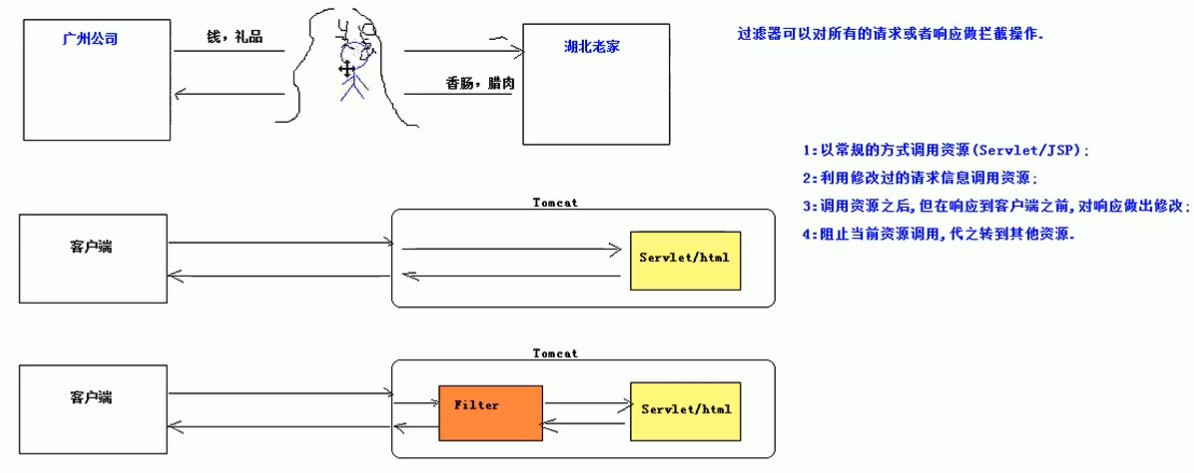
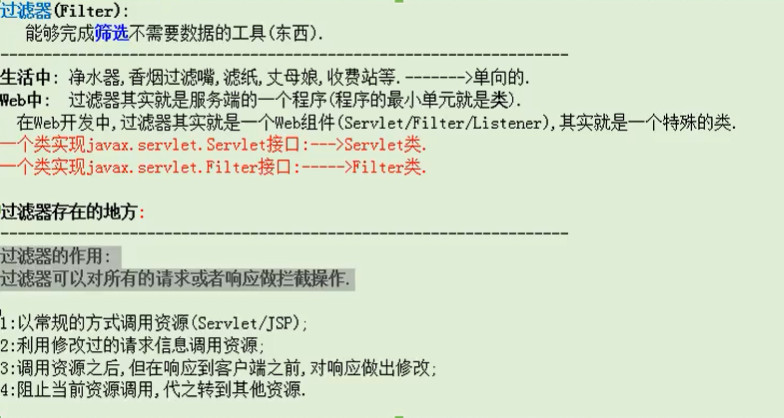
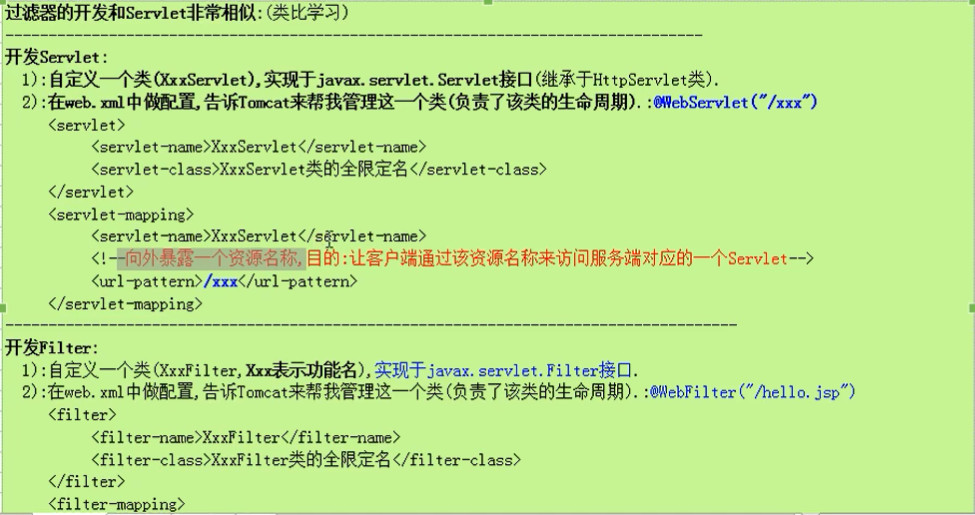
函数式接口：[函数式接口 - 希希里之海 - 博客园 (cnblogs.com)](https://www.cnblogs.com/weixuqin/p/11494894.html)







