

Trabajo colaborativo de DES, DIW y DEC.

**Aplicación bancaria**

Andrei García Cuadra

**Andrei García Cuadra**

Contents

# Preparación y puesta en marcha

La aplicación consta de 3 apartados (o subaplicaciones):

* **API REST**: Es de donde se recoge la información. Para poder usarlo se necesitan claves de la api. Está escrito en ***Java 8***.
* **Aplicación bancaria (Vista)**: Aplicación mediante la cual los clientes del banco podrán interactuar, ingresar dinero, retirarlo, abrir cuentas... Etcétera. Está escrito con ***SASS 5***, LESS
* **Aplicación externa (Vista)**: Aplicación de terceros que permite realizar ingresos en el banco.

## Requerimientos

El proyecto requiere de tener instaladas ciertas dependencias, en función de la aplicación a desarrollar.

### Dependencias de la aplicación bancaria (Vista)

* LESS ([Guía de instalación](https://malnuer.es/css/como-configurar-netbeans-para-usar-less-en-windows/)).
* SASS ([Guía de instalación](https://stackoverflow.com/questions/25965982/how-to-use-sass-with-netbeans-8-0-1)).
* Servidor web (integrado).
* NPM.

### Dependencias del api rest (Servidor)

### Dependencias de la aplicación con api externa (Vista)

## Proyecto e información

El proyecto ha sido desarrollado mediante el IDE Netbeans.

El proyecto integra la precompilación de SASS, LESS y optimizaciones a la hora de compilar el proyecto final, de modo que integra el entorno de desarrollo automáticamente en el IDE, y el entorno de producción en la compilación, facilitando así la rápida integración y despliegue de la aplicación.

## Jerarquía, organización y orden

Para entender este apartado, separaremos todo en el patrón de diseño MVC. Debemos distinguir dos tipos de programas:

* **Programas de servidor**: Su jerarquía estará ligada a M\*C, de modo que obtendremos distintas capas relacionadas con el servidor, tales como la capa de DAO para el acceso a datos, la capa de SERVICIOS para el tratamiento de datos, la capa HTTP para el acceso mediante web (si es oportuno), etc. Llevando éstos nombres las carpetas (o paquetes) contenedores.
* **Programas de cliente:** Su jerarquía será de árbol, es decir, se crearán distintas carpetas para separar los tipos de archivos y en función de su rango de importancia, siendo la **carpeta base la raíz**, la **carpeta de estilos assets**, la **carpeta de imágenes img**, etc.

# api rest

# aplicación del banco (oficial)

La idea básica de esta aplicación es que tenga los colores y el estilo del banco BBVA. Asimismo, se ha utilizado el framework [Materialize](http://materializecss.com/) para otorgar un estilo acorde a lo requerido.

El proyecto contiene varios archivos en la ruta base que pueden no ser de importancia si tan sólo se quiere la compilación. La compilación se ubica en public\_html, y contiene los ficheros \*.html, \*.css y \*.js ya compilados y minimizados por los precompiladores. La ruta base contiene los archivos para la compilación (NPM,SASS,LESS,...). Se ha elegido minificar todos estos archivos para ahorrar ancho de banda.

## SASS

El fichero que contiene las variables que se han utilizado está ubicado en *sass/components/\_variables*.scss. Se han modificado las variables nativas de materialize del siguiente modo:

* $primary-color: color("blue", "darken-1") !default;

## LESS

## Logotipo e imágenes

Como logotipo se ha utilizado el oficial del banco BBVA, el cual ha sido minimizado y comprimido con la herramienta photoshop. El resto de imágenes han sido tratadas para acordarlas al contenido de la página y, sobretodo, reducir su considerable tamaño (sobre todo en las imágenes de fondo). Todas las imágenes poseen copyrights relacionados con el banco BBVA, por lo tanto, son imágenes internas de la empresa, por lo tanto están en la más absoluta legalidad.

Para mantener la identidad de la página, se ha añadido un favicon con el mismo logo que el de la empresa.

## Menú y páginas

El menú se ha tratado de simplificar al máximo, separando tan sólo las opciones básicas. Así, se ha llegado a tener las páginas básicas (excepto conexión) en el menú.

* Conexión.
  + Al conectar se comprueba que el PIN sea de 3 dígitos y el DNI sea válido tanto en el cliente con en el servidor.
  + Envía una petición AJAX al servidor sin api-key, ya que la cabecera el dominio es la misma y el servidor lo acepta.
* Ingresos y reintegros.
* Apertura de cuentas.
* Listado de movimientos.
* Cierre de cuentas.

## Accesibilidad

## Responsive

Se han utilizado siempre medidas relativas, además de un grid de 12 celdas optimizado para pantallas largas, medianas y pequeñas. De este modo, la web se visualizará correctamente desde cualquier dispositivo.

## Optimización

## Cumplimiento de los estándares

## Cross-browser

# aplicación externa (no oficial, con acceso api)