

ACTIVIDADES UD1

En estas primeras actividades vamos a jugar con las utilidades de diferentes plataformas para probar ejemplos de código en cliente y servidor, sin necesidad de configurar la estructura que necesitaríamos en un entorno de desarrollo.

Los criterios de evaluación cubiertos por estas actividades son los siguientes:

PARTE SERVIDOR

RESULTADOS DE APRENDIZAJE	CRITERIOS DE EVALUACIÓN
RA1. Selecciona las arquitecturas y tecnologías de programación Web en entorno servidor, analizando sus capacidades y características propias.	b) Se han reconocido las ventajas que proporciona la generación dinámica de páginas Web y sus diferencias con la inclusión de sentencias de guiones en el interior de las páginas Web. f) Se han verificado los mecanismos de integración de los lenguajes de marcas con los lenguajes de programación en entorno servidor.

PARTE CLIENTE

RESULTADOS DE APRENDIZAJE	CRITERIOS DE EVALUACIÓN
RA1. Selecciona las arquitecturas y tecnologías de programación sobre clientes Web, identificando y analizando las capacidades y características de cada una.	d) Se han reconocido las particularidades de la programación de guiones y sus ventajas y desventajas y sobre la programación tradicional. e) Se han verificado los mecanismos de integración de los lenguajes de marcas con los lenguajes de programación de clientes Web.

CALIFICACIÓN

Además de estas actividades, se realizará un test online para cubrir el resto de criterios de evaluación. La aportación de cada una de las partes a los resultados de aprendizaje será:

- Test: la calificación del test aportará el 50% al RA1 de servidor y de cliente.
- Actividades prácticas: la calificación de las actividades prácticas aportará el 50% al RA1 de servidor y de cliente.

NOTA: Cualquier copia (ya sea de fuentes externas, literal de los apuntes...) que no sea una respuesta original (elaborada con tus propias palabras) será calificada con 0.

ACTIVIDAD 1 (Servidor) - Impresión de texto en PHP (2.5 puntos)

En este [enlace](#) encontrarás un ejemplo de una página web programada con PHP:

1. ¿Cuál es la etiqueta que no conoces según el HTML visto en el primer curso del ciclo? Haz una captura de pantalla y marca en color rojo lo que corresponde a código PHP, para separarlo del resto de código HTML. **(0.5 puntos)**

La etiqueta php

```
<!DOCTYPE html>
<html>
<body>

<?php
$txt1 = "Learn PHP";
$txt2 = "W3Schools.com";

print "<h2>$txt1</h2>";
print "<p>Study PHP at $txt2</p>";
?>

</body>
</html>
```

2. ¿Qué representan las líneas que empiezan por \$? Cambia los valores que aparecen a la derecha del signo "=" y pulsa el botón "Run" (no elimines el signo ";" al final de cada línea). ¿Qué sucede al cambiar estos valores? Proporciona una captura de pantalla del código modificado y el resultado. **(0.5 puntos)**

Representan variables

```
<!DOCTYPE html>
<html>
<body>

<?php
$txt1 = "Hola mundo";
$txt2 = "Alberto Martínez";

print "<h2>$txt1</h2>";
print "<p>Study PHP at $txt2</p>";
?>

</body>
</html>
```

Hola mundo

Study PHP at Alberto Martínez

3. Añade dos líneas al código: una \$txt3 con tu nombre y primer apellido, y otra que imprima un h3 (debajo del h2) con el contenido de \$txt3. Proporciona una captura del código y el resultado.

(1 punto)

```
<!DOCTYPE html>
<html>
<body>

<?php
$txt1 = "Hola mundo";
$txt2 = "Alberto Martínez";
$txt3 = "Alberto Martínez";

print "<h2>$txt1</h2>";
print "<h3>$txt3</h3>";
print "<p>Study PHP at $txt2</p>";
?>

</body>
</html>
```

Hola mundo

Alberto Martínez

Study PHP at Alberto Martínez

4. Según la descripción que se hace del lenguaje PHP en el documento teórico de la UD1, ¿dónde se ejecuta este código en una arquitectura cliente/servidor? ¿Mediante qué protocolo llega el código HTML final al cliente? Justifica tus respuestas. **(0.5 puntos)**

Se ejecuta en el servidor, llega al cliente gracias al protocolo http o https.

ACTIVIDAD 2 (Servidor) - Bucle con Django (2.5 puntos)

En este [enlace](#) encontrarás un ejemplo de una página web programada con Django (framework basado en Python).

