# Valves y Filtros
## Introducción

- Tecnologías para interceptar y pre-procesar peticiones (*request*) y respuestas (*response*) HTTP con independencia de las aplicaciones.
  - **Valves**
    - Tecnología propietaria de *Tomcat*.
  - **Filtros**
    - Tecnología definida en el API de *Servlets* (desde la versión 2.3).

# Valves y Filtros
## Introducción

- Ejemplos de utilización
  - Generar *logs* de las peticiones realizadas a un una aplicación o *Servlet*.
  - Gestionar la seguridad permitiendo accesos solo a determinadas IPs.
  - Comprimir datos antes de enviarlos al cliente.
  - Identificar la localización (país, lenguaje, …) de peticiones y responder en consecuencia
  - …

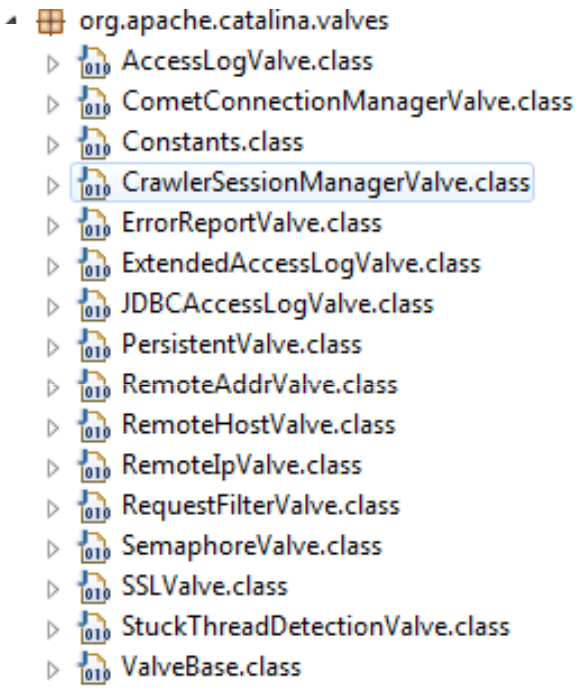
# Valves y Filtros

## Valves

- Tecnología propietaria de *Tomcat.*
- Objetos que permite interceptar y pre-procesar peticiones y respuestas HTTP.
- Creación
  - Interfaz
    - **`org.apache.catalina.Valve`**
  - Clase **abstracta** para implementar Valves.
    - **`org.apache.catalina.valves.ValveBase`**
  - Existen implementaciones predefinidas de *Valves* en *Tomcat* (RemotrAddrValve, RemoteHostValve, AccessLogValve, …)
- Web
  - http://tomcat.apache.org/tomcat-7.0-doc/config/valve.html

# Valves y Filtros
## Valves

```java
package net.daw01.valves;

import java.io.IOException;

import javax.servlet.*;

import org.apache.catalina.Valve;
import org.apache.catalina.valves.*;
import org.apache.catalina.connector.Request;
import org.apache.catalina.connector.Response;


public class LogginValve extends ValveBase {

    @Override
    public void invoke(Request request, Response response)
            throws IOException, ServletException {

        String IPRemota = request.getRemoteAddr();
        String URI = request.getRequestURI();
        System.out.println("URI " + URI + " accedida desde: "+IPRemota);
        Valve nextValve = getNext();
        if(nextValve!=null){
            nextValve.invoke(request, response);
        }
    }
}
```



# Valves y Filtros

## Valves

- Elemento **<Valve>** para configurarlos.
  - Dentro de **<Engine>**
    - Procesará las peticiones de todas las aplicaciones de todos los virtual host.
  - Dentro de **<Host>**
    - Procesará las peticiones de todas aplicaciones de ese virtual host
  - Dentro de un **<Context>**
    - Procesará las peticiones de esa aplicación.
- Observa que se puede configurar un **Valve** para múltiples aplicaciones.

# Valves y Filtros

## Valves

```
<Host name="localhost"  appBase="webapps"
      unpackWARs="true" autoDeploy="true">

  <!-- SingleSignOn valve, share authentication between web applications
       Documentation at: /docs/config/valve.html -->
  <!--
  <Valve className="org.apache.catalina.authenticator.SingleSignOn" />
  -->

  <!-- Access log processes all example.
       Documentation at: /docs/config/valve.html
       Note: The pattern used is equivalent to using pattern="common" -->
  <Valve className="org.apache.catalina.valves.AccessLogValve" directory=$
         prefix="localhost_access_log." suffix=".txt"
         pattern="%h %l %u %t &quot;%r&quot; %s %b" />

  <Valve className="net.daw01.valves.LogginValve" />

  <Valve className="org.apache.catalina.valves.RemoteAddrValve"
         allow="192.168.1.16" />
```

