

P10- Pruebas de aceptación: JMeter

Pruebas de aceptación de aplicaciones Web

El objetivo de esta práctica es automatizar pruebas de aceptación de pruebas emergentes NO funcionales sobre una aplicación Web. Utilizaremos la herramienta de escritorio JMeter.

Usaremos la aplicación JPetStore de prácticas anteriores. Proporcionamos la secuencia de entradas de nuestro driver, que nos permitirán evaluar si nuestro sistema soporta una determinada carga de usuarios (pruebas de carga).

Bitbucket

El trabajo de esta sesión también debes subirlo a *Bitbucket*. Todo el trabajo de esta práctica deberá estar en el directorio **P10-JMeter** dentro de tu espacio de trabajo, es decir, dentro de tu carpeta: ppss-Gx-apellido1-apellido2-2019.

Recuerda que si trabajas desde los ordenadores del laboratorio primero deberás configurar git y clonar tu repositorio de Bitbucket.

Aplicación a validar

Vamos a seguir trabajando con la aplicación web jpetstore de la sesión P08.

Si no hemos borrado el directorio target del proyecto maven correspondiente, para arrancar el servidor Wildfly y ejecutar la aplicación, simplemente tendremos que teclear desde un terminal (y desde el directorio **jpetstore-6**, que contiene el pom del proyecto maven):

```
> mvn cargo:run -P wildfly11
```

Con esto ya tenemos accesible la aplicación desde la URL:

```
http://localhost:8080/jpetstore
```

Opcionalmente podemos usar el servidor wildfly que ya tenemos instalado en el directorio \$HOME de la máquina virtual. Para ello necesitarás el war de nuestra aplicación jpetstore-6, que encontrarás en el directorio target del proyecto maven jpetstore-6.

Suponiendo que hemos copiado el fichero jpetstore.war en nuestro \$HOME y que estamos situados en dicho directorio, podemos desplegar la aplicación usando los siguientes comandos:

```
> cd wildfly-11.0.0.Final/bin
//arrancamos el servidor
> ./standalone.sh &
//desplegamos el war
> ./jboss-cli.sh --connect --command="deploy ../../jpetstore.war --force"
//si queremos consultar las aplicaciones desplegadas
> ./jboss-cli.sh --connect --command=deployment-info
//cuando terminemos de usar la aplicación detenemos el servidor
> ./jboss-cli.sh --connect --command=:shutdown
```

Una vez que hayamos desplegado el war, las siguientes veces simplemente tendremos que arrancar el servidor con el comando *standalone.sh*

Recuerda que la consola de administración de wildfly estará "escuchando" en el puerto 9990, por lo que podremos saber si el servidor está en marcha usando el comando *"lsof -i:9990"*.

Ejercicios

Para poder hacer los ejercicios asegúrate de que has desplegado correctamente la aplicación y que puedes acceder a ella desde Firefox.

Para iniciar jmeter debes hacerlo desde el terminal, simplemente teclea desde el terminal:

```
> jmeter
```

Nota1: Cuando arrancamos JMeter, por defecto la apariencia de la aplicación es en modo “Darcula”, con lo cual tendrá un aspecto bastaste oscuro. Si preferís cambiarlo a otro más claro, podéis hacerlo desde *Options→Look and Feel*, y seleccionar por ejemplo “Metal”.

Nota2: En nuestros tests, asumiremos que el usuario “z” tiene ya una cuenta en la tienda de animales. Por lo tanto, antes de implementar los drivers, crea una cuenta para el usuario “z” que ya hemos usado en sesiones anteriores.

⇒ Ejercicio 1: Uso de *Proxies* en JMeter

Antes de comenzar a implementar nuestros tests, vamos a explicar el **uso de “proxies”** en JMeter para averiguar los parámetros de una petición al servidor, cuando éstos no son visibles en la url correspondiente.

