

Archivos parciales

“

Existen maneras de **reutilizar** el mismo **HTML** en varias vistas, logrando así simplificar mucho el **desarrollo** de nuestro proyecto.

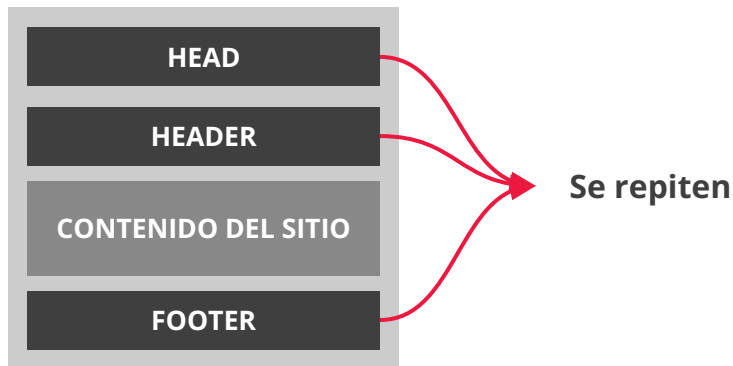
”



¿Cómo modularizar?

Lo primero a tener en cuenta es detectar aquellas porciones de **código** que vemos **repetidas** en todos los archivos del proyecto.

En una estructura corriente de HTML podemos encontrar que todos los archivos fácilmente van a contar con: un head, una barra de navegación y un footer, por mencionar algunas.

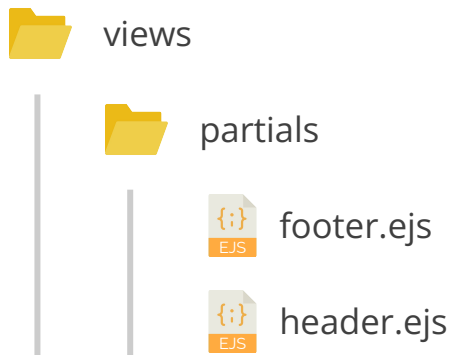


¿Cómo modularizar?

Cada una de estas partes se convertirá en un **archivo nuevo** que contendrá **únicamente** esa **porción** de código. Las llamamos vistas parciales.

Estos archivos también se guardan dentro de la carpeta **views**, o la que hayamos configurado en su lugar.

Una buena práctica es tener una carpeta aparte para vista parciales.



Vistas **parciales**

Una vez detectadas las partes que se repiten dentro de nuestras vistas:

- Cortamos cada parte y la pegamos en un nuevo archivo.
- Le asignamos un nombre que represente la parte que contiene.
- Le agregamos la extensión que corresponda, por ejemplo: `head.ejs`.

```
html1 <!-- views/partials/head.ejs -->
      <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <link rel="stylesheet" href="/css/master.css">
        <title>Mi web</title>
      </head>
```

Incluyendo las vistas parciales

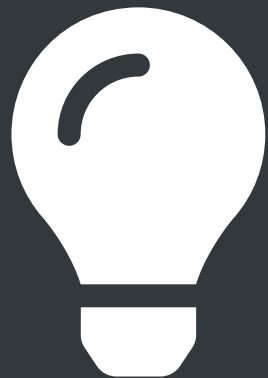
Para disponer del código que modularizamos, necesitamos hacer uso de la función `include()` que nos provee EJS. La misma recibe como parámetro un string que será la ruta hacia el archivo que queremos incluir.

Por último, deberemos escribir la etiqueta `<%- %>`, que imprimirá en el documento el contenido exacto que retorne el `include()`.

```
html  <!-- views/home.ejs -->
      <!DOCTYPE html>
      <html>
        <%- include('./partials/head') %>
        <body>
          <header>
            <div class="top-header">
```

“

Modularizar nos ofrece varios beneficios: **ordenar** nuestro código, ser menos **repetitivos**, y conseguir un mejor y más simple **mantenimiento** del código.



”

DigitalHouse>
Coding School