java.net

Java程序设计

第02-03讲 过滤器（Filter）

Java课程组

本讲教学目标

- 掌握如何配置过滤器
- 掌握如何使用过滤器

知识回顾/本讲先行知识

- 监听器（Listener）
  - ServletContext相关监听器
  - HttpSession相关监听器
  - ServletRequest相关监听器

本讲内容

- 过滤器简介
- 过滤器API
- 配置过滤器

引例图

来两张动物园门票！我要看野兽

对不起，这是张总Office

张总，面试的到了

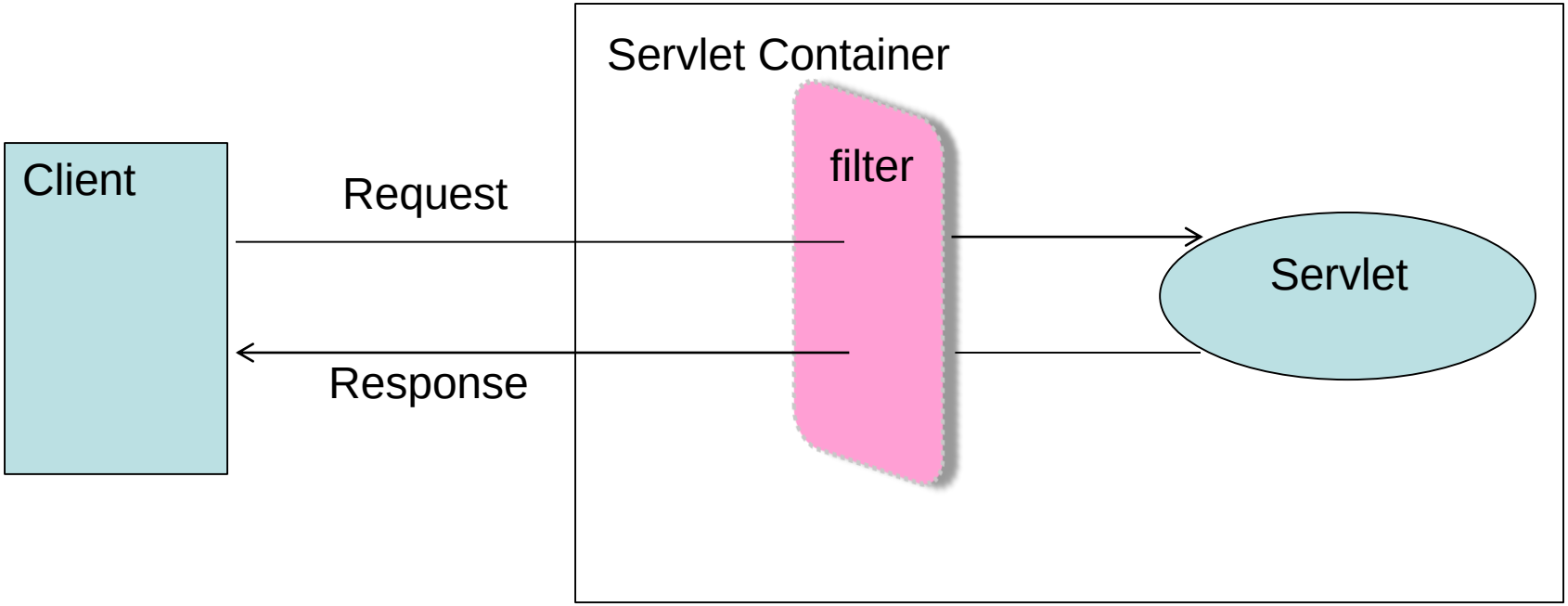
好的，安排在会议室吧

我小丸子，来你们公司面试的

你先喝杯水，张总马上过来

过滤器介绍

- 过滤器是一个用于拦截在数据源和数据目的地之间消息的一个对象



过滤器功能

- 分析请求，将请求发送给指定的资源或自己创建一个响应返回
- 在请求到达服务器端前处理请求，设置请求信息，将请求封装成符合规则的对象
- 在响应到达客户端前处理响应，将响应封装成符合规则的对象

不同功能的过滤器

- 身份验证过滤器（Authentication Filters）
- 数据压缩过滤器（Data compression Filters）
- 加密过滤器（Encryption Filters）
- 触发资源访问事件过滤器(Filters that trigger resource access events)
- 图像转换过滤器（Image Conversion Filters）
- 日志记录和审核过滤器（Logging and Auditing Filters）
- MIME-TYPE 链过滤器（MIME-TYPE Chain Filters）
- 标记化过滤器（Tokenizing Filters）
- XSL/T 过滤器（XSL/T Filters），转换 XML 内容。

