INSTITUTO TECNOLÓGICO DE TIJUANA

SUBDIRECCIÓN ACADEMICA

DEPARTAMENTO DE SISTEMAS Y COMPUTACIÓN

SEMESTRE ENERO - JUNIO 2020

CARRERA:

INGENIERIA INFORMATICA

MATERIA:

DESARROLLO DE APLICACIONES PARA DISPOSITIVO MOVILES

UNIDAD: 2

NOMBRE:

JOSUE ADAN JAIMES PEREZ

NOMBRE DEL PROFESOR:

JUAN MANUEL HERNANDEZ MARTINEZ

FECHA

01 de MAYO de 2020

Configurar nuestra aplicación iónica de YouTube y series de tv

Comenzamos con una nueva aplicación Ionic en blanco y agregamos un nuevo proveedor y página. Asegúrese de crear un archivo de módulo para esa página si su CLI no lo está haciendo.

Además, instalamos el complemento Córdova para obtener acceso al reproductor nativo de YouTube para que luego podamos reproducir nuestros videos directamente en esa aplicación.

```
ionic start devdacticYoutube blank
cd devdacticYoutube
ionic g provider yt
ionic g page playlist
ionic cordova plugin add cordova-plugin-youtube-video-player
npm install --save @ionic-native/youtube-video-player
```

La CLI actual ya no creará ningún archivo de módulo para nuestras páginas, pero como estamos usando la carga diferida aquí, puede crear fácilmente un archivo de módulo para cada página siguiendo el esquema general de ese archivo.

Para usar el complemento en Android, necesitamos agregar nuestra clave API de YouTube de antes a nuestro config.xml, así que agregue esta entrada ahora:

Por ejemplo:

```
<preference name="YouTubeDataApiKey" value="[Aqui es el api]" />
```

Finalmente, debemos asegurarnos de que todo esté conectado en consecuencia dentro de nuestro app.module.ts.

```
import { BrowserModule } from '@angular/platform-browser';
import { ErrorHandler, NgModule } from '@angular/core';
import { IonicApp, IonicErrorHandler, IonicModule } from 'ionic-angular';
import { SplashScreen } from '@ionic-native/splash-screen';
import { StatusBar } from '@ionic-native/status-bar';

import { MyApp } from './app.component';
import { HomePage } from '../pages/home/home';

import { HttpModule } from '@angular/http';
import { YtProvider } from '../providers/yt/yt';
import { YoutubeVideoPlayer } from '@ionic-native/youtube-video-player';

@NgModule({
  declarations: [
    MyApp,
    HomePage
  ],
```

```
  imports: [
    BrowserModule,
    IonicModule.forRoot(MyApp),
    HttpModule
  ],
  bootstrap: [IonicApp],
  entryComponents: [
    MyApp,
    HomePage
  ],
  providers: [
    StatusBar,
    SplashScreen,
    {provide: ErrorHandler, useClass: IonicErrorHandler},
    YtProvider,
    YoutubeVideoPlayer
  ]
})
export class AppModule {}
```

Hacer una solicitud es básicamente la URL del punto final correspondiente de la API más algunos parámetros, ¡no se necesita nada más! En nuestro caso, necesitamos agregar la clave API, y si desea especificar otras opciones, puede cambiar la respuesta, qué se incluirá o cuántos elementos obtendrá, por ejemplo.

Por lo tanto, nuestro proveedor es realmente simple, por lo tanto, cambie su src / proveedores / yt / yt.ts a:

```
import { Injectable } from '@angular/core';
import { Http } from '@angular/http';
import 'rxjs/add/operator/map';

@Injectable()
export class YtProvider {
  apiKey = 'YOURKEY';

  constructor(public http: Http) { }

  getPlaylistsForChannel(channel) {
    return this.http.get('https://www.googleapis.com/youtube/v3/playlists?key=' +
this.apiKey + '&channelId=' + channel + '&part=snippet,id&maxResults=20')
    .map((res) => {
      return res.json()['items'];
    })
  }

  getListVideos(listId) {
```

```
    return
this.http.get('https://www.googleapis.com/youtube/v3/playlistItems?key=' +
this.apiKey + '&playlistId=' + listId +'&part=snippet,id&maxResults=20')
    .map((res) => {
      return res.json()['items'];
    })
  }
}
```

Ahora necesitamos construir la lógica para hacer una llamada a la API, y tendremos una vista simple con un campo de entrada y un botón de búsqueda que activa una llamada a la API con el valor de búsqueda actual.

Después de recuperar los resultados, queremos mostrarlos, y finalmente agregamos otra función para abrir una lista de reproducción específica que empujará otra página y pasará la ID de la lista clicada para que podamos cargar los siguientes datos allí.

