

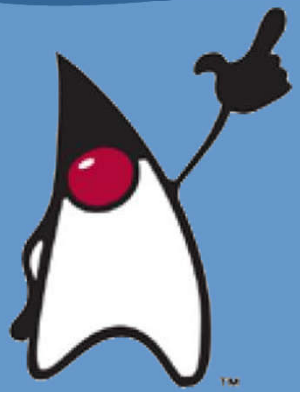


# Java Web

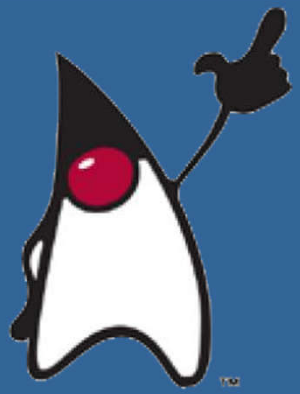
## Daves Martins

davesmartins@yahoo.com.br

Mestre em Computação de Alto Desempenho pela UFRJ
Especialista em Banco de Dados
Analista Web



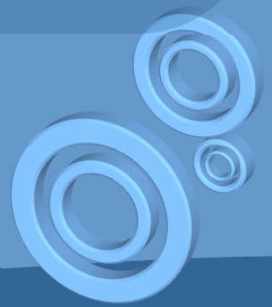
# Filtros



# Filtros

A muitos *Servlets* que se comportam como filtros, fazendo controle de acesso, geração de logs, compactação de dados, e coisas desse tipo. Utlizando filtros conseguimos separar de forma mais clara essas tarefas do resto da aplicação, além de podemos criar filtros genéricos, facilmente "plugáveis", e que poderão ser utilizados em outros lugares sem a necessidade de se alterar nenhuma linha de código.

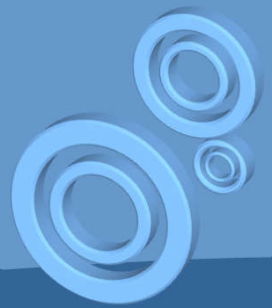
Eles foram introduzidos na API 2.3 de Servlet que está disponível a partir do Tomcat versão 4.0.4.



# Filtros

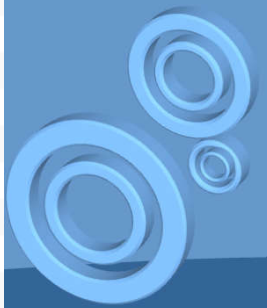
O método *init* é chamado uma vez antes do filtro entrar em operação pela primeira vez. Como parâmetro é passado um *FilterConfig* de onde se pode obter o nome do filtro, os parâmetros de inicialização, e o *ServletContext*. O método *destroy* é chamado para avisar o filtro que ele está sendo desativado, e possa liberar eventuais recursos alocados. O método *doFilter* é onde é feito todo o processamento do filtro. A sua estrutura básica é a seguinte:

```
void doFilter(ServletRequest req, ServletResponse res, FilterChain chain) throws IOException, ServletException {
    //essa parte é executada antes do request chegar ao Servlet
    chain.doFilter(request, response);
    //essa parte é executada depois que o response já foi gerado pelo Servlet
}
```



# Filtros

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class TimerFilter implements Filter {

    private ServletContext context = null;

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

    public void destroy() {
        config = null;
    }

    public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain)
        throws IOException, ServletException {

        long inicio = System.currentTimeMillis();
        chain.doFilter(request, response);
        long fim = System.currentTimeMillis();

        String nome = "";
        if (request instanceof HttpServletRequest) {
            nome = ((HttpServletRequest)request).getRequestURI();
        }
        context.log(nome + ": " + (fim - inicio) + "ms");
    }
}
```



# Filtros

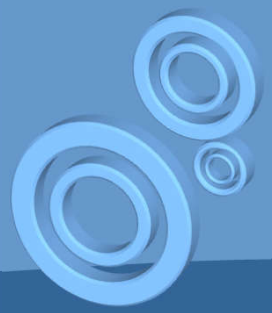
No WebXml:

```
<filter>
    <filter-name>timerFilter</filter-name>
    <filter-class>TimerFilter</filter-class>
</filter>
```

```
<filter-mapping>
    <filter-name>timerFilter</filter-name>
    <url-pattern>/my_servlet</url-pattern>
</filter-mapping>
```

# Filtros

Montar um Filtro para verificar se há alguém logado e assim pode acessar a página:



# Filtros

```
public class FiltroSeguranca implements Filter {

	public void init(FilterConfig config) throws ServletException {
	}

	public void doFilter(ServletRequest req, ServletResponse res,
			FilterChain chain) throws IOException, ServletException {
		HttpSession session = ((HttpServletRequest)req).getSession();
		Usuario usuario = (Usuario)session.getAttribute("usuario");
		if(usuario==null){
			session.setAttribute("msg","Você não está
				logado no sistema!");
			((HttpServletResponse)res).sendRedirect("../index.jsp");
		}else{
			chain.doFilter(req, res);
		}
	}

	public void destroy() {
	}
}
```

# Filtros

Edite o arquivo web.xml e adicione as seguintes linhas abaixo de </welcome-file-list>

```
	<filter>
		<filter-name>Filtro Seguranca</filter-name>

<filter-		class>br.com.javamagazine.jairelton1.FiltroSeguranca</filter-
class>
	</filter>
	<filter-mapping>
		<filter-name>Filtro Seguranca</filter-name>
		<url-pattern>/admin/*</url-pattern>
	</filter-mapping>
```