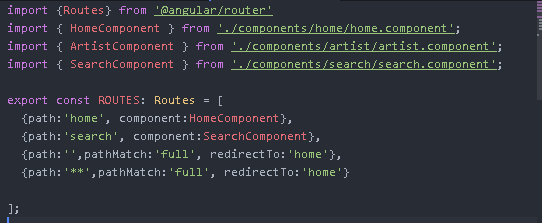
Crear componentes ng c components/home…

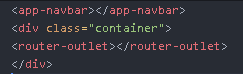
Crear rutas:  
Crear archivo app.routes.ts dentro de app que contiene las rutas donde se va a estar navegando, los componentes.  


Y en el archivo module agregarlo en los imports  




Añadiendo al html

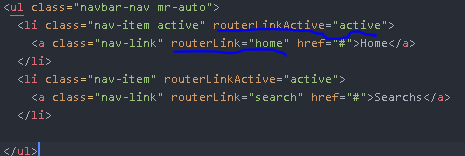
En components.html



Haciendo el que el navbar funcione

Primero añadimos una plantilla de navbar desde Bootstrap

Luego en los li



Peticiones HTTP

<https://restcountries.eu/>

Por ejemplo para hacer una petición de los países por idioma se debe hacer a :



Desde postman introducir el url para recibir un objeto json

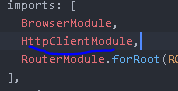


Para recibir estos servicios hay que incluir en modulos de angular HTTP



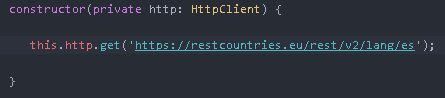
IMPORTANTE Importar también

 import { HttpModule } from '@angular/http';

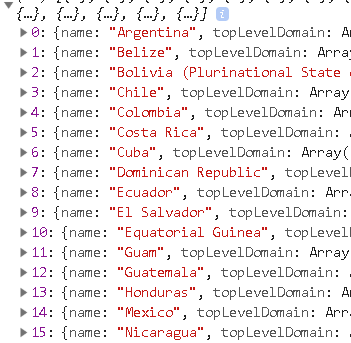
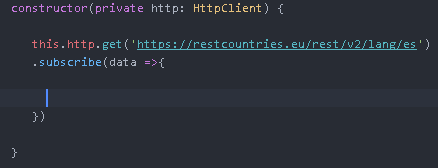


Ahora dentro del home component

Creamos la variable http de tipo HttpClient que va a ser usada dentro del constructor para llamar la peticion

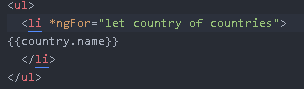


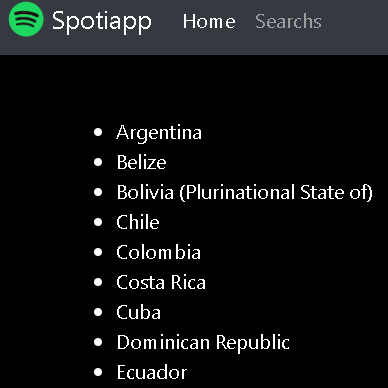
Ahora creamos lasubscripcion con la función de flecha



Ahora si por ejemplo quiero mostrar datos de los países enparticular, se declara un arreglo de tipo any , igual se cambia el valor de data a any

