



Guía de creación de plantillas

v1.1

13 / Mayo / 2014

- [1. Introducción](#)
- [2. ¿Que es una plantilla?](#)
- [3. Configuración](#)
- [4. Estructura y secciones](#)
 - [4.1 ¿Que es una sección?](#)
 - [4.1.1 Clases de sección](#)
 - [4.1.2 Definiendo secciones](#)
 - [4.1.3 Ejemplo de una sección](#)
- [5. Editando el contenido](#)
 - [5.1 Editando los textos de la plantilla](#)
 - [5.2 Editando las imagenes de la plantilla](#)
- [6. Módulos](#)
 - [6.1 ¿Que es un módulo?](#)
 - [6.2 Estructura de un módulo](#)
 - [6.3 Acciones de módulo](#)
 - [6.3.1 Añadiendo modulos](#)
 - [6.3.2 Moviendo modulos](#)
 - [6.3.3 Eliminando modulos](#)
 - [6.3.4 La clase dashed](#)
- [7. Colores](#)
 - [7.1 Formación de clases](#)
- [8. Imágenes](#)
 - [8.1 Imagenes de layout](#)
 - [8.2 Imágenes editables por el usuario](#)
- [9. Temas de iconos](#)
- [10. Probando la plantilla](#)
 - <http://templates.mdirector.com/>

1. Introducción

El editor de plantillas de MDirector es una pieza de software que permite al usuario final, a partir de una plantilla ya creada, realizar unas modificaciones de contenido y aspecto para crear una creatividad que más tarde enviarán por correo electrónico utilizando la plataforma.

El editor de plantillas permite, en el momento de escribir esta guía, realizar unas acciones muy básicas sobre las plantillas. La funcionalidad del editor irá aumentando con el paso del tiempo.

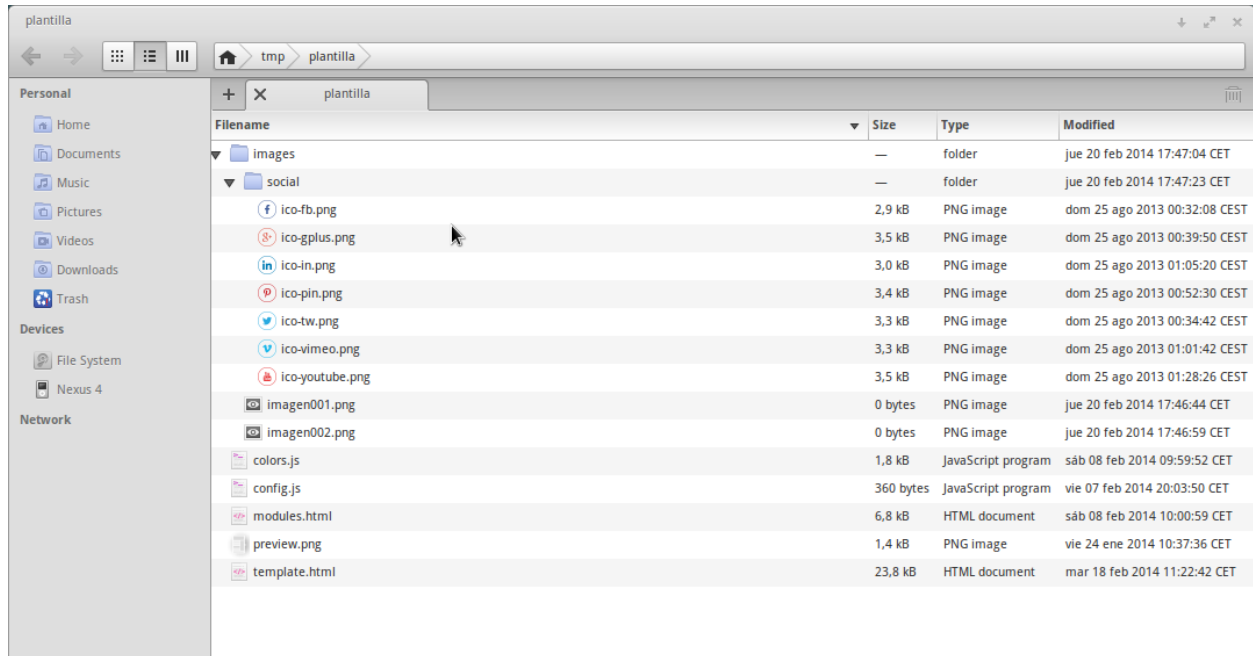
Esta guía pretende orientar a diseñadores y desarrolladores de frontend en la creación de plantillas que sean compatibles con el editor.

En el momento de escribir esta guía, la versión de desarrolladores del editor de plantillas solo es compatible con **Google Chrome**. En futuras revisiones se hará compatible con **Mozilla Firefox**. El uso de **Internet Explorer no es recomendado**.

Esta guía es un documento orientado a personas con un cierto conocimiento del desarrollo HTML, es un documento vivo que irá reflejando los cambios que se vayan produciendo al editor de plantillas de la plataforma. Si encuentras algún bug o tienes alguna sugerencia envía un mail a drin@antevenio.com.

2. ¿Que es una plantilla?

Una plantilla esta formada por un conjunto de ficheros y carpetas agrupados en un archivo .zip, dentro de este archivo pueden aparecer los siguientes ficheros y carpetas:



Filename	Size	Type	Modified
images	—	folder	jue 20 feb 2014 17:47:04 CET
social	—	folder	jue 20 feb 2014 17:47:23 CET
ico-fb.png	2,9 kB	PNG image	dom 25 ago 2013 00:32:08 CEST
ico-gplus.png	3,5 kB	PNG image	dom 25 ago 2013 00:39:50 CEST
ico-in.png	3,0 kB	PNG image	dom 25 ago 2013 01:05:20 CEST
ico-pin.png	3,4 kB	PNG image	dom 25 ago 2013 00:52:30 CEST
ico-tw.png	3,3 kB	PNG image	dom 25 ago 2013 00:34:42 CEST
ico-vimeo.png	3,3 kB	PNG image	dom 25 ago 2013 01:01:42 CEST
ico-youtube.png	3,5 kB	PNG image	dom 25 ago 2013 01:28:26 CEST
imagen001.png	0 bytes	PNG image	jue 20 feb 2014 17:46:44 CET
imagen002.png	0 bytes	PNG image	jue 20 feb 2014 17:46:59 CET
colors.js	1,8 kB	JavaScript program	sáb 08 feb 2014 09:59:52 CET
config.js	360 bytes	JavaScript program	vie 07 feb 2014 20:03:50 CET
modules.html	6,8 kB	HTML document	sáb 08 feb 2014 10:00:59 CET
preview.png	1,4 kB	PNG image	vie 24 ene 2014 10:37:36 CET
template.html	23,8 kB	HTML document	mar 18 feb 2014 11:22:42 CET

template.html (obligatorio)

Este fichero contiene el html de la plantilla con sus hojas de estilo incrustadas en las etiquetas html. Este fichero debe aparecer obligatoriamente en el archivo .zip de la plantilla.

config.js (obligatorio)

