PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
ESCUELA DE INGENIERÍA
DEPARTAMENTO DE CIENCIA DE LA COMPUTACIÓN

**IIC2233 - Programación Avanza (II/2016)**
**Tarea 7**

# 1. Objetivos

- Interacción con una API para la creación de un programa.
- Sintetizar conocimientos del curso.

# 2. Introducción

Luego de estar años en tu tanque comunicado gracias a tu Programillo logras encontrar una manera de salir. Tras décadas de no ver algo más que solo metal, te asombras con el hermoso paisaje del exterior y con la sorpresa de que los humanos ya estaban repoblando nuevamente el mundo. Tu Universidad ya estaba construida para volver a los estudios. Sin dudarlo, vuelves a la U, a tus queridas clases de programación avanzada, pero no sin antes querer volver a escuchar los nuevos *hits* musicales. Intentas hacer *login* en tu cuenta *premium* de **Spotify** cuando te das cuenta que no te logras conectar, momento en el cuál te enteras de la triste verdad: la universidad bloqueó **Spotify**. Sin nada que escuchar se te ocurre la gran idea de usar **YouTube** para hacer todo lo que querías hacer en **Spotify**.

# 3. Problema

En esta tarea deberá crear un programa que consuma la API de **YouTube** y permita realizar una serie de funcionalidades mediante una interfaz gráfica. A través de la interfaz, un usuario deberá ser capaz de conectarse con sus credenciales de **YouTube** (usuario real) y trabajar sobre esta cuenta (para probar su tarea, deberán proporcionar su cuenta y todo lo necesario para conectarse a la API[1]). Una vez conectado se deben poder observar las opciones para las funcionalidades pedidas, realizarlas y, para el caso de las estadísticas, desplegar los resultados obtenidos desde la API en la interfaz. Para las otras dos funcionalidades (explicadas más adelante), el resultado deberá verse directamente en **YouTube**.

# 4. Especificaciones

## 4.1. API

Como se mencionó anteriormente, para realizar esta tarea tendrá que usar la API de **YouTube**. Una API es una *Application Programming Interface* y consiste en un conjunto de recursos que permiten consumir funcionalidades de alguna plataforma en particular, de forma amigable para desarrolladores. Existen APIs de distintos servicios: **Facebook**, **Amazon Web Services (AWS)**, **Telegram**, etc.. Como se dijo al principio, en esta tarea deberán usar la API de **Youtub**e. Como se pueden imaginar, es de vital importancia que

[1] Se recomienda crear una cuenta provisoria para el desarrollo de esta tarea.

los clientes que consumen alguna API (es decir, ustedes) sepan exactamente qué parámetros deben entregar al realizar una solicitud, así como también qué se espera que retornen los recursos de la API. Por ello es que deben estudiar y comprender la documentación de la API de **YouTube**. En particular, recomendamos fuertemente los siguientes enlaces para que se familiaricen con la estructura de la documentación: `Getting Started`, `Code Examples`, `Obtaining Authorization Credentials`.[2]

## 4.2. Funcionalidades

Para cada una de las siguientes funcionalidades, queda estrictamente prohibido desplegar el código `html` en un *browser* para cumplir con lo pedido. Deben crear una interfaz que permita llevar a cabo estas funcionalidades.

### 4.2.1. Creación de Lista de Reproducción

En base a una lista de canciones el programa será capaz de crear una lista de reproducción en **YouTube**. Esta lista deberá ser creada en la cuenta del usuario que hizo *login* con la interfaz. Al momento de crear la lista, el programa debe verificar que las canciones pertenecientes en la lista hayan sido encontradas en **Youtube**.

En la interfaz se deberá poder seleccionar una lista de reproducción (de las existentes en el archivo `listas_de_reproduccion.json`) y en base a esta crear el equivalente a esta lista en **YouTube**. Esta lista deberá aparecer en el canal de **YouTube** del usuario.

Deberá poder seleccionarse una canción de la lista de reproducción original, y para esta mostrar el correspondiente video en **YouTube** (Título del video, *link* y espacio/botón para comentar).

### 4.2.2. Comentar un Video

Luego de crear una o más listas de reproducción, el usuario podrá comentar cualquiera de estos videos. La interfaz debe poder permitir seleccionar un video de una lista ya creada e insertar un comentario. El cambio se deberá ver reflejado en el video (debes proporcionar el *link* a el video o el nombre exacto para poder probar esta funcionalidad).