1. ¿Mediante qué etiquetas se delimita el código Python para diferenciarlo del HTML? Haz una captura de pantalla y marca en color rojo lo que NO corresponde a código HTML. **(0.5 puntos)**

{% %}

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
{% for x in fruits %}
```

```
<h1>{{ x }}</h1>
```

```
{% endfor %}
```

```
<p>In views.py you can see what the fru:
```

```
</body>
```

```
</html>
```

2. Esta página programada en Django corresponde al modelo de programación MVC. ¿Eres capaz de identificar los 3 elementos (modelo, vista y controlador)? ¿falta alguno de ellos? Fíjate en que existen 2 ficheros (template.html y views.py), y NO es necesario que entiendas todo lo que está sucediendo en el código en estos momentos. (1 punto)

El elemento vista se encuentra en template.html, los elementos modelo y controlador se encuentran en [views.py](#) (diccionario y método).

3. ¿Qué elementos haría falta configurar en un servidor para que esta aplicación, basada en **Python**, pudiese funcionar correctamente? Elige una posible opción de las citadas en el documento de teoría y explica brevemente la función de cada uno. (1 punto)

Un framework de python, como Django, que devuelva al cliente el html con la información generada dinámicamente.

ACTIVIDAD 3 (Cliente) - jsfiddle (5 puntos)

En el siguiente enlace vas a encontrar un componente programado con el framework Vue.js:

<https://jsfiddle.net/jpeter06/eov5whe9/>

Tranquilidad, aún no conoces el concepto de "componente" (profundizarás en este concepto cuando nos adentremos en el framework de entorno cliente). En estos momentos solo podemos adelantar que un componente es una cápsula de código HTML + CSS + JavaScript que puede ser reutilizada en muchas partes de la aplicación, sin necesidad de repetir el mismo código una y otra vez.

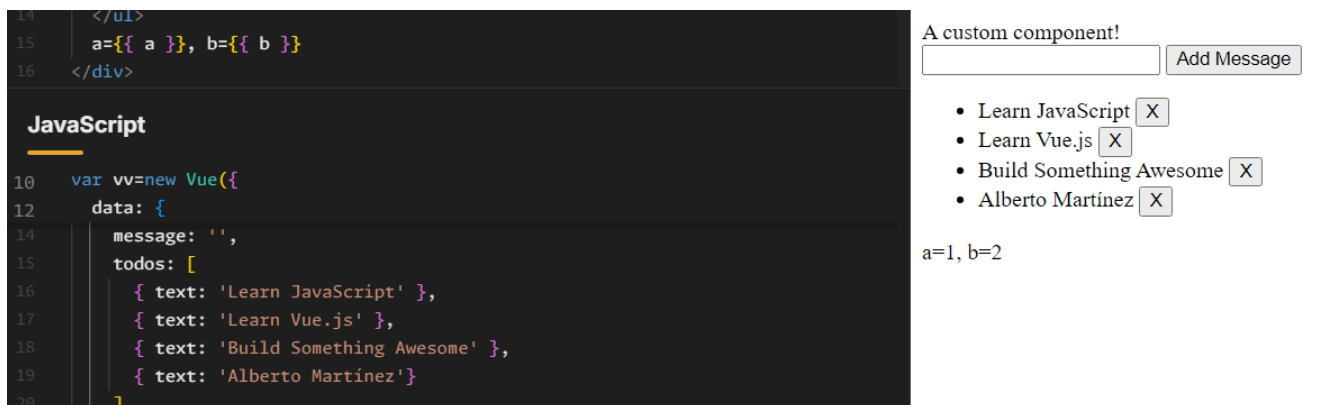
1. Identifica 3 aspectos (etiquetado, atributos, salidas de texto...) del código HTML del componente que creas que no son sintaxis pura de HTML y justifícalo. (1 punto)

```
<li v-for="todo in todos">
```

```
</div>  
a={{ a }}, b={{ b }}  
</div>
```

```
<my-component :msg='message'></my-component>
```

2. En el bloque de JavaScript, añade una entrada al array "todos" (dentro del atributo "data") con un texto nuevo, y pulsa en Run. Aporta una captura de pantalla. (Has de tener en cuenta que este array es de tipo JSON, un lenguaje de marcas que puedes revisar en este [enlace](#), y que se utilizará muchísimo a lo largo del curso, en varios módulos, y a lo largo de tu vida profesional, tanto si te dedicas al front-end, como al back-end). (1 punto)