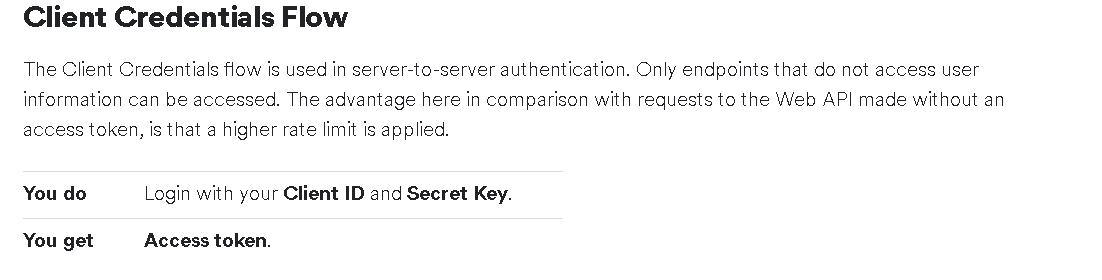
Ahora en el html





Uso del token para trabajar conel servicio de Spotify

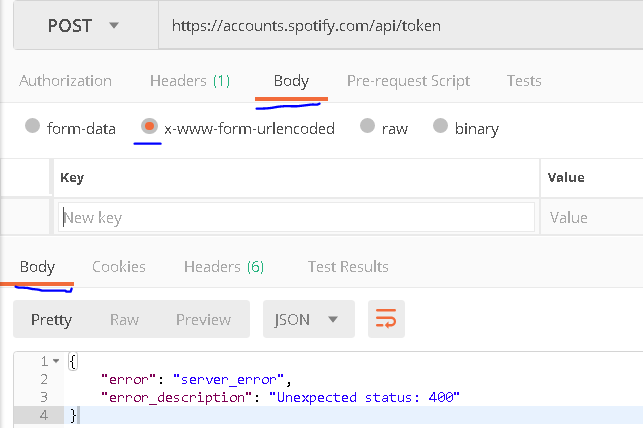
Desde la pagina de Spotify developer llegar a https://developer.spotify.com/documentation/general/guides/authorization-guide/

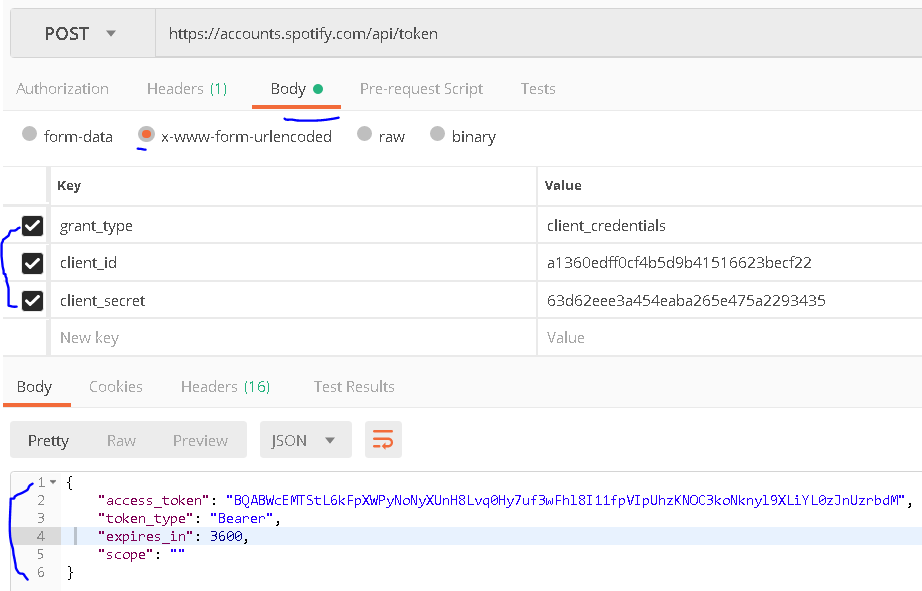


Debemos hacer la petición desde el postman con un POST de este url

<https://accounts.spotify.com/api/token>

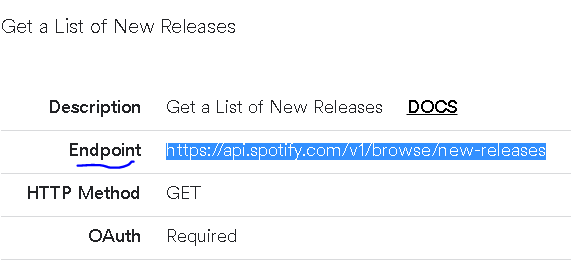
El cual nos va a tirar error por que no se ha hecho la autenticación