# Práctica

- Práctica 7.13
  - Valves.

```
<Host name="localhost"  appBase="webapps"
      unpackWARs="true" autoDeploy="true">

  <!-- SingleSignOn valve, share authentication between web applications
       Documentation at: /docs/config/valve.html -->
  <!--
  <Valve className="org.apache.catalina.authenticator.SingleSignOn" />
  -->

  <!-- Access log processes all example.
       Documentation at: /docs/config/valve.html
       Note: The pattern used is equivalent to using pattern="common" -->
  <Valve className="org.apache.catalina.valves.AccessLogValve" directory=$
         prefix="localhost_access_log." suffix=".txt"
         pattern="%h %l %u %t &quot;%r&quot; %s %b" />

  <Valve className="net.daw01.valves.LogginValve" />

  <Valve className="org.apache.catalina.valves.RemoteAddrValve"
         allow="192.168.1.16" />
```

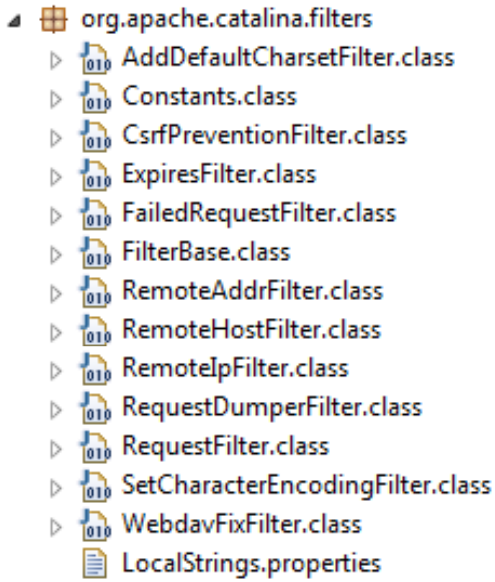
```
package net.daw01.valves;

import java.io.IOException;

import javax.servlet.*;

import org.apache.catalina.Valve;
import org.apache.catalina.valves.*;
import org.apache.catalina.connector.Request;
import org.apache.catalina.connector.Response;


public class LogginValve extends ValveBase {

    @Override
    public void invoke(Request request, Response response)
            throws IOException, ServletException {

        String IPRemota = request.getRemoteAddr();
        String URI = request.getRequestURI();
        System.out.println("URI " + URI + " accedida desde: "+IPRemota);
        Valve nextValve = getNext();
        if(nextValve!=null){
            nextValve.invoke(request, response);
        }
    }
}
```

# Valves y Filtros
## Filtros

- Definidos en el API de *Servlets* (desde la versión 2.3).
- Objetos que permite interceptar y pre-procesar peticiones y respuestas HTTP.
- Creación
  - Interfaz
    - `javax.servlet.Filter`
  - Existen implementaciones predefinidas de *Filters* en *Tomcat* (RemoteIPFilter, ExpiresFilter, RequestDumperFilter, …).
- Web
  - http://tomcat.apache.org/tomcat-7.0-doc/config/filter.html

# Valves y Filtros
## Filtros

```java
package net.daw01.curso;

import java.io.IOException;
import javax.servlet.Filter;
import javax.servlet.FilterChain;
import javax.servlet.FilterConfig;
import javax.servlet.ServletContext;
import javax.servlet.ServletException;
import javax.servlet.ServletRequest;
import javax.servlet.ServletResponse;

public class FiltroContador implements Filter {

    private FilterConfig config;

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

    public void doFilter(ServletRequest request, ServletResponse response,
            FilterChain chain) throws IOException, ServletException {
        // Contexto de sesión de almacenará el contador
        ServletContext contexto = config.getServletContext();
        // Se comprueba si el atributo contador existe dentro del contexto
        Integer contador = (Integer) contexto.getAttribute("contador");
        if (contador == null) {
            contador = new Integer(0);
        }
        contador = new Integer(contador.intValue() + 1);
        contexto.setAttribute("contador", contador);
        System.out.println();
        System.out.println("Número de accesos a curso : " + contador.intValue());
        // Para que se invoque al siguiente filtro
        chain.doFilter(request, response);
    }

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



# Valves y Filtros
## Filtros

- Los filtros se configuran e inicializan en el descriptor de despliegue de cada aplicación.
- Es posible reutilizar filtros en varias aplicaciones pero cada filtro debe ser configurado en cada aplicación por separado.

```
<filter>
    <filter-name>FiltroContador</filter-name>
    <filter-class>net.daw01.curso.FiltroContador</filter-class>
</filter>
<filter-mapping>
    <filter-name>FiltroContador</filter-name>
    <url-pattern>/*</url-pattern>
</filter-mapping>
```

# Práctica

- Práctica 7.14
  - Filtros.

