

EN ESTA EXPERIENCIA ESTUDIAREMOS

- Manejo de imágenes
- Organización de assets
- Manejo de CSS y SASS
- Utilizar las variables de bootstrap
- Integrar plantillas

DURANTE ESTA EXPERIENCIA
APRENDEREMOS 2 CONCEPTOS
MUY IMPORTANTES

1) Asset path

2) Asset pipeline

ASSET PATH

POR QUÉ ES IMPORTANTE SABER SOBRE EL ASSET PATH?

- Para saber desde que carpetas se incorporan las **imágenes**, los **CSS** y los **JS**
- Para poder crear o utilizar plantillas
- Para evitar perder muchas horas haciendo prueba y error con los assets
- Organizar de mejor forma nuestros archivos

¿QUÉ SON LOS ASSETS?

Es una palabra comodín para referirse a todo tipo de archivos que se necesitan para completar un proyecto como las imágenes los CSS, los JS, tipografías entre otros.



EL ASSET PATH

Es un array que contiene un listado de carpetas desde donde se cargan los assets


¿QUÉ CARPETAS ESTÁN EN EL
ASSET PATH?

¿QUÉ CARPETAS ESTÁN EN EL ASSET PATH?

```
print Rails.application.config.assets.paths
```


¿QUÉ CARPETAS ESTÁN EN EL ASSET PATH?

```
print Rails.application.config.assets.paths
```

A thick orange curved arrow originates from the code box and points towards the output box.

```
[  
  "/Users/gonzalosanchez/ruby/clase 11/twitter/app/assets/config",  
  "/Users/gonzalosanchez/ruby/clase 11/twitter/app/assets/images",  
  "/Users/gonzalosanchez/ruby/clase 11/twitter/app/assets/javascripts",  
  "/Users/gonzalosanchez/ruby/clase 11/twitter/app/assets/stylesheets",  
  "/Users/gonzalosanchez/.rvm/gems/ruby-2.3.1/gems/coffee-rails-4.2.1/lib/assets/  
  javascripts",  
  "/Users/gonzalosanchez/.rvm/gems/ruby-2.3.1/gems/actioncable-5.1.0/lib/assets/  
  compiled",  
  "/Users/gonzalosanchez/.rvm/gems/ruby-2.3.1/gems/actionview-5.1.0/lib/assets/  
  compiled",  
  "/Users/gonzalosanchez/.rvm/gems/ruby-2.3.1/gems/turbolinks-source-5.0.3/lib/assets/  
  javascripts"  
]
```

USANDO EL ASSET PATH

Si un archivo, por ejemplo una imagen, está dentro del `asset_path` la podemos cargar sin especificar el path

```
<%= image_tag 'imagen-prueba.png' %>
```

USANDO EL ASSET PATH

Si un archivo está **dentro de una subcarpeta** del `asset_path`

```
<%= image_tag 'subcarpeta/imagen-prueba.png' %>
```

USANDO EL ASSET PATH

Si el archivo no está dentro del `asset_path` o de una subcarpeta no lo podremos ocupar

```
[  
  "/Users/gonzalosanchez/ruby/clase 11/twitter/app/assets/config",  
  "/Users/gonzalosanchez/ruby/clase 11/twitter/app/assets/images",  
  "/Users/gonzalosanchez/ruby/clase 11/twitter/app/assets/javascripts",  
  "/Users/gonzalosanchez/ruby/clase 11/twitter/app/assets/stylesheets"  
]
```

Si el anterior es mi asset path, y tengo una imagen en
twitter/app/assets/images/template/hola.png

¿Cómo debo incluirla?

```
[  
  "/Users/gonzalosanchez/ruby/clase 11/twitter/app/assets/config",  
  "/Users/gonzalosanchez/ruby/clase 11/twitter/app/assets/images",  
  "/Users/gonzalosanchez/ruby/clase 11/twitter/app/assets/javascripts",  
  "/Users/gonzalosanchez/ruby/clase 11/twitter/app/assets/stylesheets"  
]
```

Si el anterior es mi asset path, y tengo una imagen en
twitter/app/assets/images/template/hola.png

¿Cómo debo incluirla?

```
<%= image_tag 'subcarpeta/imagen-prueba.png' %>
```

LOS HELPERS

Son métodos que nos ayudan a resolver problemas cotidianos como incluir imágenes o otro tipo de archivos

EJEMPLOS DE HELPERS

❑ `image_tag`

❑ `asset_path`

IMAGE_TAG

Es un helper específico para cargar imágenes

```
<%= image_tag "logo.jpg" %>
```

IMAGE_TAG

Es un helper específico para cargar imágenes

```
<%= image_tag "logo.jpg" %>
```



```

```

fingerprint
(lo veremos más adelante)

CLASE Y ESTILO

Al método `image_tag` se le pueden pasar como parámetros **class** y **style**

```
<%= image_tag "logo.png", class: 'logo',  
           style: 'float:left' %>
```

UN LINK CON IMAGEN

```
<%= link_to image_tag("padrino.png"), "http://www.google.cl" %>
```

En este caso, en vez de mostrar un string como link, será la imagen ingresada la que funcionará como link

EJEMPLOS DE HELPERS

☒ image_tag

☐ asset_path

EL HELPER ASSET_PATH

asset_path es un método que busca un asset dentro de todas las carpetas de la lista y devuelve el path si lo encuentra.

```
<%= asset_path "logo.png" %>
```

EL HELPER ASSET_PATH

asset_path es un método que busca un asset dentro de todas las carpetas de la lista y devuelve el path si lo encuentra.

```
<%= asset_path "logo.png" %>
```



/assets/logo-14fd73a29b1c3aa3fba9674025a2070230a67103ba30963a2bfda66904f9a7ef.png

EL HELPER ASSET_PATH

asset_path es un método que busca un asset dentro de todas las carpetas de la lista y devuelve el path si lo encuentra.

```
<%= asset_path "logo.png" %>
```



/assets/logo-14fd73a29b1c3aa3fba9674025a2070230a67103ba30963a2bfda66904f9a7ef.png

fingerprint
(lo veremos más adelante)

¿PARA QUÉ SIRVE ASSET_PATH SI TENEMOS IMAGE_TAG?

```
.header{  
  background: url('<%= asset_path "logo.png" %>')  
}
```

Para que esto funcione la extensión del archivo debe ser .css.erb

PARA AGREGAR UNA IMÁGEN CON ASSET_PATH

```
">
```

EJEMPLOS DE HELPERS

☒ image_tag

☒ asset_path

Para entender como funciona el manejo de archivos **CSS**, **SASS**, **ERB** y **coffescript**, y otros necesitamos introducir un segundo concepto muy importante

Para entender como funciona el manejo de archivos **CSS**, **SASS**, **ERB** y **coffescript**, y otros necesitamos introducir un segundo concepto muy importante

ASSET PIPELINE

¿QUÉ ES EL ASSET PIPELINE?

Es un framework encargado de preprocesar, precompilar, minificar y agregar un fingerprint a los assets.

