Tema 3

Jade

¿Qué es Jade?

- Jade se trata de un motor de generación de plantillas html muy potente que podemos utilizar en nuestros proyectos con Node.js.

- Una de sus características más importantes es que usa los espacios en blanco y las tabulaciones para organizar el código y conseguir diferentes niveles de propiedad.

- Si estamos acostumbrados a trabajar con HTML, no será difícil adaptarse a la utilización de Jade ya que la mayoría de las etiquetas que conocemos, son las mismas.

- Viene incluido como gestor de plantillas por defecto dentro de los proyectos creados con **Express**.

Tags

- Por defecto, cualquier texto que encontremos al principio de una línea en una plantilla Jade será tratado como una etiqueta Html.

- Una de las mayores ventajas de la utilización de Jade como gestor de plantillas es que una única etiqueta representa tanto al tag de apertura como al tag de cierre en html.

- Por ejemplo, el uso del tag **div** dentro de una plantilla Jade se transformaría en ** ** en su representación html.

- Si colocamos un texto, separado por un espacio, detrás de cualquier tag, se representará como el contenido existente entre la etiqueta de apertura y cierre html:

```
Jade   div Texto contenido dentro del div

Html   <div>Texto contenido dentro del div</div>
```

Tags

- Un ejemplo un poco más complejo, podría ser el siguiente:

Jade

```
body
    div
        h1 Cabecera de mi página
        p Probando las plantillas con Jade
    div
        footer Aquí los métodos de contacto
```

Html

```
<body>
    <div>
        <h1>Cabecera de mi página</h1>
        <p>Probando las plantillas con Jade</p>
    </div>
    <div>
        <footer>Aquí los métodos de contacto</footer>
    </div>
</body>
```

Variables

- Para completar el funcionamiento de nuestras plantillas Jade, podemos pasarle diferentes variables que incrustaremos dentro de las etiquetas definidas

- Para asignar el valor de alguna de las variables sobre nuestras etiquetas, utilizaremos el símbolo de igual (=).

- Más adelante veremos cómo podemos pasar estas variables al contexto de nuestras plantillas gracias a Javascript.

Jade

```
h1 = titulo
p = descripcion
```

Variables

```
{
    titulo: "Titulo de mi página",
    descripcion: "Esta es la descripción"
}
```

Html

```
<h1>Titulo de mi página</h1>
<p>Esta es la descripción</p>
```

Atributos

- Al igual que hacemos en html, al diseñar una plantilla Jade, podemos definir los atributos de cada una de las etiquetas.

- Simplemente habrá que definir los atributos que queremos usar entre paréntesis, justo después de la creación del tag.

- Los atributos que podemos usar son los mismos que utilizaríamos en html y los adicionales que nos aporte alguna librería con la que estemos trabajando. (jQuery, por ejemplo)

Atributos

- Un ejemplo, podría ser el siguiente:

Jade

```
div(id="content", class="main")
    a(href="http://www.imaginaformacion.com",
title="ImaginaFormacion", target="_blank") Web ImaginaFormacion
    form(action("/login"))
        input(type="submit", value="Salvar")
```

Html

```
<div id="content" class="main">
    <a href="http://www.imaginaformacion.com" title="ImaginaFormacion"
target="_blank">
        Web ImaginaFormacion
    </a>
    <form action="/login">
        <input type="submit" value="Salvar" />
    </form>
</div>
```

Atributos

- Podemos definir el valor de un atributo dinámicamente, en ese caso, sólo tenemos que asignar el nombre de nuestra variable al atributo donde queramos representarlo.

Jade

```
a(href=url, target="_blank") Web
input(type="checkbox", checked=isChecked)
```

Variables

```
{
    url: "http://www.google.es",
    isChecked: true
}
```

Html

```
<a href="http://www.google.es"
target="_blank">Web</a>
<input type="checkbox"
checked="checked" />
```

Ids y Class

- Los atributos **id** y **class** se pueden concatenar directamente detrás de la etiqueta a la que pertenecen con los símbolos **#** y **.**

- Si se omite el nombre de la etiqueta, se toma la etiqueta **div** por defecto.

Jade

```
div#contenido
    h1.cabecera.principal Cabecera 1
    #barra-lateral.izquierda
        span.contacto Contacto
    h1.cabecera.secundaria Cabecera 2
```

Html

```
<div id="contenido">
    <h1 class="cabecera principal">
        Cabecera 1
    </h1>
    <div id="barra-lateral" class="izquierda">
        <span class="contacto">
            Contacto
        </span>
    </div>
    <h1 class="cabecera secundaria">
        Cabecera 2
    </h1>
</div>
```

Script y Style

- Podemos definir el contenido de nuestras etiquetas **script** o **style** para poder crear código javascript o css respectivamente, dentro de nuestra página web.

