Guia de instalación del plugin

* Pasos previos a la instalación (Guia de instalación en Windows)

1. Descargar el código fuente del obs-studio del [repositorio en GitHub](https://github.com/obsproject/obs-studio)
2. Descargar el código fuente del [plugin](https://github.com/Bryshot/TFG)
3. Descargar los archivos de dependencias
   1. [Dependencias de VS2017/2019](https://obsproject.com/downloads/dependencies2017.zip)
   2. [Dependencias de Qt5(5.10.1)](https://cdn-fastly.obsproject.com/downloads/Qt_5.10.1.7z)
   3. [Versión de desarrollo de FFmpeg](https://www.gyan.dev/ffmpeg/builds/ffmpeg-git-essentials.7z)
   4. [Versión de desarrollo de x264](https://code.videolan.org/videolan/x264/)
   5. [Versión de desarrollo de mbedTLS](https://github.com/ARMmbed/mbedtls)
   6. [Versión de desarrollo de CEF](https://cef-builds.spotifycdn.com/index.html)
4. [Descargar el CMake](https://cmake.org/download/) (3.16 o superior, preferiblemente la última versión)
5. Tener instalado VS2019 (Recomendado) con la [última versión de SDK](https://developer.microsoft.com/en-us/windows/downloads/windows-10-sdk/)

Pasos en caso de querer usar Browser Source:

1. Construir el proyecto del CEF con el CMake, rellenando los campos del siguiente modo:
   1. En el campo de “where is the source code”, introducir la dirección de la carpeta fuente (Por ejemplo D:/Dependencias/CEF).
   2. En el campo de "where to build the binaries", introducir la dirección de un subdirectorio “build” que debemos crear dentro de la carpeta fuente(Por ejemplo D:/ Dependencias/CEF/Build)
2. Configurar y generar dicho proyecto(Seleccionando la versión del Visual Studio adecuada y la versión de sistema operativo).
3. Compilar dicho proyecto en modo Release y en modo Debug (Al menos el proyecto de libcef\_dll\_wrapper).

Instalación in tree

* 1. Crear dentro de la carpeta de código del obs-studio, una carpeta con nombre build.
  2. Copiar el código fuente del plugin, insertarlo dentro de la carpeta del obs/UI/frontend-plugins, y añadir el plugin al CMake (Escribir dentro del CMakeList.txt “add\_subdirectory(SceneSwitcher)” )
  3. (Opcional) Crear una carpeta con todos los archivos de dependencias, creando dos subdirectorios en dicha carpeta:
     + DependenciasGenerales, donde se debe insertar todas las versiones de desarrollo y las dependencias de VS2017/2019(Incluir solo la versión adecuada para el sistema operativo)
     + Qt5, donde se debe insertar las dependencias de Qt5
  4. Abrir el CMake, y rellenar los siguientes campos:
     + En el campo de “where is the source code”, introducir la dirección del repositorio (Por ejemplo D:/obs-studio).
     + En el campo de "where to build the binaries", introducir la dirección del subdirectorio “build” creado previamente (Por ejemplo D:/obs-studio/build)
  5. Establecer las siguientes variables dentro de CMake (o establecerlas como variables de entorno de Windows, para evitar tener que volver a ponerlo si se desea copilar de nuevo el código fuente):
     + DepsPath, variable de tipo path, que debe apuntar a la carpeta de dependenciasGeneral. (Por ejemplo, DepsPath= D:\Dependencies\ dependenciasGeneral)
     + QTDIR, variable de tipo path, que debe apuntar a la carpeta de dependencias de Qt5 (Por ejemplo, QTDIR= D:\Dependencies\Qt\ 5.10.1\msvc2017\_64)
     + (En caso de querer usar Browser Source) CEF\_ROOT\_DIR, variable de tipo path, que debe apuntar a la carpeta de dependencias de CEF (Por ejemplo, CEF\_ROOT\_DIR = D:\Dependencies\CEF
     + (En caso de querer usar Browser Source) BUILD\_BROWSER, variable de tipo booleana, debe estar a true.
  6. En CMake-gui, presionar el botón de configuración, establecer la versión de Visual Studio deseada (Y en caso de que el sistema sea Win32, establecerlo en el segundo campo)
  7. En CMake-gui, presionar el botón de generación, lo cual creara el proyecto.

El proyecto se puede abrir desde el propio CMake o desde el explorador de archivos como cualquier otro.

Imagen de la carpeta dependencias generales

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Imagen de la configuración en el cMake

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Instalación out-of-tree (Aún no operativa)

1. Crear dentro de la carpeta de código del plugin, una carpeta con nombre build.
2. (Opcional) Crear una carpeta con todos los archivos de dependencias, creando dos subdirectorios en dicha carpeta:
   1. DependenciasGeneral, donde se debe insertar todas las versiones de desarrollo y las dependencias de VS2017/2019
   2. Qt5, donde se debe insertar las dependencias de Qt5
3. Abrir el CMake, y rellenar los siguientes campos:
   1. En el campo de “where is the source code”, introducir la dirección del repositorio del plugin (Por ejemplo D:/AutoProducer).
   2. En el campo de "where to build the binaries", introducir la dirección del subdirectorio “build” creado previamente (Por ejemplo D:/AutoProducer/build)
4. Establecer las siguientes variables dentro de CMake (o establecerlas como variables de entorno de Windows, para evitar tener que volver a ponerlo si se desea copilar de nuevo el código fuente):
   * 1. DepsPath, variable de tipo path, que debe apuntar a la carpeta de dependenciasGeneral. (Por ejemplo, DepsPath= D:\Dependencies\ dependenciasGeneral)
     2. QTDIR, variable de tipo path, que debe apuntar a la carpeta de dependencias de Qt5 (Por ejemplo, QTDIR= D:\Dependencies\Qt\ 5.10.1\msvc2017\_64)
     3. BUILD\_OUT\_OF\_TREE, variable de tipo bool, seleccionándola como true
     4. LIBOBS\_INCLUDE\_DIR, variable de tipo path, que debe apuntar a la localización de la subcarpeta del obs-studio llamada libobs (Por ejemplo D:\obs-studio\libobs)
     5. LIBOBS\_FRONTEND\_INCLUDE\_DIR, variable de tipo path, que debe apuntar a la subcarpeta del obs-studio llamada obs-frontend-api (Por ejemplo D:\obs-studio\UI\obs-frontend-api)
     6. W32 ,variable de tipo path, que debe apuntar a la subcarpeta del obs-studio llamada w32-pthreads (Por ejemplo D:/obs-studio/Build/deps/w32-pthreads)
     7. LIBOBS\_LIB, variable de tipo filepath, que debe apuntar a obs.dll (Por defecto se encuentra en C:\Program Files\obs-studio\bin\64bit\obs.dll)
     8. LIBOBS\_FRONTEND\_API\_LIB, variable de tipo filepath, que debe apuntar al obs-frontend-api.dll (Por defecto se encuentra en C:\Program Files\obs-studio\bin\64bit\ obs-frontend-api.dll)
     9. W32\_PTHREADS\_LIB, variable de tipo filepath, que debe apuntar al w32-pthreads.dll (Por defecto se encuentra en C:/Program Files/obs-studio/bin/64bit/w32-pthreads.dll)
5. En CMake-gui, presionar el botón de configuración, establecer la versión de Visual Studio deseada (Y en caso de que el sistema sea Win32, establecerlo en el segundo campo)
6. En CMake-gui, presionar el botón de generación, lo cual creara el proyecto.

El proyecto se puede abrir desde el propio CMake o desde el explorador de archivos como cualquier otro.