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
    2. Descargar el código fuente del plugin
    3. Descargar los archivos de dependencias
        a. Dependencias de VS2017/2019
        b. Dependencias de Qt5(5.10.1)
        c. Versión de desarrollo de FFmpeg
        d. Versión de desarrollo de x264
        e. Versión de desarrollo de mbedTLS
        f. Versión de desarrollo de CEF
        g. Versión de desarrollo de VLC
    4. Descargar el CMake (3.16 o superior, preferiblemente la última versión)
    5. Tener instalado VS2019 (Recomendado) con la última versión de SDK

    Pasos en caso de querer usar Browser Source:

    1. Construir el proyecto del CEF con el CMake, rellenando los campos del siguiente modo:
        a. En el campo de “where is the source code”, introducir la dirección de la carpeta fuente (Por ejemplo D:/Dependencias/CEF).
        b. 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.
    - VLCPath, variable del tipo path, que debe apuntar a la carpeta de dependencias de VLC (Por ejemplo, VLCPath = D:\Dependencies\VLC).
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

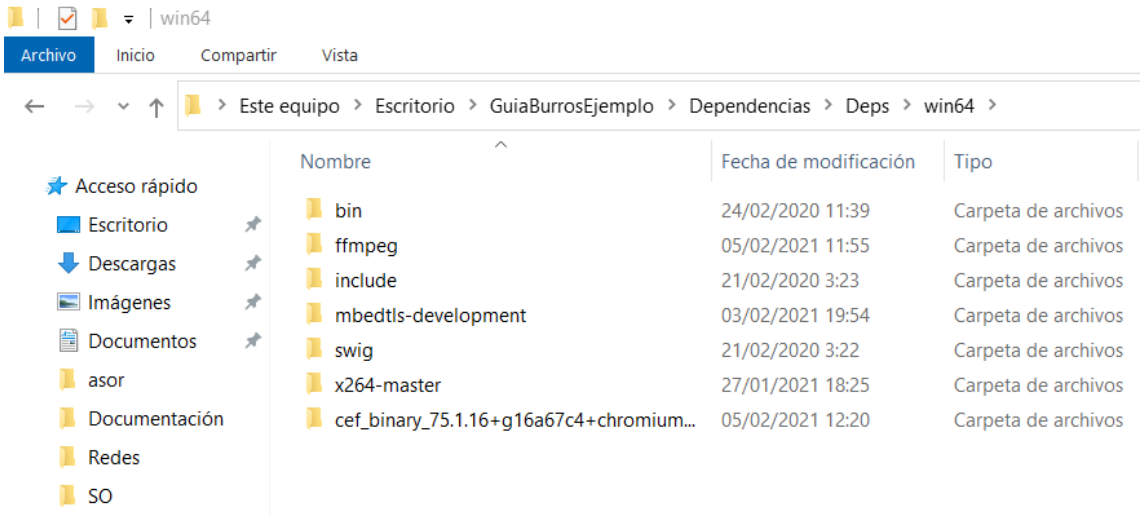
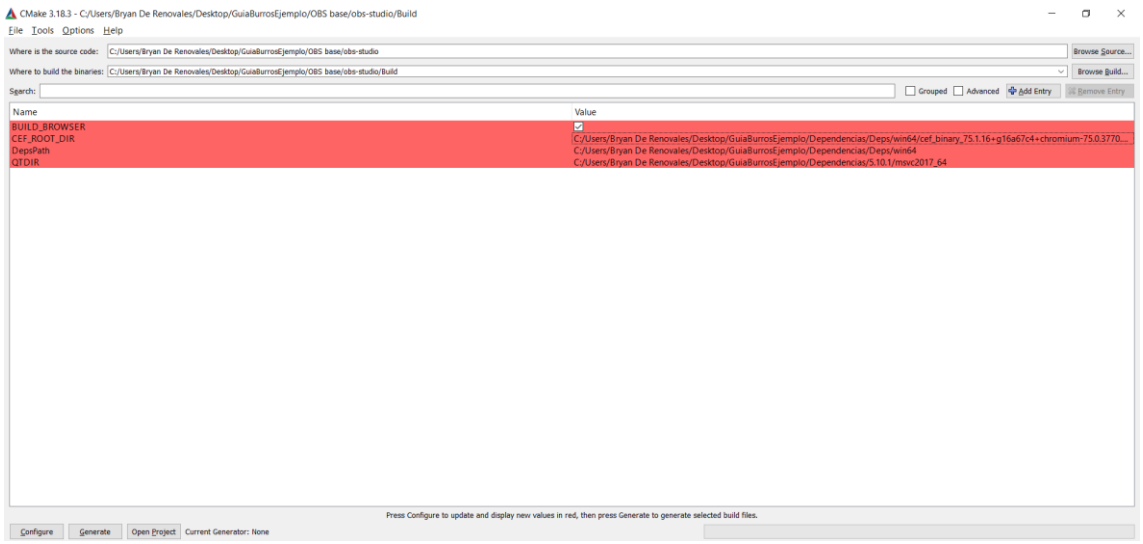