Servlet生命周期：

[Servlet 生命周期 | 菜鸟教程 (runoob.com)](https://www.runoob.com/servlet/servlet-life-cycle.html)

Servlet究竟式单例还是多例—大多数情况下是单例这也就解释的通为什么i能用来统计访问数目了（[Servlet到底是单例还是多例你了解吗？ - 沧海一滴 - 博客园 (cnblogs.com)](https://www.cnblogs.com/softidea/p/7245899.html)）

Filter也是如此：再说一遍，对象实例主要用于存储数据（类就是声明的数据结构），如果数据有大量的不同那么自然可以用多例。但是如果更多的是用来执行方法，方法都是执行类的模板，所以对象的实例化多少关系不大。（开发者将Java中的类划分为各种各样的类是有道理的，比如简单实体类，这样就能将数据与行为进行分割，而行为更多通过接口与抽象类进行约定限制，这样就大大的增加了编程与维护的灵活性。）

（tomcat是一个web容器、servlet容器，用于管理各种的servlet，如果是饿汉式加载那么servlet在容器启动时就进行创建，反之懒汉式就是需要时创建。Tomcat是一个web服务器，而本质上他就是一个带上main函数的Java程序，再到Java基础上就是Java网络编程，对应到操作系统上就是一个JVM进程，而我们写的各式各样的Java对象，文件资源，都相当于是在扩展这个tomcat程序的功能。这么一看tomcat的抽象级别还是比较高。能够让程序员灵活的扩展开发。）

（当一个远程服务请求发送时，tomcat会进行拦截并根据配置信息进行映射servlet然后交给该类进行执行，这也是mvc模型建立的好处。从这里也可以看出其实tomcat的主要事务是在进行不断的接收请求、映射处理器、加载处理器，返回结果。Tomcat本身就像一个过滤器。因为更多的是在调用类的方法，所以大多数是单例完全没问题。）

（

官方文档上说：Servlet引擎必须等待所有的service()方法完成或超时结束之后才会销毁这个servlet）

（（1）首先，能够获得大家一致认可的是，不论一个servlet保存多长时间，它的生存周期肯定会短于容器的生存期，servlet实例在容器被移除之前被销毁，即所谓的“皮之不存，毛何附焉”；

（2）servlet生存期的定义，包括加载、实例化、初始化、处理客户端请求以及如何被移除。这个生存期由javax.servlet.Servlet接口的init,service和destroy方法表达。

（3）重点是：servlet采用的是单实例，多线程模式，即在同一时刻，容器中只存在某servlet的一个实例；同时，多请求（用户）会获得多个线程来运行同一个实例；

（4）在容器生命周期内，不同时间段会生存servlet的不同实例，可以理解为：A servlet在“时间段一”存在实例A.1，在“时间段二”会存在实例A.2;容器可以根据需要对长时间没有被请求的servlet实例销毁，在必要的时候再生成。

（5）对于容器销毁servlet的时机，则需要研究servlet的内部运行机制，请高人赐教。可以肯定的是调用destroy()方法来销毁servlet实例。）

（

servlet的几个生命周期方法，都是回调方法，由容器托管，应用程序自身无法调用。就像线程的run方法一样。所以你调用了是看不出任何效果的。

至于其生命周期，确实，一般来说service方法一结束基本可以说该servlet生命周期即结束了。jsp本质就是servlet,每个请求页面结束，servlet就结束了。占有的资源有gc来处理。