```java
package net.daw01.curso;

import java.io.IOException;
import javax.servlet.Filter;
import javax.servlet.FilterChain;
import javax.servlet.FilterConfig;
import javax.servlet.ServletContext;
import javax.servlet.ServletException;
import javax.servlet.ServletRequest;
import javax.servlet.ServletResponse;

public class FiltroContador implements Filter {

    private FilterConfig config;

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

    public void doFilter(ServletRequest request, ServletResponse response,
            FilterChain chain) throws IOException, ServletException {
        // Contexto de sesión de almacenará el contador
        ServletContext contexto = config.getServletContext();
        // Se comprueba si el atributo contador existe dentro del contexto
        Integer contador = (Integer) contexto.getAttribute("contador");
        if (contador == null) {
            contador = new Integer(0);
        }
        contador = new Integer(contador.intValue() + 1);
        contexto.setAttribute("contador", contador);
        System.out.println();
        System.out.println("Número de accesos a curso : " + contador.intValue());
        // Para que se invoque al siguiente filtro
        chain.doFilter(request, response);
    }

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

```
er-name>FiltroContador</filter-name>
er-class>net.daw01.curso.FiltroContador</filter-class>

apping>
er-name>FiltroContador</filter-name>
pattern>/*</url-pattern>
mapping>
```

# Ficheros de registros (*logs*)

- Directorio que almacena los ficheros de logs
  - **`$CATALINA_BASE/log (enlace /var/log/tomcat7)`**

```
alumno@ServidorLinux01:/var/lib/tomcat7$ ls -l
total 16
drwxr-xr-x 3 tomcat7 tomcat7 4096 jun  2 18:57 common
lrwxrwxrwx 1 root    root      12 abr 11 14:26 conf -> /etc/tomcat7
lrwxrwxrwx 1 root    root      17 abr 11 14:26 logs -> ../../log/tomcat7
drwxr-xr-x 3 tomcat7 tomcat7 4096 jun  2 18:57 server
drwxr-xr-x 3 tomcat7 tomcat7 4096 jun  2 18:57 shared
drwxrwxr-x 7 tomcat7 tomcat7 4096 jun  5 01:41 webapps
lrwxrwxrwx 1 root    root      19 abr 11 14:26 work -> ../../cache/tomcat7
alumno@ServidorLinux01:/var/lib/tomcat7$
```

- Web
  - http://tomcat.apache.org/tomcat-7.0-doc/logging.html

# Ficheros de registros (*logs*)

- Ficheros de logs
  - *Logs* de <Engine>
    - catalina<fecha>.log
    - catalina.out
      - System.out
      - System.err
  - Logs de <Host>
    - Errores
      - <nombre del host>.<fecha>.log
    - Accesos
      - <nombre del host>_access_log.<fecha>.txt (Valve)

```
alumno@ServidorLinux01:/var/lib/tomcat7/logs$ ls
catalina.2012-06-02.log   localhost.2012-06-04.log
catalina.2012-06-03.log   localhost.2012-06-05.log
catalina.2012-06-04.log   localhost_access_log.2012-06-02.txt
catalina.2012-06-05.log   localhost_access_log.2012-06-03.txt
catalina.out              localhost_access_log.2012-06-04.txt
localhost.2012-06-02.log  localhost_access_log.2012-06-05.txt
localhost.2012-06-03.log
```

# Ficheros de registros (*logs*)

- Configuración y personalización
  - **Valves**
  - Fichero s
    - **$CATALINA_BASE/conf/logging.properties**
    - **WEB-INF/classes/logging.properties**

```
<Valve className="org.apache.catalina.valves.AccessLogValve"
       directory="logs"
       prefix="localhost_access_log." suffix=".txt"
       pattern="%h %l %u %t &quot;%r&quot; %s %b" />
```

```
handlers = 1catalina.org.apache.juli.FileHandler, 2localhost.org.apache.juli.Fi$

.handlers = 1catalina.org.apache.juli.FileHandler, java.util.logging.ConsoleHan$

############################################################
# Handler specific properties.
# Describes specific configuration info for Handlers.
############################################################

1catalina.org.apache.juli.FileHandler.level = FINE
1catalina.org.apache.juli.FileHandler.directory = ${catalina.base}/logs
1catalina.org.apache.juli.FileHandler.prefix = catalina.

2localhost.org.apache.juli.FileHandler.level = FINE
2localhost.org.apache.juli.FileHandler.directory = ${catalina.base}/logs
2localhost.org.apache.juli.FileHandler.prefix = localhost.

java.util.logging.ConsoleHandler.level = FINE
java.util.logging.ConsoleHandler.formatter = java.util.logging.SimpleFormatter
```

# Ficheros de registros (*logs*)

- Fichero **catalina.out**
  - Puede crecer mucho y afectar la rendimiento y la estabilidad de *Tomcat.*
  - En entornos de producción es una buena práctica personalizarlo.
  - Es posible deshabilitarlo a nivel global o deshabilitar las aplicaciones que no queramos que escriban en el.

```
<Context path="/curso" swallowOutput="true">
```

# Práctica

- Práctica 7.15
  - Ficheros de registros *(logs)*.

```
alumno@ServidorLinux01:/var/lib/tomcat7/logs$ ls
catalina.2012-06-02.log   localhost.2012-06-04.log
catalina.2012-06-03.log   localhost.2012-06-05.log
catalina.2012-06-04.log   localhost_access_log.2012-06-02.txt
catalina.2012-06-05.log   localhost_access_log.2012-06-03.txt
catalina.out              localhost_access_log.2012-06-04.txt
localhost.2012-06-02.log  localhost_access_log.2012-06-05.txt
localhost.2012-06-03.log
```