Configurar un proyecto en Ortools usando VS

Tutorial by amigo de Cueto.cpp

Requisitos

- MinGW (U otro compilador de c++)
- Cmake (Se necesita instalar en el path)
- Visual Studio (Enterprise/Community 2022/Enterprise 2022)

Paso 1:

Ir al repositorio y descargar el proyecto

Descargar

Paso 2:

> **Importante**
> Ejecutar todo con el símbolo de sistema de herramientas nativas x64 (no el símbolo de desarrolladores)

En este tutorial se usará el **MinGW Command Prompt**

Entra a la carpeta de Ortools

```
cd C:\Users\HP\Desktop\Ortools
```

Paso 3:

Entra al proyecto que deseas probar, ejemplo

```
cd C:\Users\HP\Desktop\Ortools\examples\assignment_groups_mip
```

Paso 4 (Opcional):

Crea un directorio y trabaja desde ahí

```
mkdir build
cd build
```

Paso 5:

Buildea el proyecto con:

```
cmake -DCMAKE_BUILD_TYPE=Release -DCMAKE_PREFIX_PATH="C:\Users\HP\Desktop\Ortools" -S .. -B .
```

OJO :

> Donde esta DCMAKE_PREFIX_PATH = " (Aquí debe estar la ruta de tu Ortools) " -S .. -B .

Ese es tu path y es donde se saca las librerías y todo lo que se necesita para buildear un proyecto

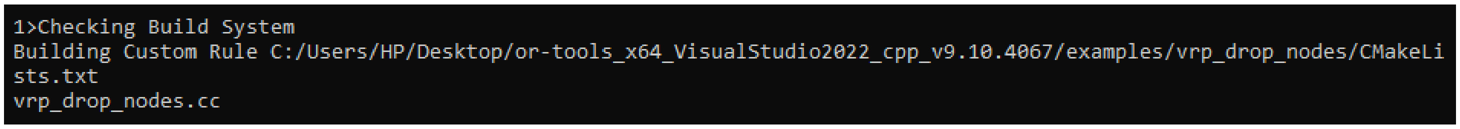
Paso 6

Compila el proyecto con lo siguiente

Solo debería detectar un archivo **.cc**

```
cmake --build . --config Release
```

Ejemplo:



```
1>Checking Build System
Building Custom Rule C:/Users/HP/Desktop/or-tools_x64_VisualStudio2022_cpp_v9.10.4067/examples/vrp_drop_nodes/CMakeLi
sts.txt
vrp_drop_nodes.cc
```