Imagen de la configuración en el cMake



Instalación out-of-tree (no operativo)

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:
    a. DependenciasGeneral, donde se debe insertar todas las versiones de desarrollo y las dependencias de VS2017/2019
    b. Qt5, donde se debe insertar las dependencias de Qt5
3. Abrir el CMake, y rellenar los siguientes campos:
    a. En el campo de “where is the source code”, introducir la dirección del repositorio del plugin (Por ejemplo D:/AutoProducer).
    b. 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):
    i. DepsPath, variable de tipo path, que debe apuntar a la carpeta de dependenciasGeneral. (Por ejemplo, DepsPath= D:\Dependencies\ dependenciasGeneral)
    ii. 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)
    iii. (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
    iv. (En caso de querer usar Browser Source) BUILD_BROWSER, variable de tipo booleana, debe estar a true.
    v. VLCPath, variable del tipo path, que debe apuntar a la carpeta de dependencias de VLC (Por ejemplo, VLCPath = D:\Dependencies\VLC).
    vi. BUILD_OUT_OF_TREE, variable de tipo bool, seleccionándola como true
    vii. 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)
    viii. 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)
    ix. 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)
    x. 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)
    xi. 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)
    xii. 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.

Formato del archivo para importar la información de los equipos.

El formato de fichero necesario para la correcta importación de toda la información relativa a las fuentes de los equipos y la de la clasificación, es la siguiente :

Cada línea contendrá únicamente un cadena de información. La primera línea contendrá la cantidad de equipos presentes en el torneo, continuado por la dirección web de la clasificación. Tras esto, para cada equipo se debe seguir el siguiente patrón:

- En primer lugar, el nombre del equipo tal y como se refleje en el torneo (necesario para la vinculación).
- A continuación, la dirección IP de la retransmisión del directorio del equipo, seguido por la dirección IP de la retransmisión de la cámara del equipo (con el formato adecuado para las fuentes del VLC (Ejemplo: http://192.168.0.1::8000)).
- En último lugar, la ruta donde se halla el icono del equipo (Los iconos deben tener una resolución de 110x110 px, de ser otra, seria necesario redimensionar la fuente en OBS, pero es necesario que todas tengan al menos la misma resolución)

Formato del archivo JSON para importar/exportar la configuración del plugin.

El formato de fichero necesario para realizar una correcta importación de la configuración del plugin, así como el del fichero resultante al exportar la configuración del plugin; es la siguiente:

El archivo comenzara con una llave de apertura “{” y terminara con una llave de cierre “}”; entre estos caracteres, separados por comas, pueden aparecer los siguientes elementos:

- contestName: String, es el nombre del torneo.
- contestServerWebsite: String, dirección web de la api del Domjudge.
- cycleSize: Integer, duración del ciclo de visionado.
- passwordContestServer: String, contraseña de administrador en el api del Domjudge.
- userContestServer: String, usuario administrador en el api del Domjudge.
- delayIP: Integer, tiempo de espera en ms antes de hacer efectivo un cambio de VLC.
- delayJugdement: Integer, tiempo en ms entre comprobaciones de nuevos veredictos.
- interval: Integer, tiempo de espera en ms entre ciclos del plugin.
- numPendingWeight: Float, peso en la heurística del número de envíos pendientes.
- rankWeight: Float, peso en la heurística de la posición en la clasificación.
- timeInStreamWeight: Float, peso en la heurística del número de apariciones previas.
- speedRotativeText: Integer, velocidad de rotación del texto móvil.
- startHotkey: Array, conjunto de teclas que ejercen de hotkey de inicio del plugin.
- stopHotkey: Array, conjunto de teclas que ejercen de hotkey de inicio del plugin.
- toggleHotkey: Array, conjunto de teclas que ejercen de hotkey de inicio del plugin.

- threadPriority: Integer [0-6], la prioridad que tienen los QThread que forman el plugin.
- verbose: Booleano, si se debe mostrar información de debug por consola.
- generalTabPos: Integer [0-2], posición que ocupa en el gui la pestaña General.
- heuristicTabPos: Integer [0-2], posición que ocupa en el gui la pestaña Heurística.
- timerTabPos: Integer [0-2], posición que ocupa en el gui la pestaña Timers.