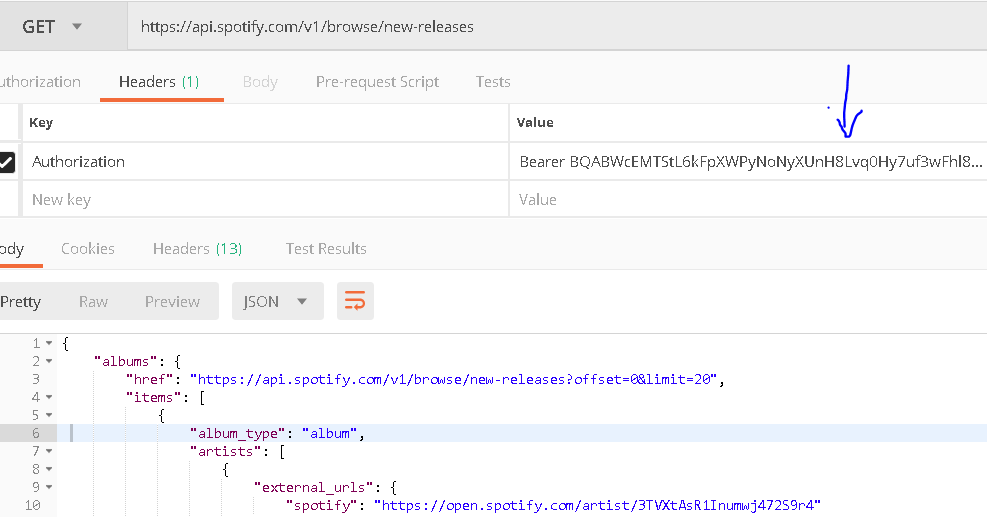


Para la primera interaccion de Spotify vamos al develover de Spotify a web console

Get new releases



Utilizamos este url para ir a postman y probarlo  
Necesitamos ingresar el url y un key con la palabra authorization con valor Bearer y el token genereado anteriormente



Ahora para visualizar todo esto en angular

Al necesitar traer la información una practica es crear un servicio para centralizarla



Es necesario registrarlo en el app.modules como provider:



Agregar todo esto solo es necesario si el servicio no cuenta con la siguiente sentencia, de lo contrario podemos únicamente importarlo en el constructor del componente.

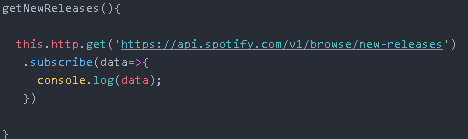
<-homeComponent

Se crea un servicio pero para esto hay que ocupar hacer la petición http.get

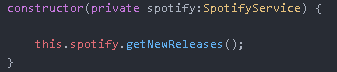
Asi que dentro del constructor del servicio



Y dentro de la función creada getNewReleases pasamos como parámetro el url de la petición que nos da spotify



Para acceder a esta función desde el componente deseado, en este caso el home, dentro del constructor



Solo que va a mandar un error pidiendo el token.

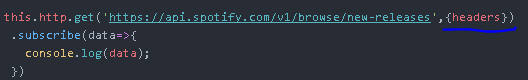
En el servicio hay que importar headers al lado del HttpClient  


Se crea una constante headers de tipo HttpHeaders accediendo a su constructor creado un objeto con los headers necesarios.



Dandole los valores de llave – valor que utilizamos en postman

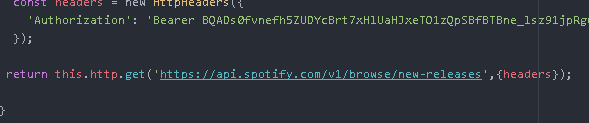
Ahora dentro de la petición añadimos otro parámetro que será la variable headers que declaramos.



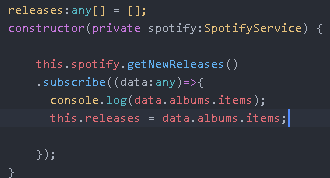
Consumiendo información del servicio de Spotify

Agregamos una vista de card en el home html

Editamos el servicio para que no tenga elsubscribe y solo retorne el valor de los releases.