En Rails el asset pipeline es implementado por la gema **sprockets-rails**

¿POR QUÉ ES IMPORTANTE EL ASSET PIPELINE?

Nos ayuda a optimizar el proceso de manejo,
transformación y carga de assets

¿POR QUÉ ES IMPORTANTE EL ASSET PIPELINE?

¿POR QUÉ ES IMPORTANTE EL ASSET PIPELINE?

Nos ayuda a cargar más rápido nuestras aplicaciones

USO DEL ASSET PIPELINE

Sprocket ve todos los archivos de estilo en la carpetas del asset path, si uno de estos archivos ocupa SASS lo **transforma** a CSS, luego junta todos los archivos, los **minifica** y les agrega un **fingerprint**.

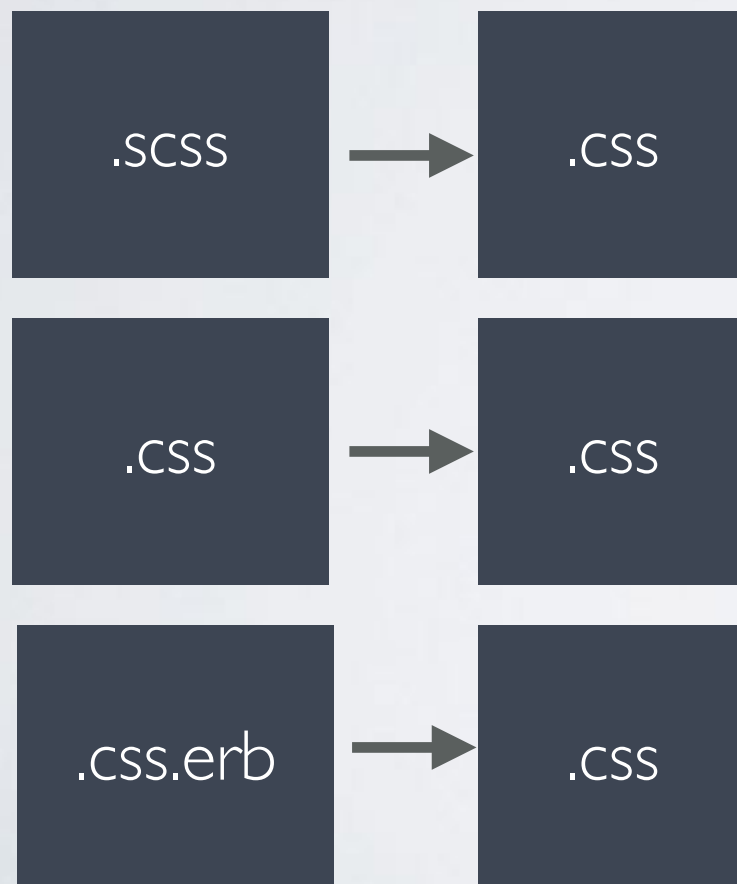
.SCSS

.CSS

.css.erb

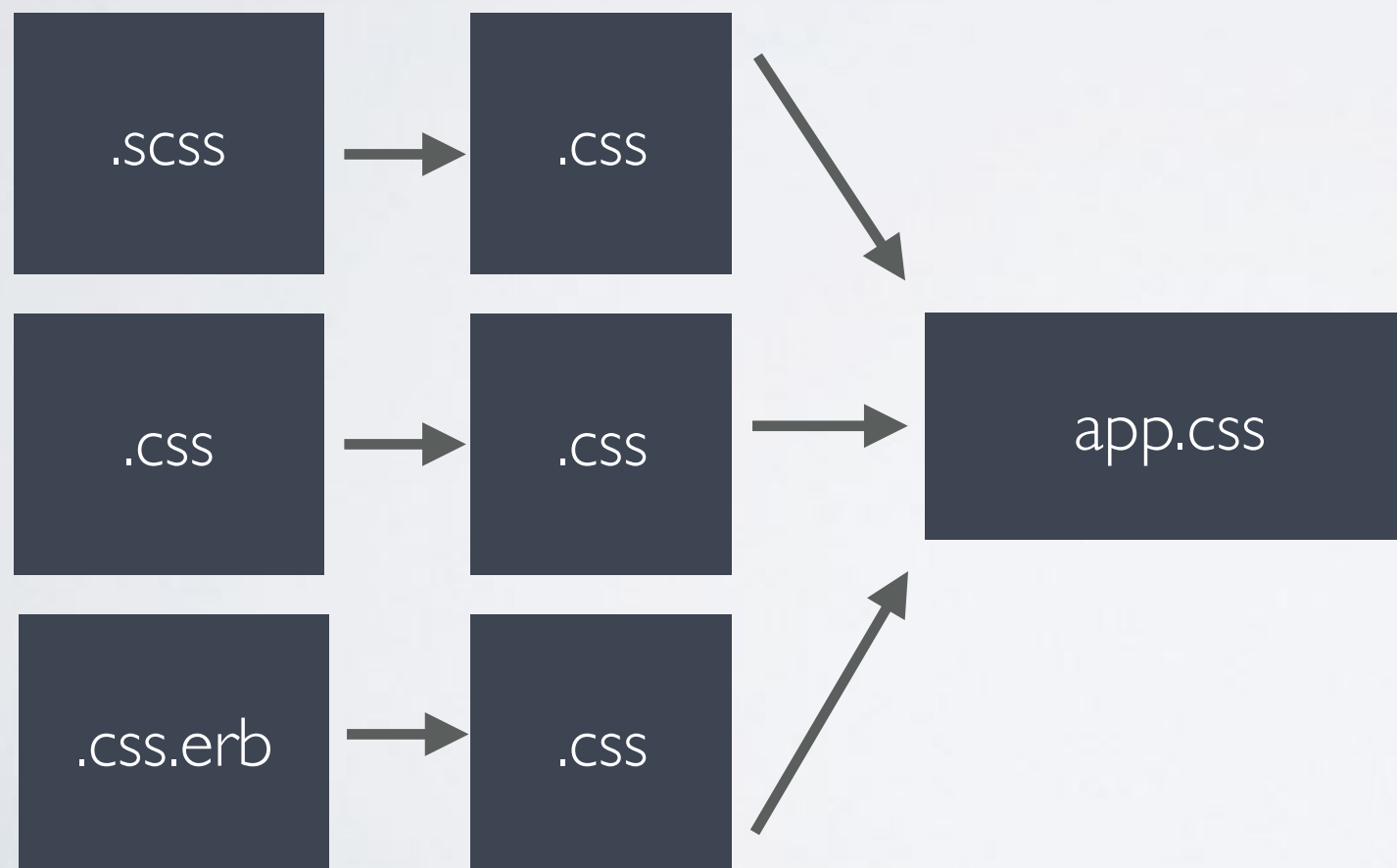
USO DEL ASSET PIPELINE

Sprocket ve todos los archivos de estilo en la carpetas del asset path, si uno de estos archivos ocupa SASS lo **transforma** a CSS, luego junta todos los archivos, los **minifica** y les agrega un **fingerprint**.