过滤器的执行

- 过滤器处理请求采取以下3种行动之一
  - 过滤器自身产生一个响应，返回给客户端
  - 把请求转发给过滤器链中的下一个过滤器，如果是最后一个过滤器则将请求转发给请求指定资源
  - 把请求转发到另一资源

过滤器示例

- AuthorityFilter
  - 编码（实现Filter接口）
  - 部署（在web.xml配置）
    - 注册过滤器
    - 配置虚拟路径
  - 运行

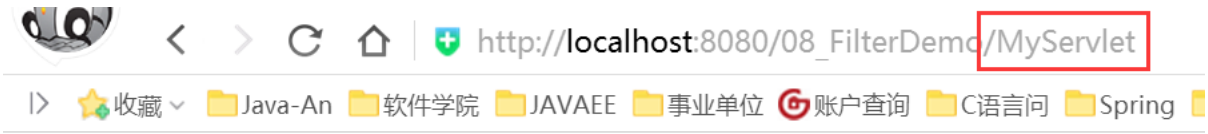
过滤器示例

```
<filter>
  <filter-name>CharseFilter</filter-name>
  <filter-class>net.onest.CharsetFilter</filter-class>
</filter>
```

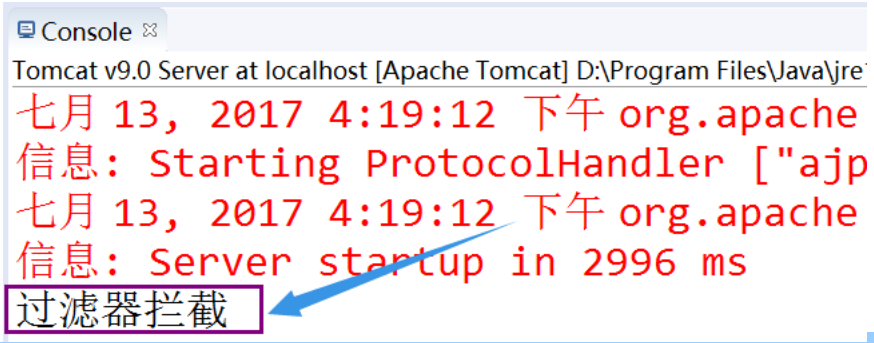
注册过滤器

```
<filter-mapping>
  <filter-name>CharseFilter</filter-name>
  <url-pattern>/MyServlet</url-pattern>
</filter-mapping>
```

配置拦截虚拟路径



Served at: /08_FilterDemo



本讲内容

- 过滤器简介
- 过滤器API
- 配置过滤器

Filter接口

- init()方法
  - Servlet容器最先调用过滤器的init()方法，初始化过滤器，生命周期中只调用一次
- doFilter()方法
  - Servlet容器调用doFilter()方法处理请求
- destroy()方法
  - Servlet容器最后调用destroy()方法，释放过滤器所占的资源

过滤器API

- 过滤器相关接口、类

接口/类	所处包	描述
Filter	javax.servlet	过滤器类实现此接口
FilterChain	javax.servlet	该接口实现对象实例由容器负责创建
FilterConfig	javax.servlet	提供初始化参数
ServletRequestWrapper	javax.servlet	ServletRequest接口的便利实现
ServletResponseWrapper	javax.servlet	ServletResponse接口的便利实现
HttpServletRequestWrapper	javax.servlet.http	HttpServletRequest接口的便利实现
HttpServletResponseWrapper	javax.servlet.http	HttpServletResponse接口的便利实现

本讲内容

- 过滤器简介
- 过滤器API
- 配置过滤器

配置Filter

配置过滤器

<filter>

- <filter-name>
- <filter-class>
- <description>
- <init-param>
  - <param-name>
  - <param-value>

<filter-mapping>

- <filter-name>
- <url-pattern>

过滤器示例


```
<filter>
  <filter-name>CharseFilter</filter-name>
  <filter-class>net.onest.CharsetFilter</filter-class>
  <init-param>
      <param-name>encodeType</param-name>
      <param-value>GB2312</param-value>
  </init-param>
</filter>
<filter-mapping>
  <filter-name>CharseFilter</filter-name>
  <url-pattern>/MyServlet</url-pattern>
</filter-mapping>
```