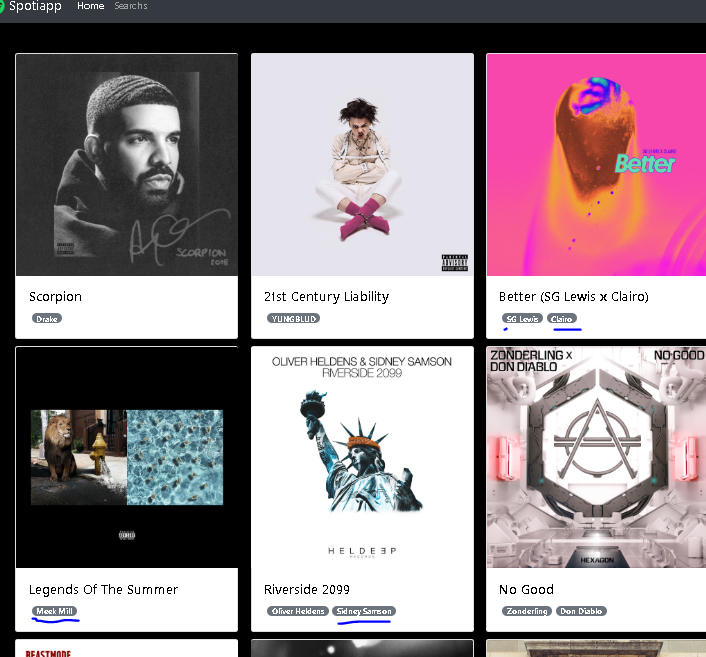


En el home component declaramos un arreglo de tipo any, y al hacer el llamado a la función del servicio, le adicionamos el subscribe y dándole el valor que trae en data al arreglo local, para poder usarlo en el cuerpo del html.



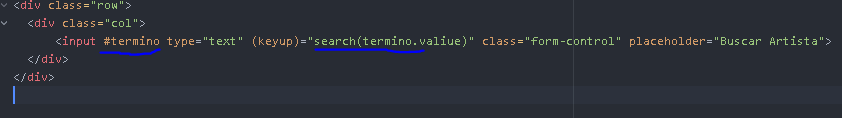
Finalmente en el html

Haciendo que repita con un for según el tamaño del arreglo de artistas que participan en el álbum y luego imprimiendo el valor actual del artista.name



Search

El html

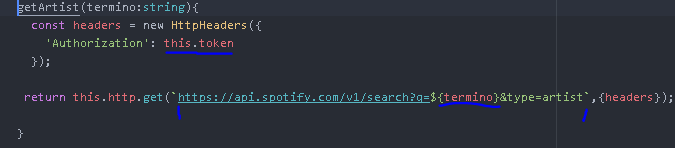


Donde se declara un valor #termino que recibe la entrada, y se manda como parámetro a la función que se activa cuando se presione una tecla con su value

Siendo básicamente para recibir el valor en la función del component



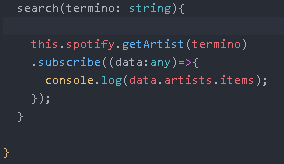
Modificando el servicio añadiéndole la funcipn de getArtist , donde utiliza la misma estructura anterior para conectarse a la api de Spotify,cambiando solamente la parte del get  
La variable this.token se crea para no repetir varias veces el key



En el constructor de el search component se trae la variable del servicio para poder llamar sus funciones.



Se añade el subscribe para asi obtener la data de la búsqueda.



Finalmente el html

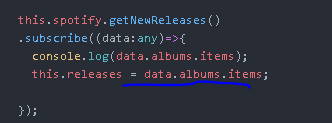


Optimizaciones.

Operador map

Home component.ts, El operador se adjunta a la petición y filtra para ver únicamente lo que en el caso sirve.

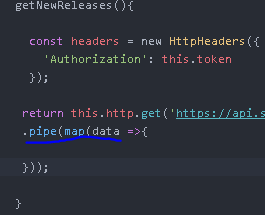
Desde el home deberiamos poder acceder a la información de una forma más fácil, no asi:



Por esto vamos a modificar el servicio

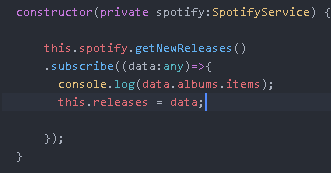


Haciendo un pipe que recibecomo parametro la función map que dentro lleva la data, y función de flecha



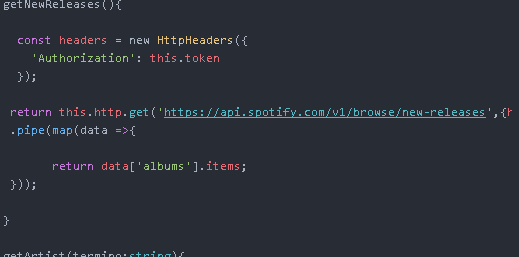
Para no cambiar a data a tipo any se puede poner dentro de llave cerrada como una opción a buscar dentro de data.