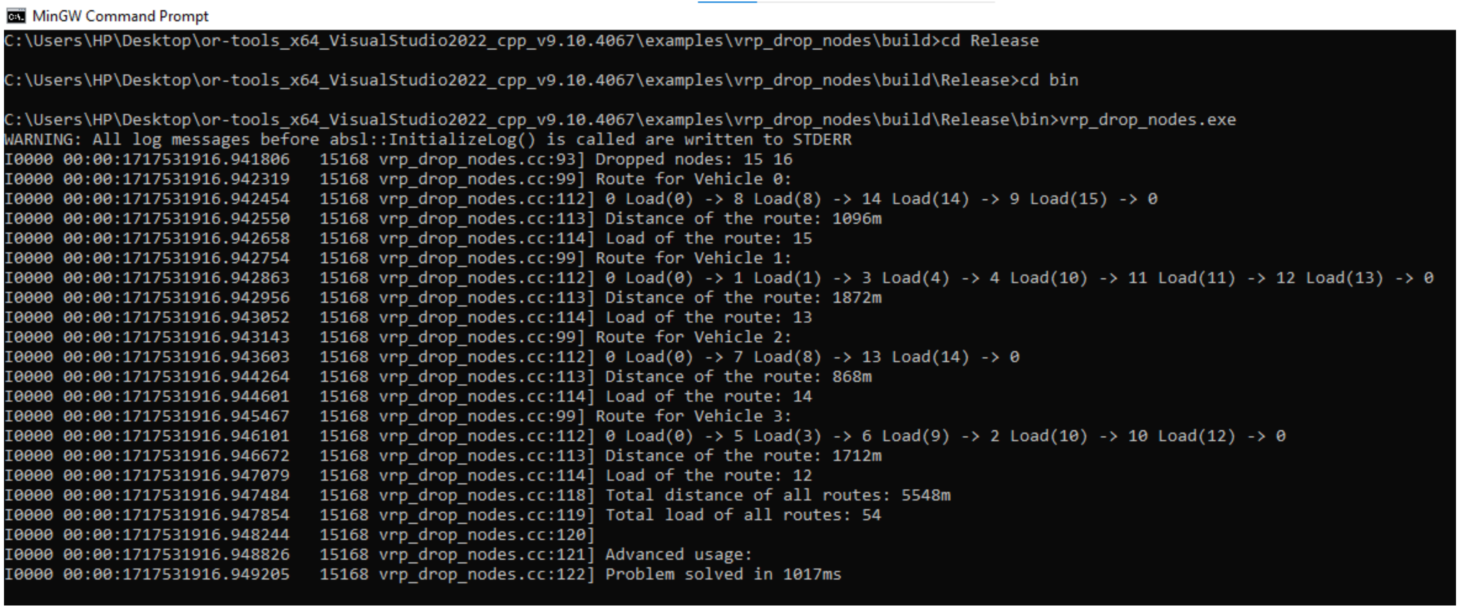
Paso 7

Si no ha aparecido ningún error hasta ahí se debería poder ejecutar el .exe generado en el paso 6. Para eso

Si creaste un build

```
cd Release
cd bin
NombreDelArchivo.exe
```

Resultado:



MinGW Command Prompt

```
C:\Users\HP\Desktop\or-tools_x64_VisualStudio2022_cpp_v9.10.4067\examples\vrp_drop_nodes\build>cd Release

C:\Users\HP\Desktop\or-tools_x64_VisualStudio2022_cpp_v9.10.4067\examples\vrp_drop_nodes\build\Release>cd bin

C:\Users\HP\Desktop\or-tools_x64_VisualStudio2022_cpp_v9.10.4067\examples\vrp_drop_nodes\build\Release\bin>vrp_drop_nodes.exe
WARNING: All log messages before absl::InitializeLog() is called are written to STDERR
I0000 00:00:1717531916.941806   15168 vrp_drop_nodes.cc:93] Dropped nodes: 15 16
I0000 00:00:1717531916.942319   15168 vrp_drop_nodes.cc:99] Route for Vehicle 0:
I0000 00:00:1717531916.942454   15168 vrp_drop_nodes.cc:112] 0 Load(0) -> 8 Load(8) -> 14 Load(14) -> 9 Load(15) -> 0
I0000 00:00:1717531916.942550   15168 vrp_drop_nodes.cc:113] Distance of the route: 1096m
I0000 00:00:1717531916.942658   15168 vrp_drop_nodes.cc:114] Load of the route: 15
I0000 00:00:1717531916.942754   15168 vrp_drop_nodes.cc:99] Route for Vehicle 1:
I0000 00:00:1717531916.942863   15168 vrp_drop_nodes.cc:112] 0 Load(0) -> 1 Load(1) -> 3 Load(4) -> 4 Load(10) -> 11 Load(11) -> 12 Load(13) -> 0
I0000 00:00:1717531916.942956   15168 vrp_drop_nodes.cc:113] Distance of the route: 1872m
I0000 00:00:1717531916.943052   15168 vrp_drop_nodes.cc:114] Load of the route: 13
I0000 00:00:1717531916.943143   15168 vrp_drop_nodes.cc:99] Route for Vehicle 2:
I0000 00:00:1717531916.943603   15168 vrp_drop_nodes.cc:112] 0 Load(0) -> 7 Load(8) -> 13 Load(14) -> 0
I0000 00:00:1717531916.944264   15168 vrp_drop_nodes.cc:113] Distance of the route: 868m
I0000 00:00:1717531916.944601   15168 vrp_drop_nodes.cc:114] Load of the route: 14
I0000 00:00:1717531916.945467   15168 vrp_drop_nodes.cc:99] Route for Vehicle 3:
I0000 00:00:1717531916.946101   15168 vrp_drop_nodes.cc:112] 0 Load(0) -> 5 Load(3) -> 6 Load(9) -> 2 Load(10) -> 10 Load(12) -> 0
I0000 00:00:1717531916.946672   15168 vrp_drop_nodes.cc:113] Distance of the route: 1712m
I0000 00:00:1717531916.947079   15168 vrp_drop_nodes.cc:114] Load of the route: 12
I0000 00:00:1717531916.947484   15168 vrp_drop_nodes.cc:118] Total distance of all routes: 5548m
I0000 00:00:1717531916.947854   15168 vrp_drop_nodes.cc:119] Total load of all routes: 54
I0000 00:00:1717531916.948244   15168 vrp_drop_nodes.cc:120]
I0000 00:00:1717531916.948826   15168 vrp_drop_nodes.cc:121] Advanced usage:
I0000 00:00:1717531916.949205   15168 vrp_drop_nodes.cc:122] Problem solved in 1017ms
```