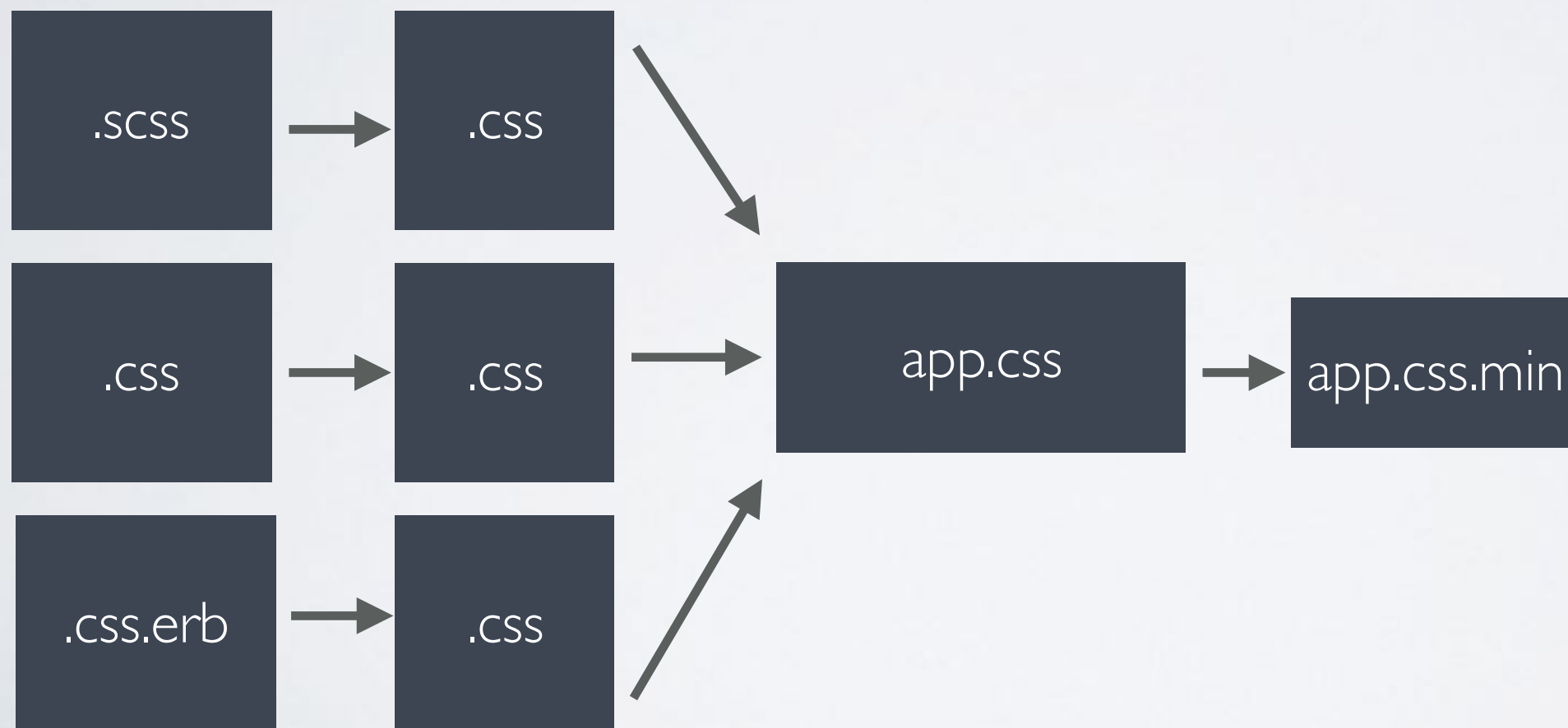
USO DEL ASSET PIPELINE

Sprocket ve todos los archivos de estilo en la carpetas del asset path, si uno de estos archivos ocupa SASS lo **transforma** a CSS, luego junta todos los archivos, los **minifica** y les agrega un **fingerprint**.



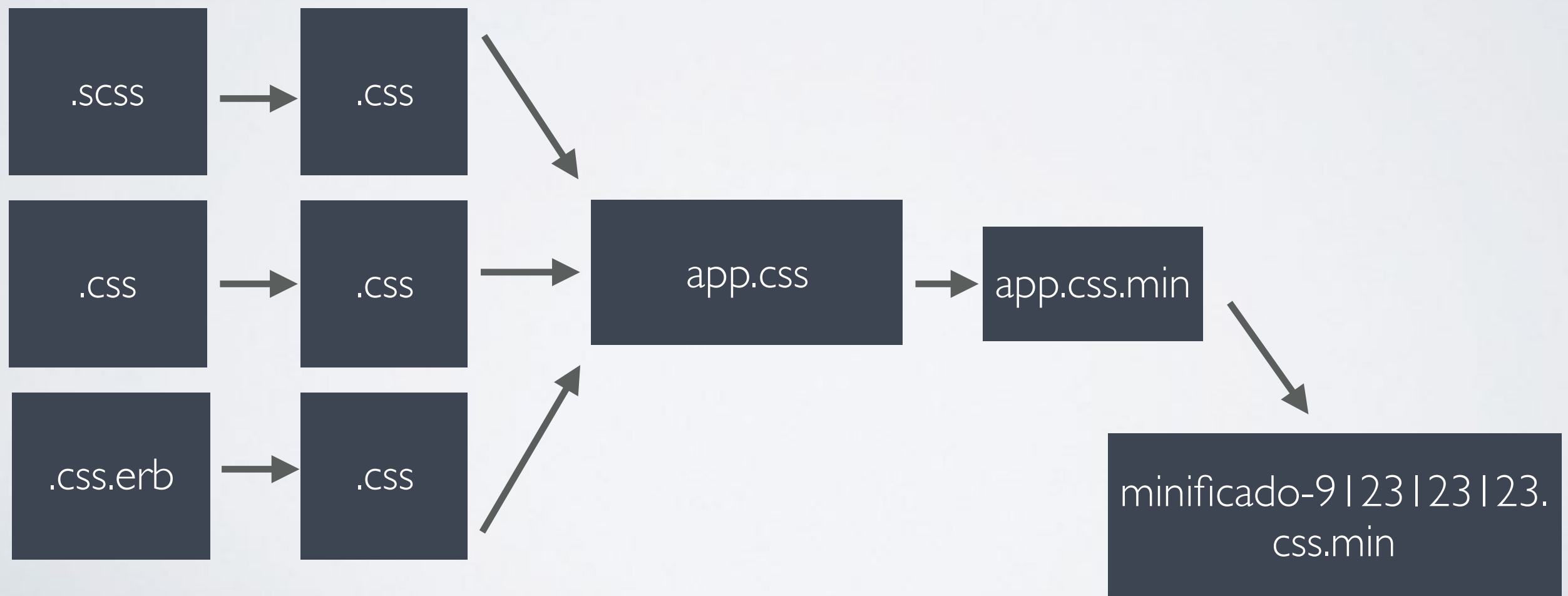
USO DEL ASSET PIPELINE

Sprocket ve todos los archivos de estilo en la carpetas del asset path, si uno de estos archivos ocupa SASS lo **transforma** a CSS, luego junta todos los archivos, los **minifica** y les agrega un **fingerprint**.