- Simplemente tenemos que acompañar la etiqueta con un punto y tabular el contenido.

```
script.
    console.log("Hola Mundo");

style.
    body{
        background-color: red;
    }
```

Comentarios

- Tenemos dos tipos de comentarios dentro de nuestras plantillas Jade, según nuestra necesidad.
- Si utilizamos los comentarios típicos de javascript (**//**), conseguimos que se muestren dentro de la web cuando generemos el html.
- Si queremos que los comentarios no se visualicen, debemos usar aparte de las dos barras, el guión (**//-**)

Jade

```
//Comentario Html
p Este párrafo irá precedido de un comentario
//- Comentario sólo para visualizarlo en la plantilla
p#footer Copyright 2015
```

Html

```
<!-- Comentario Html -->
<p>Este párrafo irá precedido de un comentario</p>
<p id="footer">Copyright 2015</p>
```

Uso de Javascript

- Al contrario de lo explicado anteriormente, podemos utilizar código javascript en tiempo de renderizado de nuestra plantilla.
- Con este tipo de sentencias podemos generar plantillas dinámicas que se adapten al contenido de ciertas variables o incluíso utilizar bucles y condicionales.
- Simplemente tendríamos que preceder la sentencia javascript por un guión.

```
- var arr = ['a', 'b', 'c']
ul
    - for (var i = 0; i<arr.length, i++)
       li
          span= i
```

Condicionales

- Aparte de los condicionales propios de Javascript, podemos usar la sentencia **if** propia de Jade, la cual es mucho más sencilla, eliminando las llaves y los paréntesis.

```
- var user = "admin"
if user == "admin"
   p Eres el usuario administrador
else
   p No has entrado como administrador
```

Iteraciones

- De la misma manera que los condicionales anteriores, en Jade, podemos utilizar la sentencia **each** para poder iterar ciertas estructuras de tipo array.

Jade

```
- var lenguajes = ['php', 'javascript', 'java']
div
    each value, index in lenguajes
        p= index + "." +value
```

Html

```
<div>
    <p>1.php</p>
    <p>2.javascript</p>
    <p>3.java</p>
</div>
```

Interpolación

- Si queremos intercalar el valor de cualquier variable dentro de cualquier texto de nuestra plantilla, podemos utilizar la siguiente estructura **#{variable}**

```
- var titulo = "Titulo de la Pagina"
p Titulo: #{titulo}
```

Case

- Podemos utilizar la clausula **case** para poder diferenciar entre varios valores de una variable.

```
- var monedas = Math.round(Math.random()*10)
case monedas
   when 0
      p No tienes dinero
   when 1
      p Tienes una moneda
   default
      p Tienes #{monedas} monedas
```

Include

- Para evitar la duplicidades de código, podemos incluir el código de alguna de nuestras páginas dentro de otras.

- Para ello tenemos la cláusula **include** mediante la cual indicamos qué plantilla es la que vamos a incluir.

include ./ruta/al/fichero/fichero

Extends

- Para realizar plantillas complejas podemos crear un fichero base a partir del cual generaremos la estructura básica y posteriormente las demás páginas heredarán de esta plantilla base completando los diferentes bloques.

- Para definir los diferentes bloques, podemos utilizar la palabra reservada **block** seguida del nombre del bloque.

base.jade

```
block header
  p texto por defecto
block content
  p Cargando ...
block footer
  p copyright
```

main.jade

```
extends base
block header
  p texto especifico
block content
  .main-content
```

Jade y Express

- Por defecto, jade es el motor de gestión de plantillas que viene incluido en Express.

- Al generar un proyecto con Express, en el fichero **app.js** encontramos las líneas que incluyen jade en nuestra aplicación.

```
// view engine setup
app.set('views', path.join(_dirname, 'views'));
app.set('view engine', 'jade');
```

- Dentro del directorio que hemos fijado (en este caso views) debemos incluir nuestras plantillas .jade a las que luego accederemos a partir de las rutas definidas dentro de nuestro proyecto.

Jade y Express

- Cuando capturamos una petición dentro de nuestro proyecto, podemos renderizar una plantilla jade gracias al método **render**, incluyendo en la llamada las diferentes variables que luego reemplazaremos dentro de la plantilla.

```
app.get('/', function(req, res){
  res.render('index', {
    title: 'Titulo de la pagina'
  })
})
```