



Clase 10. Programación Backend

PUG & EJS



OBJETIVOS DE LA CLASE

- Conocer el motor de plantillas Pug: sintaxis y uso, e integrarlo a Express.
- Conocer el motor de plantillas Ejs: sintaxis y uso, e integrarlo a Express.

CRONOGRAMA DEL CURSO

Clase 4



Motores de Plantillas

Clase 5



Pug & Ejs

Clase 6



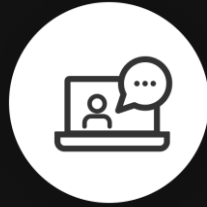
Websockets

Repasando... 

- Un motor de plantillas **lee** un **archivo de texto** (plantilla) que contiene la presentación ya preparada en un **lenguaje Pseudo HTML** e **inserta** en él la **información dinámica** que le ordena el "controlador" (la C de MVC) que representa la parte que **une la vista con la información**.
- La **sintaxis** a utilizar **depende del motor** de plantillas utilizado.
- Los **motores** de plantillas suelen tener un pequeño **lenguaje de script** que permite generar **código dinámico**.

Motores de Plantillas

Vamos al IDE... 



***¿Alguna pregunta hasta
ahora?***

- Es una **herramienta** que permite **generar** un **proyecto base** usando **express js**.
- Este proyecto base ya implementa algunas **características** de express js como **archivos estáticos, routers, vistas por motores de plantilla**, entre otros.
- Algunos de los motores de plantilla compatibles con express son: **ejs, handlebars, pug**, entre otros.

***Express
Generator***

Vamos al IDE... 



***¿Alguna pregunta hasta
ahora?***

Pug



CODER HOUSE



¿Qué es Pug?



- Pug JS es un **motor de plantillas** que nos permite **utilizar archivos estáticos** como plantillas, enviar **valores** para **reemplazar variables** dentro de las mismas y **transformar** estos archivos en **páginas HTML** que se envían al cliente.
- Express permite trabajar con muchos motores de plantillas, entre los que se encuentra Pug JS.
- Pug es muy **fácil de implementar**, solo bastará un par de líneas de código para indicarle a *express* que use Pug JS como motor de plantillas.

Pug: sintaxis

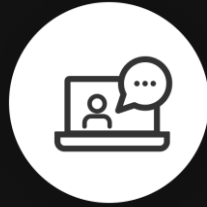
```
app.js | inicio.pug
1 extends layout.pug
2
3 block contenido
4   h1 Curso Node Básico
5
```

- Pug JS utiliza su **propia sintaxis** para declarar atributos html sin necesidad de abrir y cerrar etiquetas. En cambio se usa la **tabulación** para indicar que una etiqueta pertenece o está dentro de otra.

Para mayor información visitar el sitio web oficial de pugjs:

<https://pugjs.org/api/getting-started.html>

Vamos al IDE... 



***¿Alguna pregunta hasta
ahora?***



BREAK

¡5/10 MINUTOS Y VOLVEMOS!



PUG

Vamos a practicar lo aprendido hasta ahora

Tiempo: 10 minutos

CODER HOUSE



- 1) Realizar un servidor que reciba por query params (mediante la ruta get `'/datos'`) el valor que debe representar una barra de medición (usando el tag de html *meter*).
- 2) Asimismo recibirá además los valores mínimos y máximos permitidos y el título que se pondrá por arriba de la barra, en un elemento `h1` en color azul (debe permitir formato HTML).

Ejemplo de petición:

`http://localhost:8080/datos?min=10&nivel=15&max=20&titulo=<i>Medidor</i>`

- 1) Como respuesta a este request, el servidor devolverá al frontend una plantilla armada con los datos recibidos.
- 2) Utilizar pug integrado a express, manejando una plantilla común y una particular con la representación requerida.



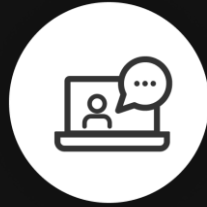
Mi plantilla de medidor en Pug JS x



localhost:8080/datos?min=5&nivel=15&max=20&titulo=<i>Medidor</i>

Medidor

5  20



***¿Alguna pregunta hasta
ahora?***

EJS

<%= EJS %>

Effective JavaScript templating.



CODER HOUSE

<%= **EJS** %>

¿Qué es EJS?



- EJS se encuentra entre los **motores de visualización temáticos más populares para node.js y express** con 5k estrellas en github y más de 8 millones de descargas por semana en npm.
- EJS significa **plantillas de JavaScript incrustadas** y podemos usarlo tanto en el **lado del servidor** como en el **del cliente**. En esta presentación, nos centraremos en el lado del servidor.
- EJS es **fácil de configurar** y podemos incluir las partes repetibles de nuestro sitio (parciales) y pasar los datos a nuestras vistas.