Este fichero contiene la configuración de la plantilla, es un fichero javascript compatible con [require.js](#) y contiene una función que devuelve un objeto JSON con los campos de la configuración de la plantilla. Más adelante se entrará en detalle en el contenido de este fichero. Este fichero debe aparecer obligatoriamente en el archivo .zip de la plantilla.

colors.js (obligatorio)

Este fichero contiene las combinaciones o temas de color disponibles para la plantilla, al igual que el fichero config.js, es un fichero javascript compatible con **require.js** y contiene una función que devuelve un objeto JSON con la definición de los diferentes temas de color para la plantilla. Más adelante se entrará en detalle en el contenido del fichero y en la creación de los temas de color. Este fichero debe aparecer obligatoriamente en el archivo .zip de la plantilla.

modules.html (opcional)

Este fichero contiene la definición de los diferentes modulos que se pueden añadir a una plantilla en diferentes los diferentes huecos de la plantilla habilitados específicamente. La plantilla puede tener o no tener módulos para añadir por lo que este fichero no es obligatorio para el funcionamiento de la plantilla. Más adelante se entrará en detalle en la definición de módulo y la manera de añadir, mover y eliminar éstos.

preview.png (obligatorio)

Este fichero es una miniatura de la plantilla, puede ser esquemático, diferenciando los diferentes bloques y la estructura básica de la plantilla, o puede ser una captura de la propia plantilla. El tamaño de esta imagen ha de ser de 150x200 píxeles. Este fichero debe aparecer obligatoriamente en el archivo .zip de la plantilla.

carpeta images (opcional)

Esta carpeta contiene las imágenes que forman parte de la estructura de la plantilla, no del contenido. Estas imágenes no son editables por el usuario y no pueden ser reemplazadas cuando se utilice la plantilla en un envío. Una plantilla puede no necesitar de ninguna imagen por lo que esta carpeta es opcional y puede no aparecer en el archivo de la plantilla. Más adelante se entrará en detalle en cómo utilizar estas imágenes dentro de la plantilla.

carpeta images/social (opcional)

Esta carpeta contiene los iconos para las redes sociales, más adelante se entrará en detalle de cómo se crea un tema de iconos. Esta carpeta es opcional y no tiene por qué aparecer en el archivo de la plantilla.

3. Configuración

La configuración de la plantilla se lleva a cabo utilizando el fichero config.js, como dijimos anteriormente este fichero javascript tiene que ser una función compatible con require.js y devolver un objeto JSON con la configuración de la plantilla. Este fichero es obligatorio y la plantilla no puede funcionar correctamente sin él.

Un fichero estándar tendría el formato que se ve a continuación:

```
define([],function() {  
  
    return {  
        "id": "basic-one-column",  
        "author": "MDirector Team",  
        "name": "Básico 1 Columna",  
        "description": "Template basico de una columna",  
        "responsive": false,  
        "supported": ["firefox", "chrome", "IE9+"],  
        "sections": ["general", "header", "content", "footer", "social"],  
        "tags": ["ocio", "turismo", "viajes"]  
        "social_icon_theme": "cute"  
    };  
  
});
```

a continuación pasamos a explicar cada una de las propiedades de configuración de una plantilla, aunque la mayoría son autoexplicativos, todos los campos han de aparecer en el fichero de configuración de la plantilla, aunque sea a su valor por defecto:

id

Es el identificador único de la plantilla, puede ser cualquier cadena alfanumérica, puede contener guiones altos y bajos, es un identificador de uso interno, no puede repetirse entre plantillas. No tiene valor por defecto.

author

El nombre del autor de la plantilla

name

El nombre de la plantilla, es un identificador que se utilizará en la aplicación, es el nombre que verán los usuarios de la aplicación para referirse a la plantilla, aparece tanto en el selector de plantillas como en el propio editor.

description

Breve descripción de la plantilla, indicando sus principales características y cualidades en un texto corto. Aparece en el selector de plantillas.

responsive

Indica si la plantilla es responsive y se ajusta a diferentes anchos de pantalla y dispositivos o si no lo es, este campo debe tomar el valor true o false.

```

1  define([],function() {
2
3      return {
4          "id": "basic-one-column",
5          "author": "MDirector Team",
6          "name": "Básico 1 Columna",
7          "description": "Ideal para comunicaciones simples y claras",
8          "responsive": true,
9          "supported": ["firefox", "chrome", "IE9+"],
10         "sections": ["general", "header", "content", "footer", "social"],
11         "tags": ["ocio", "turismo", "viajes", "general"]
12         "social_icon_theme": "default"
13     };
14
15 });
16

```



supported

Indica que navegadores estan soportados por la plantilla, a priori todas las plantillas tienen que soportar Chrome, Firefox e Internet Explorer a partir de la versión 9. El valor por defecto para este campo es el que se ve en el código mostrado anteriormente.

tags

Contiene una serie de tags que definen los sectores a los que aplica la plantilla, se utilizaran para poder agrupar, filtrar y buscar plantillas usando estos tags, de esta manera podremos sectorizar las plantillas y ajustarlas a diferentes sectores o grupos de interés. Este campo no tiene valor por defecto y el valor mínimo que debe tener es el de un array vacío []

social_icon_theme

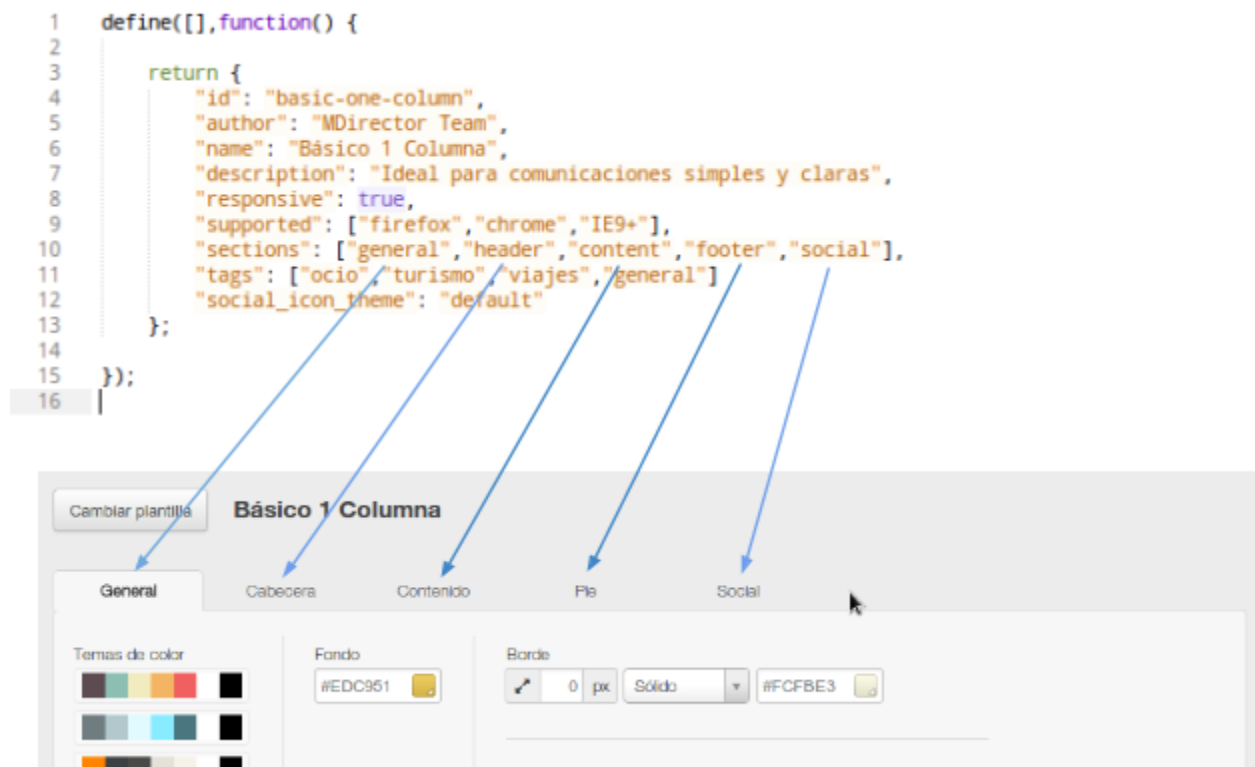
Especifica el tema de iconos para el contenedor de redes sociales, solo se aplica si esta activa la sección social, puede tomar el valor de uno de los temas de iconos predefinidos (**default**, **8bits**, **circled**, **cute**, **flat**, **rounded**, **shaded**, **shadow**, **stained**, **styled**) o bien puede tomar el valor **custom** si la plantilla lleva asociado un tema propio de iconos para redes sociales. En el capítulo dedicado a la creación de temas de iconos entraremos en más detalle en este aspecto.

