

Под windows обычно так делаю:

1. [Скачиваю Qt](#), сейчас актуальна версия [Qt 5.1.1 for Windows 32-bit \(MinGW 4.8, OpenGL, 666 MB\)](#),
2. [Скачиваю CMake](#), сейчас актуальная версия [cmake-2.8.11.2-win32-x86.exe](#),
3. Прописываю пути в PATH до cmake (можно задать при установке), до mingw (идет вместе с Qt, сейчас ставится в C:\Qt\Qt5.1.1\Tools\mingw48_32\bin), до qmake (C:\Qt\Qt5.1.1\5.1.1\mingw48_32\bin),
4. Проверяю из консоли, что пути выставлены правильно:

```
>cmake --version
```

```
cmake version 2.8.11.2
```

```
>g++ --version
```

```
g++.EXE (rev2, Built by MinGW-builds project) 4.8.0
```

```
Copyright (C) 2013 Free Software Foundation, Inc.
```

```
This is free software; see the source for copying conditions. There is NO  
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

```
>qmake --version
```

```
QMake version 3.0
```

```
Using Qt version 5.1.1 in C:\Qt\Qt5.1.1\5.1.1\mingw48_32\lib
```

5. Добавляю переменную окружения QTDIR=C:\Qt\Qt5.1.1\5.1.1\mingw48_32 (понадобится потом в Eclipse),
6. Создаю тестовый проект, все файлы ввожу вручную без всяких IDE (можно хоть в блокноте, но мне far удобнее):

CMakeLists.txt:

Код Code

```
1 cmake_minimum_required(VERSION 2.8.8)
2
3 cmake_policy(SET CMP0020 NEW)
4
5 project (test)
6
7 set(CMAKE_AUTOMOC ON)
8
9 set(CMAKE_INCLUDE_CURRENT_DIR ON)
10
11 find_package(Qt5Widgets REQUIRED)
12
13 set (test_sources
14     "main.cpp"
15 )
16
17 add_executable(test WIN32 ${test_sources})
18
19 qt5_use_modules(test Widgets)
20
```

7. **main.cpp:**

```

1
2 #include <QApplication>
3 #include <QPushButton>
4
5 int main(int argc, char* argv[])
6 {
7     QApplication app(argc, argv);
8     QPushButton button("Hello!");
9     button.show();
10    return app.exec();
11 }
12

```

8. Создаю директорию **build** для сборки, захожу в нее и запускаю cmake:

```

>cmake -G "MinGW Makefiles" -DCMAKE_BUILD_TYPE=Debug ..
-- The C compiler identification is GNU 4.8.0
-- The CXX compiler identification is GNU 4.8.0
-- Check for working C compiler: C:/Qt/Qt5.1.1/Tools/mingw48_32/bin/gcc.exe
-- Check for working C compiler: C:/Qt/Qt5.1.1/Tools/mingw48_32/bin/gcc.exe -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working CXX compiler: C:/Qt/Qt5.1.1/Tools/mingw48_32/bin/g++.exe
-- Check for working CXX compiler: C:/Qt/Qt5.1.1/Tools/mingw48_32/bin/g++.exe -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Configuring done
-- Generating done
-- Build files have been written to: .../test/build

```

9. Запускаю сборку через mingw32-make:

```

>mingw32-make
Scanning dependencies of target test_automoc
[ 33%] Automoc for target test
[ 33%] Built target test_automoc
Scanning dependencies of target test
[ 66%] Building CXX object CMakeFiles/test.dir/main.cpp.obj
[100%] Building CXX object CMakeFiles/test.dir/test_automoc.cpp.obj
Linking CXX executable test.exe
[100%] Built target test

```

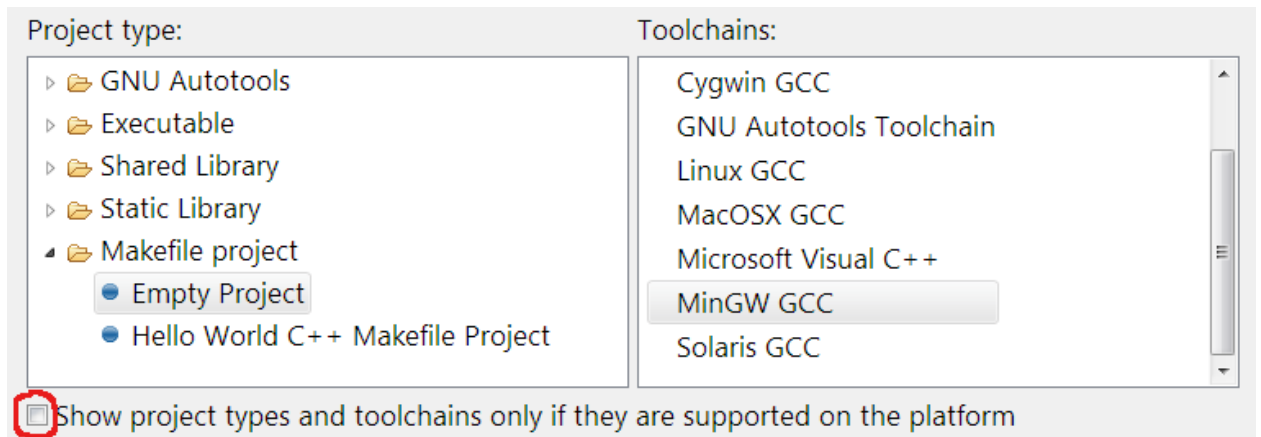
10. Запускаю **test.exe**, чтобы проверить собранный exe.

Все это нужно сделать один раз, чтобы проверить корректность установки Qt/CMake/MinGW.

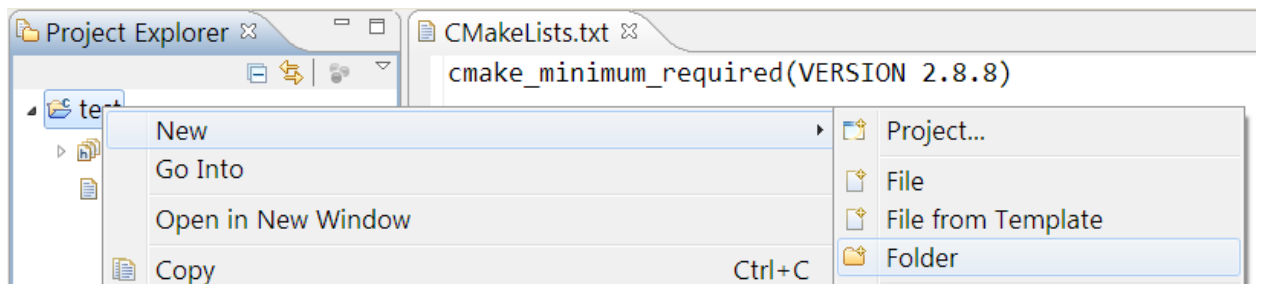