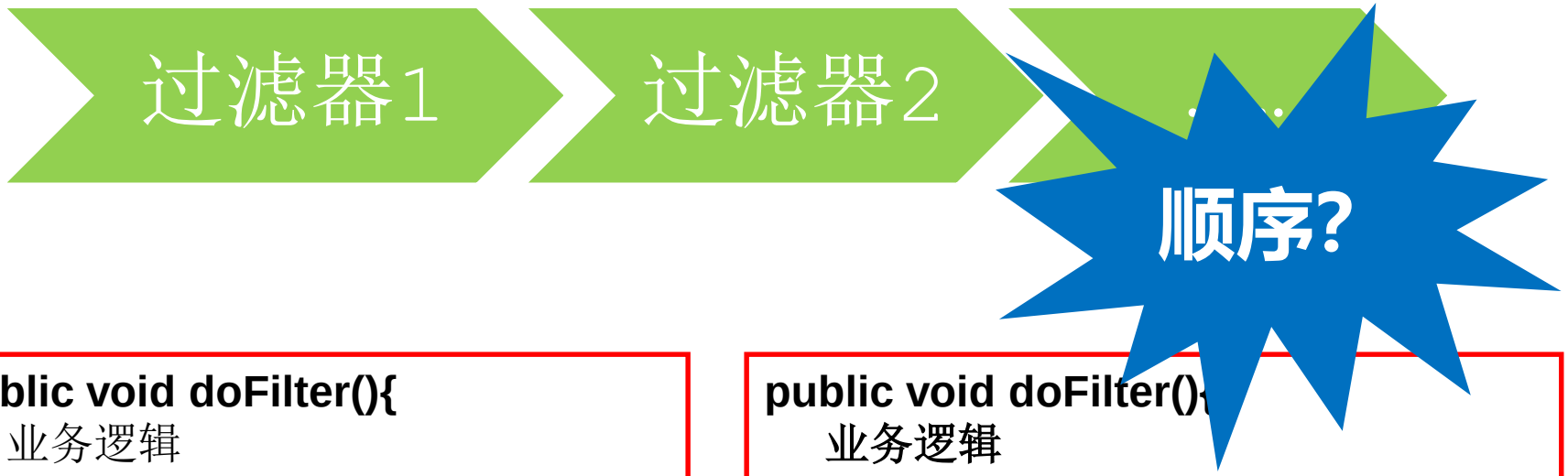
初始化参数

别名

拦截虚拟路径

Filter过滤器链

- 多个过滤器共同过滤相同的资源




```
public void doFilter(){
    业务逻辑
    chain.doFilter(request, response);

}
```

```
public void doFilter(){
    业务逻辑
    chain.doFilter(request, response);

}
```

Filter过滤器链

- 示例

```
<filter>
  <filter-name>FilterTest</filter-name>
  <filter-class>net.onest.FilterTest</filter-class>
</filter>
<filter-mapping>
  <filter-name>FilterTest</filter-name>
  <url-pattern>/MyServlet</url-pattern>
</filter-mapping>
<filter>
  <filter-name>CharsetFilter</filter-name>
  <filter-class>net.onest.CharsetFilter</filter-class>
</filter>
<filter-mapping>
  <filter-name>CharsetFilter</filter-name>
  <url-pattern>/MyServlet</url-pattern>
</filter-mapping>
```

