

Desarrollo web en entorno servidor 2023-24 (DAW-DUAL-A)

[Taboleiro](#) / [Os meus cursos](#) / [DWCS-23-24-DUAL-A](#) / UD7: Generación dinámica de páginas web con AJAX / [AJAX](#)

AJAX



1. AJAX ¿Qué es?

1.4. Ejemplo básico de una SPA

Vamos a ver un ejemplo de cómo podríamos imitar el funcionamiento de una SPA con parte de la infraestructura que tenemos organizada en MVC:

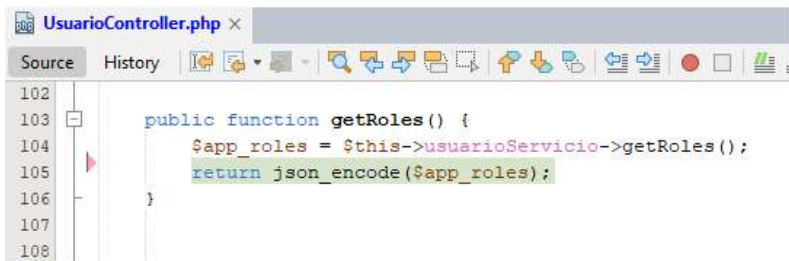
Está disponible en la URL: https://github.com/dudwcs/UD7_Ejemplo_SPA.git

El **FrontController.php** se ha modificado para que en lugar de crear una página HTML completa con header + mainView + footer, sirva una página **spa_view.php** si no recibe parámetros controller ni action.

Si los recibe, llamará al método correspondiente, pero indicando en la cabecera que la respuesta esta vez será en formato JSON y se escribirá directamente en la salida:

```
37
38 //Se preparan los datos para que estén disponibles en la vista
39 $dataToView["data"] = array();
40
41 /* Check if method is defined */
42 if (method_exists($controller, $GET["action"])) {
43     $allowed = AuthorizationManager::isUserAuthorized($controllerName, $GET["action"]);
44     if ($allowed) {
45         //Se llama a la action
46         $dataToView["data"] = $controller->{$GET["action"]}();
47         SessionManager::updateLastAccess();
48
49         ob_clean();
50         //https://www.php.net/manual/en/function.header.php
51         header("Content-Type: application/json; charset=utf-8", true);
52         echo $dataToView["data"];
53     }
54 } else {
55     header("Content-Type: text/html; charset=utf-8", true);
56     require_once dirname(__FILE__, 2) . DIRECTORY_SEPARATOR . 'view/template/spa_view.php';
57 }
58 } else {
59     header("Content-Type: text/html; charset=utf-8", true);
60     require_once dirname(__FILE__, 2) . DIRECTORY_SEPARATOR . 'view/template/spa_view.php';
61 }
62
63 ob_end_flush();
```

En los métodos de cada controlador devolveremos los datos en formato JSON con la función `json_encode` de PHP:



```

102
103 public function getRoles() {
104     $app_roles = $this->usuarioServicio->getRoles();
105     return json_encode($app_roles);
106 }
107
108

```

spa_view.php es una página HTML con los siguientes apartados destacados:

- header
- section login: se mostrará al inicio
- main: estará inicialmente oculta y se irá rellenando con los datos que corresponda en función de la interacción con el usuario.
- scripts que realizarán las llamadas al servidor:
 - global.js: las variables globales y la función onceLoaded() que establecerá los escuchadores de eventos de la sección login
 - manejarSesion.js: las funciones para iniciar y cerrar sesión (de usuario) y sus funciones auxiliares
 - cargarDatos.js con las funciones que permitirán obtener datos del servidor y sus funciones auxiliares



```

2 <html>
3 <head>
4     <meta charset="utf-8">
5     <title id="headTitle">Login</title>
6     <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css">
7
8     <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.min.js" integrity="sha384-11e-c92467J5X6/R4n0R02884c53u8356A5j8p1bk260077f668482e8487">
9 </head>
10 <body>
11
12 <section class="container-fluid" id="spa">
13     <...15 lines /> <header>...</header>
14
15     <...36 lines /> <section id="login">...</section>
16
17     <!-- d-none => display:none https://getbootstrap.com/docs/5.1/utilities/display/ -->
18     <...4 lines /> <main id="main">...</main>
19
20 </section>
21
22 <script src=".../js/sesion.js" type="text/javascript"></script>
23 <script src=".../js/global.js" type="text/javascript"></script>
24 <script src=".../js/cargarDatos.js" type="text/javascript"></script>
25 <script src=".../js/manejarSesion.js" type="text/javascript"></script>
26 </script>
27 </body>
28 </html>

```

En global.js, además de establecer las funciones de algunos eventos se llama a getRoles(), definido en cargarDatos.js.

getRoles() realiza una petición GET con el API Fetch a UsuarioController para obtener los roles en formato JSON. Cuando los recibe, crea los elementos <option> que se mostrarán en la lista desplegable del apartado de login.

La función login() de manejarSesion.js simula una petición POST al método login de UsuarioController, también con el API Fetch. Para que los datos enviados estén disponible en \$_POST, es necesario enviarlos con FormData().

```

let login_url = "?controller=Usuario&action=login";

//preparamos los datos que se enviarían al servidor como si se enviasen por POST desde el formulario
const data = new FormData();
data.append('email', email);
data.append('pwd', pwd);
data.append('rol', rol);

const request = new Request(base_url + login_url, {
  method: "POST",
  body: data
});

```

En el servidor ya no tiene sentido utilizar redirecciones, pues causarían la recarga de toda la página, por lo que se perdería el objeto de una SPA.

Si hay algún problema, devolveremos un JSON {error: true}. En caso contrario devolveremos un objeto { userId: 1, email: "user1@edu.es" }

```

29  // references | overrides
30  public function login()
31  {
32      //Para simplificar la implementación del ejemplo de SPA vamos a obviar la redirección en caso de q
33
34      $this->page_title = 'Inicio de sesión';
35      $this->view = self::VIEW_FOLDER . DIRECTORY_SEPARATOR . 'login';
36
37      if (isset($_POST["email"]) && isset($_POST["pwd"]) && isset($_POST["rol"])) {
38          $email = $_POST["email"];
39          $pwd = $_POST["pwd"];
40          $roleId = $_POST["rol"];
41
42          $userResult = $this->usuarioServicio->login($email, $pwd, $roleId);
43
44          if ($userResult == null) {
45              //400 Bad Request
46              http_response_code(400);
47              $response["error"] = true;
48              return json_encode($response);
49          } else {
50
51              SessionManager::iniciarSesion();
52              $_SESSION["userId"] = $userResult->getId();
53              $_SESSION["email"] = $userResult->getEmail();
54              $_SESSION["roleId"] = $roleId;
55              $_SESSION["ultimoAcceso"] = time();
56
57
58              $response["userId"] = $userResult->getId();
59              $response["email"] = $userResult->getEmail();
60              return json_encode($response);
61          }
62      } else {
63          //400 Bad Request
64          http_response_code(400);
65          $response["error"] = true;
66          return json_encode($response);
67      }
68  }

```



◀ Foro UD7

Ir a...

Vostede accedeu como Joaquín Lafuente Espino (Saír)
DWCS-23-24-DUAL-A

Resumen da retención de datos
Obter a apli móbil