La creación de un plan de pruebas con JMeter puede presentar cierta dificultad cuando se ven implicadas *queries* y/o formularios complejos, peticiones https POSTs, así como peticiones javascript, en las que los parámetros que se envían por la red NO son visibles en la URL.

JMeter proporciona un **servidor HTTP proxy**, a través del cual podemos utilizar el navegador para realizar las pruebas, y JMeter “grabará” las peticiones http generadas, tal y como se enviarán al servidor, y creará los correspondientes HTTP *samplers*. Una vez que hayamos “guardado” las peticiones HTTP que necesitemos, podemos utilizarlas para crear un plan de pruebas.

Vamos a ver paso a paso cómo utilizar dicho servidor *proxy*:

- A) JMeter “grabará” todas las acciones que realicemos en el navegador en un controlador de tipo “recording”, por lo que tendremos que incluir un controlador de este tipo en nuestro plan de pruebas. Lo primero que haremos será **añadir un grupo de hilos** en nuestro plan. Desde el menú contextual del nodo “Test Plan”, elegiremos *Add→Threads (Users)→Thread Group*. A continuación, desde el menú contextual del grupo de hilos que acabamos de añadir tendremos que seleccionar *Add→Logic Controller→Recording Controller*.
- B) Ahora vamos a **añadir un servidor proxy** para realizar peticiones HTTP. Desde el menú contextual del elemento *Test Plan*, seleccionando *Add→NonTest Elements→HTTP(S) Test Script Recorder*. En la configuración del *proxy* tendremos que asegurarnos de que el **puerto** no está ocupado con algún otro servicio en nuestra máquina local, (ver el valor del campo *Global Settings→Port* de la configuración del elemento *HTTP(S) Test Script Recorder*, por defecto tiene el valor 8888). En nuestro caso, los puertos 8080 y 9990 están ocupados por el servidor de aplicaciones Wildfly, por lo que en principio el puerto 8888 estará libre.

Cómo averiguar qué puertos están siendo utilizados

Para averiguar qué conexiones están activas y qué procesos y puertos están abiertos podemos usar el comando:

```
> ss -tap (https://www.binarytides.com/linux-ss-command/)
```