USO DEL ASSET PIPELINE

Sprocket ve todos los archivos de estilo en la carpetas del asset path, si uno de estos archivos ocupa SASS lo **transforma** a CSS, luego junta todos los archivos, los **minifica** y les agrega un **fingerprint**.



USO DEL ASSET PIPELINE

La gema sprocket-rails hace todo esto automáticamente



4 FUNCIONES DE SPROCKETS

- Preprocesar
- Concatenar
- Minificar
- Añadir Fingerprint

PREPROCESADO

- Evaluar ERB
- Convertir SASS o LESS en CSS
- Convertir Coffescript en Javascript
- Convertir SLIM o HAML en HTML

CONCATENAR

La concatenación no sucede en entorno de desarrollo (pero puede activarse)

- Convertir todos los CSS en uno solo
- Convertir todos los javascript en uno solo

SIN CONCATENAR

▼ assets

- action_cable.self-5454023407ffec0d29137c7110917e1e745525ae9afbc05f52104c4cd6597429.js?body=1
- jquery.self-bd7ddd393353a8d2480a622e80342adf488fb6006d667e8b42e4c0073393abee.js?body=1
- jquery_ujs.self-784a997f6726036b1993eb2217c9cb558e1cbb801c6da88105588c56f13b466a.js?body=1
- turbolinks.self-c5acd7a204f5f25ce7a1d8a0e4d92e28d34c9e2df2c7371cd7af88e147e4ad82.js?body=1
- application.self-6ee4e4f81779c3e3e689732220d321ad30a8825a9648259bfb4cf9ec2aa0aa32.css?body=1
- css1.self-e3b0c44298fc1c149afb4c8996fb92427ae41e4649b934ca495991b7852b855.css?body=1
- css2.self-e3b0c44298fc1c149afb4c8996fb92427ae41e4649b934ca495991b7852b855.css?body=1
- pages.self-e3b0c44298fc1c149afb4c8996fb92427ae41e4649b934ca495991b7852b855.css?body=1
- logo-404b0a5b93422cf8cf467d746efa62a992b947687215ddbe423bfd71c9a36af3.jpg

CON CONCATENAR

▼ assets

- application-a069cda85c0cc4a533f211c9843971b399941213f42fe8eb8605c2f7eb436fe6.js
- application-6ee4e4f81779c3e3e689732220d321ad30a8825a9648259bfb4cf9ec2aa0aa32.css
- logo-404b0a5b93422cf8cf467d746efa62a992b947687215ddbe423bfd71c9a36af3.jpg

MINIFICAR (MINIMIFICAR)

- Eliminar los espacios en blanco y retornos de carro de los archivos css y js, de forma de hacer optimizar el peso.

FINGERPRINT

- Agrega un identificador único al archivo para optimizar el caching de este.

FINGERPRINT

- Agrega un identificador único al archivo para optimizar el caching de este.

`/assets/logo-14fd73a29b1c3aa3fba9674025a2070230a67103ba30963a2bfda66904f9a7ef.png`

FINGERPRINT

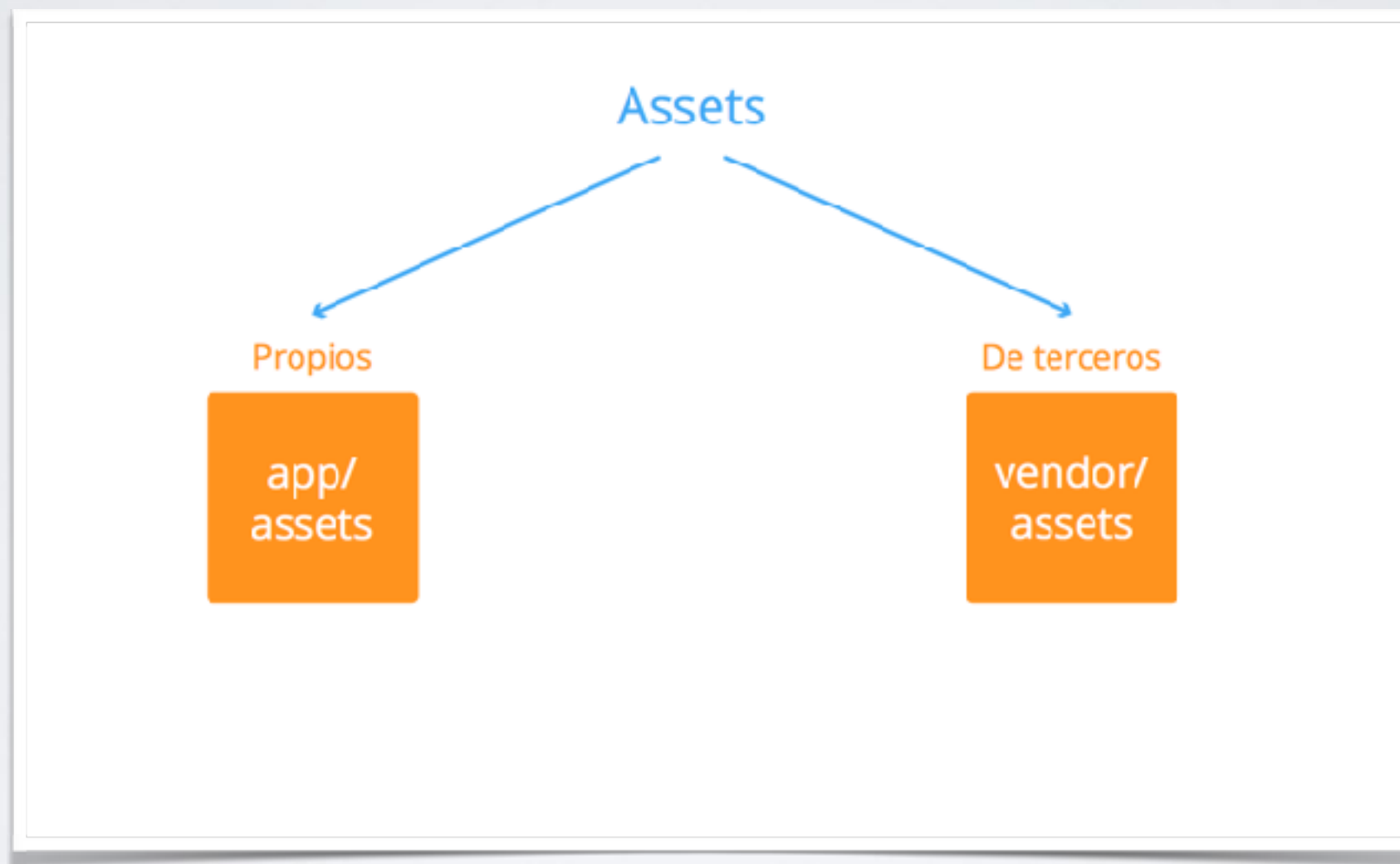
- Agrega un identificador único al archivo para optimizar el caching de este.