Теперь про то как работаю в Eclipse:

1. Запускаю Eclipse с новым workspace.

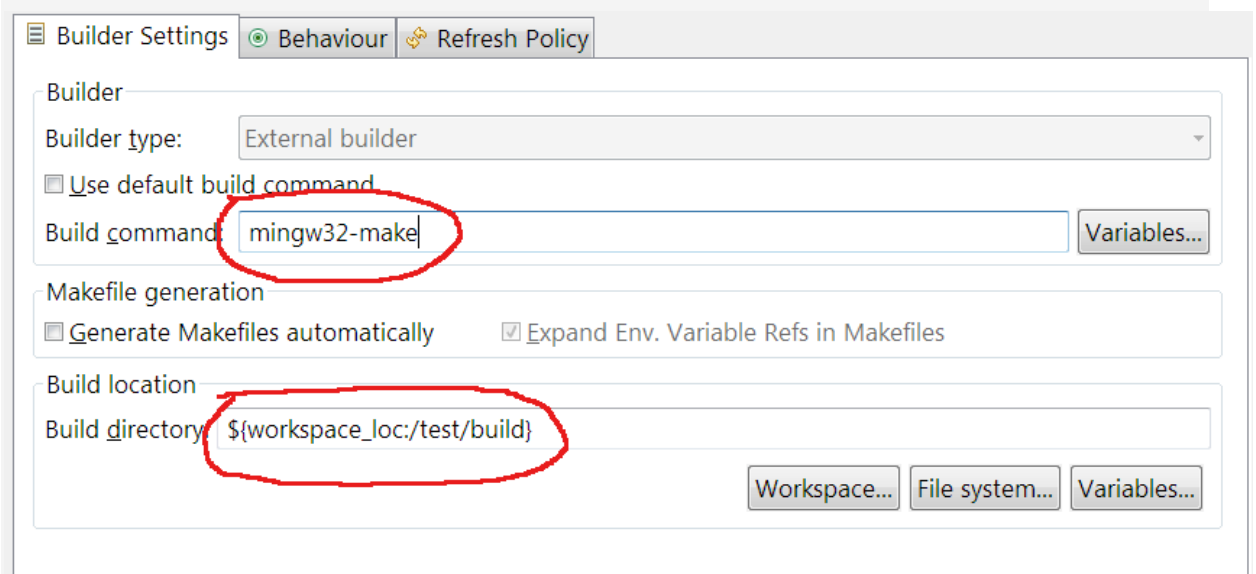
2. Сразу отключаю Project->Build Automatically (это только для Java работает нормально):
3. Создаю новый проект через File->New->C++ Project (приходится снимать галочку почему-то):



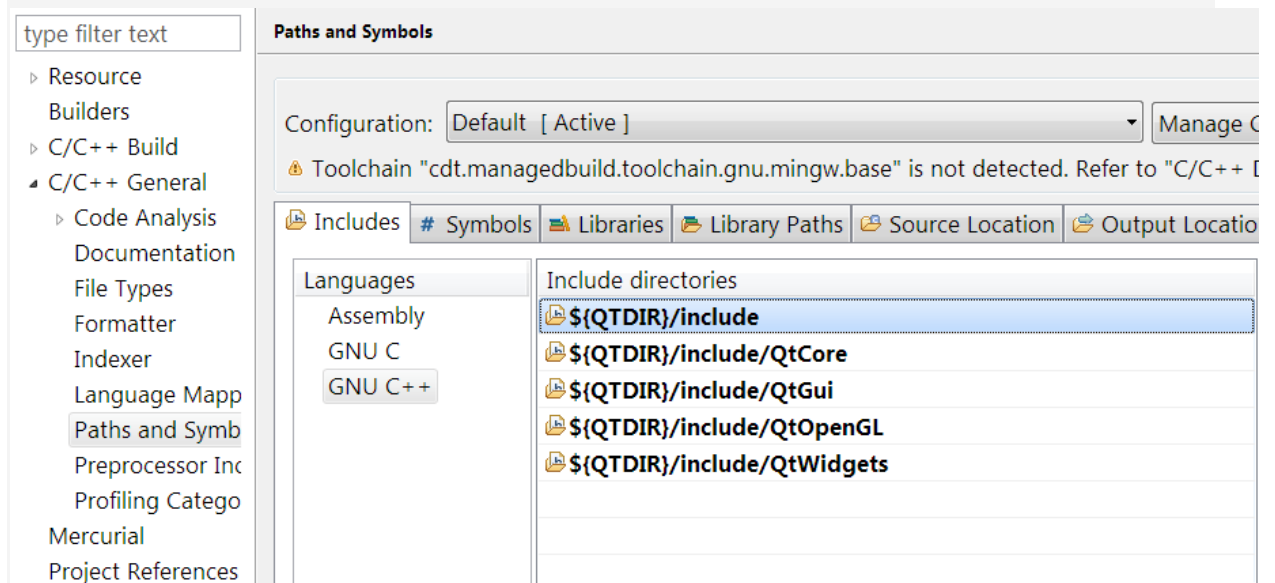
4. Создаю директорию build через контекстное меню:



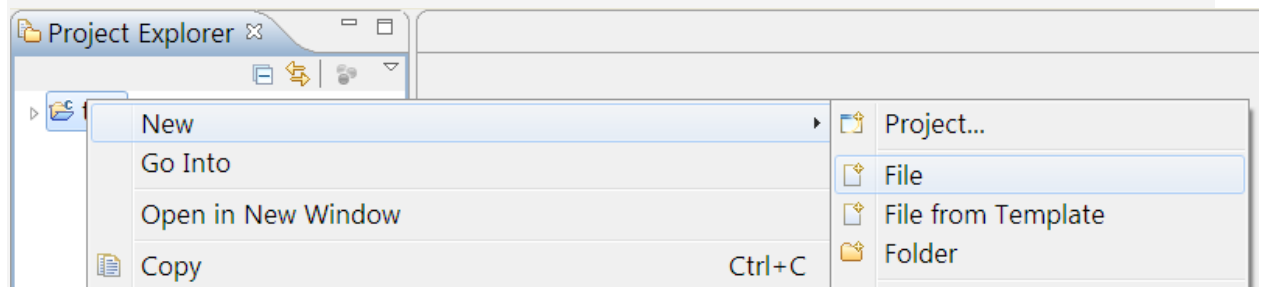
5. В настройках проекта выставляю команду mingw32-make для сборки в директории build:



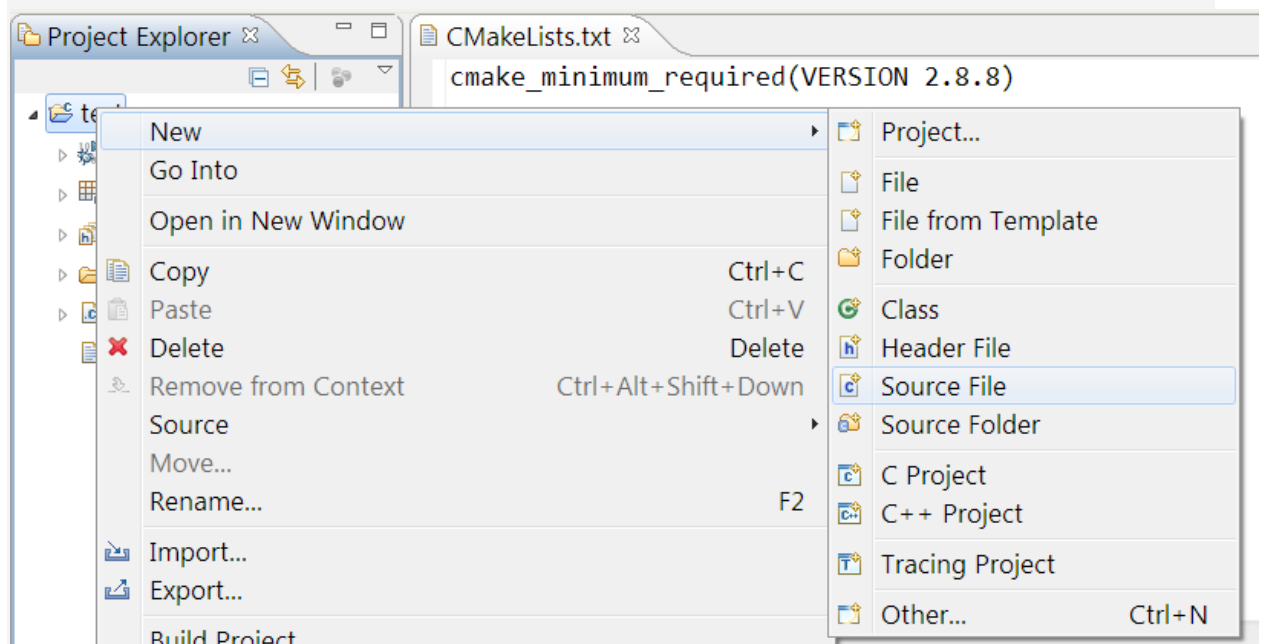
6. В настройках проекта добавляю пути до Qt:



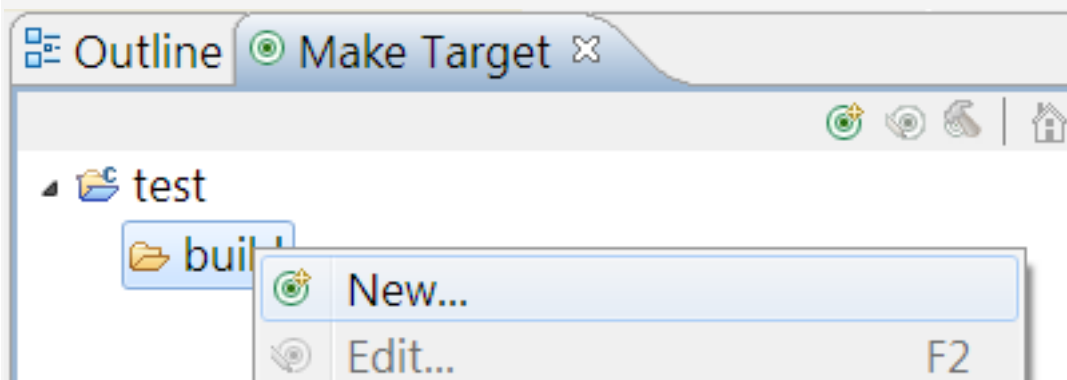
7. Добавляю новый файл CMakeLists.txt через контекстное меню:



8. Содержимое то же, что и выше было. Добавляю новый файл main.cpp:



9. Содержимое то же, что было выше. Добавляю команду для запуска CMake в Make Target:



Target name:

Make Target

☐ Same as the target name

Make target:

Build Command

☐ Use builder settings

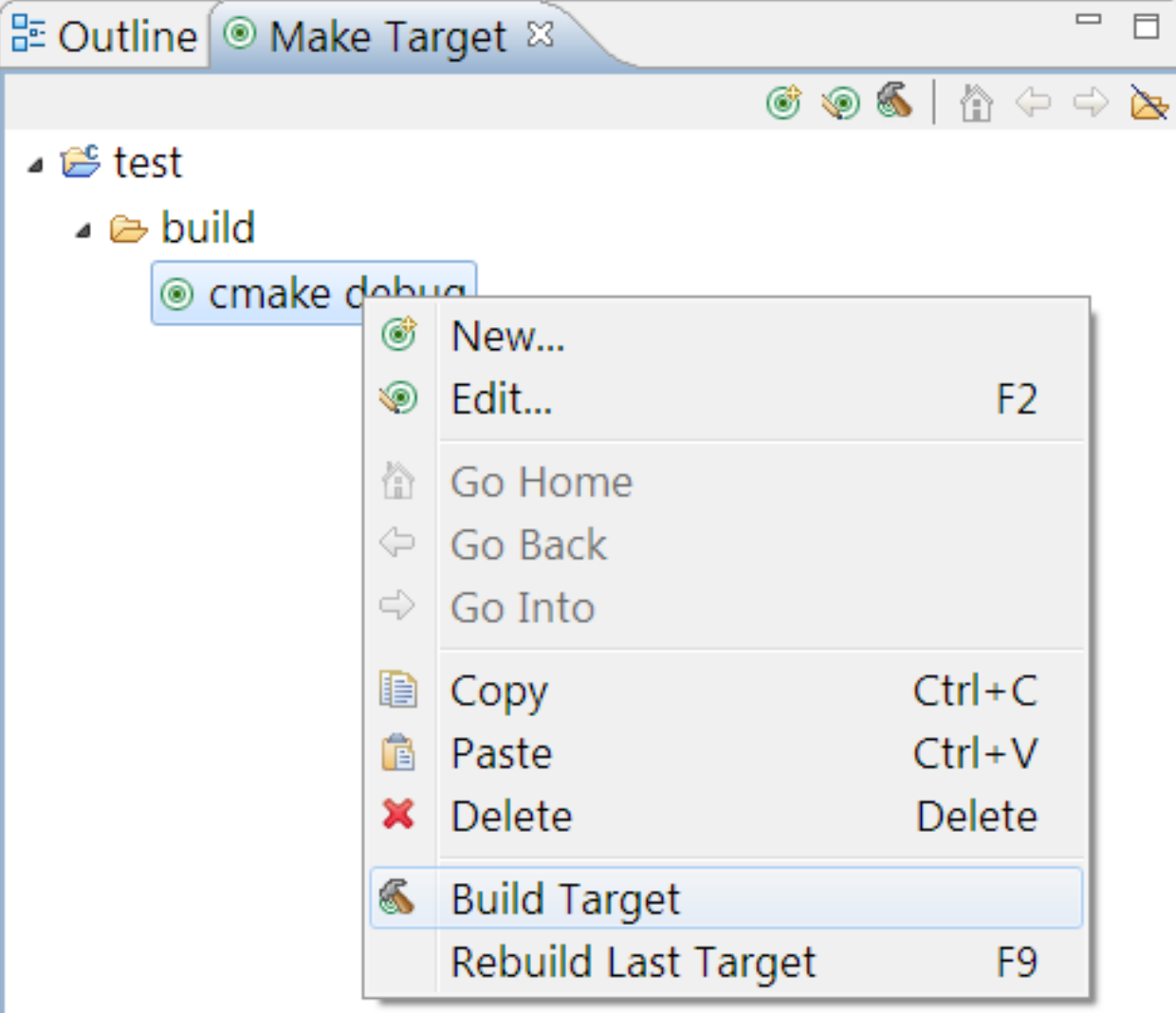
Build command:

Build Settings

☒ Stop on first build error

☒ Run all project builders