O, de forma alternativa, también puedes usar el comando

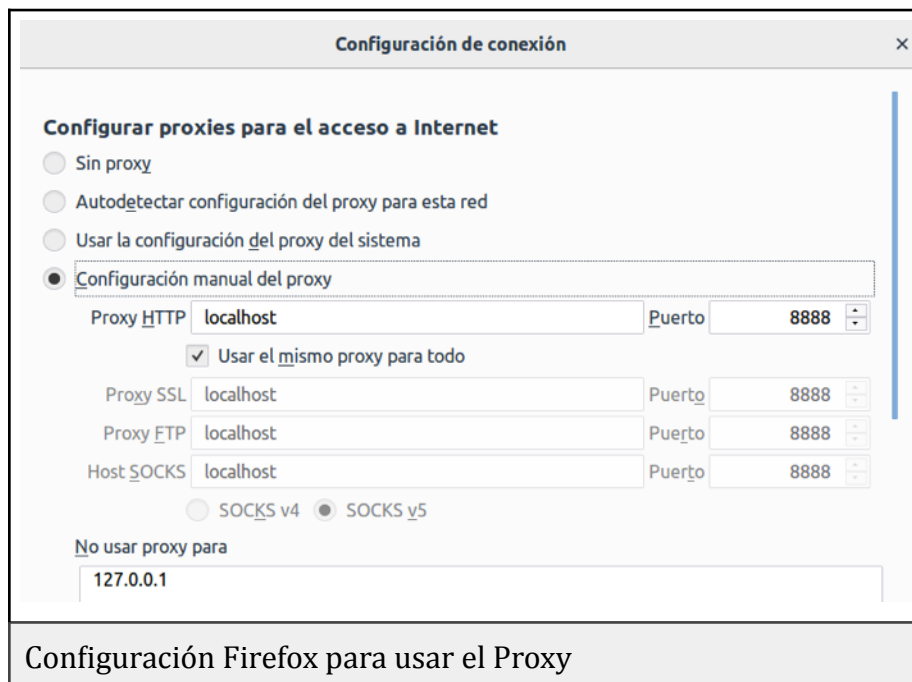
```
> lsof -i tcp
```

Por defecto, el servidor proxy (elemento *HTTP(S) Test Script Recorder*) interceptará todas las peticiones http dirigidas a nuestra aplicación y las “grabará” en un controlador de tipo *recording* (ver el valor del campo *Test plan content→Target Controller*, en la pestaña *Test Plan Creation* de la configuración del *HTTP(S) Test Script Recorder*). Ya hemos incluido el controlador de grabación en nuestro plan. Por defecto, JMeter grabará cualquier “cosa” que envíe o reciba el navegador, incluyendo páginas HTML, ficheros javascript, hojas de estilo css, imágenes, etc, la mayoría de las cuales no nos serán de mucha utilidad para nuestro plan de pruebas. Para “filtrar” el tipo de

contenido que queremos (o no queremos) utilizaremos patrones URL a Incluir (o a Excluir), desde la pestaña *Request Filtering* de la configuración del Servidor Proxy.

Por ejemplo, vamos a **excluir** del proceso de grabación las **imágenes** y las **hojas de estilo** (desde la pestaña *Request Filtering*, y más concretamente desde el elemento *URL Patterns to exclude*). Para ello tendremos que pulsar con el ratón sobre el botón *Add*, y a continuación hacer doble click en la nueva fila en blanco añadida (en color blanco). Tenemos que añadir dos líneas, y escribiremos las expresiones regulares `*.gif` y `*.css` como patrones URL a excluir (cada patrón en una línea diferente). NO hay que poner comillas al introducir cada patrón. Para más información sobre expresiones regulares podéis consultar: <http://rubular.com>).

- C) A continuación tendremos que **configurar el navegador** (en este caso Firefox) para utilizar el puerto en donde actuará el proxy JMeter. Para ello iremos al menú de “Preferencias→General→Configuración de red→Configuración...”. Desmarcamos la opción actual (“Usar la configuración proxy del sistema”), y marcamos “Configuración manual del proxy”. Como valor de Proxy HTTP pondremos localhost, y el puerto el 8888. Acuérdate de marcar la casilla “Usar el mismo proxy para todo”. También tenemos que asegurarnos de que en el campo “No usar proxy para” NO no figure “localhost”. A continuación mostramos la configuración Inicial de Firefox, y la configuración que debemos usar para utilizar el proxy JMeter.



Vamos a utilizar el *proxy* de JMeter para ver cuáles son las peticiones http que tenemos que utilizar para “loguearnos” en el sistema. La petición de *login* es una petición http de tipo POST, y los

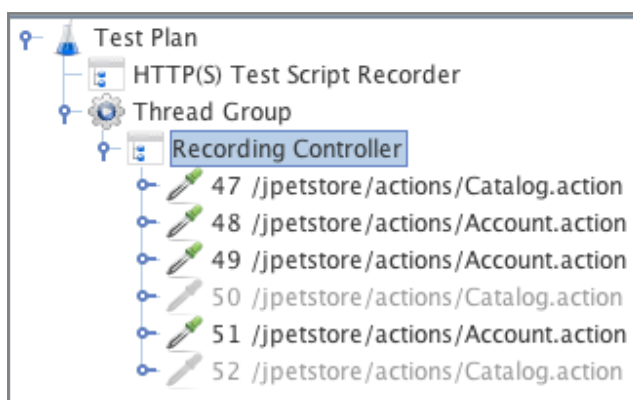
parámetros de esta petición NO son visibles en el navegador, por lo que averiguaremos dichos valores a través del *proxy*.

