



Filtro HTTP para forçar o download de arquivos

Filtros HTTP.....	2
Forçando o Download de Arquivos.....	2
Criando o DownloadFilter.....	2



Filtros HTTP

Desde a versão J2EE 1.3, as aplicações Web em Java tem a possibilidade de implementar a filtragem das requisições de rede ao serviço HTTP do WebContainer presente nos servidores de aplicação Java Enterprise.

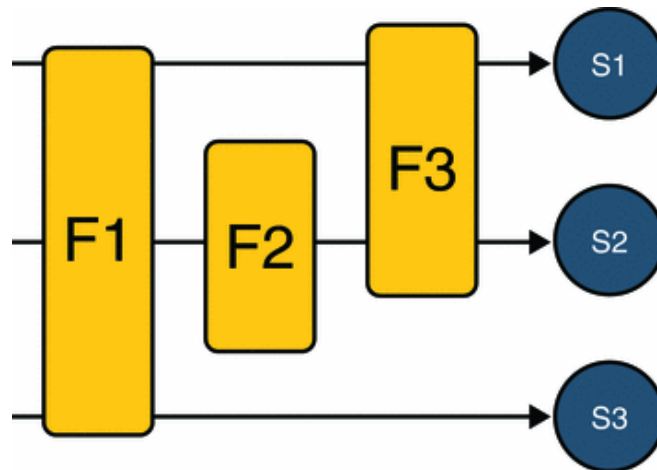


Figura 1. (retirada do Java EE 5 Tutorial)

O uso destes filtros (F1, F2 e F3) permite realizar ações com a requisição antes de ela ser encaminhada ao componente requisitado (S1, S2 e S3), com mostra a figura 1.

Forçando o Download de Arquivos

Algumas aplicações geram dinamicamente arquivos, principalmente nos formatos PDF, XLS e XML, e que dependendo do tamanho deles a exibição se torna extremamente demorada, pois enquanto o download não estiver completo os aplicativos não exibem o arquivo trazendo para o usuário um grande desconforto no uso da aplicação.

Dessa forma podemos usar um Filtro HTTP para inserir o Header: `Content-disposition` no processo da requisição para que o navegador interprete e exiba a caixa de diálogo com a opção de abrir diretamente ou fazer o download do arquivo.

Criando o DownloadFilter

Precisamos criar uma classe concreta que implementa a interface `javax.servlet.Filter` tratando as opções dos arquivos que se deseja forçar o download no método `doFilter`



Código fonte da classe do filtro HTTP para download:

```
package web.filter;

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

public class DownloadFilter implements Filter {

    private FilterConfig config;

    public void init(FilterConfig config) throws ServletException {
        this.config = config;
    }

    public void destroy() {
        this.config = null;
    }

    public void doFilter(ServletRequest request, ServletResponse response,
        FilterChain chain) throws IOException, ServletException {
        if ( request instanceof HttpServletRequest && response instanceof
        HttpServletResponse ) {
            HttpServletRequest httpReq = (HttpServletRequest) request;
            HttpServletResponse httpResp = (HttpServletResponse) response;
            String fileName = httpReq.getRequestURI().substring(
            httpReq.getRequestURI().lastIndexOf("/"), httpReq.getRequestURI().length() );
            if ( fileName.toLowerCase().endsWith("pdf") ) {
                httpResp.setContentType ("application/pdf");
            } else if ( fileName.toLowerCase().endsWith("xls") ) {
                httpResp.setContentType ("application/vnd.ms-excel");
            }
            httpResp.setHeader ("Content-Disposition", "attachment;
            filename=\"\" + fileName + "\"");
            chain.doFilter(request, response);
        }
    }
}
```

Depois de codificar o componente é preciso registrá-lo no descritor de deploy da J2EE, inserido as seguintes linhas no arquivo web.xml (localizado na pasta WEB-INF do módulo WAR)

```
<!--Registro do Componente DownloadFilter -->
<filter>
    <filter-name>DownloadFilter</filter-name>
    <filter-class>web.filter.DownloadFilter</filter-class>
</filter>

<!--Registro das URLs que serão filtradas pelo DownloadFilter -->
<filter-mapping>
    <filter-name>DownloadFilter</filter-name>
    <url-pattern>*.pdf</url-pattern>
</filter-mapping>

<filter-mapping>
    <filter-name>DownloadFilter</filter-name>
    <url-pattern>*.xls</url-pattern>
</filter-mapping>
```

Pronto, basta gerar um novo pacote WAR, fazer o deploy no servidor de aplicações e testar o download destes tipos de arquivos.