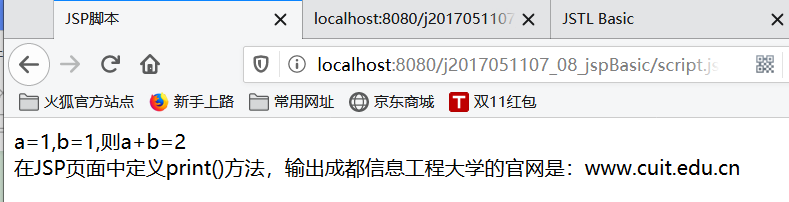
1. out.jsp



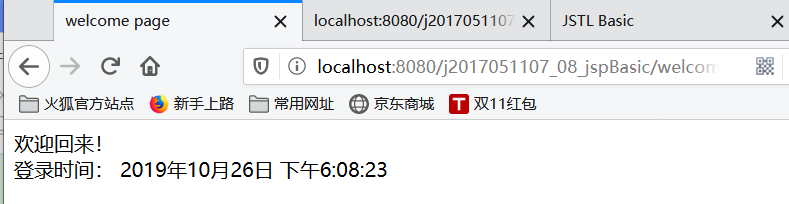
1. pageContext.jsp



1. script.jsp



1. welcome.jsp



错误1：filter映射错误

错误原因：在web.xml中配置filter时映射配置错误，将url-pattern配置成了filter本身。

解决方案：将url-pattern设置为需要过滤的文件，在此处设置为MyServlet

错误2：服务器启动失败

错误原因：就是由于错误1导致的

解决方案：解决掉错误1后，该问题自动解决

错误3：在AutoLoginFilter中拆分autologin时，出现数组下标越界错误

错误原因：在设置自动登录的cookie时，autologin的值任然是cookie有效期

解决方案：在LoginServlet中创建自动登录的cookie时，autologin的值设置为“用户名-密码”的组合

错误4：url-patter冲突

错误原因：同时采用了注解方式和web.xml方式配置servlet，并且两个的url-patter一样

解决方案：删除掉注解方式，只留下web.xml方式

错误5：

错误原因：在index.jsp获取从LoginServlet中传过来的session对象中的数据时，直接用request.getAttribute("user")，忘了获取session

解决方案：先获取session在取值

错误5：服务器内部错误

错误原因：没有在error.jsp和exception.jsp文件设置isErrorPage和errorPage属性

解决方案：分别在error.jsp和exception.jsp文件设置isErrorPage和errorPage属性

收获：

通过错误1明白，在web.xml中配置filter时，url-pattern中设置的是需要被过滤的资源。在没明白这个之前，我还在纠结filter是怎么实现过滤的，它怎么知道要去过滤哪个文件。现在通过这个错误，也明白了这个问题。

filter过滤顺序：过滤的执行顺序为在web.xml中配置时各个过滤器配置的先后顺序。比如说，我在配置时，先配置MyFilter01，再配置MyFilter02，那么，在执行时，就是先执行MyFilter01，再执行MyFilter02。在退出时，则是后执行的先退出，整个执行过程就像是栈一样，遵循“后进先出”原则。

cookie的存储分为临时存储与定时存储。

临时存储：在创建cookie时不指定有效期，即不设置setMaxAge(int ms)。在我们访问带有cookie的servlet时，cookie被创建，但是他是存储在浏览器的运行内存中的，所以，当我们关闭浏览器后，创建的cookie就被释放掉了，再次访问时就没有了。

cookie定时存储：如果我们在创建cookie时，设置了有效期，那么这个cookie将会被保存在客户端的硬盘中。这样，只要在cookie的有效期以内，即使我们关闭了浏览器，这个cookie任然存在，当我们下次再次访问对应网站的时候，在请求中同样会带着这个cookie。

Cookie有效路径：通过调用方法void setPath(String path）：设置cookie的有效访问路径。浏览器在有效路径下访问服务器时就会带着cookie信息，否则不带cookie信息。

servlet映射路径中使用通配符，有两种映射方法：

方法一：做路径映射，即以”/’开头和以”/\*”结尾

方法二：做拓展映射，即以前缀”\*.”开头

通配符匹配原则：当客户端访问一个Servlet时，如果请求的URL地址，能够匹配多个虚拟路径 ，Tomcat将采取**最具体匹配原则**，来查看与请求URL最接近的虚拟路径，即匹配的优先级是：**匹配的范围越大优先级越低**

带后缀的没懂，PPT这样说：

如果url最后一段包含扩展，容器将会根据扩展选择合适的servlet

还有那个默认的servlet也没懂

多重映射（有两种映射方法）：

方法一：配置多个<servlet-mapping>元素

方法二：在一个<servlet-mapping>元素下配置多个<url-pattern>子元素

JSP相关：

1. 声明变量和方法：

①声明部位：在”<%!”与”%>”标记符号之间

②作用域：声明该变量和方法的JSP页面

1. Java程序块：

①编写部位：在”<%”与”%>”标记符号之间

②注意事项：每条语句必须以”;”结尾

1. JSP表达式：

①编写部位：在”<%=”与”%>”标记符号之间

②作用：

Ⅰ计算变量或表达式的值

Ⅱ将值转换成字符串

Ⅲ用out.println发送标签，按照先后顺序依次将结果输出到页面上JSP表达式所在 的位置

③注意：

Ⅰ在”<%=”与”%>”标记符号之间不可插入语句，即不能以”;”结尾

Ⅱ”<%”与“=”之间不要有空格

Ⅲ变量或表达式中包含的变量必须是已声明的

1. JSP三大指令

①page指令：

在page指令中针对本次实验着重说明三个属性：

Ⅰ.import属性：用于导入java软件包或者类名列表

Ⅱ.isErrorPage属性：指定当前页面是否可以作为错误处理页面

Ⅲ.errorPage属性：指定当JSP页面发生异常时，需要转向的错误处理页面

②include指令

Ⅰ.作用：包含其他文件——JSP文件、HTML文件、文本文件

Ⅱ.语法：<%@ include file=”path” %>

③taglib指令

1. JSP九个内置对象

①request对象

②response对象

③out对象

④session对象

⑤pageContext对象：

使用该对象可以访问页面中的共享数据，即通过他可以获取JSP页面的out、request、response、session、application等对象。

⑥application对象

⑦config对象

⑧page对象

⑨exception对象

为什么response对象比out对象先输出