- D) Una forma de asegurarnos de que el navegador está utilizando el proxy de JMeter es: estando “parado” el proxy, intentar navegar desde Firefox. Si el navegador sigue sirviendo las páginas es que NO está utilizando el proxy. Una vez que nos hemos asegurado de que hemos configurado adecuadamente Firefox para utilizar el proxy de JMeter, iremos al panel de configuración de nuestro *Test Script Recorder* y en el panel “State” veremos el botón “Start”. Al pulsar dicho botón **pondremos en funcionamiento el proxy** y podemos comenzar la grabación (no podremos arrancar el proxy si previamente no hemos añadido el controlador de grabación en nuestro plan).
- E) Al arrancar el proxy nos aparecerá una alerta indicando que se ha creado un certificado temporal en un directorio de JMeter (simplemente tenemos que pulsar sobre “OK” en dicha ventana). También veremos una ventana con el nombre “Recorder: Transactions Control” que permanecerá abierta hasta que detengamos el proxy. Ahora, cualquier acción que hagamos desde el navegador, quedará “registrada” en el controlador de grabación que hemos añadido en nuestro plan de pruebas. Vamos a realizar las siguientes acciones en el navegador:
- Entramos en la aplicación de la tienda de animales (<http://localhost:8080/jpetstore/actions/Catalog.action>), a continuación pinchamos sobre Sign In.
 - Introducimos las credenciales del usuario z que ya tenemos creado y pulsamos sobre “Login”.
Nota: si no has creado antes el usuario tal y como te hemos indicado, hazlo ahora, aunque entonces se grabarán también esas acciones, las cuales no vamos a necesitar en nuestro driver. No te preocupes, porque podrás borrarlas.
 - Finalmente pulsamos sobre “Sign Out” en nuestra aplicación web y PARAMOS la grabación desde el proxy de JMeter.

Ahora podemos ver las peticiones http que han quedado grabadas en el “Controlador Grabación” de nuestro plan. Selecciona cada una de ellas y observa la ruta y parámetros de petición (**comprueba que cuando hacemos “login” en el sistema se envían 5 parámetros, y que es una petición POST**). Utilizaremos esta información en el siguiente ejercicio para confeccionar nuestro plan de pruebas.

Observarás que hay un par de peticiones que están deshabilitadas (aparecen en gris claro). Se trata de redirecciones realizadas por la aplicación, que también quedan grabadas, pero que no se lanzarán cuando ejecutemos el test.

Nuestro plan de pruebas JMeter cuando finalicemos el ejercicio tendrá un aspecto similar a éste (los números que ves a la izquierda de la imagen mostrada se corresponde con una secuencia de acciones que no incluyen la creación del usuario z. Dependiendo del número de acciones que hayas grabado tendrás más o menos elementos hijo del nodo Controlador Grabación):



Observaciones sobre el CONTROLADOR DE GRABACIÓN: Cuando ejecutemos el plan de pruebas, el “Recording Controller” no tiene ningún efecto sobre la lógica de ejecución de las acciones que contiene, es como el controlador “Simple”, es decir, actúa simplemente como un nodo que “agrupa” un conjunto de elementos.

Guarda el plan de pruebas que hemos creado, lo puedes hacer desde “File→Save Test Plan as”, y ponle el nombre **Plan-Ejercicio1-proxy..**

Una vez terminado el ejercicio recuerda volver a la configuración inicial de red en Firefox para poder acceder a Internet sin utilizar el proxy de JMeter.

➡ ➡ Ejercicio 2: Driver y validación de carga

Vamos a crear un plan de pruebas a partir del plan del ejercicio 1. Queremos evaluar el rendimiento de nuestra aplicación y comprobar si es capaz de soportar una determinada carga de usuarios.

