

DESARROLLO WEB

2017-I



SEMANA 11

ANGULARJS – INTEGRACIÓN CON PHP

Eric Gustavo Coronel Castillo

ecoronel@uch.edu.pe - gcoronelc.blogspot.com

ANGULARJS – INTEGRACION CON PHP

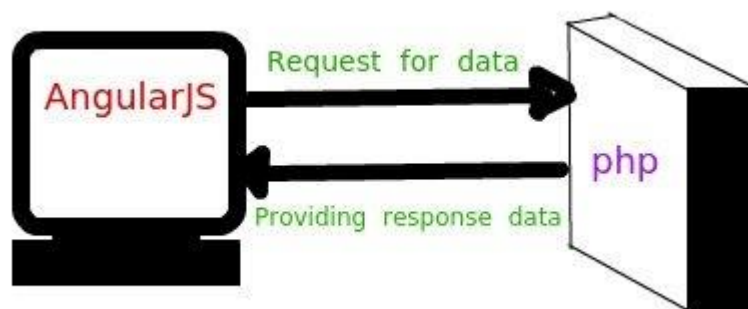
El objetivo de esta semana es que integres AngularJS con PHP utilizando JSON.

ÍNDICE

JSON EN PHP	5
PHP: JSON_ENCODE()	5
EJEMPLO 1	5
EJEMPLO 2	6
EJEMPLO 3	7
EJEMPLO 4	8
PROCESAR PARAMETROS JSON.....	10
PHP: JSON_DECODE()	10
EJEMPLO 5	10
<i>Ejecución 1: Pasar un arreglo JSON de números</i>	<i>11</i>
<i>Ejecución 2: Pasar un objeto JSON</i>	<i>12</i>
<i>Ejecución 3: Pasar un objeto JSON</i>	<i>13</i>
EJEMPLO 6	14
<i>Ejecución 1: Pasar un arreglo JSON de números</i>	<i>15</i>
<i>Ejecución 2: Pasar un objeto JSON</i>	<i>16</i>
<i>Ejecución 3: Pasar un objeto JSON</i>	<i>17</i>
PHP: PHP://INPUT	18
EJEMPLO 7	18
<i>Prueba 01</i>	<i>20</i>
<i>Prueba 02</i>	<i>21</i>
PROYECTO EJEMPLO.....	22
ESTRUCTURA DEL PROYECTO:	22
PROGRAMA: INDEX.HTML	23
PROGRAMA: APPCONTROLLER.PHP	24

PROGRAMA: APP.JS.....	24
PRUEBA.....	25

JSON EN PHP



Una de las ventajas que tiene PHP, es que tiene integrada funciones para convertir arreglo y objetos a formato JSON y viceversa, haciendo de esta manera su integración con AngularJS muy fácil.

PHP: `json_encode()`

Esta función retorna la representación JSON del dato que se le pasa como parámetro.

Ejemplo 1

En este ejemplo se ilustra cómo pasa un arreglo simple a un arreglo JSON.

```
<?php  
  
$datos = array(20, 30, 15, 50);  
  
var_dump($datos);  
  
$json = json_encode($datos);  
  
var_dump($json);  
  
?>
```

El resultado es el siguiente:

```
array (size=4)
  0 => int 20
  1 => int 30
  2 => int 15
  3 => int 50

string '[20,30,15,50]' (length=13)
```

Ejemplo 2

En este ejemplo se aprecia cómo se para un arreglo asociativo a formato JSON.

```
<?php

$datos = array(
    'codigo'=>"123",
    'nombre'=>"TV",
    'precio'=>4500.00,
    'stock'=>500);

var_dump($datos);

$json = json_encode($datos);

var_dump($json);

?>
```

El resultado es el siguiente:

```
array (size=4)
  'codigo' => string '123' (length=3)
  'nombre' => string 'TV' (length=2)
  'precio' => float 4500
  'stock' => int 500

string '{"codigo":"123","nombre":"TV","precio":4500,"stock":500}' (length=56)
```