Servlet生命周期结束：1）容器重启关闭，2）servlet的service方法结束返回

目前没听说过可以控制Servlet的生命周期长短。有人说在web.xml的默认30分钟的控制，那个好像是HttpSession的默认超时时间吧

）

Filter的生命周期：（[(2条消息) JAVA中级(十三)fliter过滤器基本使用,filter生命周期.\_小冷猫的博客-CSDN博客](https://blog.csdn.net/qq_42577241/article/details/89558587)）

（[servlet生命周期一共分几步？ (baidu.com)](http://baijiahao.baidu.com/s?id=1684616619505078970)）

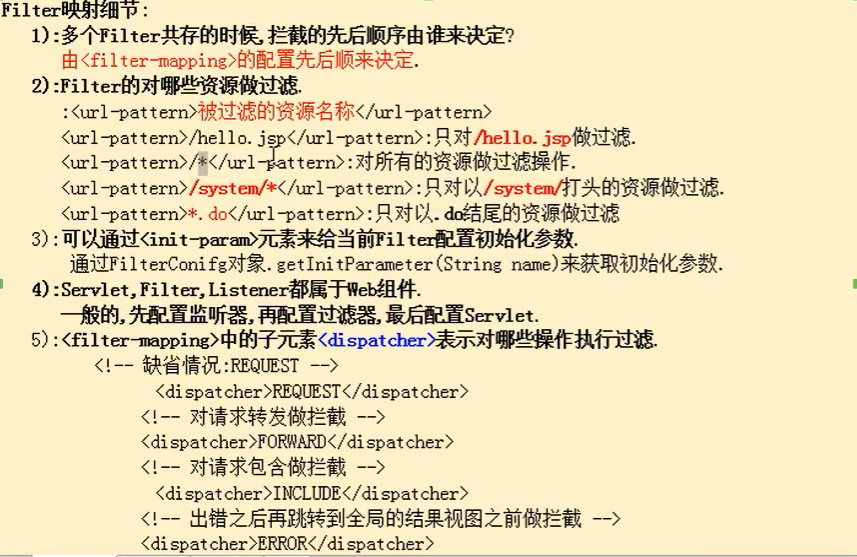


过滤器执行流程如下：（所有过滤按照配置文件中的顺序形成一个双向环链。）

1、请求来进入第一个过滤器，执行dofilter中chain.dofliter()之前的代码块，执行完这个过滤器的操作后通过调用chain的操作又将执行转到下一个过滤器的dofliter方法。

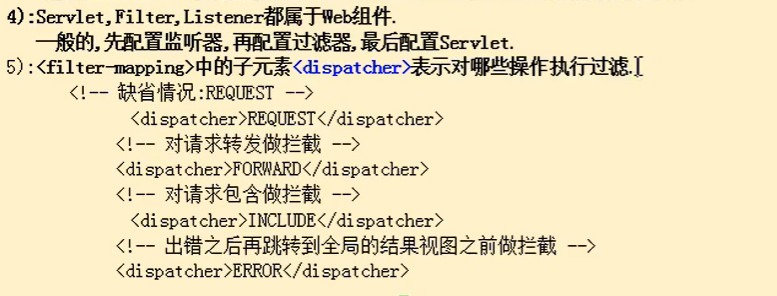
2、当chain到末尾后，会逆向返回去执行每个fliter的chain.dofliter之后的代码块。

3、最后相应请求。



Tomcat热部署：

[Tomcat热部署 - Amber- - 博客园 (cnblogs.com)](https://www.cnblogs.com/almm/p/11202468.html)



请求转发、请求重定向、请求包含：（此篇足以）

配置过滤器的较好的方式：（过滤器的配置就像一条主流与各式各样的支流。）

1、将过滤的请求进行分类。

2、每一类进行分组进行所有的过滤器进行配置。

3、所有请求共享的过滤器应该在过滤器最先处配置。

[(2条消息) 请求重定向、请求转发、请求包含的特点与区别详解\_丹丹的后花园-CSDN博客](https://blog.csdn.net/qq_34666857/article/details/104863508)

[**web.xml中的filter标签\_136.la**](https://www.136.la/nginx/show-12086.html)

REQUEST：非内部访问资源形式都要拦截。

INCLUDE：包含请求方式访问资源的形式进行拦截。

FORWARD：请求转发形式访问资源被拦截。

ERROR：通过异常处理机制调用资源的方式进行拦截。