/assets/logo-14fd73a29b1c3aa3fba9674025a2070230a67103ba30963a2bfda66904f9a7ef.png

fingerprint

¿CÓMO SE ORGANIZAN LOS ASSETS?

Por defecto los assets se organizan en 2 carpetas principales



Nosotros podemos agregar nuevas carpetas y subcarpetas al path

¿CÓMO SE ORGANIZAN LOS ASSETS?

app/assets

Los assets dentro de esta carpeta se cargan automáticamente :)

vendor/assets

Los assets dentro de esta carpeta los tenemos que agregar nosotros :(

¿CÓMO AGREGAR UNA CARPETA AL ASSET_PATH?

Dentro del archivo config/initializers/assets.rb

```
Rails.application.config.assets.paths << Rails.root.join("app", "assets", "images")
```



Rails.root es la raíz del proyecto

REINICIAR EL SERVIDOR

Los initializers se cargan al levantar el servidor de rails,
si hacemos un cambio tenemos que reiniciar el
servidor

REVISAMOS QUE HAYA SIDO AGREGADA

```
exit (de la consola)
```

```
rails c
```

```
print Rails.application.config.assets.paths
```

A SPROCKET HAY QUE ESPECIFICARLE LAS CARPETAS Y LOS TIPOS DE ARCHIVOS QUE QUEREMOS CARGAR

Dentro del archivo `config/initializers/assets.rb`

```
Rails.application.config.assets.precompile += %w( banner.jpg )
```

EL MANIFIESTO ES UN ARCHIVO QUE ESPECIFICA QUE ASSETS CARGAR

```
<!DOCTYPE html>
<html>
<head>
  <title>Asset Path </title>
  <%= stylesheet_link_tag      'application', media: 'all', 'data-turbolinks-track' => true %>
  <%= javascript_include_tag  'application', 'data-turbolinks-track' => true %>
  <%= csrf_meta_tags %>
</head>

<body>
  <%= yield %>
</body>
</html>
```

Manifiestos



Dentro del layout vemos que vienen integrado dos manifiestos, uno para **CSS** y otro para **JS** por defecto.

APPLICATION.CSS

Este archivo es el manifiesto, se compone de directivas

- ***= require_tree .**

carga todos los css dentro de la carpeta stylesheets

- ***= require_self**

carga el contenido del mismo archivo applications.css

Directivas

A red curved arrow originates from the word "Directivas" and points to the two list items, indicating that they are directives.

DIRECTIVAS

- El orden de las directivas importan, porque todos los archivos se concatenan y el orden de las directivas es el orden de carga de los archivos.
- Se pueden hacer requires específicos y luego **require_tree** para cargar los CSS en el orden que uno quiera y luego el resto.
- El orden es importante porque en CSS la última definición manda por sobre la primera.

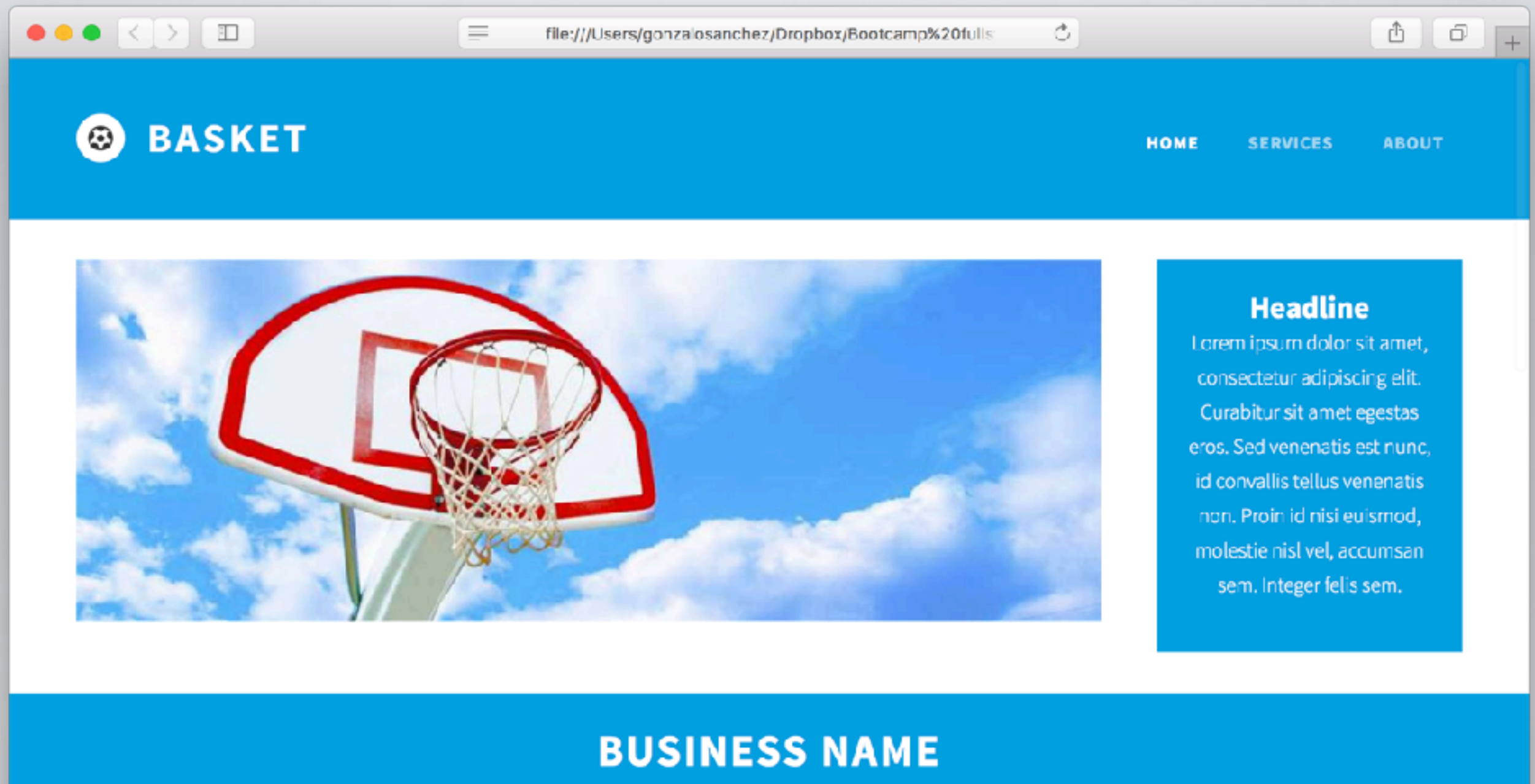
EXISTEN DIVERSOS TIPOS DE DIRECTIVAS

- **require**
- require_directory
- **require_tree**
- require_self
- link
- depend on assets
- stub