Ejemplo 3

En este ejemplo se ilustra como pasar un objeto a formato JSON.

```
<?php

$datos = (object) array(
    'codigo'=>"123",
    'nombre'=>"TV",
    'precio'=>4500.00,
    'stock'=>500);

var_dump($datos);

$json = json_encode($datos);

var_dump($json);

?>
```

El resultado es el siguiente:

```
object(stdClass) [1]
  public 'codigo' => string '123' (length=3)
  public 'nombre' => string 'TV' (length=2)
  public 'precio' => float 4500
  public 'stock' => int 500

string '{"codigo":"123","nombre":"TV","precio":4500,"stock":500}' (length=56)
```

Ejemplo 4

En este ejemplo se tiene un arreglo a arreglos asociativos, y se puede apreciar cómo se convierte a formato JSON.

```
<?php

$datos = array(
    array('codigo'=>"P01",'nombre'=>"Gustavo Coronel"),
    array('codigo'=>"P02",'nombre'=>"Ricardo Marcelo"),
    array('codigo'=>"P03",'nombre'=>"Claudia Ramirez"),
    array('codigo'=>"P04",'nombre'=>"Alejandra Ramos")
);

var_dump($datos);

$json = json_encode($datos);

var_dump($json);

?>
```

El resultado es el siguiente:

```
array (size=4)
  0 =>
    array (size=2)
      'codigo' => string 'P01' (length=3)
      'nombre' => string 'Gustavo Coronel' (length=15)
  1 =>
    array (size=2)
      'codigo' => string 'P02' (length=3)
      'nombre' => string 'Ricardo Marcelo' (length=15)
  2 =>
    array (size=2)
      'codigo' => string 'P03' (length=3)
      'nombre' => string 'Claudia Ramirez' (length=15)
  3 =>
    array (size=2)
      'codigo' => string 'P04' (length=3)
      'nombre' => string 'Alejandra Ramos' (length=15)

string '[{"codigo":"P01","nombre":"Gustavo Coronel"},
{"codigo":"P02","nombre":"Ricardo Marcelo"},
{"codigo":"P03","nombre":"Claudia Ramirez"},
{"codigo":"P04","nombre":"Alejandra Ramos"}
]' (length=177)
```


PROCESAR PARAMETROS JSON

PHP: json_decode()

Convierte una cadena codificado en JSON a su equivalente en PHP.

Ejemplo 5

En este ejemplo se tiene un programa en PHP que recibe un dato JSON y lo convierte a un elemento PHP.

```
<?php

$json = $_GET["json"];

echo "<h2>CADENA RECIBIDA</h2>";
var_dump($json);

$datos = json_decode($json);

echo "<h2>EQUIVALENTE EN PHP</h2>";
var_dump($datos);

?>
```

Ejecución 1: Pasar un arreglo JSON de números

La llamada podría ser así:

```
demo05.php?json=[100,400,432,567,234]5}
```

El resultado es el siguiente:

CADENA RECIBIDA

```
string '[100,400,432,567,234]' (length=21)
```

EQUIVALENTE EN PHP

```
array (size=5)  
  0 => int 100  
  1 => int 400  
  2 => int 432  
  3 => int 567  
  4 => int 234
```

En este caso un arreglo JSON se convierte en un arreglo PHP.

Ejecución 2: Pasar un objeto JSON

La llamada podría ser así:

```
demo05.php?json={"n1":"15","n2":"18","n3":"16","n4":"18"}
```

El resultado es el siguiente:

CADENA RECIBIDA

```
string '{"n1":"15","n2":"18","n3":"16","n4":"18"}' (length=41)
```

EQUIVALENTE EN PHP

```
object(stdClass) [1]  
  public 'n1' => string '15' (length=2)  
  public 'n2' => string '18' (length=2)  
  public 'n3' => string '16' (length=2)  
  public 'n4' => string '18' (length=2)
```

En este caso, por defecto la función **json_decode()** lo convierte en un objeto PHP.

