Clase #6 - Redirecciones / Objeto Request / Session

...

Redirecciones

Qué son las redirecciones?

Una redirección es una forma de enviar a un usuario o buscadores a una URL diferente a la que estos han especificado en la petición. Los tres códigos de estado mas utilizados en redirecciones son 301 (Movido permanentemente) y 302 (Encontrado / Movido Temporalmente).

Por qué son importantes las Redirecciones?

La razón más importante para utilizar redirecciones es para evitar perder el tráfico obtenido mediante buscadores online (e.g Google, Bing). Esto hace mucho más fácil cambiar de dominios, renovar la imagen de la empresa (Re-Brand), y cambiar URLs a otros más amigables para el usuario (`user friendly`) o con mejores prospectos para buscadores online (Search Engine Optimization - SEO).

301 - Movido Permanentemente

Cada vez que se introduce el código 301 estamos ante una redirección de **tipo permanente**, que se utiliza cuando la URL de una página se ha modificado. Asimismo, esta fórmula también puede resultar útil cuando creamos una nueva página (por ejemplo, un post) con contenido actualizado de otra ya existente. Cuando aplicamos una redirección 301 a una URL que ya está bien posicionada en los buscadores de internet, la próxima vez que el robot (o bot) de Google visite esa página, entenderá que la URL ya está obsoleta y la sustituirá por la nueva URL a la que apunta la redirección.

301 - Sus ventajas

La ventaja primordial del código 301, es que pasa entre el 90-99% del tráfico a tu nueva URL. Esto significa que al utilizar el redireccionamiento 301, se mantiene el tráfico que tenía la URL original y por tanto la viabilidad de la misma. Los buscadores online (Google, Bing) actualiza la URL a la nueva URL especificada por el código 301.

301 - Cuando usarla

- Al momento de hacer un cambio de **dominio**.
- Al momento de cambiar una URL por otra más amigable al usuario o más fácil de leer. Ejemplo, cuando tenemos `/cont` y cambiamos a `/contacto`.
- Al momento de aprovechar el posicionamiento de algún URL antiguo, por ejemplo, cuando un post viejo tiene muy buen posicionamiento en la web (es muy popular) y queremos actualizar su información, manteniendo el flujo de usuarios.

302 - Movido Temporalmente

El código 302 señala una **redirección temporal.** Uno de los rasgos más notables que la diferencian de una redirección 301 es que, en el caso de las redirecciones 302, **no se traspasa fuerza del SEO a la nueva URL.** Dígase, que se espera que en algún punto esta redirección desaparezca (por tanto - temporal) y sea reemplazada por el contenido original ya modificado o arreglado.

302 - Casos de Uso

- Cuando encontramos algún **error** o problema en nuestra URL. Mientras este se trabaja y soluciona, es buena idea hacer una redirección 302 con el objetivo de que el usuario no se encuentre con el problema.
- Si existe un ataque al sitio, este tipo de redirección puede utilizarse mientras se recuperan algunas de las páginas o URLs.

El Objeto Request

Que es?

El objeto Request, o `Request Object` es un objeto que contiene toda la información de la petición realizada por el usuario al servidor. Contiene una amplia gama de funciones y métodos que nos permiten adquirir la información que ha enviado el usuario a nuestras manos,

Métodos (Parte I)

```
request.attributes();              // Retorna la lista de todos los atributos
request.attribute("foo");          // Retorna el valor del atributo `foo`
request.attribute("A", "V");       // Establece el valor del atributo `A` a `V`.
request.body();                    // Retorna el cuerpo del Request como lo envió el usuario
request.bodyAsBytes();             // Retorna el cuerpo del request en bytes.
request.contentLength();           // Longitud del Cuerpo (Body)
request.contentType();             // MimeType de el cuerpo (body) del request.
request.contextPath();
```

Métodos (Parte II)

```
request.cookies();              // Cookies en el Request enviados por el usuario.
request.headers();              // Retorna una lista de cabeceras HTTP
request.headers("BAR");         // Retorna el valor de la cabecera “BAR”
request.host();                 // Retorna el host o domino (ejemplo -> ejemplo.com)
request.ip();                   // Retorna la IP del cliente
request.params("foo");          // Retorna el valor del parametro `foo`
request.params();               // Retorna un mapa de todos los parametros
request.pathInfo();             // Informacion sobre la ruta (path)
request.port();                 // Puerto del servidor.
request.protocol();             // Protocolo en el que corre el servidor, e.g. HTTP/1.1
```

Métodos (Parte III)

```
request.queryMap();               // Un mapa de datos representando el Query
request.queryMap("foo");          // Un mapa de datos representando un parámetro
request.queryParams();            // La lista de parametros del query
request.queryParams("FOO");       // El valor del parámetro `FOO` en el query
request.queryParamsValues("FOO")  // Todos los valores del parametro `FOO`.
request.raw();                    // El request en bruto, así como lo maneja Jetty
request.requestMethod();          // Metodo de acceso HTTP (Get, Post, Put...)
request.scheme();
request.servletPath();            // Ruta para el servlet, e.g. /result.jsp
request.session();                // Retorna el objeto session para ser manipulado
request.splat();                  // Retorna un arreglo de parámetros que representan el parametro wildcard (*)
```

Métodos (Parte IV)

```
request.uri();              // Retorna el URI
request.url();              // Retorna la URL a la que se realizó la petición.
request.userAgent();            // Retorna informacion sobre el cliente o navegador.
```

Session

Que es?

El objeto de Sesión provee una forma de identificar un usuario a lo largo de más de una petición o respuesta del usuario. Hemos de recordar que el protocolo HTTP **no mantiene estados**. Por tanto, se hace necesario una forma de mantener (al menos temporalmente) algunos parámetros, variables o valores que nos sean de utilidad, como, por ejemplo, el usuario que se ha ingresado en nuestra página web.

Para estos valores temporales, hacemos uso de la Sesión.

Métodos de Session en Spark Java

```
request.session(true);                    // Crea y retorna la session creada.
request.session().attribute("user");      // Retorna el atributo `user` de la session/
request.session().attribute("user","foo"); // Establece el valor del atributo `user`.
request.session().removeAttribute("user"); // Remueve el atributo `user`
request.session().attributes();           // Retorna todos los atributos de la session.
request.session().id();                   // Retorna el identificador de la session
request.session().isNew();                // Verifica si la session es nueva.
request.session().raw();                  // Retorna el objeto servlet
```

Bibliografias

- https://www.inboundcycle.com/blog-de-inbound-marketing/que-son-las-redirecciones-301-y-302-y-como-configurarlas
- https://www.inboundcycle.com/blog-de-inbound-marketing/que-son-las-redirecciones-301-y-302-y-como-configurarlas
- https://docs.oracle.com/javaee/5/api/javax/servlet/http/HttpSession.html
- http://sparkjava.com/documentation#session
- http://sparkjava.com/documentation#request