Usar las librerías de ortools a conveniencia

Para crear y usar las librerías del ortools a conveniencia es necesario que crees una carpeta con lo siguiente

```
Carpeta
|
|
|__ Archivo.cc
|
|__ CMakeLists.txt
```

Paso 0

Si no sabes como crear un archivo cmake, solo copia y pega esto:

> Nota: Solo debes cambiar 2 cosas aquí (Dependiendo del nombre de tu archivo .cc) **Cambia todos los "hola" que veas en el siguiente código**

```
cmake_minimum_required(VERSION 3.18)

option(CMAKE_EXPORT_COMPILE_COMMANDS "Export compile command" OFF)

project(hola VERSION 1.0 LANGUAGES CXX)
message(STATUS "${PROJECT_NAME} version: ${PROJECT_VERSION}")

if(MSVC)
  set(CMAKE_CXX_STANDARD 20)
else()
  set(CMAKE_CXX_STANDARD 17)
endif()
set(CMAKE_CXX_STANDARD_REQUIRED ON)
set(CMAKE_CXX_EXTENSIONS OFF)
set(CMAKE_INSTALL_RPATH_USE_LINK_PATH True)

get_property(isMultiConfig GLOBAL PROPERTY GENERATOR_IS_MULTI_CONFIG)
if(isMultiConfig)
  if(NOT CMAKE_CONFIGURATION_TYPES)
    set(CMAKE_CONFIGURATION_TYPES "Release;Debug" CACHE STRING
    "Choose the type of builds, options are: Debug Release RelWithDebInfo MinSizeRel. (default:
    FORCE)
  endif()
  message(STATUS "Configuration types: ${CMAKE_CONFIGURATION_TYPES}")
else()
  if(NOT CMAKE_BUILD_TYPE)
    set(CMAKE_BUILD_TYPE "Release" CACHE STRING
    "Choose the type of build, options are: Debug Release RelWithDebInfo MinSizeRel. (default:
    FORCE)
  endif()
  message(STATUS "Build type: ${CMAKE_BUILD_TYPE}")
endif()

include(GNUInstallDirs)
if(UNIX)
  option(BUILD_SHARED_LIBS "Build shared libraries (.so or .dylib)." ON)
  set(CMAKE_BUILD_WITH_INSTALL_RPATH TRUE)
  set(CMAKE_LIBRARY_OUTPUT_DIRECTORY ${CMAKE_BINARY_DIR}/${CMAKE_INSTALL_LIBDIR})
  set(CMAKE_ARCHIVE_OUTPUT_DIRECTORY ${CMAKE_BINARY_DIR}/${CMAKE_INSTALL_LIBDIR})
  set(CMAKE_RUNTIME_OUTPUT_DIRECTORY ${CMAKE_BINARY_DIR}/${CMAKE_INSTALL_BINDIR})
  foreach(OutputConfig IN LISTS CMAKE_CONFIGURATION_TYPES)
    string(TOUPPER ${OutputConfig} OUTPUTCONFIG)
    set(CMAKE_LIBRARY_OUTPUT_DIRECTORY_${OUTPUTCONFIG} ${CMAKE_BINARY_DIR}/${OutputConfig}/${CM
    set(CMAKE_ARCHIVE_OUTPUT_DIRECTORY_${OUTPUTCONFIG} ${CMAKE_BINARY_DIR}/${OutputConfig}/${CM
    set(CMAKE_RUNTIME_OUTPUT_DIRECTORY_${OUTPUTCONFIG} ${CMAKE_BINARY_DIR}/${OutputConfig}/${CM
  endforeach()
