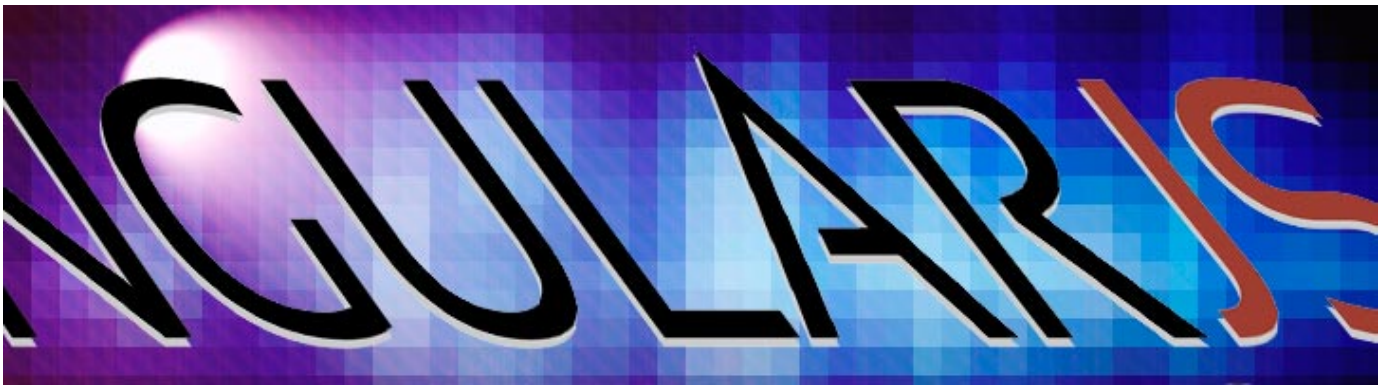
Operación post HTTP con $http en AngularJS

Cómo realizar una operación POST con Ajax, para enviar datos al servidor por medio del HTTP con la librería AngularJS y el service $http.

Hemos visto cómo realizar operaciones básicas de Ajax con AngularJS, en las que recibimos información del servidor, por medio de get. Ahora veamos también cómo se enviarían datos al servidor por medio de la operación post del HTTP. Como siempre las cosas con AngularJS se hacen muy sencillas.

El servicio ("service" en la terminología de Angular) $http nos ofrece una serie de métodos "shortcut" enfocados en realizar las operaciones típicas implementadas dentro del protocolo HTTP. Para enviar datos post disponemos de $http.post(). En ese método podemos enviar como parámetro, aparte de la URL del servidor donde haremos el post, un objeto con los datos que se desean enviar.



```
$http.post("recibe.php", {uno: 1, fruta: "manzana"});
```

Como ves, la primera dirección es la URL relativa donde enviar los datos (también podría ser absoluta) y el segundo parámetro son los datos que queremos enviar al destino.

Este método devuelve como resultado un objeto sobre el que podemos implementar con el patrón promesa algunas operaciones en diversas situaciones con respecto a esa conexión con el servidor, algo que ya viste en lo ejemplos de Ajax anteriores. Lo que vamos a aprender de momento es realizar acciones en caso de éxito y para ello tenemos que indicarlas con una función en "success".

```
var conAjax = $http.post("recibe.php", {uno: 1, fruta: "manzana"});
conAjax.success(function(respuesta){
     console.log(respuesta);
});
```

En la función que asociamos al caso success, como ya habrás visto muchas veces, recibimos un parámetro con la respuesta que nos devuelve el servidor. En este caso simplemente la volcamos a la consola de Javascript.

Nota: Este código lo encontrarás habitualmente encadenando llamadas, sin necesidad de declarar la variable "conAjax". Luego haremos ejemplos que usen esa común forma de codificar.

Recibir los datos en el servidor

Los datos que estás enviando por post Angular los empaqueta como JSON y te llegarán al servidor, aunque no por el método común POST, en pares clave/valor como quizás estás acostumbrado. En realidad nos lo envía como content-type, "application/json" en un único objeto, lo que es útil porque nos acepta datos más complejos, con anidación de objetos JSON.