Sintaxis básica (etiquetas)



- **<%=** Incrusta en la plantilla el valor tal cual está
- **<%-** Incrusta en la plantilla el valor renderizado como HTML
- **<%** 'Scriptlet': Admite instrucciones en JS para declaración de variables y control de flujo

Ejemplo

```
<% const estilo = "color:crimson;"; %>
<% if (message) { %>
  <h2 style=<%=estilo%>><%= message.name %></h2>
<% } %>
```

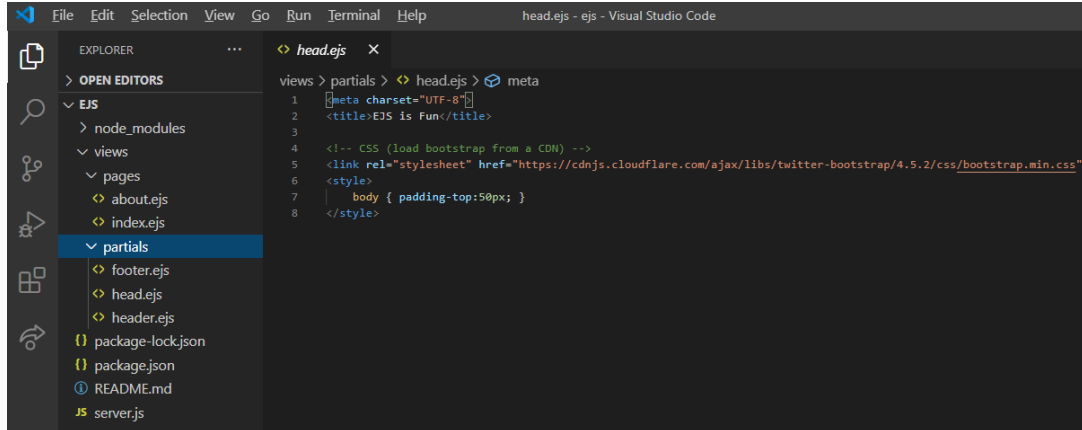
Vamos al IDE... 



***¿Alguna pregunta hasta
ahora?***

Creando nuestras plantillas Parciales

- Al igual que muchas aplicaciones que creamos, hay mucho **código que se reutiliza**. En EJS llamamos a estos códigos **parciales**
- En el ejemplo que mostramos a continuación, los definimos dentro de la **carpeta 'partials'**



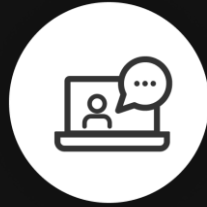
The screenshot shows the Visual Studio Code interface. On the left, the Explorer sidebar displays a project structure with folders like 'EJS', 'node_modules', 'views', and 'pages'. Under 'views', there is a 'partials' folder which is currently selected. The main editor area shows the 'head.ejs' file within the 'partials' folder. The code in 'head.ejs' includes a meta charset declaration, a title, a CSS link to Bootstrap from a CDN, and a style block for the body padding.

```
head.ejs - ejs - Visual Studio Code

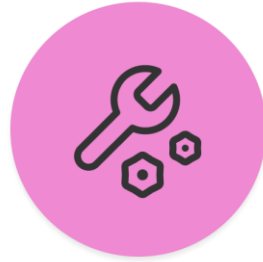
EXPLORER
  > OPEN EDITORS
  > EJS
    > node_modules
    > views
      > pages
        < head.ejs
        < index.ejs
        < partials
          < footer.ejs
          < head.ejs
          < header.ejs
        {} package-lock.json
        {} package.json
        @ README.md
        JS server.js

views > partials > < head.ejs > meta
1 <meta charset="UTF-8">
2 <title>EJS is Fun</title>
3
4 <!-- CSS (load bootstrap from a CDN) -->
5 <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/4.5.2/css/bootstrap.min.css">
6 <style>
7   |   body { padding-top:50px; }
8 </style>
```

Vamos al IDE... 



***¿Alguna pregunta hasta
ahora?***

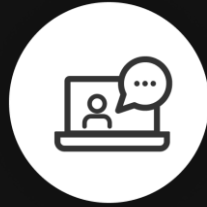


EJS

Realizar el mismo ejercicio que en el desafío anterior, utilizando ejs.