Ejecución 3: Pasar un objeto JSON

Esta ejecución es similar a la anterior, la diferencia está en que los datos son numéricos:

```
demo05.php?json={"n1":15,"n2":18,"n3":16,"n4":18}
```

El resultado es el siguiente:

CADENA RECIBIDA

```
string '{"n1":15,"n2":18,"n3":16,"n4":18}' (length=33)
```

EQUIVALENTE EN PHP

```
object(stdClass) [1]  
  public 'n1' => int 15  
  public 'n2' => int 18  
  public 'n3' => int 16  
  public 'n4' => int 18
```

En este caso, la función **json_decode()** lo convierte en un objeto PHP, pero los campos son numéricos.

Ejemplo 6

En este ejemplo se tiene un programa en PHP que recibe un dato JSON y lo convierte a arreglo PHP.

```
<?php

$json = $_GET["json"];

echo "<h2>CADENA RECIBIDA</h2>";
var_dump($json);

$datos = json_decode($json);

echo "<h2>EQUIVALENTE EN PHP</h2>";
var_dump($datos);

?>
```

Ejecución 1: Pasar un arreglo JSON de números

La llamada podría ser así:

```
demo06.php?json=[100,400,432,567,234]5}
```

El resultado es el siguiente:

CADENA RECIBIDA

```
string '[100,400,432,567,234]' (length=21)
```

EQUIVALENTE EN PHP

```
array (size=5)
  0 => int 100
  1 => int 400
  2 => int 432
  3 => int 567
  4 => int 234
```

En este caso un arreglo JSON se convierte en un arreglo PHP.

Ejecución 2: Pasar un objeto JSON

La llamada podría ser así:

```
demo06.php?json={"n1":"15","n2":"18","n3":"16","n4":"18"}
```

El resultado es el siguiente:

CADENA RECIBIDA

```
string '{"n1":"15","n2":"18","n3":"16","n4":"18"}' (length=41)
```

EQUIVALENTE EN PHP

```
array (size=4)
  'n1' => string '15' (length=2)
  'n2' => string '18' (length=2)
  'n3' => string '16' (length=2)
  'n4' => string '18' (length=2)
```

En este caso, la función **json_decode()** está convirtiendo un objeto JSON en un arreglo asociativo de PHP.

Ejecución 3: Pasar un objeto JSON

Esta ejecución es similar a la anterior, la diferencia está en que los datos son numéricos:

```
demo06.php?json={"n1":15,"n2":18,"n3":16,"n4":18}
```

El resultado es el siguiente:

CADENA RECIBIDA

```
string '{"n1":15,"n2":18,"n3":16,"n4":18}' (length=33)
```

EQUIVALENTE EN PHP

```
array (size=4)
  'n1' => int 15
  'n2' => int 18
  'n3' => int 16
  'n4' => int 18
```

En este caso, la función **json_decode()** está convirtiendo un objeto JSON en un arreglo asociativo de PHP.

PHP: php://input

Se trata de un flujo de sólo lectura que te permite leer los datos enviados por medio de una solicitud de tipo POST.

En caso de enviar datos en formato JSON, se debería procesar de la siguiente manera:

```
$postdata = file_get_contents("php://input");  
$data = json_decode($postdata, true);
```

En la primera línea se recibe los datos enviados en formato JSON, y en la segunda línea se decodifica a un arreglo asociativo.

