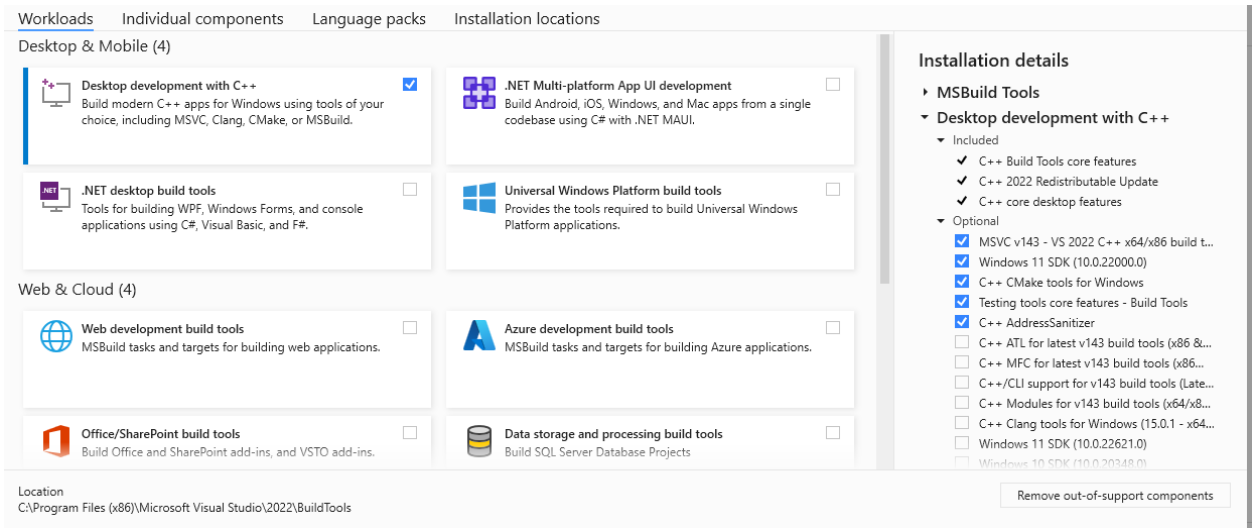
# Build libmuParser dll for Windows 10 & 11

1. Download and Install VS Code (if not already installed): Download
2. Launch VS Code and Install the C/C++ plugins
    a. C/C++ by Microsoft (ms-vscode.cpptools)
    b. C/C++ Extension Pack by Microsoft (ms-vscode.cpptools-extension-pack)
3. Download and install the Build Tools for VIsual Studio 2022 (v17.4.3): Download
    a. Only the Desktop Development with C++ workload needs to be installed. This includes MSVC and CMake.




4. Download a copy of the libmuParser source code and extract it to a folder: https://github.com/beltoforion/muparser/releases
5. Open the folder in VS Code
6. Allow the cmake to configure the project
7. Set the build variant to Release
8. Select the build kit:
    a. Use “Visual Studio Build Tools Release 2022 - amd64” to build a 64-bit dll
    b. Use “Visual Studio Build Tools Release 2022 - x86” to build a 32-bit dll
9. Build
10. The dll will be output to the build subfolder if successful

# Build libmuParser for Ubuntu 20.04

1. Install VS Code (if not already installed). Can be done through the Ubuntu Software manager.
2. Launch VS Code and Install the C/C++ plugins
    - C/C++ by Microsoft (ms-vscode.cpptools)
    - C/C++ Extension Pack by Microsoft (ms-vscode.cpptools-extension-pack)
3. Install the GCC, G++, and CMAKE

```
sudo apt update
sudo apt install build-essential cmake ninja-build
```

4. (optional) Install gcc-multilib if you want to build for an x86 target

```
sudo apt install gcc-multilib g++-multilib
```

5. Download a copy of the libmuparser source code and extract it to a folder: https://github.com/beltoforion/muparser/releases
6. Open the folder in VS Code
7. Allow the cmake to configure the project
8. Set the build variant to Release
9. Select the build kit:
    - GCC 9.4.0 x86_64-linux-gnu
10. (Optional) If you want to build for an x86 target, you need to modify the project's CMakeLists.txt file:
    - Change:

```
option(ENABLE_SAMPLES "Build the samples" ON)
```

      To:

```
option(ENABLE_SAMPLES "Build the samples" OFF)
```

    - Change:

```
set(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} -Wall -Wno-long-long -pedantic")
```

      To:

```
set(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} -m32 -Wall -Wno-long-long -pedantic")
```

11. Build
12. The shared object file will be output to the build subfolder if successful

# Modifications made to libmuParser distributed with LV-muParser

muParserDLL.cpp

1. Added callback function for Not operation:

```
muFloat_t Not(muFloat_t v) { return v == 0; }
```

2. Modified mupCreate function to add ":" to allowable variable name characters and add "!" for Not operation:

```
API_EXPORT(muParserHandle_t) mupCreate(int nBaseType)
{
	muParserHandle_t handle;
	switch (nBaseType)
	{
	case  muBASETYPE_FLOAT:
		handle = (void*)(new ParserTag(muBASETYPE_FLOAT));
		mupDefineNameChars(handle, _T("0123456789_:abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ"));
		mupDefineInfixOprt(handle, _T("!"), Not, 0);
		return handle;
	case  muBASETYPE_INT:
		handle = (void*)(new ParserTag(muBASETYPE_INT));
		mupDefineNameChars(handle, _T("0123456789_:abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ"));
		mupDefineInfixOprt(handle, _T("!"), Not, 0);
		return handle;
	default:		return nullptr;
	}
}
```