Tiempo: 10 minutos

CODER HOUSE



***¿Alguna pregunta hasta
ahora?***

EJS: Incorporando Datos en Vistas

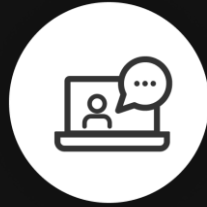
Vamos al IDE... 



***¿Alguna pregunta hasta
ahora?***

EJS: Incorporando Datos en Parciales

Vamos al IDE... 



***¿Alguna pregunta hasta
ahora?***



FORMULARIO + HISTORIAL

Tiempo: 15 minutos



- 1) Desarrollar un servidor basado en node.js, express y ejs que disponga de un **formulario** en su ruta raíz (creado con una plantilla de ejs) para ingresar los siguientes datos de una persona: nombre, apellido y edad.
- 2) La información será enviada mediante el **método post** al endpoint '/personas
- 3) Representar por debajo del mismo formulario los **datos históricos** ingresados más el actual en forma de tabla. En el caso de no encontrarse información mostrar el mensaje **'No se encontraron datos'** en lugar de la tabla.

Se sugiere el uso de bootstrap para los estilos de las plantillas. Ejemplos a continuación:



Mi plantilla de Ingreso de Datos x +

localhost:8080

Ingrese datos

Nombre

Apellido

Edad

Enviar

Historial

Nombre	Apellido	Edad
Daniel	Sánchez	52
Ana	Mei	23
Diego	Suarez	34

Mi plantilla de Ingreso de Datos x +

localhost:8080

Ingrese datos

Nombre

Apellido

Edad

Enviar

Historial

no se encontraron datos



***¿Alguna pregunta hasta
ahora?***



Motores de Plantillas

Incorporando Handlebars

Formato: link a un repositorio en Github con los tres proyectos en carpetas separadas. No incluir los node_modules.

Desafío
entregable



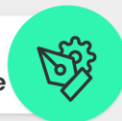
>> Consigna:

- 1) Utilizando la misma API de productos del proyecto entregable de la clase anterior, construir un web server (no REST) que incorpore:
 - a) Un formulario de carga de productos en la ruta raíz (configurar la ruta '/productos' para recibir el POST, y redirigir al mismo formulario).
 - b) Una vista de los productos cargados (utilizando plantillas de handlebars) en la ruta GET '/productos'.
 - c) Ambas páginas contarán con un botón que redirija a la otra.

Incorporando Handlebars

Formato: link a un repositorio en Github con los tres proyectos en carpetas separadas. No incluir los node_modules.

Desafío
entregable



>> Consigna:

- 2) Manteniendo la misma funcionalidad reemplazar el motor de plantillas handlebars por **pug**.
- 3) Manteniendo la misma funcionalidad reemplazar el motor de plantillas handlebars por **ejs**.
- 4) Por escrito, indicar cuál de los tres motores de plantillas prefieres para tu proyecto y por qué.

Incorporando Handlebars

Formato: link a un repositorio en Github con los tres proyectos en carpetas separadas. No incluir los node_modules.

Desafío
entregable



>> Aspectos a incluir en el entregable:

- Realizar las plantillas correspondientes que permitan recorrer el array de productos y representarlo en forma de tabla dinámica, siendo sus cabeceras el nombre de producto, el precio y su foto (la foto se mostrará como un imagen en la tabla)
- En el caso de no encontrarse datos, mostrar el mensaje: 'No hay productos'.

>> Sugerencias:

- Utilizar iconfinder (https://www.iconfinder.com/free_icons) para obtener la url de las imágenes de los productos (click derecho sobre la imagen -> copiar dirección de la imagen)

Incorporando Handlebars

Formato: link a un repositorio en Github con los tres proyectos en carpetas separadas. No incluir los node_modules.

Desafío
entregable



>> Opcional:

- Utilizar bootstrap para maquetar la vista creada por dicho motor de plantillas y el formulario de ingreso de productos.

Ejemplos a continuación





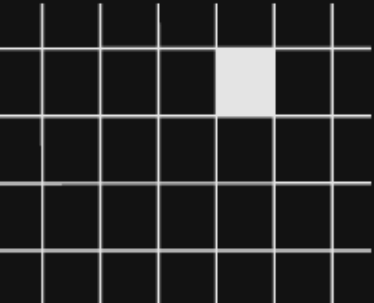
¿PREGUNTAS?





¡MUCHAS GRACIAS!

Resumen de lo visto en clase hoy:

- Motor de plantillas Pug
 - Motor de plantillas EJS
- 



OPINA Y VALORA ESTA CLASE

#DEMOCRATIZANDO LA EDUCACIÓN