Llamaremos a nuestro plan de pruebas “**Plan JMeter**”. Lo primero que haremos será añadir un **grupo de hilos**. En este caso, ya lo tenemos del ejercicio anterior. Como ya hemos visto, JMeter trabaja simulando un conjunto de usuarios concurrentes realizando varias tareas sobre la aplicación. Cada usuario simulado se representa como un hilo. Por lo tanto, el número de hilos representa el número de usuarios simultáneos generados por el grupo de hilos. De momento dejaremos los valores por defecto: un hilo, un periodo de subida de 1 y un bucle con una única iteración. Borraremos el elemento *Script Recorder* puesto que ya no nos hace falta. A continuación añadiremos los siguientes elementos:

- A) Primero vamos a incluir dos **elementos de configuración** en nuestro plan de pruebas. Los elementos de configuración evitan que tengamos que introducir repetidamente alguna información de configuración de los *samplers*. Por ejemplo, si en el plan de pruebas incluimos varias llamadas a algunas páginas que están protegidas mediante una autenticación http básica, se podría compartir la información de usuario y password, para no tener que repetir estos datos para cada *sampler* dentro del plan de pruebas. Para pruebas de aplicaciones Web, el elemento de configuración más importante es “**HTTP Request Defaults**”. Lo añadimos desde el menú contextual del grupo de hilos (*Add→ Config Element→HTTP Request Defaults*). Estableceremos el nombre del servidor a **localhost**, y el número de puerto a **8080**.

Otro elemento de configuración muy útil es “**HTTP Cooke Manager**”. Éste almacena cookies y hace que estén disponibles para subsecuentes llamadas al mismo sitio, tal y como haría un navegador. Dado que cada hilo representa un usuario diferente, las cookies no se compartirían entre hilos. Lo añadiremos desde el menú contextual del grupo de hilos (*Add→ Config Element→HTTP Cookie Manager*). En este caso usaremos la configuración por defecto de este elemento.

Si nos equivocamos al crear un elemento, en cualquier momento podemos borrar/modificar/mover cualquier elemento del plan. Para **borrar** un elemento lo haremos desde el menú contextual de dicho elemento. Para **modificar** su configuración simplemente seleccionaremos dicho elemento. Para **mover** un elemento a otra posición dentro del plan, seleccionaremos y arrastraremos dicho elemento a su nueva posición.

- B) Ahora vamos a añadir un **sampler** para hacer peticiones de páginas **http** a nuestro servidor de aplicaciones y así simular la acción de los usuarios (desde el menú contextual del grupo de hilos: *Add→Sampler→HTTP Request*). La URL inicial de la tienda es “http://localhost:8080/jpetstore”. Se trata de una petición GET. El servidor y el puerto ya los hemos indicado en un elemento de configuración (localhost y 8080), por lo que no es necesario indicarlos. Después del contexto inicial (cuyo valor es *jpetstore*), vendría el nombre del recurso, en este caso, la página html a la que queremos acceder. Observamos que no aparece esta información (no es necesaria ya que por defecto se accederá a *index.html*). Por lo tanto, lo único que haremos será indicar en el campo *Path* el valor “**/jpetstore/**” (no hay que poner las comillas). Vamos a hacer uso de varios *samplers* http, por lo que será conveniente indicar un nombre adecuado para poder diferenciarlos y hacer más “legible” nuestro código. Por ejemplo, para este primer *sampler* podemos especificar como nombre el valor “**Home**”
- C) Para asegurarnos de que estamos en la página correcta, vamos a añadir una **aserción** (desde el menú contextual de la petición HTTP Home: *Add→Assertions→Response Assertion*). Nos aseguraremos de marcar “**Text Response**”, y el patrón a probar contendrá la cadena “*Welcome to JPetStore 6*”. Recuerda que primero tienes que pinchar sobre el botón “**Add**”, y después hacer doble click en la fila en blanco añadida para editar su contenido. El texto se pone SIN comillas. Otra opción es copiar el texto anterior (con ctrl-C) y directamente seleccionar el botón **Add From Clipboard**.
- D) Ahora añadiremos **otra petición http** GET (desde el grupo de hilos), en este caso, puesto que la página mostrada nos muestra el catálogo de animales, podemos asignarle como nombre “**Catalog**”. Utiliza el navegador y pincha en el hiperenlace (“enter the Store”) para ver la URL de la nueva petición. En este caso puedes ver que la ruta tendrá el valor “*/jpetstore/actions/Catalog.action*”. para verificar que entramos en la página correcta añadiremos una aserción igual que en el caso anterior con el patrón a probar “*Saltwater*”.
- E) Desde la página anterior, vamos a acceder al menú de Reptiles (por ejemplo pinchando sobre la imagen correspondiente). Tendremos, por lo tanto, que añadir otra **petición http** (le pondremos el nombre “**Reptiles**”). Podemos comprobar que la URL del navegador, es la siguiente: “http://