10. Запускаю cmake:



The screenshot shows an IDE window with a project tree on the left. The tree structure is as follows:

- test
 - build
 - cmake debug

The 'cmake debug' target is selected, and a context menu is open over it. The menu items are:

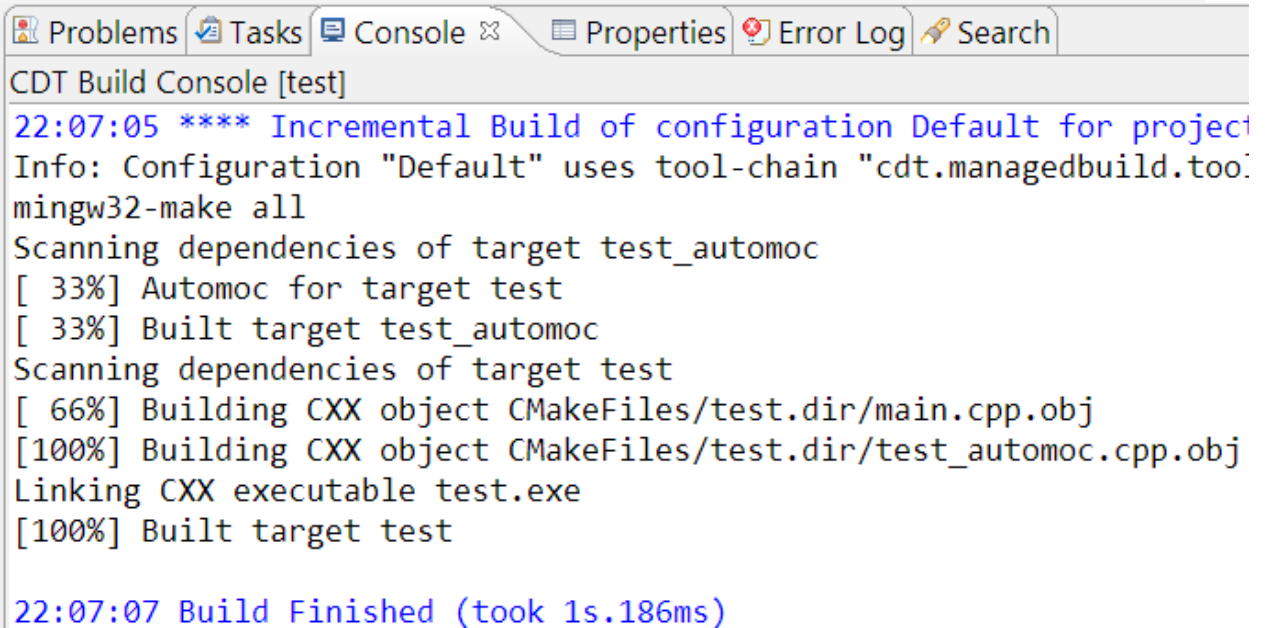
- New...
- Edit... (F2)
- Go Home
- Go Back
- Go Into
- Copy (Ctrl+C)
- Paste (Ctrl+V)
- Delete (Delete)
- Build Target (highlighted)
- Rebuild Last Target (F9)

At the bottom of the IDE, the 'CDT Build Console [test]' is open, displaying the following output:

```
22:05:40 **** Build of configuration Default for project test ****
Info: Configuration "Default" uses tool-chain "cdt.managedbuild.toolchain.gnu.mingw.base"
cmake -G "MinGW Makefiles" -DCMAKE_BUILD_TYPE=Debug ..
-- The C compiler identification is GNU 4.8.0
-- The CXX compiler identification is GNU 4.8.0
-- Check for working C compiler: C:/Qt/Qt5.1.0/Tools/mingw48_32/bin/gcc.exe
-- Check for working C compiler: C:/Qt/Qt5.1.0/Tools/mingw48_32/bin/gcc.exe -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working CXX compiler: C:/Qt/Qt5.1.0/Tools/mingw48_32/bin/g++.exe
-- Check for working CXX compiler: C:/Qt/Qt5.1.0/Tools/mingw48_32/bin/g++.exe -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Configuring done
-- Generating done
-- Build files have been written to: [redacted]/test/build

22:05:42 Build Finished (took 1s.545ms)
```

11. Собираю проект через Ctrl+B:



The screenshot shows the CDT Build Console interface. At the top, there is a toolbar with icons for Problems, Tasks, Console, Properties, Error Log, and Search. Below the toolbar, the console title is "CDT Build Console [test]". The output text is as follows:

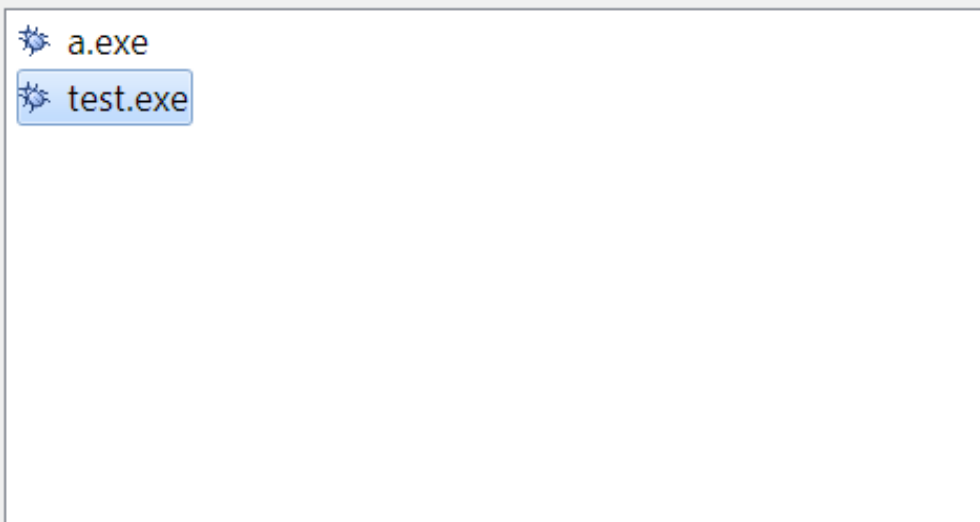
```
22:07:05 **** Incremental Build of configuration Default for project test
Info: Configuration "Default" uses tool-chain "cdt.managedbuild.toolchain.gcc-x86_64-linux-gnu"
mingw32-make all
Scanning dependencies of target test_automoc
[ 33%] Automoc for target test
[ 33%] Built target test_automoc
Scanning dependencies of target test
[ 66%] Building CXX object CMakeFiles/test.dir/main.cpp.obj
[100%] Building CXX object CMakeFiles/test.dir/test_automoc.cpp.obj
Linking CXX executable test.exe
[100%] Built target test

