Responses en Laravel 5

Qué son las responses, uno de los elementos fundamentales de Laravel 5 y algunos ejemplos de uso.

Las responses son las "respuestas" que nos debe devolver Laravel ante cualquier solicitud que se realice al servidor. Según la documentación oficial "Cualquier ruta [del sistema de routing] y controlador debe devolver algún tipo de respuesta al navegador del usuario". Existen diversos modos de devolver respuestas y en este artículo analizaremos algunos de ellos.

El sistema de Response depende de la clase Illuminate\Http\Response aunque muchas veces nos saltamos realmente el uso de esa clase y simplemente le dejamos a Laravel que la use internamente. Por ejemplo cuando escribimos salida desde el sistema de routing:

```
Route::get('ruta/de/ejemplo',function(){
        return"Respuesta desde sistema de routing.";
});
```

En este caso, como decimos, internamente Laravel recibirá esa cadena de respuesta la convertirá en un HTTP response para enviarla al cliente.

Cuando devolvemos una vista, ya sea desde un controlador o un closure en el sistema de routing también se pone en marcha el sistema de response de Laravel sin que el programador necesite usarlo directamente.

```
Route::get('cualquier/ruta',function(){
returnview('una_vista');
});
```

¿Entonces se puede abordar directamente el sistema de Response de Laravel y en ese caso, qué utilidad tiene? Efectivamente, nosotros podemos usar directamente una instancia de Response y ello nos permite personalizar todavía más elementos de la respuesta HTTP, enviando códigos de status particulares o cabeceras de HTTP.

Nota: La clase Response de Laravel hereda directamente de la clase Symfony\Component\HttpFoundation\Response que ya provee una buena cantidad de métodos para construir respuestas HTTP. En este caso nos pueden interesar las documentaciones de Response de Laravely por supuesto de Response de Symfony.

Ejemplo de respuesta usando una instancia de Response

A través del helperresponse() podemos crear fácilmente una instancia del objeto Response con el que podemos hacer varios ejemplos de uso de personalización de la respuesta.

En este primer ejemplo tenemos una respuesta exactamente igual a la que conseguiríamos con un simple return 'Hola respuesta'.

```
Route::get('respuesta',function(){
        returnresponse('Hola respuesta',200);
});
```

El helperresponse(), como puedes ver, recibe dos parámetros:

1. Contenido de la respuesta
2. Código de status

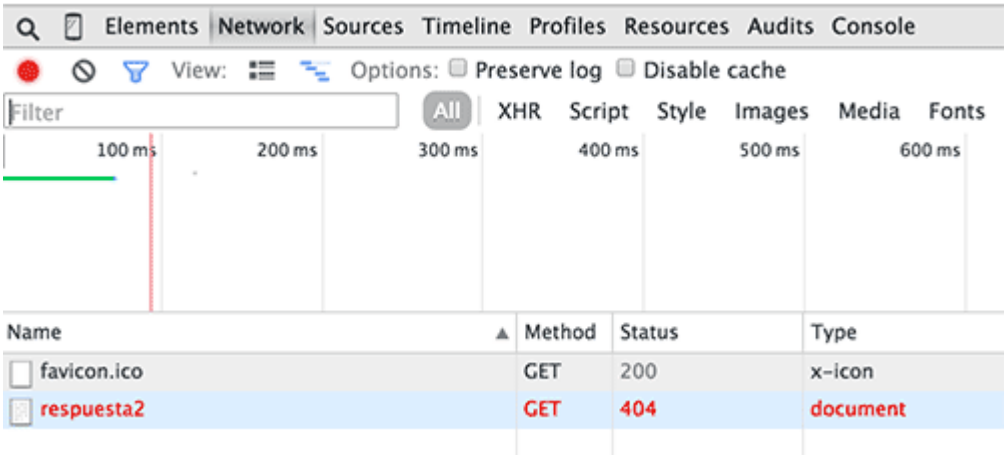
El contenido es el texto que devuelve como respuesta la solicitud HTTP. El código de status son los típicos que debes de conocer del protocolo HTTP. 200 significa "todo correcto".