Ahora dentro del homecomponent



Centralizar las peticiones a Spotify

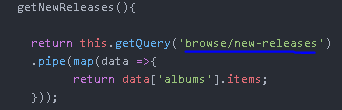
Haciendo una función que realice la petición a Spotify se ahorra código.:



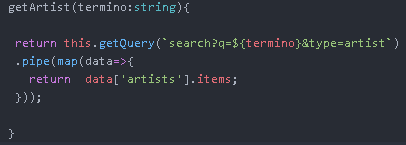
Creamos la función de query centralizada:



Y ahora la función solamente la llama pasándole el query donde cambia según la información a pedir del servicio de Spotify.



Al igual que para buscar artistas

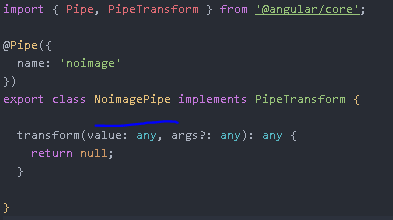


Verificación de imágenes y pipe para el manejo de imágenes

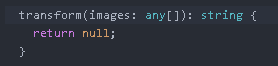
Para esto creamos un pipe personalizado



El que originalmente se crea con este contenido

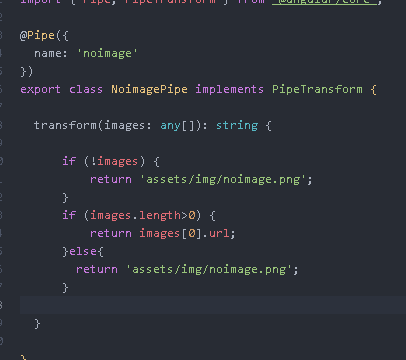


Cambiando el transform a una nueva estructura de parámetros que recibe



Siendo al final este el pipe que recibe la imagen y de ser nula retorna una imagen precargada nula,

O de existir, retorna la primera, o de venir el arreglo de imágenes pero sin imágenes de igual forma retorna la precargada nula.



Y para aplicar el pipe en el html



Componente de tarjetas

En este momento en los html de search y home existe duplicado la plantilla que muestra los artistas.

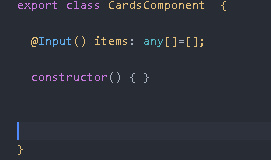
Creamos un componente nuevo que va a manejar las cards

Pegamos el formato de card del home  
Ahora el home quedaría llamando el componente de cards con su tag, y enviándole la información como un ítem.

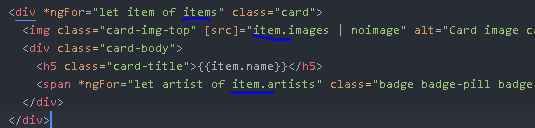


Ahora en el cards component.ts vamos a recuperar esta información mandada





Ahora que recuperamos el valor lo utilizamos en el html del cards teniendo asi el mismo efecto.



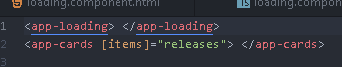
Ahora para utilizar la vista de tarjetas en el search component

 El elemento a enviar es la variable de arreglo que tenemos definida en el component.ts

Creando un loading component

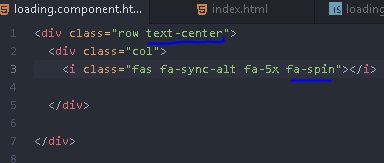


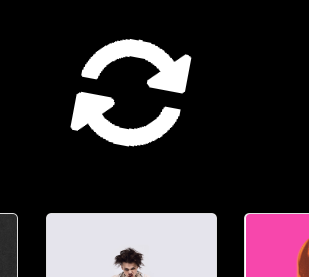
Y esteloading lo incluimos en el home antes de los elementos que deben cargar



Utilizaremos fontawesome.com

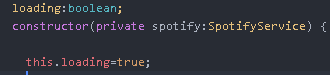
Añadimos el link de estilo al index y ya luego en el html que se desea implementar una imagen



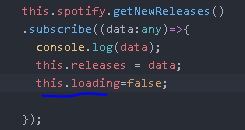


Ahora deseamos controlar cuando mostrar este loading, con un ng if

En el componente donde vamos a implementar creamos una variable loading detipo booleana para luego inicializarla en true en el constructor.