sections

Define que secciones estan presentes en la plantilla, no todas las plantillas tiene que tener todas las secciones activas, es posible que una plantilla tenga cabecera, contenido y pie y otra ademas pueda tener un sidebar o un contenedor de iconos para redes sociales, actualmente las secciones que podemos especificar en este campo son las siguientes:

- ✓ **general**
- ✓ **header**
- ✓ **content**
- ✓ **footer**
- ✓ **social**
- ✓ **gallery.**

Dependiendo de las secciones que presentes en el fichero de configuración el interfaz del editor de plantillas presentará más o menos pestañas, apareciendo solo las pestañas de configuración para las secciones que estén activas.



4. Estructura y secciones

Las plantillas de MDirector deben tener una estructura mínima y unas secciones definidas para poder ser modificadas por el usuario una vez este personalizando nuestra plantilla.

4.1 ¿Que es una sección?

Una sección es una agrupación lógica de código html que se ve afectada de manera conjunta por los cambios en los elementos de configuración de cada una de las pestañas del editor de plantillas, es decir, nuestra plantilla podrá contener una sección **header** que se verá afectada por los cambios que realicemos en la pestaña **Cabecera** de nuestro editor.

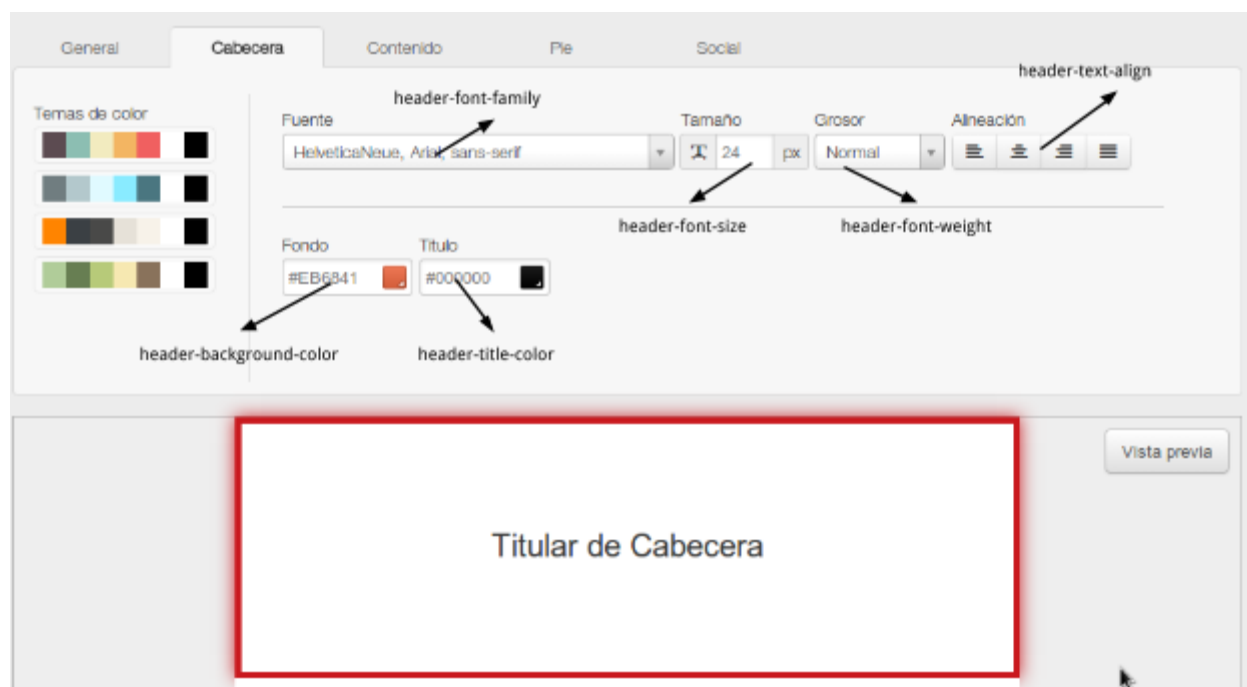
Como es lógico estas secciones son las mismas que aparecen en el fichero de configuración que hemos descrito en el punto anterior. Ninguna sección es obligatoria, pero si en el código de nuestra plantilla no aparecen los tags que definen las diferentes secciones los cambios que realice el usuario cuando use el editor de plantillas no se verán reflejados en nuestra plantilla.

4.1.1 Clases de sección

Para poder modificar el aspecto visual de los elementos englobados dentro de una sección vamos a usar una serie de clases que se forman siguiendo el siguiente patrón:

- todas las clases de una sección empiezan con el nombre de la sección,

A continuación vemos las clases disponibles para la sección header y su correspondencia con los elementos del editor de plantillas.