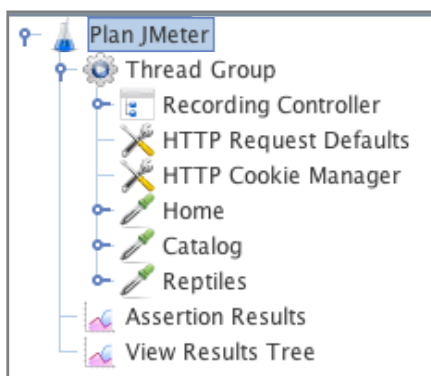
localhost:8080/jpetstore/actions/Catalog.action?viewCategory=&categoryId=REPTILES” En este caso, vemos que la petición tiene dos parámetros (los parámetros se especifican con pares nombre=valor, separados por ‘&’): un primer parámetro con nombre viewCategory (que en este caso no tiene valor asociado, pero que es necesario incluir), y un segundo parámetro con nombre categoryId (cuyo valor es REPTILES). Por lo tanto, indicaremos, además de la **ruta** (/jpetstore/actions/Catalog.action), los **parámetros** anteriores con el valor correspondiente para categoryId. Acuérdate de que primero tienes que pulsar el botón añadir, y hacer doble click en las entradas añadidas para introducir los valores. No tienes que poner comillas). Añade una **aserción de respuesta textual** con el patrón “Reptiles”.

Nota: Como acabamos de ver, cuando utilizamos **samplers http**, para averiguar la “ruta” y parámetros de la petición URL podemos fijarnos en la ruta que aparece en el navegador al ejecutar la aplicación. Esta aproximación funciona bien cuando se trata de peticiones GET “simples” y parámetros “sencillos” de “manipular”. Páginas que contengan formularios grandes pueden requerir docenas de parámetros. Además, si se utilizan peticiones HTTP POST (como por ejemplo el envío de login y password), los valores de los parámetros no aparecerán en la URL, por lo que tendremos que descubrirlos de alguna otra forma. Una aproximación consiste en utilizar un **Proxy JMeter**, que grabará el caso de prueba por nosotros, tal y como hemos visto en el ejercicio 1, y que luego podremos modificar según nuestras necesidades.

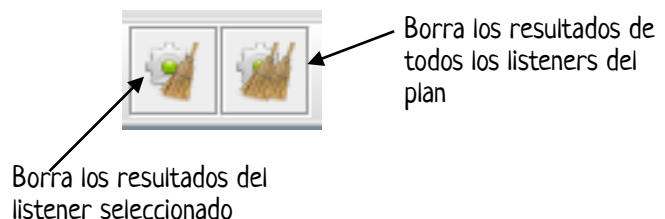
- F) Antes de continuar vamos a **grabar nuestro plan** de pruebas (como medida de precaución es recomendable grabar a menudo). Lo podemos hacer desde “File→Save Test Plan as”, y podemos ponerle el nombre “**Plan-Ejercicio2**”. Dado que podemos ejecutar el plan en cualquier momento, vamos a hacerlo ahora, pero antes nos aseguraremos de que en las propiedades del grupo de hilos solamente se genera un hilo, con un periodo de subida de 1 segundo y una única vez.