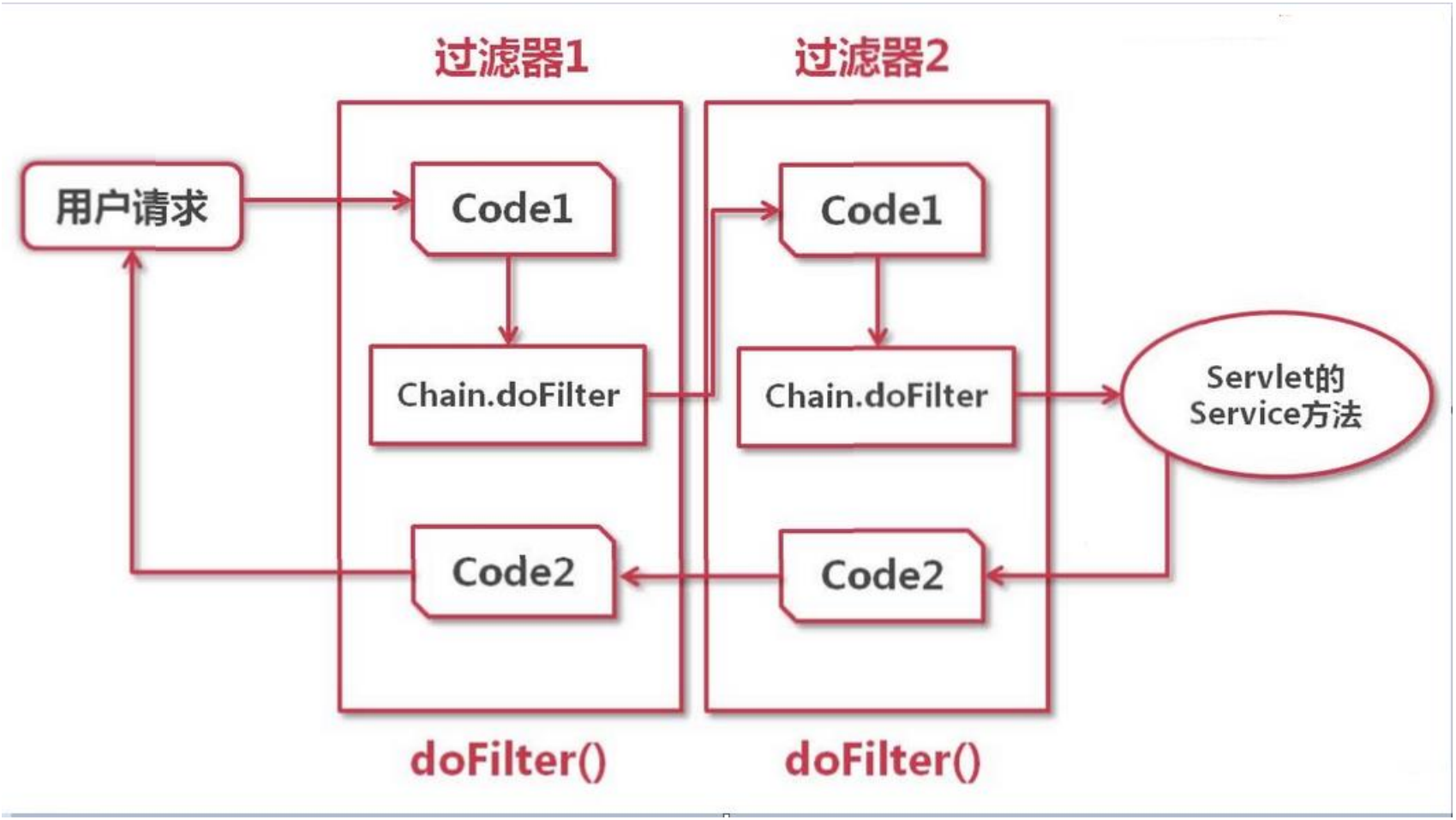
过滤器1

过滤器2

Filter过滤器链

- Web容器加载的顺序：web.xml文件中配置的Filter的倒序
- 拦截执行顺序：web.xml文件中配置的Filter的正序
- 销毁顺序：web.xml文件中配置的Filter的倒序

Filter过滤器链


过滤器1
过滤器2
用户请求
Code1
Code1
Chain.doFilter
Chain.doFilter
Servlet的
Service方法
Code2
Code2
doFilter()
doFilter()

Filter注意事项

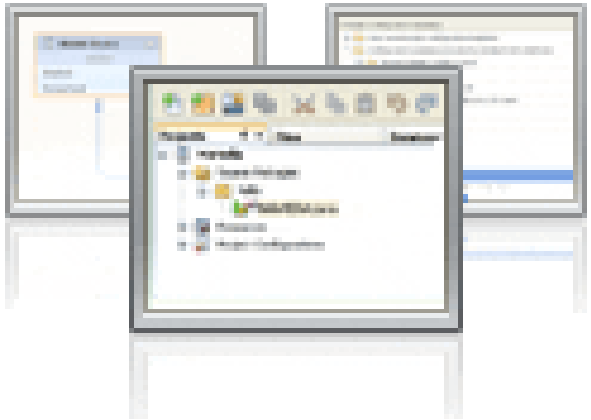
- Filter配置过滤器链
- Servlet容器可以对同一个过滤器对象运行多个线程来同时处理多个请求

小结

- 过滤器简介
- 过滤器API
- 配置过滤器

实验

- 实验



本讲结束

- 谢谢大家