### 4.2.3. Estadísticas

Dada una lista de reproducción creada, tu programa debe desplegar el mejor/peor video según algún parámetro especificado más adelante. El despliegue incluye como mínimo mostrar el título del video, *link* y la cantidad del parámetro a buscar. Por ejemplo, si utilizamos el parámetro `comentarios`, el mejor video será el que tiene la mayor cantidad de comentarios y el peor será el que tiene la menor cantidad de estos. Para el despliegue se mostrarán sus títulos, *links* y cantidad de comentarios para cada video. En caso de existir empate queda a su criterio como lo resuelven (por ejemplo, puede utilizarse un segundo parámetro).

Los parámetros son los siguientes:

- Cantidad de comentarios
- Cantidad de *likes*
- Cantidad de *dislikes*
- Cantidad de reproducciones

[2] Estos enlaces son sólo puntos de partida en el viaje que deberán hacer a través de la documentación de la API de **YouTube**. Se espera que sean capaces de buscar información eficientemente en cualquier motor de búsqueda decente (**Google**, **DuckDuckGo**, etc.).

## 4.3. listas_de_reproduccion.json

Para la búsqueda de las canciones se les proporcionó un archivo `.json` que contiene la información de las listas de reproducciones a crear. El formato de este archivo es de la siguiente forma:

**Listing 1** JSON example

```
{
    "playlists": {
        "Top 50 Hits": {
            "name": "Top 50 Hits",
            "id": "1ut7asYQJ6VhATGVtcxhnb"
            "tracks": [
                {
                  "id": "5iJzuuTd5Fl2uosN4WG5I4",
                  "uri": "spotify:track:5iJzuuTd5Fl2uosN4WG5I4",
                  "name": "Love My Life",
                  "album": "Love My Life",
                  "artists": [
                    "Robbie Williams"
                    ]
                },
                {
                  "id": "1vvNmPOiUuyCbgWmtc6yfm",
                  "uri": "spotify:track:1vvNmPOiUuyCbgWmtc6yfm",
                  "name": "My Way",
                  "album": "My Way",
                  "artists": [
                    "Calvin Harris"
                    ]
                }
            ]
        }
    },
      "saved": [
      ],
      "starred": [
      ]
}
```

- `playlists:` Es un diccionario que tiene como llaves los nombres de distintas listas de reproducción. Cada lista corresponde a otro diccionario que contiene la información relevante a ella, es decir, el nombre, id y las canciones que tiene. Finalmente, las canciones corresponden a una lista de diccionarios que contiene el id, uri, nombre, álbum y una lista con los artistas que colaboran en la canción.

## 4.4. Bonus (0.5 c/u)

### 4.4.1. Incrustar video

Una vez seleccionado un video (para comentarlo por ejemplo), se deberá poder reproducir el video respectivo en la interfaz.

#### 4.4.2. Spotify

Además de trabajar con el archivo `listas_reproduccion.json`, se deberá dar la opción de obtener las listas de reproducción directamente de la API de **Spotify** (puede ser cualquier set de listas de reproducción, no necesariamente las que se encuentran en `listas_de_reproduccion.json`).

## 5. Restricciones y alcances

- Solo se corregirá la tarea si ésta fue inscrita en el cuestionario habilitado del Siding.
- Tu programa debe ser desarrollado en Python 3.5
- Esta tarea es estrictamente individual, y está regida por el Código de Honor de la Escuela: Clickear para Leer.
- Su código debe seguir la guía de estilos `PEP8`
- Si no se encuentra especificado en el enunciado, asuma que el uso de cualquier librería Python está prohibido. Pregunte por foro si se pueden usar librerías específicas.
- El ayudante puede castigar el puntaje[3] de tu tarea, si le parece adecuado. Se recomienda ordenar el código y ser lo más claro y eficiente posible en la creación algoritmos.
- Debe adjuntar un archivo `README.md` donde comente sus alcances y el funcionamiento de su sistema (*i.e.* manual de usuario) de forma *concisa* y *clara*.
- Cree un módulo para cada conjunto de clases. Divídalas por las relaciones y los tipos que poseen en común. **Se descontará hasta un punto si se entrega la tarea en un solo módulo[4].**
- Cualquier aspecto no especificado queda a su criterio, siempre que no pase por sobre otro.

## 6. Entrega

- **Fecha/hora:** 27 de Noviembre - 23:59 hrs
- **Lugar:** GIT - Carpeta: Tareas/T07

Tareas que no cumplan con las restricciones señaladas en este enunciado tendrán la calificación mínima (1.0).

[3] Hasta −5 décimas.
[4] No agarre su código de un solo y lo divida en dos módulos, module su código de forma inteligente