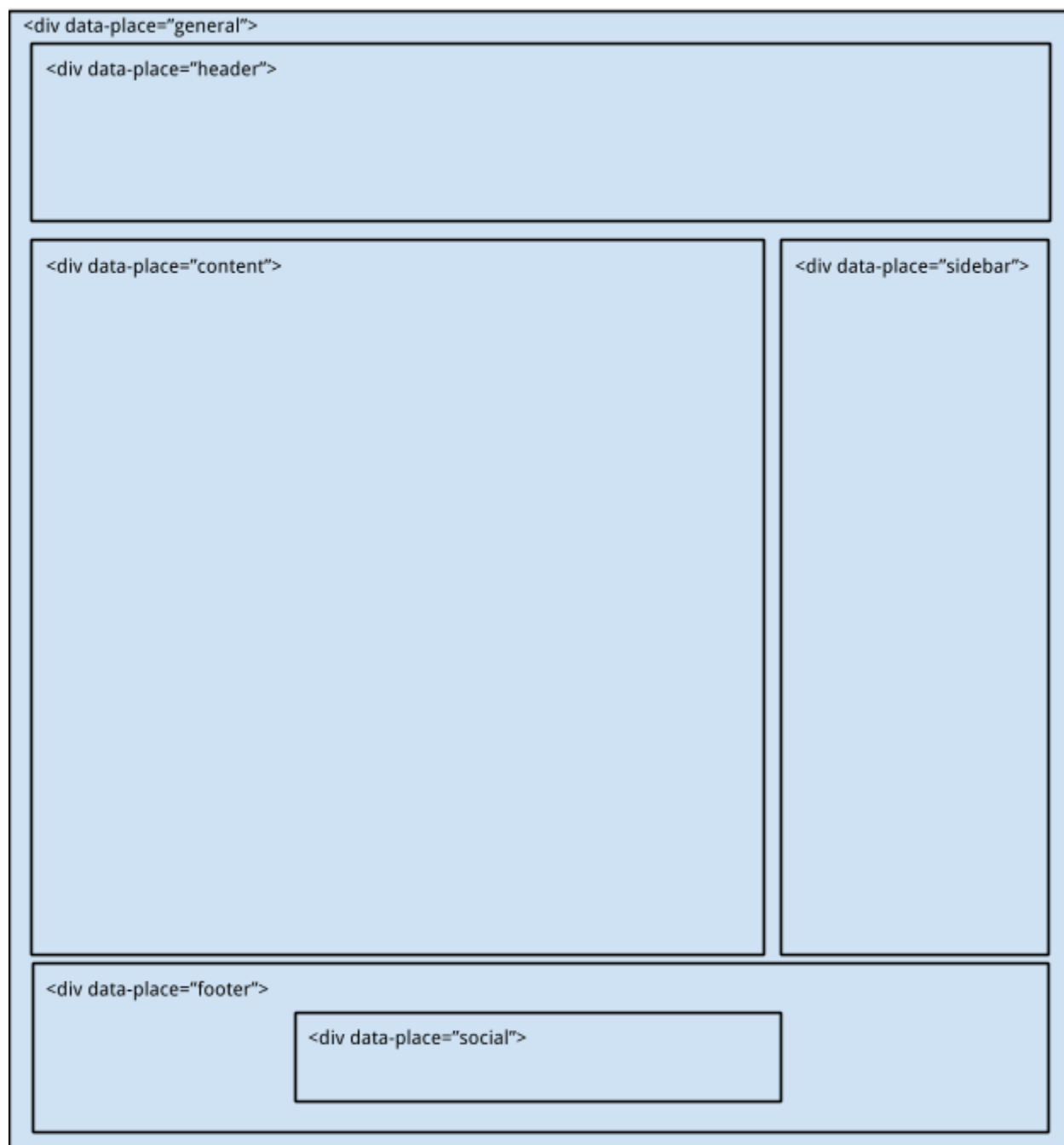


Como hemos podido ver en el ejemplo anterior, las clases disponibles en cada sección están íntimamente relacionadas con los elementos de configuración para cada una de las secciones, el que añadamos una clase

que no existe en una sección no afecta al buen funcionamiento de la plantilla, simplemente es ignorada.

4.1.2 Definiendo secciones

Para definir las diferentes secciones vamos a usar un *data attribute* de html, más concretamente el atributo **data-place**, todo lo que esté dentro de un tag con un atributo data-place se verá afectado por la configuración de esa sección: enlaces, textos, bordes, tipos y tamaños de letra, etc, etc.



Una sección puede contener otra sección dentro de ella, como podemos ver en el esquema, la sección **social** que

contiene los enlaces a redes sociales, está contenida dentro de la sección **footer**.

4.1.3 Ejemplo de una sección

A continuación se muestra un ejemplo de una sección de una plantilla de las que ya están funcionando en el editor de MDirector, con sus clases aplicadas y su atributo **data-place** fijado convenientemente.

```
<table width="100%" cellpadding="0" cellspacing="0" border="0" align="left" class="table
footer-background-color footer-font-family footer-font-size footer-text-color footer-text-align"
data-place="footer" style="font-family: HelveticaNeue, Arial, sans-serif; font-size: 12px; color:
#393939; background-color: #fff; width: 100%; ">
  <tbody>
    <tr>
      <td valign="middle" class="footer-font-size footer-text-color footer-text-align"
style="padding: 20px;margin:20px;color:inherit; font-size: inherit; font-family: inherit;text-align:
inherit;line-height: 1em;">
        <div contenteditable="true" class="contenteditable">
          © 2014 <span>MDirector España</span>
          |
          <a href="#" target="_blank" class="footer-link-color" style="text-decoration:
underline;color: #393939;">Política de Privacidad</a>
          |
          <a href="#" target="_blank" class="footer-link-color" style="text-decoration:
underline;color: #393939;">Aviso Legal</a>
          |
          <a href="#" target="_blank" class="footer-link-color" style="text-decoration:
underline;color: #393939;">Contacte con nosotros</a>
        </div>
      </td>
    </tr>
  </tbody>
</table>
```

Más allá de las clases específicas de cada sección y el atributo data-place, el HTML de cada sección es normal y corriente, teniendo siempre en cuenta las consideraciones y las restricciones de los lectores de correo actuales, así podemos ver que en el ejemplo anterior, la práctica totalidad de los elementos HTML llevan un atributo style que define su estilo inicial en la plantilla.

5. Editando el contenido

Una de las funcionalidades más importantes de las plantillas es que el usuario pueda editar el contenido de las mismas, principalmente y en el momento de escribir esta guía, el usuario puede editar dos tipos de contenidos: textos e imágenes.

A continuación veremos cómo habilitar la edición de contenidos, tanto textos como imágenes.

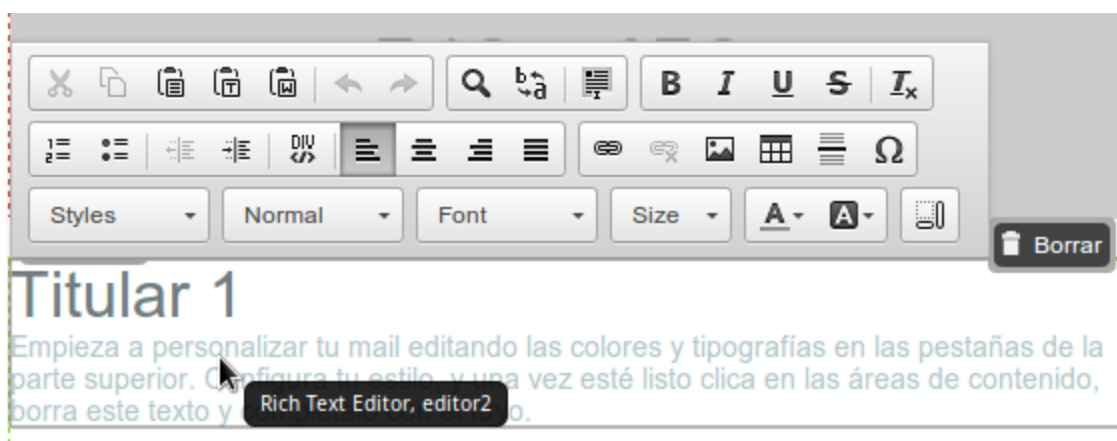
5.1 Editando los textos de la plantilla

Para habilitar la edición de textos vamos a usar una clase y un atributo en el contenedor del texto que queramos hacer editable, el contenedor editable no tiene por qué contener solamente texto, puede contener HTML, el usuario será capaz de editarlo utilizando la herramienta. La clase que debemos utilizar es `contenteditable` y el atributo se llama también `contenteditable` y debe adquirir valor `true` en la plantilla y valor `false` en el fichero de módulos `modules.html`.

