



UTN.BA

UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Motores de templates

Express.js



Introducción a los motores de templates

Motores de templates

Qué es una plantilla HTML?

Es un modelo escrito en código HTML que se utiliza como base para crear nuevas páginas web.

Cuando se genera una página web basada en una plantilla, la página web tendrá similar diseño pero distinto contenido.

Cada persona puede generar sus propias plantillas o utilizar algunas ya existentes.

Motores de templates

Qué es una plantilla HTML?

Ejemplo de porción de código de una posible plantilla HTML

```
<div class="estructura">  
  <h1>{{título}}</h1>  
  <div class="cuerpo">  
    {{cuerpo}}  
  </div>  
</div>
```

{{título}} , **{{cuerpo}}** son **marcadores**. Espacio variable donde irá el contenido de cada página generada a partir de este template.

Motores de templates

Motores de plantillas.

Programas que conocen un tipo de plantillas, escritas en un lenguaje en particular y que pueden mediante petición, generar páginas web con similar diseño al de la plantilla e información diferente (la enviada por el sistema) para ser entregadas al sistema que las solicitó.

Nosotros vamos a usar el motor de plantillas **Handlebars**

Motores de templates

Motores de plantillas

Volviendo al ejemplo anterior:

Nombre de la plantilla: ejemploPlantilla.hbs

```
<div class="estructura">
  <h1>{{titulo}}</h1>
  <div class="cuerpo">
    {{cuerpo}}
  </div>
</div>
```

El motor reemplazará de forma automática los campos `{{titulo}}` y `{{cuerpo}}` por los que arroje la lógica de programación.

Motores de templates

Motores de plantillas

Un circuito posible sería:

1. El sistema solicita una página al motor de plantilla enviándole la siguiente información:
Nombre de la plantilla a utilizar: ejemploPlantilla.hbs
título: "Mi primer plantilla"
cuerpo: "Hola, estoy usando por primera vez una plantilla!"
2. El motor busca **ejemploPlantilla.hbs** y genera un archivo HTML **"ejemploPlantilla.html"** que contiene el mismo código de la plantilla pero sustituyendo los campos título y cuerpo por lo recibido desde el sistema
3. Devuelve al sistema, la página creada



Instalación de Handlebars

Motores de templates

Instalar Handlebars

Paso 1: Crear la carpeta del proyecto

```
mkdir <nombre_de_la_carpeta>
```

o hacerlo desde el explorador de Windows.

Paso 2: Agregar node a la carpeta del proyecto (crea el archivo package.json)

```
$ npm init
```

Paso 3: instalar

```
$ npm install --save express-handlebars
```

Uso de motores de templates

Estructura de archivos

```
.
├── app.js
└── views
    ├── home.handlebars
    └── layouts
        └── main.handlebars
```

Ejemplo de uso de express

Ejemplo de app.js para uso de Express

```
var express = require('express');
var exphbs  = require('express-handlebars');
var app = express();
app.engine('handlebars', exphbs());
app.set('view engine', 'handlebars');

app.get('/', function (req, res) {
  res.render('home');
});

app.listen(3000);
```



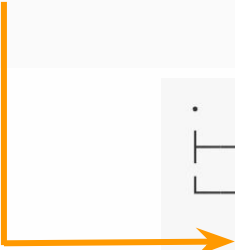
UTN.BA

FACULTAD
REGIONAL
BUENOS AIRES

El usuario ingresa a <http://localhost:3000/>

El servidor Express ejecuta el código de

```
app.get('/', function (req, res) {  
  res.render('home');  
});
```



```
.  
├─ app.js  
└─ views  
    ├─ home.handlebars  
    └─ layouts  
        └─ main.handlebars
```

Ejemplo de views/layouts/main.handlebars

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>App de ejemplo</title>
</head>
<body>

  {{{body}}}

</body>
</html>
```



Ejemplo de views/home.handlebars

```
<h1>Mi app de ejemplo!!!</h1>
```


HTML generado

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>App de ejemplo</title>
</head>
<body>
<h1>Example App: Home</h1>
</body>
</html>
```