22:07:07 Build Finished (took 1s.186ms)
```

12. Запускаю полученный exe файл через Ctrl+F11 (при этом создается новая конфигурация для запуска):

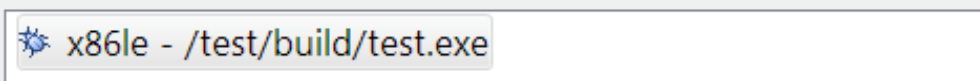
Choose a local application to run

Binaries:

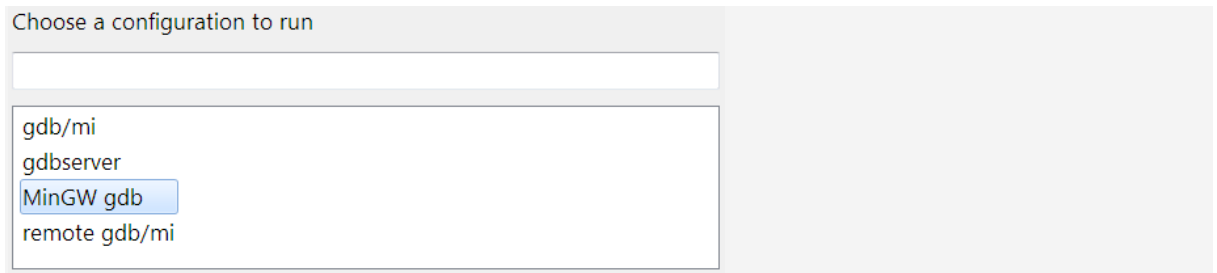


A list of binaries is shown, each with a gear icon to its left. The items are "a.exe" and "test.exe". The "test.exe" item is highlighted with a blue selection bar.

Qualifier:



A text input field containing the qualifier "x86le - /test/build/test.exe". A gear icon is visible to the left of the text.



При добавлении новых файлов через menu/class wizard в Eclipse нужно вручную их добавлять в CMakeLists.txt вот сюда:

Код Code