A continuación un ejemplo de un contenedor de texto editable:

```
<div contenteditable="true" class="contenteditable">  
  <span class="content-title-color general-font-family content-font-family">  
    Título del modulo  
  </span>  
  <br />  
  <p class="content-text-color general-font-family content-font-family">  
    Empieza a personalizar tu mail editando los colores y tipografías en las pestañas de la parte superior. Configura tu estilo, y una vez esté listo clicka en las áreas de contenido, borra este texto y complétalo con el tuyo.  
  </p>  
</div>
```

cuando el usuario haga clic dentro de un contenedor editable verá un pequeño editor basado en [CKEditor](#) que le permitirá editar con todo detalle el contenido del mismo, como se muestra en la siguiente imagen:



5.2 Editando las imagenes de la plantilla

El capítulo 8 de esta guía entra en detalle en el uso de imagenes dentro de las plantillas.

6. Módulos

Las diferentes secciones de una plantilla están compuestas de HTML, dentro de este código se pueden habilitar huecos específicos que permiten la manipulación por parte del usuario de determinadas partes del código que se denomina módulos. A continuación veremos que es un módulo y que podemos hacer con él.

6.1 ¿Que es un módulo?

Un módulo es un pedazo de código HTML que puede ser insertado, eliminado o movido por el usuario dentro de la plantilla. Los módulos que forman parte de la plantilla vienen definidos en el fichero **modules.html**. Al código de un módulo se le aplican las mismas reglas que al HTML del resto de la plantilla, es decir, debemos aplicar las mismas clases que al resto de la plantilla si queremos que el usuario pueda modificar el aspecto del contenido del módulo.

En el momento de escribir esta guía no existen huecos en la plantilla que puedan cargar más de un tipo de módulo diferente, está planificado que se puedan insertar diferentes tipos de módulo en un mismo slot de la plantilla.

6.2 Estructura de un módulo

Un módulo no es más que un pedazo de HTML que habita en el fichero **modules.html** y que tiene un nombre para que el editor sea capaz de encontrarlo e insertarlo dentro de la plantilla.

Para dotar de un nombre a un módulo debemos asignar una clase con un nombre único dentro de esa plantilla, es decir, dentro de la misma plantilla no puede haber dos módulos con el mismo nombre.

A continuación se muestra un ejemplo de módulo:

```
<div class="mod-newimage">
  <div class="removeFromHere movable">
    <table width="100%" cellpadding="0" cellspacing="0" border="0" align="left" class="mod-imagen
dashed" data-button="borrar mover" style="margin-bottom: 20px;">
      <tbody>
        <tr>
          <td style="" valign="top">
            <div contenteditable="true" class="contenteditable image" data-maxWidth="563"
data-maxHeight="150">
              
            </div>
          </td>
        </tr>
      </tbody>
    </table>
    <br style="clear:both;">
  </div>
</div>
```

En el ejemplo anterior la clase que identifica al módulo dentro de la plantilla es **mod-newimage** y como vemos este módulo añade una imagen editable por el usuario, de momento nos vamos a olvidar de las clases **removeFromHere** y **movable** que vemos en el primer div del módulo, pero diremos que son las encargadas de habilitar las acciones de mover y eliminar el módulo.

6.3 Acciones de módulo

En el momento de escribir esta guía se pueden llevar a cabo tres acciones básicas con un módulo dentro de una plantilla: **añadir**, **mover** y **eliminar**,

También se puede editar su contenido igual que el resto del contenido de la plantilla.

A continuación veremos cómo habilitar estas tres acciones en la plantilla.

6.3.1 Añadiendo módulos

Para habilitar la posibilidad de añadir módulos en la plantilla debemos habilitar en nuestra plantilla el botón de **Añadir**, para ello debemos añadir el siguiente código al final del contenedor donde vayamos a insertar el módulo, es decir, cuando el usuario haga clic en el botón añadir el nuevo módulo aparecerá inmediatamente encima del botón **Añadir**.

```
<div class="button-container" id="add-module-trigger">
  <button type="button" class="anadir" data-module-type="mod-newimage">
    <i class="ico-add-content"></i>&nbsp;<span>Añadir</span>
  </button>
</div>
```

Como vemos, el botón añadir tiene una clase que indica la acción del botón, en este caso **anadir**, y *data attribute* que indica cual es el módulo que vamos a insertar, en nuestro caso vemos que el **data-module-type** vale **mod-newimage**, que es el módulo que hemos definido con anterioridad en el fichero **modules.html** de nuestra plantilla. Cuando el usuario haga click en el botón Añadir, el editor de plantillas buscare el modulo mod-newimage en el fichero modules.html, hará una copia de su contenido y lo insertará en nuestra plantilla.

6.3.2 Moviendo módulos

Para habilitar la posibilidad de mover módulos dentro de un contenedor debemos hacer uso, una vez más, de clases y atributos específicos. El usuario no puede mover los módulos a su antojo de un sitio a otro de la plantilla, solo poder intercambiar unos módulos con otros si estos comparten el mismo contenedor habilitado al efecto. Es decir, debemos crear un contenedor de módulos intercambiables y dentro de este habilitar el botón mover dentro de los módulos.

Para habilitar el botón mover debemos utilizar el atributo `data-button` dentro del módulo y añadir la clase `movable` al contenedor de nuestro módulo, exactamente igual que hicimos para habilitar el borrado de módulos.

Un ejemplo de módulos intercambiables quedaría de la siguiente manera:

```
<div class="mod-twoimages">
  <div class="moveArea">
    <div class="movable" id="image-01">
      <table width="100%" cellpadding="0" cellspacing="0" border="0" align="left"
class="mod-imagen dashed" data-button="mover" style="margin-bottom: 20px;">
        <tbody>
          <tr>
            <td style="" valign="top">
              <div contenteditable="true" class="contenteditable image"
data-maxWidth="563" data-maxHeight="150">
                
              </div>
            </td>
          </tr>
        </tbody>
      </table>
      <br style="clear:both;">
    </div>
    <div class="movable" id="image-02">
      <table width="100%" cellpadding="0" cellspacing="0" border="0" align="left"
class="mod-imagen dashed" data-button="mover" style="margin-bottom: 20px;">
        <tbody>
          <tr>
            <td style="" valign="top">
              <div contenteditable="true" class="contenteditable image"
data-maxWidth="563" data-maxHeight="150">
                
              </div>
            </td>
          </tr>
        </tbody>
      </table>
    </div>
  </div>
</div>
```

vemos el contenedor con la clase **moveArea** y dentro de ella, dos divs con la clase **movable** y dentro de ellos en algún lugar de nuestro módulo, más concretamente en el tag `table`, vemos el atributo **data-button** con valor `mover`. El usuario al pasar por encima de cualquiera de las dos imágenes habilitaría el botón Mover y podría arrastrar las imágenes e intercambiarlas entre ellas.