Después de la invocación al helperresponse() recibimos como valor de devolución un objeto de la clase Response. Éste objeto es el que devuelve el closure y que Laravel toma para componer la respuesta HTTP.

En este segundo ejemplo devolvemos un error de página no encontrada, status 404, con contenido 'Esto es un error'.

```
Route::get('respuesta2',function(){
        returnresponse('Esto es un error',404);
});
```

Nota: Los códigos de status los puedes ver en las herramientas para desarrolladores de tu navegador de preferencia. En Chrome por ejemplo los podrás apreciar en la sección "Network" de las developertools, en la columna "Status".



Si quieres conocer otros códigos de respuesta puedes consultar este artículo de la MDN: Response codes.

En este ejemplo realizamos algo también simple pero más elaborado, ya que enviamos una respuesta a la que le agregamos una cabecera adicional para especificar que el contenido es una hoja de cálculo, archivo CSV.

```
Route::get('respuesta3',function(){
        returnresponse("1,2,3,4\n5,6,7,8",200)
                ->header('Content-Type','text/csv');
});
```

Como decíamos, el helperresponse() devuelve un objeto de la clase Response y sobre él estamos encadenando la invocación de su método header(), que sirve para agregar nuevas cabeceras a la respuesta HTTP. En este caso el nombre de la cabecera es 'Content-Type' y el valor que estamos aplicando es 'text/csv'.

En el ejemplo siguiente probamos otra cabecera diferente, para realizar una redirección. Tómalo simplemente como un test, puesto que existe otro método más rápido y sencillo para enviar responses de tipo redirect.

```
Route::get('respuesta4',function(){
        returnresponse("",301)
                ->header('location','http://desarrolloweb.com');
});
```

Nota: Existe un helper llamado redirect() que forma parte de los componentes del sistema de response de Laravel y está pensado para hacer ese trabajo, aportando diversos tipos de redirecciones posibles. Un ejemplo podría ser este:

```
returnredirect('/uri/de/redireccion');
```

En esta ocasión lo usamos para redirigir a otra URI del mismo sitio web, pero veremos otros ejemplos en futuros artículos.

Podemos encadenar varias llamadas al método header() si quisiéramos agregar varias cabeceras diferentes a la respuesta. como vemos en el siguiente ejemplo:

```
Route::get('respuesta5',function(){
        returnresponse("Esta página se refrescará en 5 segundos
hacia...",200)
                ->header('Cache-Control','max-age=3600')
                ->header('Refresh','5; url=http://www.desarrolloweb.com');
});
```

Hemos puesto dos cabeceras de HTTP response, la primera para definir la antigüedad máxima del elemento cacheado y la segunda para decir que en 5 segundos se refresque la página enviando al navegador a una nueva URL.

Enviar una vista mediante una instancia Response

Habrás advertido que resulta especialmente incómodo enviar como respuesta una cadena en el primer parámetro del helperresponse(). Si la respuesta es muy larga para colocarla, así tal cual, en una cadena, o simplemente prefieres separar la salida en una vista, tal como has aprendido, tenemos una alternativa muy útil.

Se trata de invocar al helper de response() sin enviar ningún parámetro. La respuesta (objeto Response) estará limpia para configurar todos sus detalles. Uno de los métodos del objeto Response que te devuelve el helperresponse() es view(), en el que indicamos la vista que queremos procesar.

Luego podemos indicar nuevas cabeceras HTTP en la respuesta, encadenando métodos, tal como hemos visto en ejemplos anteriores en este artículo.

```
Route::get('respuesta6',function(){
        returnresponse()
                ->view('error')
                ->header('status',404)
                ->header('Refresh','5; url=/');
});
```

En el código anterior tendrías un ejemplo de response en la que cargamos una vista llamada "error" y luego enviamos dos cabeceras adicionales, una para mandar un código de status (404 de error "página no encontrada") y una redirección pasados 5 segundos a la home del dominio.

Conclusión

De momento dejamos por aquí esta introducción al sistema de Response de Laravel, hay bastante más que ver en los siguientes artículos, como responder en formato JSON, generar cookies, enviar archivos para descarga, etc. pero como nuestro objetivo de momento es la presentación de los componentes principales del framework, es más que suficiente.