Ahora abra su src / pages / home / home.ts y cámbielo a:

```
import { YtProvider } from './../../providers/yt/yt';
import { Component } from '@angular/core';
import { NavController, AlertController} from 'ionic-angular';
import { Observable } from 'rxjs/Observable';

@Component({
  selector: 'page-home',
  templateUrl: 'home.html'
})
export class HomePage {
  channelId = 'UCZZPgUIorPao48a1tBYSDgg'; // Devdactic Channel ID
  playlists: Observable<any[]>;

  constructor(public navCtrl: NavController, private ytProvider: YtProvider,
private alertCtrl: AlertController) { }

  searchPlaylists() {
    this.playlists = this.ytProvider.getPlaylistsForChannel(this.channelId);
    this.playlists.subscribe(data => {
      console.log('playlists: ', data);
    }, err => {
      let alert = this.alertCtrl.create({
        title: 'Error',
        message: 'No Playlists found for that Channel ID',
        buttons: ['OK']
      });
      alert.present();
    })
  }
```

```
  openPlaylist(id) {
    this.navCtrl.push('PlaylistPage', {id: id});
  }
}
```

La vista para nuestra primera página también es súper simple, solo un campo de entrada y el botón más la lista que usa nuestras listas de reproducción Observable de la clase que contiene las listas de reproducción del canal que buscamos.

Como hemos incluido información para cada lista de reproducción, podemos mostrar una miniatura, el título de esa lista e incluso cuando se publicó la lista.

Continúe y termine la primera parte cambiando su src / pages / home / home.html a:

```
<ion-header>
  <ion-navbar color="danger">
    <ion-title>
      Playlist Search
    </ion-title>
  </ion-navbar>
</ion-header>

<ion-content>
  <ion-item>
    <ion-label stacked>Channel ID</ion-label>
    <ion-input type="text" [(ngModel)]="channelId"></ion-input>
  </ion-item>
  <button full ion-button (click)="searchPlaylists()" [disabled]="channelId === ""
color="danger">Search Playlists</button>

  <ion-list no-padding>
    <button ion-item *ngFor="let list of playlists | async"
(click)="openPlaylist(list.id)">
      <ion-thumbnail item-start>
      <img [src]="list.snippet.thumbnails.standard.url">
    </ion-thumbnail>
    <h2>{{ list.snippet.title }}</h2>
    <p>{{ list.snippet.publishedAt | date:'short' }}</p>
    </button>
  </ion-list>
</ion-content>
```

Cargando videos para una lista de reproducción

La última parte del tutorial que falta es mostrar los videos de una lista de reproducción seleccionada, pero es muy sencillo después de nuestros esfuerzos iniciales.

¡Obtenemos la ID de la página que ya pasó a través de navParams, por lo que solo necesitamos hacer otra llamada a través de nuestro proveedor para recibir todos los videos de una lista de reproducción específica!

Finalmente, también agregamos una función para abrir un video específico por su ID a través del complemento Cordova que instalamos antes. También agregué un respaldo para el navegador que simplemente abrirá el video en una nueva pestaña para que pueda probar fácilmente el código sin un dispositivo.

Siga adelante y cambie su src / pages / playlist / playlist.ts a:

```
import { YtProvider } from './../../providers/yt/yt';
import { Observable } from 'rxjs/Observable';
import { Component } from '@angular/core';
import { IonicPage, NavController, NavParams, Platform } from 'ionic-angular';
import { YoutubeVideoPlayer } from '@ionic-native/youtube-video-player';

@IonicPage()
@Component({
  selector: 'page-playlist',
  templateUrl: 'playlist.html',
})
export class PlaylistPage {
  videos: Observable<any[]>;

  constructor(private navParams: NavParams, private ytProvider: YtProvider,
private youtube: YoutubeVideoPlayer, private plt: Platform) {
    let listId = this.navParams.get('id');
    this.videos = this.ytProvider.getListVideos(listId);
  }

  openVideo(video) {
    if (this.plt.is('cordova')) {
      this.youtube.openVideo(video.snippet.resourceId.videoId);
    } else {
      window.open('https://www.youtube.com/watch?v=' +
video.snippet.resourceId.videoId);
    }
  }
}
```

Para mostrar todos los videos, usamos nuevamente una lista de iones y agregamos una miniatura y un título como antes.

El evento click llamará a nuestra función con la identificación del video y mostrará el reproductor nativo de YouTube o la ventana del navegador dependiendo de dónde ejecute su aplicación.

Termine cambiando su src / pages / playlist / playlist.html a:

```
<ion-header>
  <ion-navbar color="danger">
    <ion-title>Playlist</ion-title>
  </ion-navbar>
</ion-header>

<ion-content>
  <ion-list>
    <button ion-item *ngFor="let video of videos | async"
(click)="openVideo(video)" detail-none>
      <ion-thumbnail item-start>
      <img [src]="video.snippet.thumbnails.standard.url">
    </ion-thumbnail>
    <h2>{{ video.snippet.title }}</h2>
    <p>{{ video.snippet.description }}</p>
    </button>
  </ion-list>
</ion-content>
```

Ahora está listo para buscar la lista de reproducción de un canal e incluso mostrar los videos de esa lista de reproducción con una pequeña miniatura, ¡directamente desde YouTube!

Conclusión

No siempre necesita un flujo completo de OAuth dentro de su aplicación Ionic para usar los Servicios de Google. Con una clave API puede realizar solicitudes a algunos servicios.