6.3.3 Eliminando módulos

Para habilitar la posibilidad de eliminar un módulo añadido previamente o un módulo de los originales que conforman nuestra plantilla hemos de habilitar el botón **Eliminar** en el módulo que deseamos tenga la posibilidad de ser eliminado de la plantilla, para habilitar los botones de acción dentro de un módulo se utiliza un *data attribute* específico y una clase.

Para poder eliminar un módulo éste debe estar englobado en un div con la clase **removeFromHere** y en algún elemento de su contenido debe aparecer el atributo data-button que debe contener el valor **borrar**.

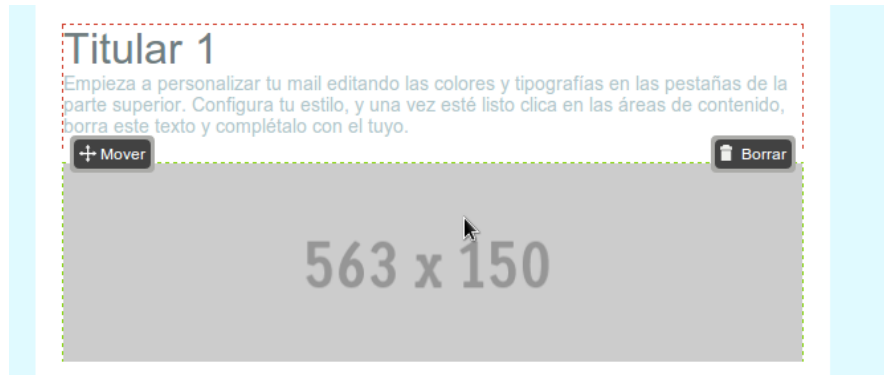
```
<div class="mod-newimage">
  <div class="removeFromHere">
    <table width="100%" cellpadding="0" cellspacing="0" border="0" align="left" class="mod-imagen
dashed" data-button="borrar" style="margin-bottom: 20px;">
      <tbody>
        <tr>
          <td style="" valign="top">
            <div contenteditable="true" class="contenteditable image" data-maxWidth="563"
data-maxHeight="150">
              
            </div>
          </td>
        </tr>
      </tbody>
    </table>
    <br style="clear:both;">
  </div>
</div>
```

Volviendo al ejemplo de nuestro módulo, vemos que el contenido está englobado en un div que contiene las clases **removeFromHere**, y además, vemos que el tag **table** contiene el atributo **data-button** con valor **borrar**.

Cuando el usuario haga clic en el botón Borrar del módulo, el editor de plantillas buscará el div padre del botón Borrar que contenga la clase **removeFromHere** y lo eliminara de la plantilla.

6.3.4 La clase dashed

Para indicar al usuario que puede realizar acciones sobre un módulo, vamos a utilizar la clase **dashed**, como vemos en el ejemplo anterior, el tag table del módulo tiene la clase dashed entre otras, lo que hace esta clase es añadir un borde rojo y discontinuo a nuestro módulo, al pasar por encima con el ratón, el borde se vuelve verde , indicando al usuario que puede interactuar con esa parte de la plantilla.



7. Colores

Las plantillas de MDirector soportan diferentes variaciones de colores o temas, los temas de color para cada plantilla vienen definidos en el fichero **colors.js**, este fichero es obligatorio y debe contener al menos un tema de color para cada plantilla.

La estructura del fichero colors.js, como se mencionó anteriormente es la de un fichero javascript compatible con require.js que implementa una función que devuelve un objeto JSON con los diferentes temas de color para una plantilla.

Un fichero colors.js estándar tendría el siguiente formato:

```
1  /**
2   * Created by diego on 11/28/13.
3   */
4  define([],function() {
5
6      return {
7          "VintageCard": {
8              "colors": ["#5C4B51", "#8CBEB2", "#F2EBBF", "#F3B562", "#F06060", "#FFFFFF", "#000000"],
9              "general": {
10                 "background": "#F2EBBF",
11                 "text": "#8CBEB2",
12                 "link": "#F06060",
13                 "title": "#5C4B51"
14             },
15             "header": {
16                 "background": "#8CBEB2",
17                 "text": "#000000",
18                 "link": "#F06060",
19                 "title": "#5C4B51"
20             },
21             "content": {
22                 "background": "#FFFFFF",
23                 "text": "#8CBEB2",
24                 "link": "#F06060",
25                 "title": "#5C4B51"
26             },
27             "footer": {
28                 "background": "#8CBEB2",
29                 "text": "#000000",
30                 "link": "#FFFFFF",
31                 "title": "#000000"
32             },
33             "social": {
34                 "background": "#FFFFFF",
35                 "text": "#8CBEB2",
36                 "link": "#F06060",
37                 "title": "#5C4B51"
38             }
39         }
40     }
41 }
42 |
43 });
44
```

Podemos ver que cada tema de color va identificado con un nombre, en la imagen superior vemos que esta definido el tema **VintageCard**, cada tema de color define un array de siete colores, que son los colores que forman el tema, aparte de este array se define un objeto por cada sección y de uno a n valores dentro de cada objeto de sección, los valores que podemos modificar para cada sección son los siguientes:

- ✓ **background:** define el color del fondo de esa sección
- ✓ **text:** define el color del texto para esa sección
- ✓ **link:** define el color de los enlaces para esa sección
- ✓ **title:** define el color de los títulos para esa sección
- ✓ **border:** define el color del borde para esa sección (actualmente solo soportado en la sección general)

no todas las secciones tienen que implementar todos los valores, por ejemplo, podemos solo querer modificar color del texto y de los enlaces y no tocar el resto.

The image shows a code editor on the left and a design tool on the right. The code editor displays a JSON configuration for a theme named 'VintageCard'. The configuration includes a 'colors' array with seven color codes and objects for different sections: 'general', 'header', 'content', 'footer', and 'social'. Each section object contains properties for 'background', 'text', 'link', and 'title'. The design tool on the right shows a 'Temas de color' (Color Themes) palette with various color swatches. Below the palette, there's a 'Fuentes' (Fonts) section with a font family dropdown set to 'HelveticaNeue, Arial, sans-serif' and a font size of 24 px. At the bottom, there are color pickers for 'Fondo' (Background), 'Texto' (Text), 'Títulos' (Titles), and 'Enlaces' (Links). Arrows connect the code properties to the corresponding UI elements: 'background' to the 'Fondo' picker, 'text' to the 'Texto' picker, 'link' to the 'Enlaces' picker, and 'title' to the 'Títulos' picker. The 'Fondo' picker is set to '#FFFFFF', 'Texto' to '#000000', 'Títulos' to '#009FB0', and 'Enlaces' to '#FAFABE'.