else()
  option(BUILD_SHARED_LIBS "Build shared libraries (.dll)." OFF)
  set(CMAKE_LIBRARY_OUTPUT_DIRECTORY ${CMAKE_BINARY_DIR}/${CMAKE_INSTALL_BINDIR})
  set(CMAKE_ARCHIVE_OUTPUT_DIRECTORY ${CMAKE_BINARY_DIR}/${CMAKE_INSTALL_BINDIR})
  set(CMAKE_RUNTIME_OUTPUT_DIRECTORY ${CMAKE_BINARY_DIR}/${CMAKE_INSTALL_BINDIR})
  foreach(OutputConfig IN LISTS CMAKE_CONFIGURATION_TYPES)
    string(TOUPPER ${OutputConfig} OUTPUTCONFIG)
    set(CMAKE_LIBRARY_OUTPUT_DIRECTORY_${OUTPUTCONFIG} ${CMAKE_BINARY_DIR}/${OutputConfig}/${CM
    set(CMAKE_ARCHIVE_OUTPUT_DIRECTORY_${OUTPUTCONFIG} ${CMAKE_BINARY_DIR}/${OutputConfig}/${CM
    set(CMAKE_RUNTIME_OUTPUT_DIRECTORY_${OUTPUTCONFIG} ${CMAKE_BINARY_DIR}/${OutputConfig}/${CM
  endforeach()
endif()

find_package(ortools REQUIRED CONFIG)

add_executable(${PROJECT_NAME} hola.cc)
target_include_directories(${PROJECT_NAME} PUBLIC ${CMAKE_CURRENT_SOURCE_DIR})
target_compile_features(${PROJECT_NAME} PUBLIC
  $<IF:$<CXX_COMPILER_ID:MSVC>,cxx_std_20,cxx_std_17>)
target_link_libraries(${PROJECT_NAME} PRIVATE ortools::ortools)

set_property(GLOBAL PROPERTY CTEST_TARGETS_ADDED 1)
include(CTest)
if(BUILD_TESTING)
  add_test(NAME test_${PROJECT_NAME} COMMAND ${PROJECT_NAME} )
endif()

include(GNUInstallDirs)
install(TARGETS ${PROJECT_NAME})
```

Paso 1

Entra a la carpeta que tengas

```
cd C:\Users\HP\Desktop\hola
```

> **Importante**
> Ejecutar todo con el símbolo de sistema de herramientas nativas x64 (no el símbolo de desarrolladores)

En este tutorial se usará el **MinGW Command Prompt**

Paso 2 (Opcional)

Crea un directorio y trabaja desde ahí

```
mkdir build
cd build
```

Paso 3

Buildea el proyecto, ya en la url que tengas debes escribir lo siguiente

```
cmake -DCMAKE_BUILD_TYPE=Release -DCMAKE_PREFIX_PATH="C:\Users\HP\Desktop\Ortools" -S .. -B .
```

OJO :

> Donde esta DCMAKE_PREFIX_PATH = " (Aquí debe estar la ruta de tu Ortools) " -S .. -B .

Ese es tu path y es donde se saca las librerías y todo lo que se necesita para buildear un proyecto

Paso 4

Compila el proyecto con lo siguiente

Solo debería detectar un archivo **.cc**

```
cmake --build . --config Release
```

Paso 5

Si creaste un build

```
cd Release
cd bin
NombreDelArchivo.exe
```

> Nota: En teoría ya debería estar todo compilado