Java EE 6 nos ha añadido muchas nuevas funciones que a veces nos pasan desapercibidas . Una de las que más destaca es la capacidad de registrar dinámicamente Servlets o Filtros (dynamic filters) dependiendo de las características de nuestra aplicación. Vamos a ver un ejemplo sencillo. El primer paso es construir un “utility project” o jar que contenga una clase que implemente Filter y nos haga un log de las diferentes páginas que vayamos a solicitar.

```
package com.arquitecturajava;

import java.io.IOException;

import javax.servlet.Filter;
import javax.servlet.FilterChain;
import javax.servlet.FilterConfig;
import javax.servlet.ServletException;
import javax.servlet.ServletRequest;
import javax.servlet.ServletResponse;
import javax.servlet.annotation.WebFilter;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;

public class FiltroLog implements Filter {
public FiltroLog() {
// TODO Auto-generated constructor stub
}
public void destroy() {
// TODO Auto-generated method stub
}
public void doFilter(ServletRequest request, ServletResponse response,
```

```
FilterChain chain) throws IOException, ServletException {

HttpServletRequest peticion= (HttpServletRequest)request;
System.out.println(peticion.getRequestURI());
chain.doFilter(request, response);
}
public void init(FilterConfig fConfig) throws ServletException {
// TODO Auto-generated method stub
}

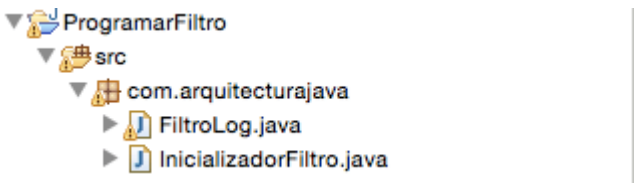
}
```

El código es sencillo de entender simplemente usamos HttpServletRequest y su método getRequestURI para imprimir por la consola la URI.

```
23-dic-2014 10:21:01 org.apache.catalina.core.StandardContext reload
INFO: Se ha completado la recarga de este Contexto
/WebDinamicaTest/index.jsp
```

Filtros y Registro

Eso sí, el filtro tiene una peculiaridad, no esta registrado a nivel de web.xml y ni siquiera lleva la anotación de @WebFilter así que de entrada no funcionará. Esto se debe a que vamos a registrarlo dinámicamente . ¿Cómo se realiza esta operación?. Vamos a incorporar otra clase a nuestro proyecto que se denomina IniciadorFiltro.



Esta clase implementa el interface ServletContextInitializer e incluirá el código que nos permite registrar de una forma dinámica el filtro en una aplicación Web.

```
package com.arquitecturajava;

import java.util.EnumSet;
import java.util.Set;

import javax.servlet.DispatcherType;
import javax.servlet.FilterRegistration;
import javax.servlet.ServletContainerInitializer;
import javax.servlet.ServletContext;
import javax.servlet.ServletException;

public class InicializadorFiltro implements
ServletContainerInitializer {

@Override
public void onStartup(Set<Class<?>> arg0, ServletContext
servletContext)
throws ServletException {

if (servletContext.getContextPath().contains("Test")) {

System.out.println("filtro activado");

FiltroLog filtro= servletContext.createFilter(FiltroLog.class);
FilterRegistration registro= servletContext.addFilter("filtro",
filtro);
```

```
registro.addMappingForUrlPatterns(EnumSet.of(DispatcherType.REQUEST,
DispatcherType.FORWARD),
true, "/*");

}else {
System.out.println("filtro desactivado");
}
}

}
```

Como podemos ver esta clase registra un filtro a nivel programático. Para ello revisa si nuestra aplicación cuando se despliega incluye la palabra “Test” en su nombre. Si este es el caso el filtro se registrará y grabará todas las peticiones que se realicen.

```
23-dic-2014 10:22:51 org.apache.catalina.core.StandardContext reload
INFO: Se ha completado la recarga de este Contexto
/WebDinamicaTest/pagina1.jsp
/WebDinamicaTest/pagina2.jsp
```

En cambio si renombramos la aplicación como “WebDinamica” y reiniciamos el filtro quedará desactivado y no se grabará nada ya que la palabra Test no es parte del nombre de la aplicación.

```
23-dic-2014 10:22:51 org.apache.catalina.core.StandardContext reload
INFO: Se ha completado la recarga de este Contexto
```

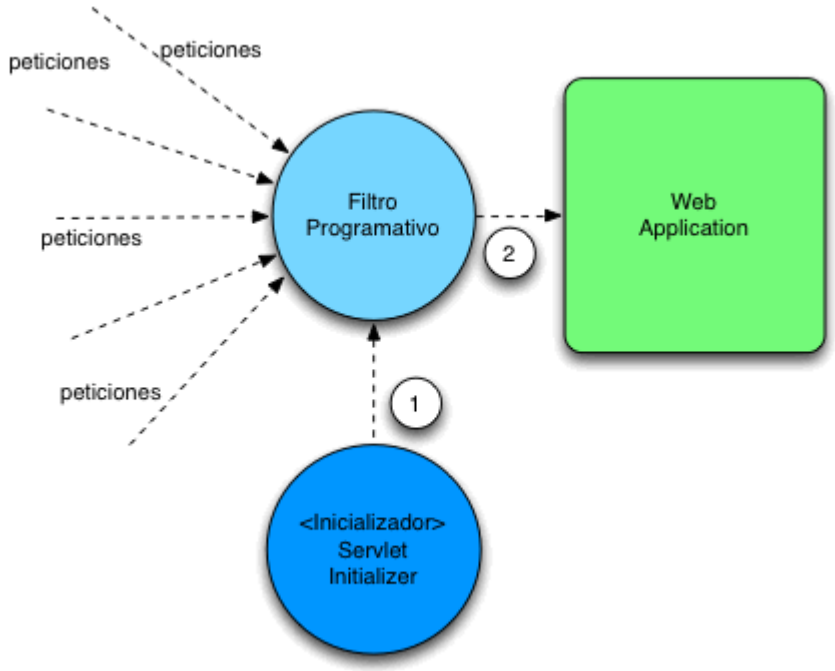
Para que el registro dinámico funcione correctamente debemos añadir en la carpeta META-INF de nuestro “utility project” una carpeta services que contiene el siguiente fichero.



Este fichero nos sirve para saber que clases debemos inicializar cuando se inicializa el containier. En este caso su contenido será el siguiente:

com.arquitecturajava.InicializadorFiltro

De esta forma hemos registrado un filtro dinámicamente dependiendo de nuestras necesidades.



De esta forma podríamos registrar los Filtros dependiendo totalmente de nuestras

necesidades.

Otros artículos relacionados : ServletContext , UtilityProject ,Filter