Ejemplo 7

En este procederá a crear un archivo HTML que envía una solicitud en formato JSON con JQuery.

```
<!DOCTYPE html>  
<html>  
<head>  
  <meta charset="utf-8">  
  <title>DEMO</title>  
  <script src="jquery.min.js"></script>  
</head>  
<body>  
  <h1>DEMO</h1>  
  <label>Texto JSON</label><br/>  
  <textarea name="" id="json" cols="50" rows="3">textarea>  
  <button type="button" id="boton">Procesar</button><br/>  
  <h2>Respuesta</h2>  
  <div id="rpta"></div>  
  <script>  
    $("#boton").click(function(){  
      var json = $("#json").val();  
      $.post("demo07.php",json,function(respuesta){  
        $("#rpta").html(respuesta);  
      });  
    });
```

```
</script>
</body>
</html>
```

Este formulario se muestra de la siguiente manera:

DEMO

Texto JSON

Procesar

Respuesta

En el campos **Texto JSON** debes ingresar un datos en formato JSON, y en al área de respuesta obtendrás como los interpreta PHP.

El programa PHP es el siguiente:

```
<?php

$postdata = file_get_contents("php://input");

echo "<h2>DATOS RECIBIDOS</h2>";
var_dump($postdata);

$data = json_decode($postdata, true);

echo "<h2>EQUIVALENTE EN PHP</h2>";
var_dump($data);

?>
```

Prueba 01

Enviar un arreglo JSON.

DEMO

Texto JSON

Procesar

Respuesta

DATOS RECIBIDOS

```
string '[230,657,234,7650]' (length=18)
```

EQUIVALENTE EN PHP

```
array (size=4)
  0 => int 230
  1 => int 657
  2 => int 234
  3 => int 7650
```

Como puedes apreciar en la imagen, si envías un arreglo JSON, PHP lo convierte en arreglo PHP.

Prueba 02

Enviar un objeto JSON:

DEMO

Texto JSON

```
{ "codigo": "P7865", "nombre": "Televisor", "precio": 2345.68, "stock": 500 }
```

Respuesta

DATOS RECIBIDOS

```
string '{"codigo":"P7865","nombre":"Televisor",  
      "precio":2345.68,"stock":500}' (length=68)
```

EQUIVALENTE EN PHP

```
array (size=4)  
  'codigo' => string 'P7865' (length=5)  
  'nombre' => string 'Televisor' (length=9)  
  'precio' => float 2345.68  
  'stock' => int 500
```

Como puedes apreciar en la imagen, un objeto JSON, PHP lo convierte en un arreglo asociativo de PHP.

PROYECTO EJEMPLO

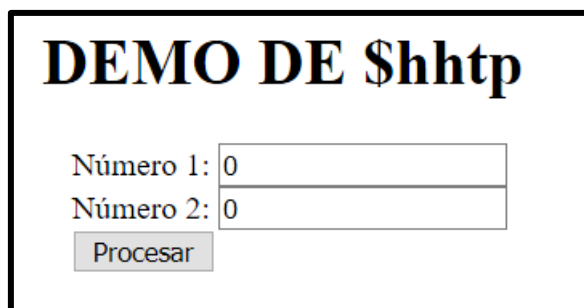
En este caso veras un proyecto ejemplo ilustrativo.

Para ilustrar la integración con AngularJS, se desarrollara un programa que sume dos números.

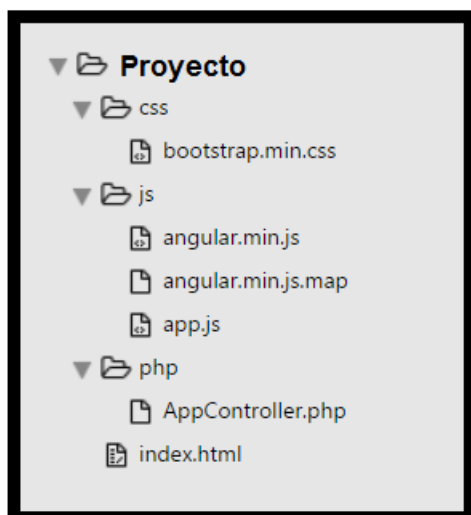
Para este proyecto se está utilizando **AngularJS** versión **1.6.4**, este dato es muy importante.