Para poder “ver” lo que ocurre al ejecutar las pruebas es imprescindible añadir uno o varios “**Listeners**” que nos muestren (y/o graben los resultados de las pruebas en algún fichero). Utilizaremos, de momento, dos de ellos. Desde el menú contextual del plan de pruebas elegimos **Add→Listener→Assertion Results**. También añadiremos un segundo Listener: **View Results Tree**. El primero de ellos nos mostrará en pantalla las etiquetas de los samplers ejecutados, y los resultados de las aserciones de nuestro plan, para cada uno de los samplers. El segundo nos muestra las peticiones y respuestas de cada uno de los *samplers*.

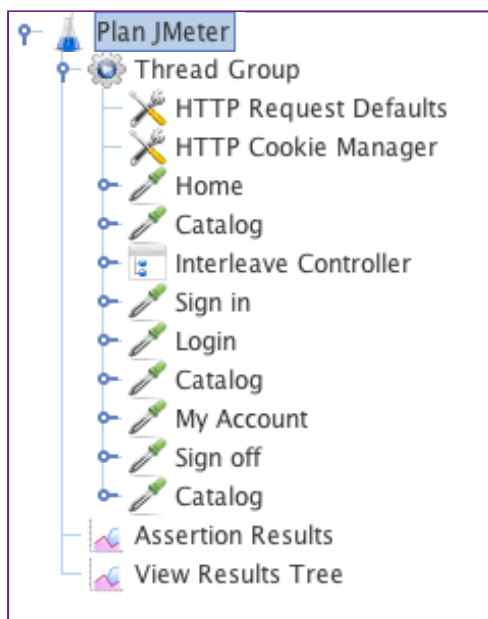
Para **ejecutar el plan** tienes que ir al menú de JMeter y seleccionar **Run→Start** (también puedes utilizar el icono con forma de **triángulo verde**). En el extremo derecho de la barra de herramientas de JMeter verás un pequeño cuadrado. Durante la ejecución del plan, verás que el cuadrado se muestra de color verde. Cuando acaba la ejecución vuelve a estar gris. Selecciona cualquiera de los listeners que hemos añadido y observa los resultados. Si ejecutas el plan con un listener seleccionado, verás cómo se va actualizando durante la ejecución. Realiza varias ejecuciones para familiarizarte con la información que proporcionan estos dos listeners. (si todas las aserciones se evalúan a “true” verás que el listener correspondiente no muestra ninguna información. Prueba a cambiar la aserción para que de “fallo”, y podrás observar que el listener muestra el error producido). Verás que los listeners muestran los resultados de la ejecución “a continuación” de los resultados de ejecuciones anteriores. Podemos “limpiar” el panel de resultados pinchando sobre el icono correspondiente. A continuación mostramos el aspecto del plan de pruebas hasta este momento, así como la imagen del icono para “limpiar” los resultados de los listeners de ejecuciones anteriores.