7.1 Formación de clases

A partir del fichero colors.js se forman una serie de clases que podemos utilizar en las diferentes secciones de nuestra plantilla, la formación de clases sigue un patrón como el que se muestra en la imagen siguiente.

```
27      "footer": {
28          "background": "#8CBEB2", footer-background-color: #8CBEB2
29          "text": "#000000", footer-text-color: #000000
30          "link": "#FFFFFF", footer-link-color: #FFFFFF
31          "title": "#000000" footer-title-color: #000000
32      },
33      "social": {
```

las clases solo estarán disponibles en su sección de la plantilla a la que pertenecen, siguiendo el ejemplo tendríamos las clases footer-background-color, footer-text-color, footer-link-color y footer-title-color, que debemos aplicar a los elementos correspondientes de nuestra sección footer, siguiendo con otro ejemplo, nuestro pie de una plantilla quedaría de la siguiente manera con sus clases ya aplicadas, en **negrita** las clases del ejemplo anterior.

```
<div class="removeFromHere">
  <table width="100%" cellpadding="0" cellspacing="0" border="0" align="left" class="table
  footer-background-color general-font-family footer-font-family footer-font-size footer-text-color footer-text-align"
  data-place="footer" style="font-family: HelveticaNeue, Arial, sans-serif; font-size: 12px; color: #393939;
  background-color: #fff; width: 100%; ">
    <tbody>
      <tr>
        <td valign="middle" class="dashed footer-font-size footer-text-color footer-text-align"
        data-button="borrar" style="padding: 20px; margin: 20px; color: inherit; font-size: inherit; font-family:
        inherit; text-align: inherit; line-height: 1em; ">
          <div contenteditable="true" class="contenteditable">
            <a href="#" target="_blank" class="footer-link-color" style="text-decoration: underline; color:
            #393939;" >Política de privacidad</a>
            |
            <a href="#" target="_blank" class="footer-link-color" style="text-decoration: underline; color:
            #393939;" >Terminos legales</a>
          </div>
        </td>
      </tr>
    </tbody>
  </table>
</div>
```

Para que una clase tenga efecto en el plantilla ha de estar definida en el fichero de colores, es decir, que por mucho que utilicemos la clase footer-background-color en nuestra plantilla si no hay una entrada background en la sección footer del fichero, no podremos utilizarla

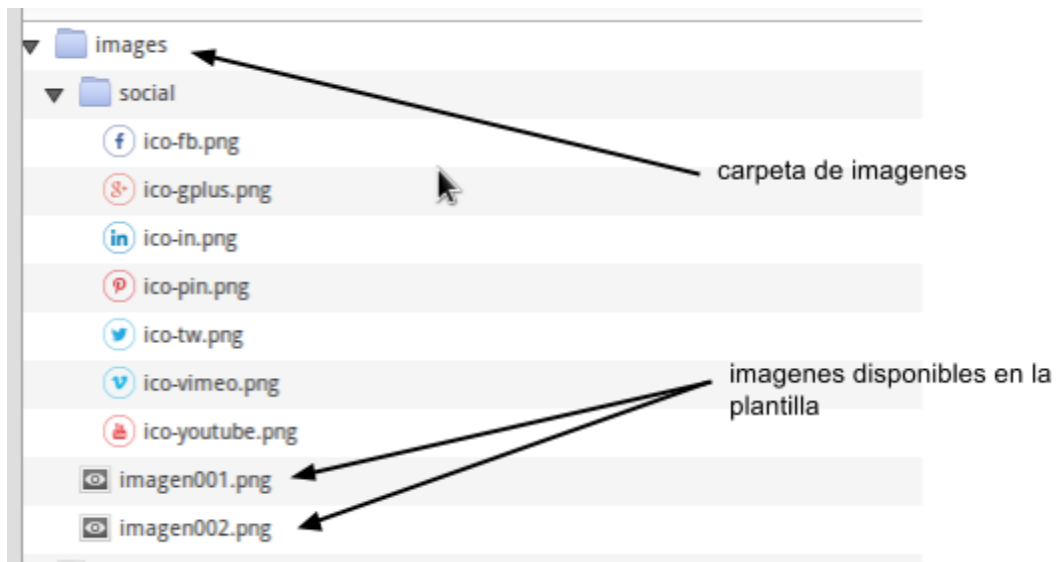
8. Imágenes

Existen dos tipos de imágenes que podemos usar en una plantilla, las imágenes de layout y las imágenes de contenido, las primeras forman parte de la estructura de la plantilla, por ejemplo, iconos, bordes y fondos, estas imágenes no son editables por el usuario.

El segundo tipo de imágenes son las que forman parte del contenido de la plantilla, estas imágenes son editables por el usuario utilizando la herramienta de recortar y subir imágenes del editor, a continuación explicaremos como hacer uso de ambos tipos de imágenes.

8.1 Imágenes de layout

Estas imágenes forman parte de la estructura de la plantilla, para poder hacer uso de estas imágenes hemos de colocarlas dentro de la carpeta **images** que va dentro del fichero zip de la plantilla, dentro de esta carpeta podremos meter un número ilimitado de imágenes que luego podremos usar desde dentro de la plantilla.



Para poder utilizar las imágenes que hemos subido en nuestra plantilla las debemos referenciar de la siguiente forma:

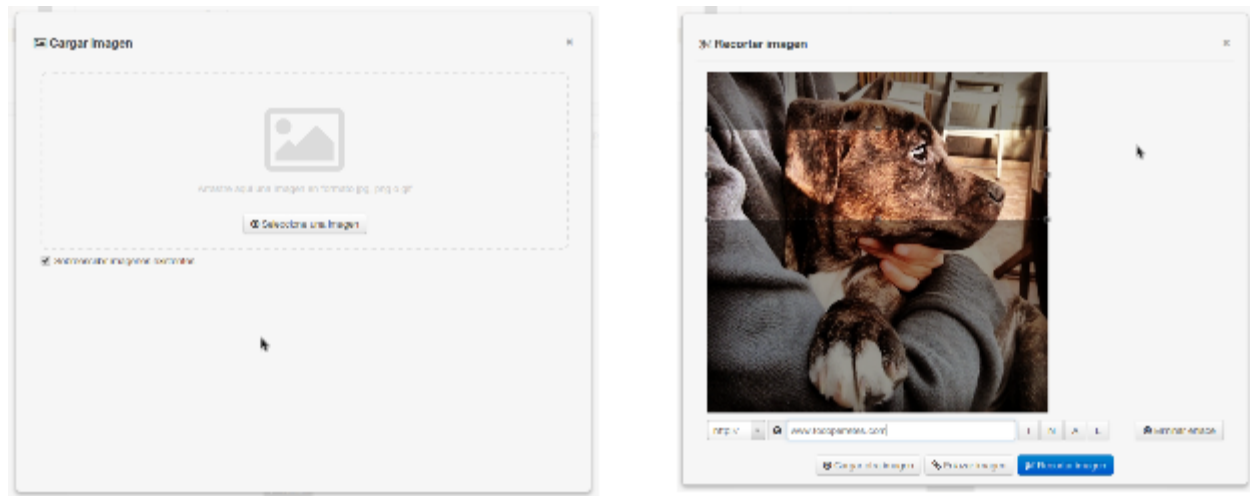
```

```

Las imágenes de layout deben llevar **obligatoriamente** la clase **layoutImage**. El tag `<%= imageDir %>` es sustituido por el directorio real que contiene las imágenes cuando se procesa la plantilla y se convierte en una creatividad válida.

8.2 Imágenes editables por el usuario

El segundo tipo de imágenes que podemos utilizar en nuestras plantillas son aquellas que más tarde el usuario, cuando esté haciendo uso de nuestra plantilla, podrá modificar estas imágenes al pinchar sobre ellas utilizando el diálogo de **Recortar y Subir Imagen**.