```
14 </div>
15 a={{ a }}, b={{ b }}
16 </div>
```

JavaScript

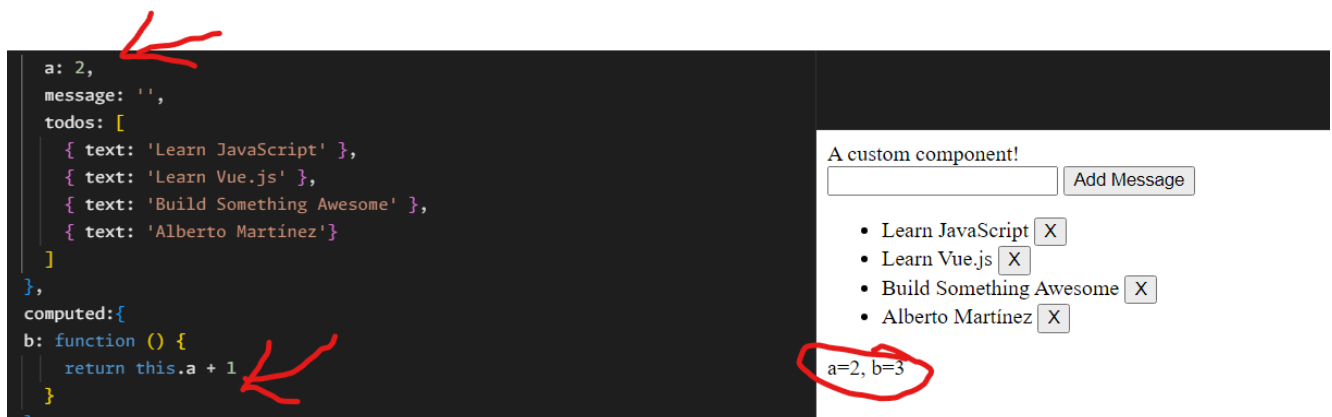
```
10 var vv=new Vue({
12   data: {
14     message: '',
15     todos: [
16       { text: 'Learn JavaScript' },
17       { text: 'Learn Vue.js' },
18       { text: 'Build Something Awesome' },
19       { text: 'Alberto Martínez' }
20     ]
11   }
12 })
```

A custom component!

- Learn JavaScript ☒
- Learn Vue.js ☒
- Build Something Awesome ☒
- Alberto Martínez ☒

a=1, b=2

3. Cambia el valor de "a" (línea 13) y comprueba la salida (pulsa en Run). ¿Cómo se ve alterado "b"? Aporta una captura de pantalla del código y el resultado. Identifica la línea en la cual se modifica el valor de "b" a partir del de "a". Prueba a alterar esta relación entre a y b para que b se incremente en un número mayor a 1, pruébalo y aporta una captura de pantalla del código y el resultado. (1 punto)



```
a: 2,
message: '',
todos: [
  { text: 'Learn JavaScript' },
  { text: 'Learn Vue.js' },
  { text: 'Build Something Awesome' },
  { text: 'Alberto Martínez' }
],
computed: {
  b: function () {
    return this.a + 1
  }
}
```

A custom component!

- Learn JavaScript ☒
- Learn Vue.js ☒
- Build Something Awesome ☒
- Alberto Martínez ☒

a=2, b=3

```

{ text: 'Learn JavaScript' },
{ text: 'Learn Vue.js' },
{ text: 'Build Something Awesome' },
{ text: 'Alberto Martínez' }
]
},
computed: {
  b: function () {
    return this.a + this.a
  }
},
methods: {
  addMessage: function () {

```

A custom component!

Add Message

- Learn JavaScript X
- Learn Vue.js X
- Build Something Awesome X
- Alberto Martínez X

a=2, b=4

4. En el cuadro de resultado (abajo a la derecha), añade un par de mensajes, utilizando el campo de texto y el botón "Add Message". Si te fijas verás que, conforme tecleas el mensaje en el campo de entrada, el texto que está sobre este campo se actualiza también; y también, al añadir un mensaje, se refleja automáticamente en la lista (sin refrescarse la pantalla). Esto se debe a que se ha utilizado un framework reactivo (Vue.js). Busca otros dos ejemplos de frameworks reactivos (aporta el enlace/s en el que te basas). ¿Qué empresas/organizaciones están detrás de cada uno de ellos (tanto de Vue.js como de los otros dos que encuentres)? (1 punto)

Aaaaaaa custom component!

aaaaaa
Add Message

- Learn JavaScript X
- Learn Vue.js X
- Build Something Awesome X
- Alberto Martínez X
- Holaaaaaa X

a=2, b=4

Angular, creado por Google, React, creado por Facebook y [Vue.js](https://vuejs.org/) mantenido por una comunidad liderada por Evan You.

<https://www.unir.net/revista/ingenieria/frameworks-javascript/>

5. ¿Dónde acabará ejecutándose este código (en el servidor o en el cliente)? ¿a qué tipo de modelo crees que pertenece este tipo de aplicaciones (MVC o servicios REST)? Justifica tus respuestas
(1 punto)

Es un servicio Rest ya que en el servidor se encuentra la API que se encargará de enviar todo lo necesario al cliente para que se ejecute allí.