Plan de pruebas hasta este momento



- G) Ahora añadiremos **controladores lógicos** a nuestro plan de pruebas para “alterar” el comportamiento secuencial del plan. Vamos a incluir un “Controlador Interleave” desde el menú contextual del Grupo de Hilos: **Add→Logic Controller→Interleave Controller**. Este controlador “alterna” la ejecución de uno de sus hijos en cada iteración de cada uno de los hilos. Podéis consultar el funcionamiento de este elemento y de cualquier componente de JMeter en (http://jmeter.apache.org/usermanual/component_reference.html). Situaremos el controlador justo después del elemento Catalog, e incluiremos como nodos hijo tres peticiones http. Una de ellas es la que hemos denominado Reptiles. Las otras dos son el acceso a los submenús Dogs, y Birds, respectivamente. Puedes crearlos rápidamente “duplicando” el nodo Reptiles y editando los elementos correspondientes.
- H) A continuación añadiremos las **peticiones http** para “loguearnos” en el sistema (como hijos del grupo de hilos), y las llamaremos “**Sign in form**”, “**Login**” respectivamente. La primera es la petición que realizamos cuando “pinchamos” sobre el enlace “Sign in” en el navegador, y la segunda corresponde con la introducción del login y password y pinchar sobre el enlace “Login” en el navegador Firefox. Estas acciones las tenemos “grabadas” en el controlador Grabación del ejercicio anterior. Identificalas y cópialas (o muévelas) a continuación del controlador Interleave. Seguidamente introducimos una **petición para mostrar el catálogo** (podemos, por ejemplo duplicar el elemento “Catalog” que ya teníamos antes (fíjate que, en el navegador Firefox, después de “pinchar” sobre “Login”, se llama de nuevo a la página que muestra el catálogo general de especies). El nuevo elemento “Catalog” tendrá asociada una **aserción de Respuesta** con dos patrones: el patrón “Sign Out”, y el patrón “Welcome z”. (IMPORTANTE: JMeter es sensible a las mayúsculas/minúsculas, asegúrate de que el patrón coincide exactamente con el mensaje que aparece en la página).
- Añade otra **petición Http** con el nombre “**My Account**” que es la petición que se genera cuando en el navegador “pinchamos” sobre “My Account”. Asociaremos una **aserción de respuesta** con el patrón “User Information”. Finalmente añadimos una última petición para salir de nuestra cuenta (utiliza la petición que tenemos grabada) a la que llamaremos “**Sign off**”, seguida de otra petición “**Catalog**”, esta vez con una aserción de respuesta con el patrón “Sign In”.
- I) Vamos a **borrar el controlador Grabación**, ya que ya no nos hace falta, simplemente lo hemos utilizado para “averiguar” la configuración de las peticiones http que teníamos que realizar. Guardamos el plan y ejecutamos (Todas las peticiones y aserciones correspondientes tiene que aparecen “en verde”. Si no es así, corrige los posibles errores en tu plan). Podemos ver los resultados en los “listeners” Resultados de la Aserción y Ver árbol de resultados. El plan resultante debe tener este aspecto:



- J) Finalmente vamos a **añadir dos listeners más** (desde el nodo raíz de nuestro plan de pruebas): un ***Aggregate Report***, que proporciona un resumen general de la prueba; y un ***Graph Results***, que “dibuja” los tiempos registrados para cada una de las muestras. El informe agregado muestra una tabla con una fila para cada petición, mostrando el tiempo de respuesta, el número de peticiones, ratio de error, ... El gráfico de resultados muestra el rendimiento de la aplicación (throughput) como el número de peticiones por minuto gestionadas por el servidor.
- Vuelve a ejecutar el plan y observa los resultados mostrados por los nuevos listeners que hemos añadido. Reduce al máximo el número de aplicaciones “abiertas” en el ordenador cuando estés ejecutando JMeter, ya que afectarán a los datos obtenidos. En un caso de prueba “real” solamente deberíamos tener en ejecución la aplicación JMeter en una máquina.
- K) Ahora vamos a suponer que está prevista una carga de 25 usuarios concurrentes. Nuestras especificaciones estipulan que con 25 usuarios concurrentes, el tiempo de respuesta debe ser inferior a 1 segundos por página. Cambia los parámetros del grupo de hilos de nuestro plan para validar si nuestra aplicación cumple con los requisitos de rendimiento acordados con el cliente. Explica qué valores necesitas utilizar para el número de hilos, el periodo de subida y el número de iteraciones en el bucle. Ten en cuenta que en cualquier momento podemos parar la ejecución del plan.