Ejemplo con parámetros (app.js)

```
var express = require('express');
var exphbs  = require('express-handlebars');
var app = express();
app.engine('handlebars', exphbs());
app.set('view engine', 'handlebars');

app.get('/', function (req, res) {
  res.render('home', {nombre: 'Un nombre', apellido: 'Un apellido'});
});

app.listen(3000);
```



Ejemplo de views/home.handlebars

```
<h1>Bienvenido {{nombre}} {{apellido}}</h1>
```

```
app.get('/', function (req, res) {  
  res.render('home', {nombre: 'Un nombre', apellido: 'Un apellido'});  
});
```



```
<h1>Bienvenido {{nombre}} {{apellido}}</h1>
```

Mostrar un valor

```
{{nombre_variable}}
```

Condicional

```
{{#if <condición>}}
```

```
{{/if}}
```

Bucles

```
{{#each <lista>}}  
  {{this}}<br>  
{{/each}}
```

Envío de mails

Express.js



Envío de mails

Instalamos el paquete

```
$ npm install nodemailer --save
```


Envío de mails

```
var express = require('express');  
  
//Requerimos el paquete  
var nodemailer = require('nodemailer');  
  
var app = express();  
  
//Creamos el objeto de transporte  
var transporter = nodemailer.createTransport({  
  service: 'gmail',  
  auth: {  
    user: 'tucorreo@gmail.com',  
    pass: 'tucontraseña'  
  }  
});  
  
var mensaje = "Hola desde nodejs...";
```

```
//Creamos el objeto de opciones de envío  
var mailOptions = {  
  from: 'tucorreo@gmail.com',  
  to: 'mi-amigo@yahoo.com',  
  subject: 'Asunto Del Correo',  
  text: mensaje  
};  
  
//Enviamos el mail  
transporter.sendMail(mailOptions, function(error, info){  
  if (error) {  
    console.log(error);  
  } else {  
    console.log('Email enviado: ' + info.response);  
  }  
});
```



PRÁCTICA

Práctica 01

Ejercicio 1

El objetivo es lograr un pequeño sitio web con una página estática inicial (tipo index.html) que tenga un enlace a una página de registración.

En la página de registración, el usuario completa un formulario de registro y oprime el botón para registrarse.

Tu servidor tiene que verificar que la info ingresada por el usuario sea completa, es decir, que ningún campo se haya enviado vacío.

Si algún dato no se ingresó, tiene que devolver una página que te muestre los datos que ingresaste y los que te faltan ingresar y que contenga un enlace a la página de registro.

Si todos los datos fueron correctos, tiene que devolver otra página que te muestra todos los datos ingresados y te dice que tu registro fue exitoso. Esta página tiene que contener un enlace a la página inicial.



Dado el JSON de la página

<https://jsonplaceholder.typicode.com/todos>

Descargar el mismo, asignarlo a una variable llamada listado

Hacer la página de listado de tareas (con template handlebars) cuando se acceda a la url

<http://localhost:3000/tareas>

Se debe mostrar el título de la tarea y si está finalizada

A partir del ejercicio 02.

Crear un template handlebars y ruta express que cuando el usuario ingrese a <http://localhost:3000/tareas/3> muestre el template con la tarea cuyo id sea 3

Tener en cuenta que el id puede cambiar (y que tu código debe retornar la tarea con el id que te solicite el usuario)

Práctica 04

Realizar la práctica 02 y 03 pero retornando un JSON en vez de un template. Las URLs deben ser

<http://localhost:3000/api/tareas>

<http://localhost:3000/api/tareas/4>

Dado el JSON de la página

<https://jsonplaceholder.typicode.com/users>

Debes realizar los mismos pasos que para las prácticas 02,03,04.

El listado de usuarios debe incluir: nombre, usuario, email, y nombre de la compañía. Mientras que la visualización debe incluir todos los datos.