También se estará utilizando el servicio **\$http** para realizar llamadas AJAX.

El formulario inicial se mostrará de la siguiente manera:



Estructura del proyecto:



En este proyecto debes construir los siguientes programas.

Programa: index.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>Servicios - $http</title>
  <script src="js/angular.min.js"></script>
  <script src="js/app.js"></script>
</head>
<body ng-app='appsuma' ng-controller='AppController'>
  <div>
    <h1>DEMO DE $http</h1>
    <div style="margin: 15px;">
      <form ng-submit="procesar()">
        <div>
          <label>Número 1:</label>
          <input type="text" ng-model="datos.num1" >
        </div>
        <div>
          <label>Número 2:</label>
          <input type="text" ng-model="datos.num2">
        </div>
        <div>
          <input type="submit" value="Procesar">
        </div>
      </form>
    </div>
    <div style="margin: 15px;" ng-show="mostrar">
      <h2>Reporte</h2>
      <table>
        <tr>
          <td>Número 1:</td>
          <td>{{repo.num1}}</td>
        </tr>
        <tr>
          <td>Número 2:</td>
          <td>{{repo.num2}}</td>
        </tr>
      </table>
    </div>
  </div>
</body>
</html>
```

```
<tr>
  <td>Suma:</td>
  <td>{{repo.suma}}</td>
</tr>
</table>
</div>
</div>
</body>
</html>
```

Programa: ApplicationController.php

```
<?php

// Datos
$postdata = file_get_contents("php://input");
$data = json_decode($postdata, true);

// Proceso
$data["suma"] = $data["num1"] + $data["num2"];

// Reporte
print_r(json_encode($data));

?>
```

Programa: App.js

```
var app = angular.module('appsuma', []);

app.controller('AppController', function($scope, $http){

  $scope.mostrar = false;
  $scope.datos = {"num1":0, "num2":0};
  $scope.repo = {};
```

```
$scope.procesar = function(){  
  
    $http.post('php/AppController.php', $scope.datos)  
    .then(function(response){  
        console.log(response.data);  
        $scope.repo = response.data;  
        $scope.mostrar = true;  
    });  
  
}  
  
})
```

Prueba

Como puedes apreciar en la imagen de la prueba realizada, se envía dos número en formato JSON y se recibe otro JSON con el resultado.

DEMO DE \$hhttp

Número 1:	<input type="text" value="78"/>
Número 2:	<input type="text" value="45"/>

Procesar

Reporte

Número 1:	78
Número 2:	45
Suma:	123

Herramientas de desarrollo - Servicios - \$http - http://localhost/AngularJS/Seccion093/

Inspe... Cons... Depura... Editor de es... Rendimi... Mem...

Todos HTML CSS JS XHR Tipografía Imágenes Medios Flash WS Otros Una solicitud, 0.04 KB, 0.00 s Filtrar las UF

Estado	Método	Archivo	Cabeceras	Cookies	Parámetros	Respuesta	Tiempos	Vista preliminar
200	POST	AppController.php	Filtrar los parámetros de la petición JSON num1: "78" num2: "45"					

Red CSS JS Seguridad Registro Servidor Filtrar salida

Object { num1: "78", num2: "45", suma: 123 } app.js:13:9

Herramientas de desarrollo - Servicios - \$http - http://localhost/AngularJS/Seccion093/

Inspe... Cons... Depura... Editor de es... Rendimi... Mem...

Todos HTML CSS JS XHR Tipografía Imágenes Medios Flash WS Otros Una solicitud, 0.04 KB, 0.00 s Filtrar las UF

Estado	Método	Archivo	Cabeceras	Cookies	Parámetros	Respuesta	Tiempos	Vista preliminar
200	POST	AppController.php	Filtrar propiedades JSON num1: "78" num2: "45" suma: 123					

Red CSS JS Seguridad Registro Servidor Filtrar salida

Object { num1: "78", num2: "45", suma: 123 } app.js:13:9