Para activar el diálogo de **Recortar y Subir Imagen** debemos añadir una serie de clases y seguir una serie de indicaciones a la hora de escribir el html de la plantilla.

```
<div contenteditable="true" class="contenteditable image" data-maxWidth="563" data-maxHeight="150">
  
</div>
```

Como vemos en el trozo de código HTML anterior, para poder hacer modificable una imagen de nuestra plantilla debemos cumplir los siguientes requisitos:

- ✓ Las imágenes pueden formar parte de la plantilla y deben ser incluidas en la carpeta `images` del zip de la plantilla. En caso de no querer adjuntar imágenes editables por el usuario en nuestra plantilla deberemos usar un placeholder como se indica en el punto siguiente.
- ✓ La imagen puede ser un placeholder proporcionado por <http://placeholder.it> (En el momento de escribir esta guía solo se soporta `placeholder.it`, en futuras versiones se añadirá soporte para cualquier tipo de imagen)
- ✓ La imagen a editar debe estar contenida en un `div`, este `div` debe tener las clases **contenteditable** e **image**, además, debe tener el atributo **contenteditable** a **true**. Si la imagen forma parte de un módulo del fichero `modules.html` el atributo `contenteditable` deberá estar fijado a **false**.
- ✓ El `div` contenedor debe tener **obligatoriamente** los data attributes **maxWidth** y **maxHeight**, estos atributos contienen el ancho y alto de la imagen, las imágenes se escalan proporcionalmente a estas medidas si son más grandes. Si el usuario intenta subir una imagen más pequeña que estas medidas recibirá un error.

9. Temas de iconos

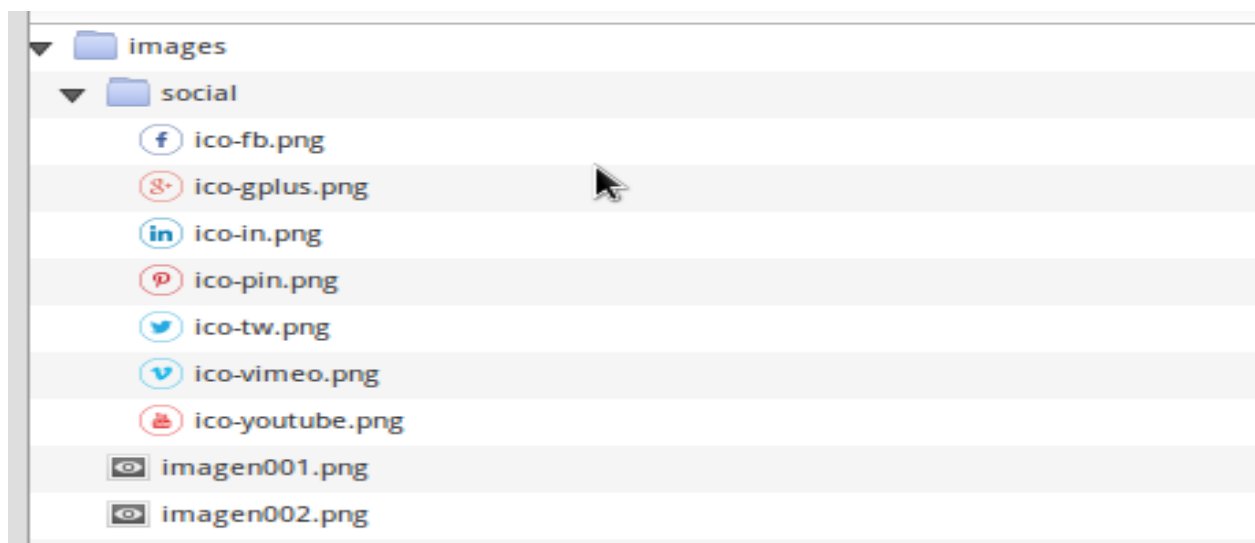
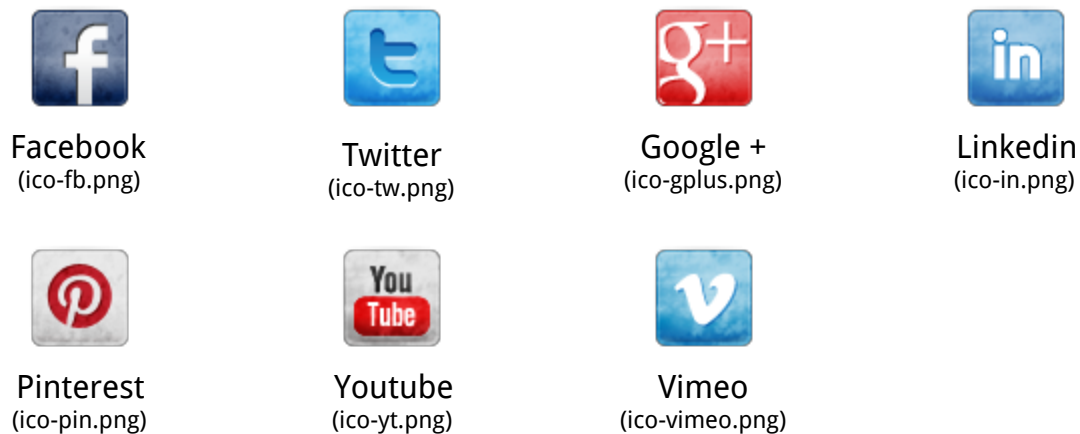
Un tipo de imagen particular dentro de las plantillas son los iconos de redes sociales, podemos añadir nuestro propio tema de iconos, en el momento de escribir este documento solo esta soportado un tema de iconos personalizado por plantilla, pero en futuras actualizaciones, se podrán añadir ilimitados.

Para poder añadir un tema de iconos de redes sociales a nuestra plantilla debemos incluir el directorio **social** con nuestros iconos, dentro del directorio images del zip de nuestra plantilla, dentro de esta carpeta habra un icono por cada red social soportada por la plataforma.

El formato de los iconos ha de ser siempre el de imágenes png, el tamaño puede variar entre los tamaños recomendados que van de **16x16** a **48x48**.

Para conseguir una coherencia estética todos los iconos del tema deberían tener el mismo tamaño. El nombre de los iconos tiene que ajustarse a los requerimientos de la plataforma.

A continuación se detallan las redes sociales soportadas y los nombres de los iconos que deben aparecer en la carpeta del tema:



10. Probando la plantilla

Para probar el diseño de nuestra plantilla y que todo funciona correctamente podemos usar la versión de desarrollo del editor de plantillas, para ello debemos solicitar un acceso al equipo de desarrollo de MDirector y una vez que nos lo hayan concedido podremos acceder al tester de plantillas en la siguiente url:

<http://templates.mdirector.com/>

Una vez allí, podremos cargar nuestra plantilla en formato zip, usando el boton **Cargar Plantilla** de la esquina superior izquierda, si nuestra plantilla está correctamente formada, podremos interactuar con ella de manera en que lo haría un usuario final en la herramienta, ya que la versión de desarrollo del editor tiene el mismo comportamiento que la versión del usuario final, con alguna mínima diferencia.