PODEMOS ESPECIFICAR EL ORDEN DE CARGA

- `*= require primero.css`
- `*= require ultimo.css`
- `*= require_tree .`
- `*= require_self`

INTEGREMOS NUESTRA PRIMERA PLANTILLA



http://www.css3templates.co.uk/templates/css3_basket/index.html

PUEDES DESCARGAR LA
PLANTILLA DESDE

http://www.css3templates.co.uk/downloads/css3_basket.zip

CREAR NUEVO PROYECTO DE RAILS

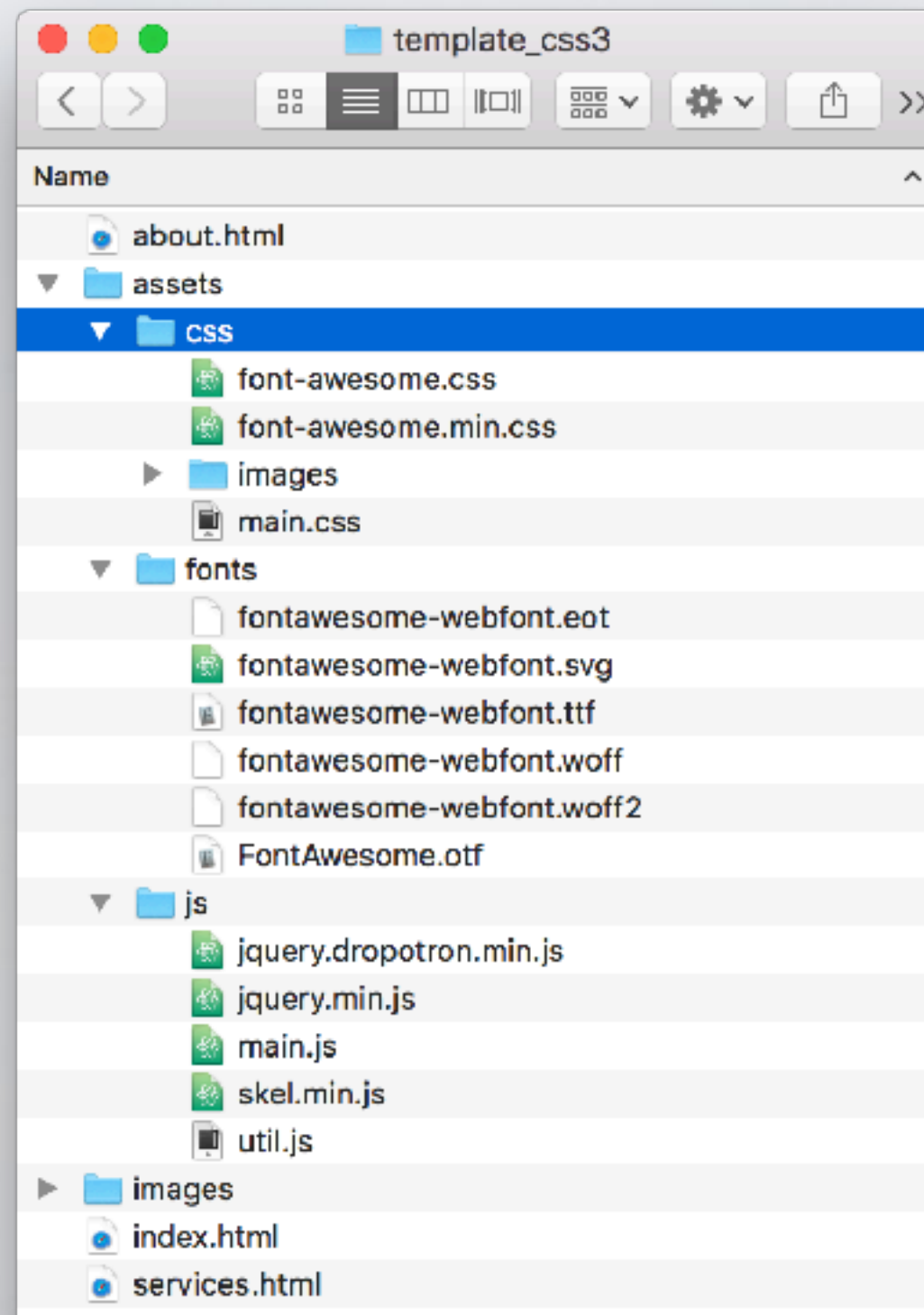
rails new template