Nota: Para que nos entendamos, en un lenguaje como PHP, cuando enviamos datos por post los recoges con el array $_POST. Ese array no aceptaría diversos niveles de anidación para recibir datos complejos, osea, podemos tener claves con valores simples. Pero se nos queda un poco corto para aplicaciones modernas. En PHP si accedes a $_POST encontrarás que el array está vacío.
Este asunto de los niveles de anidación en los datos que se envían con JSON se ve por ejemplo en este código, perfectamente válido y habitual.

```
$http.post("recibe2.php", {
            nombre: "Miguel",
            fechaNacimiento: "21/02/1975",
            sitiosPreferidos: [
                "DesarrolloWeb.com",
                "Guiarte.com"
            ],
            direccion: {
                calle: "De la alegría",
                numero: 18,
                ciudad: "Villadigital"
            }
        })
            .success(function(respuesta){
                console.log(respuesta);
            });
```

Ese objeto complejo JSON lo recibirás en el servidor tal cual. El código para recogerlo dependerá de tu lenguaje de servidor. Por ejemplo en PHP lo haría de esta manera:

```
file_get_contents("php://input")
```

Eso nos devolvería una cadena de texto, que si quieres volcar a un objeto nativo de PHP usarás la función json_decode(). Tu código podría quedarte parecido a este:

```
$objDatos = json_decode(file_get_contents("php://input"));
```

A partir de ese momento encontrarás que puedes acceder a los datos del JSON recibido como estás acostumbrado en el trabajo con objetos PHP.

```
echo $objDatos->nombre;
echo $objDatos->sitiosPreferidos[0];
echo $objDatos->direccion->calle;
```

Enviar un formulario por POST

Si lo que quieres es enviar los datos que se encuentran en un formulario por POST al servidor, para recibirlos en un JSON, la verdad es que el procedimiento es bien parecido a lo que has visto. Simplemente tendremos que crear nuestro modelo con los datos del formulario, algo que hace Angular por ti agregando la directiva ngModel, y enviarlo por $http.post().

Echa un vistazo a este formulario:

```
<div ng-app="app" ng-controller="appCtrl as vm">
  <h1>Pruebo Ajax</h1>
  <section>
      <form ng-submit="vm.enviar()">
        Nombre: <input type="text" ng-model="vm.fdatos.nombre">
        <br>
        Edad: <input type="text" ng-model="vm.fdatos.edad">
        <br>
        <input type="submit" value="Enviar">
      </form>
  </section>
</div>
```

Observa que en ng-model hemos volcado los campos del formulario dentro de un objeto llamado "fdatos". Osea, en el modelo de la vista "vm" tenemos un objeto "datosf" y dentro ya encontraremos los datos de nuestro formulario.

También repara en la directiva ngSubmit que hemos colocado en la etiqueta FORM. Verás que la hemos asociado con una función de nuestro modelo: vm.enviar(). En esa función que veremos a continuación es donde debes escribir el código para poder enviar el formulario.

Nota: Esto no estás obligado a hacerlo así necesariamente, porque puedes crear la estructura que desees en el modelo. Simplemente nos resultará más cómodo colocar directamente los datos vinculados a un objeto de datos del formulario. Al servidor por post no vamos a enviar todo el modelo de la vista, sino únicamente los datos del formulario. Como esta estructura ya nos permite tener los datos del formulario en un objeto independiente, nos ahorrará el tener que hacerlo "a mano". Enseguida lo verás mejor.
Para que esto funcione en tu Javascript debes inicializar "fdatos" como un objeto, aunque sea vacío. Por ello en tu controller deberías incluir este código.

```
vm.fdatos = {};
```

Es un código parcial, luego lo verás en el contexto del controlador completo. Observa que en tu "vm" has inicializado el objeto "fdatos" con un literal de objeto vacío, expresado con las dos llaves.

Esto ya nos deja en muy buena situación para enviar ese formulario de una manera muy limpia. Ahora te mostramos el código completo para crear nuestro controlador.

```
angular
        .module('app', [])
        .controller('appCtrl', ['$http', controladorPrincipal]);

    function controladorPrincipal($http){
        var vm=this;

        //inicializo un objeto en los datos de formulario
        vm.fdatos = {};

        // declaro la función enviar
        vm.enviar = function(){
          $http.post("recibe-formulario.php", vm.fdatos)
            .success(function(res){
              console.log(res);
              //por supuesto podrás volcar la respuesta al modelo con algo como vm.res
= res;
            });
        }
    }
```

Con esto creo que lo tendrás todo claro para poder enviar datos a un servidor por medio de post, datos que podrán ser todo lo complejos que necesites en tu aplicación. Como puedes comprobar la llamada a $http.post() te hace todo el trabajo de la solicitud HTTP por medio de Ajax, por lo que solo tendrás que programar el comportamiento específico para tus necesidades, tanto en el cliente como en el servidor.