```
1 set (test_sources
2     "main.cpp"
3 )
```

Примерно вот так у меня происходит настройка Eclipse для Qt/CMake. Может быть можно и проще все сделать...

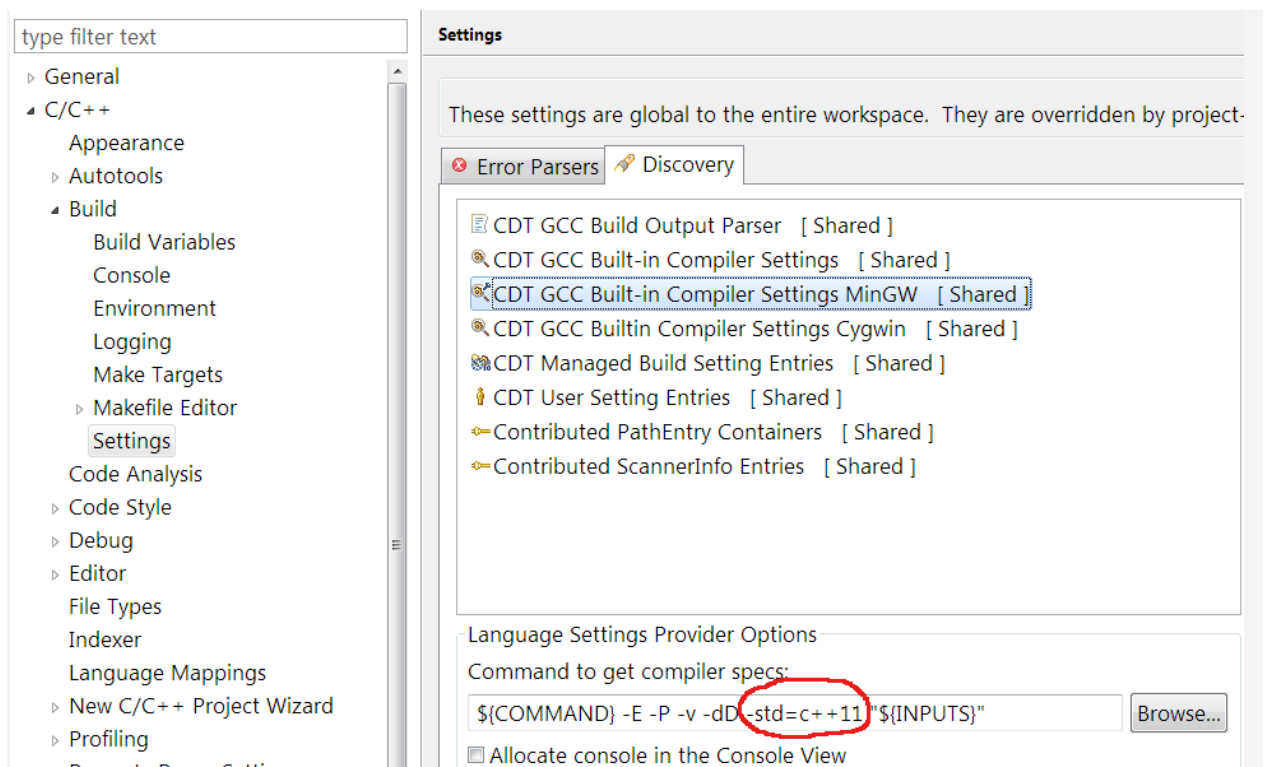
Чтобы использовать C++11 нужно еще вот что сделать:

- добавить в CMakeLists.txt строку:

Код Code

```
1 set (CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} -std=c++11")
2
```

- Добавить в Eclipse ключ в настройки компилятора (через Windows->Preferences):



Есть еще способ генерировать с помощью CMake сразу файлы проекта под Eclipse CDT через запуск:

Код Code

```
1
2 cmake -G "Eclipse CDT4 - MinGW Makefiles" -
3 DCMAKE_BUILD_TYPE=Debug <path/to/CMakeLists.txt>
```

Тогда по настройкам проекта не нужно лазить, но все равно не удобно получается, т.к. build директорию нельзя рядом с CMakeLists.txt создать, нужно на уровень выше. Кроме того в проекте получаются linked resources, с которым контроль версий в Eclipse не умеет работать.