CREAREMOS UN CONTROLLER CON NUESTRA PRIMERA PÁGINA

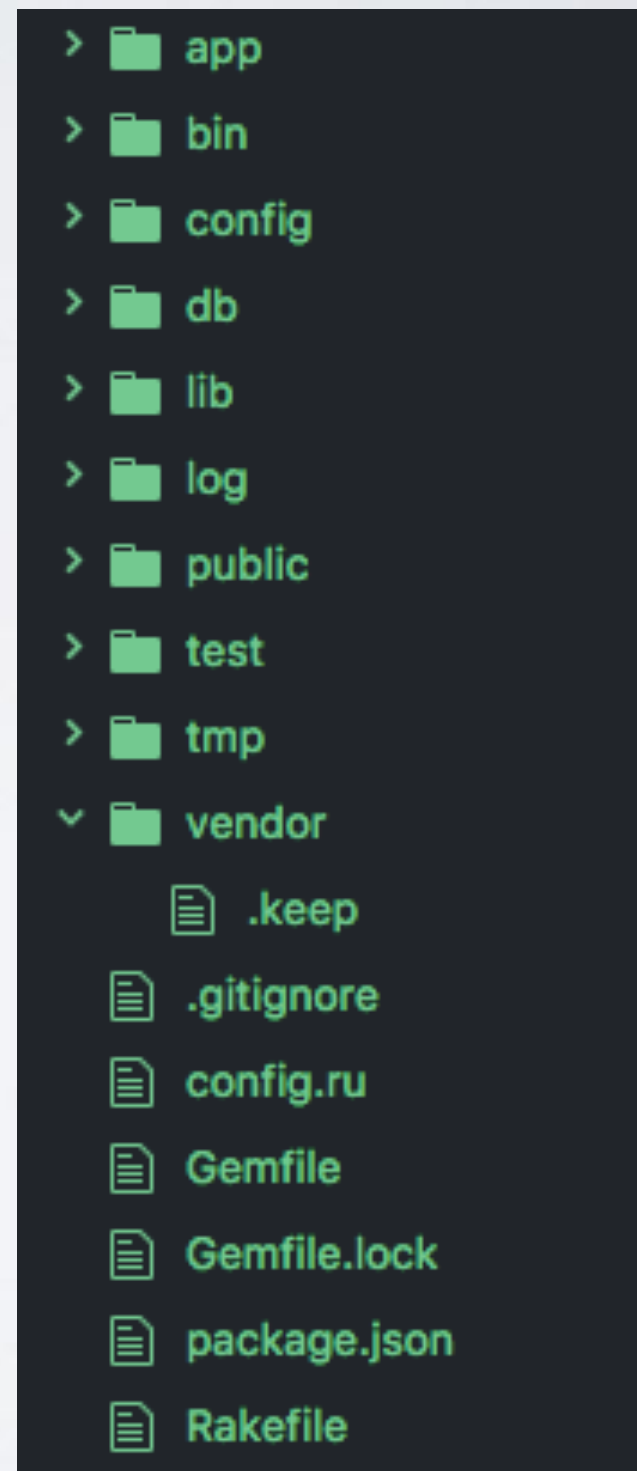
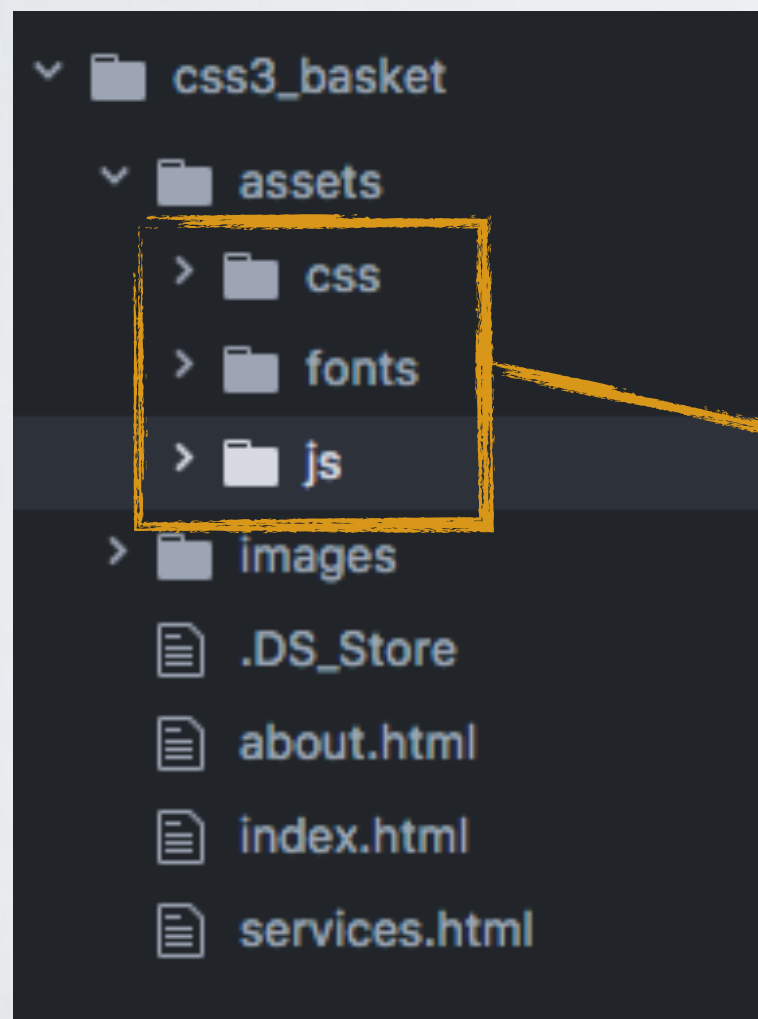
rails g controller pages home

Podemos aprovechar de dejar home como página de inicio

PRIMERO ANALIZAREMOS LA PLANTILLA



COPIAMOS LOS ASSETS IMPORTANTES DENTRO DE VENDOR



¿POR QUÉ EN VENDOR?

Sería perfectamente válido agregar las carpetas dentro de assets, pero por regla general los assets de 3° deberían ir dentro de vendor

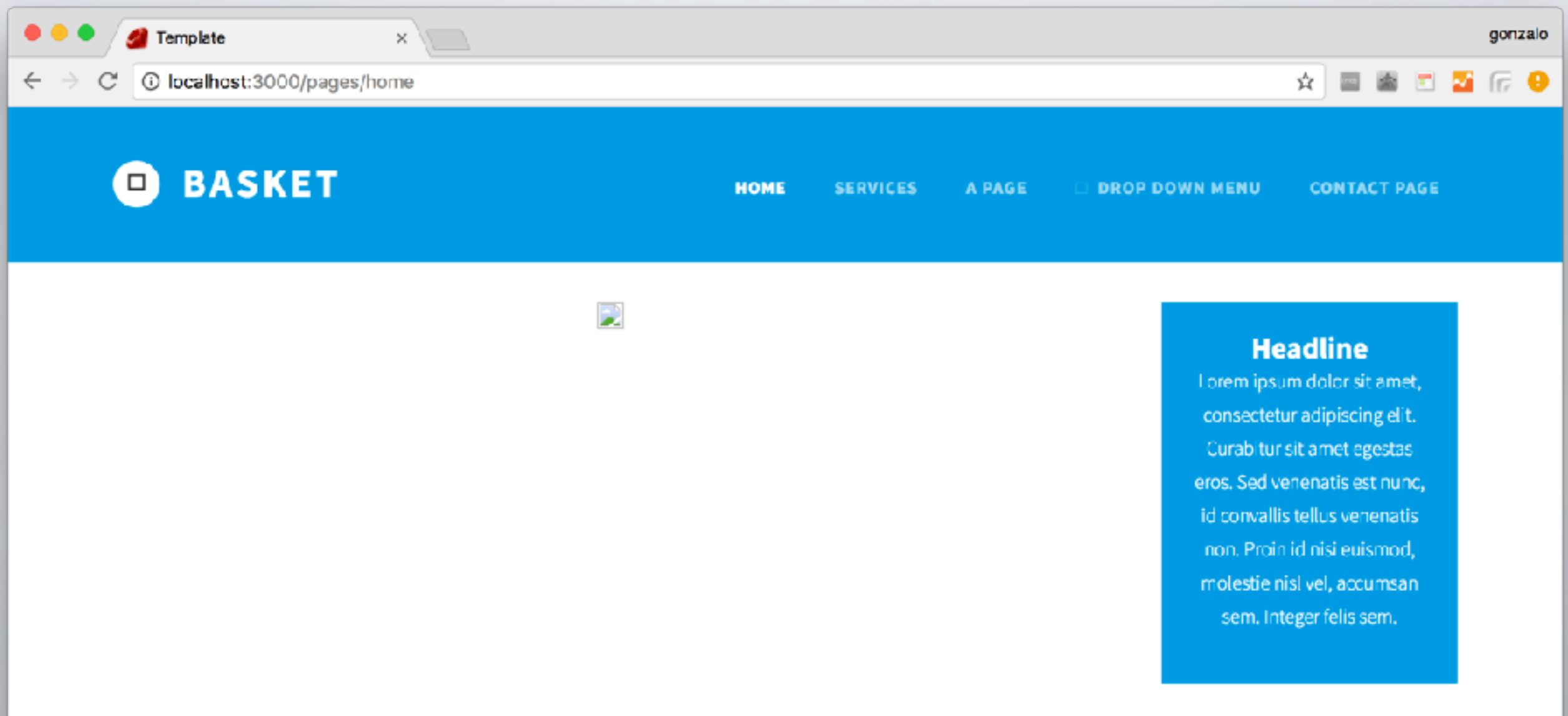
LA CARPETA VENDOR NO SE CARGA AUTOMÁTICAMENTE EN EL MANIFIESTO

Agregamos la directiva

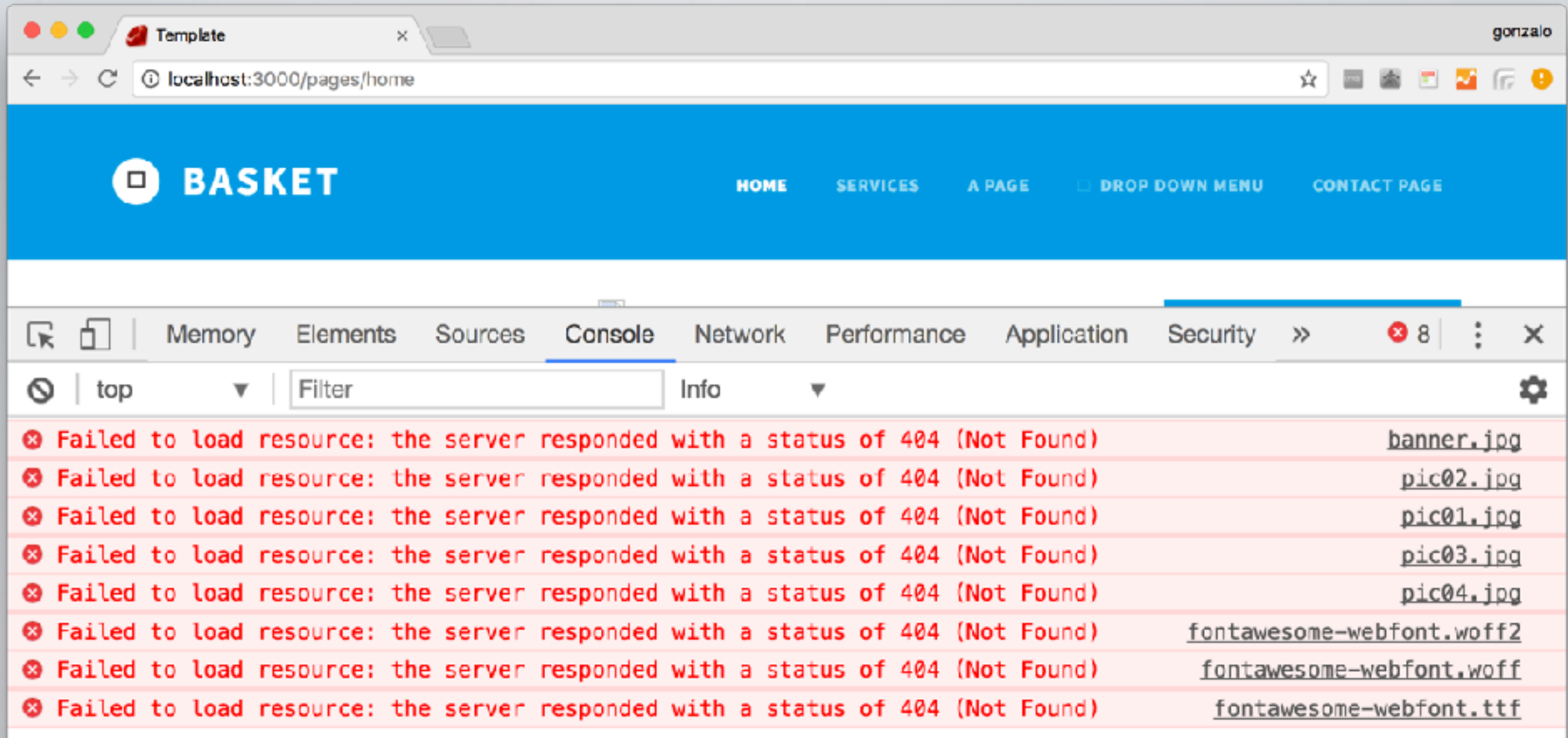
`*= require main.css`

AHORA AGREGAREMOS LA ESTRUCTURA DE LA PÁGINA

AHORA AGREGAREMOS LA ESTRUCTURA DE LA PÁGINA



¿Y LAS IMÁGENES?



Tenemos que ocupar el helper `image_tag`

¿Y LAS IMÁGENES?

Agregamos la carpeta al asset path

```
Rails.application.config.assets << Rails.root.join('vendor', 'assets', 'images')
```

Le especificamos a Sprocket que archivos incluir

```
Rails.application.config.assets.precompile += %w( banner.jpg )
```

PODEMOS UTILIZAR * COMO COMODÍN

```
Rails.application.config.assets.precompile += %w( *.jpg, *.png )
```

¿Y LAS TIPOGRAFÍAS?

PARA AGREGAR FUENTES DEBEMOS AÑADIRLAS EN EL ASSET PIPELINE

En config/initializers/assets.rb

```
Rails.application.config.assets.paths << Rails.root.join('vendor', 'assets', 'fonts')  
Rails.application.config.assets.precompile += %w( *.svg *.eot *.woff *.ttf *.woff2)
```

Después de modificar archivos de configuración
hay que reiniciar el servidor

BUSCAMOS LAS FUENTES EN EL CSS TENEMOS QUE CAMBIAR LAS REFERENCIAS

```
url('../fonts/glyphicons-halflings-regular.eot');  
url('../fonts/glyphicons-halflings-regular.eot?#iefix') format('embedded-opentype'),  
url('../fonts/glyphicons-halflings-regular.woff2') format('woff2'),  
url('../fonts/glyphicons-halflings-regular.woff') format('woff'),  
url('../fonts/glyphicons-halflings-regular.ttf') format('truetype'),  
url('../fonts/glyphicons-halflings-regular.svg#glyphicons_halflingsregular') format('svg');
```

Ahí es donde tenemos que eliminar el ../fonts y luego envolverlo
ocupando `<%= asset_path "nombrefuente" %>`

DEBERÍA QUEDAR ASÍ:

```
url('<%= asset_path "glyphicons-halflings-regular.eot" %>');  
url('<%= asset_path "glyphicons-halflings-regular.eot?#iefix" %>') format('embedded-opentype'),  
url('<%= asset_path "glyphicons-halflings-regular.woff2" %>') format('woff2'),  
url('<%= asset_path "glyphicons-halflings-regular.woff" %>') format('woff'),  
url('<%= asset_path "glyphicons-halflings-regular.ttf" %>') format('truetype'),  
url('<%= asset_path "glyphicons-halflings-regular.svg#glyphicons_halflingsregular" %>') format('svg');
```

```
<span class="glyphicon-cloud"> hola </span>
```