Y simplemente cambiar el valor cuando ya ha cargado los ítems a visualizar



Página del artista, parámetro por url y el servicio

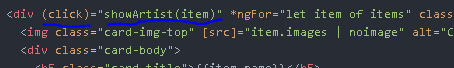
Creando ruta para acceder a la pagina del artista en routes



La declaramos como va a recibir un parámetro que es el id

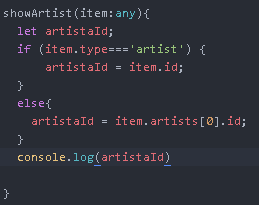
Ahora en el cards Component hacemos que cada card tenga el método click showArtist( ítem)

Donde pasará el ítem seleccionado



Ahora creamos el método dentro del component

Donde creamos lavariable artista que recibirá valores dependiendo de donde detecte este posicionada la página, de ser el tipo artista se sabe que la estructura es ítem.id pero de ser un álbum estará en la primera posición de artists

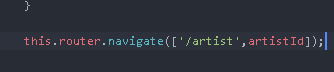


Ahora que ya tenemos el id podemos redireccionar a la pagina del artista, primero debemos importar el router





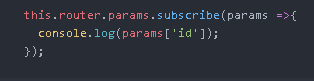
En donde para direccionar accedemos a la función de router navigate que recibe como parámetros el nombrede la página donde se dirige, y la variable que contiene el valor a enviar.



Ahora necesitamos recuperar el valor que se envio por el url, importamos en artista component



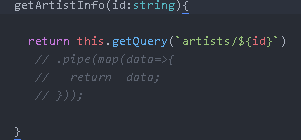
En el constructor declaramos la variable router de tipo ActivatedRoute para dentro recuperar el valor del id uscribiendonos mediante la función params de router. Donde params tendrá el valor de todos los parámetros que sean recibidos por el url



Al poner llaves cuadradas accede directamente a la clave y su valor contenido.

Obteniendo el artista:

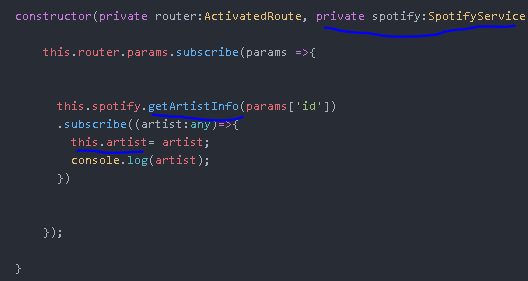
Agregamos en el servicio una función que recupere los datos del artista dependiendo de lo quenos pida el api de Spotify como query: en este caso no es necesario un pipe ya que regresa los datos como son deseados.



En el artist component declaramos el servicio para poder acceder a la función de getArtistInfo que esta dentro del servicio.

Luego como ya tenemos los datos del id accedemos a la función getArtistInfo pasándole el id que necesita para recuperar la info.

Y nos subscribimos a los datos que trae asignándolo a una variable local a la clase artista component para luego poder utilizarla en el html.

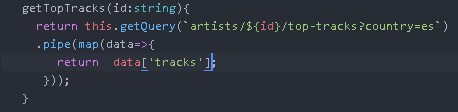


Top tracks service:

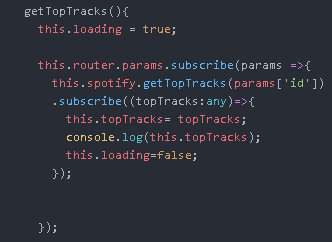
Primero vemos la documentación del servicio en la pagina de Spotify.



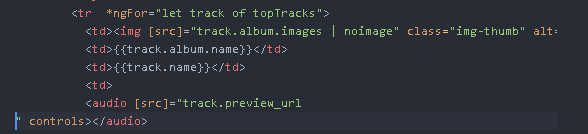
En el servicio



En el artist component



html



Manejo de errores

El subscribe tiene dos funciones, la sucess y cuando falla:

