

Ravesli [Ravesli](#)

- [Уроки по C++](#)
- [OpenGL](#)
- [SFML](#)
- [Qt5](#)
- [RegExpr](#)
- [Ассемблер](#)
- [Купить .PDF](#)

Конфигурация компилятора: Выбор стандарта языка C++

👤 [Юрий](#) |

- [Уроки C++](#)

|

📅 Обновл. 29 Авг 2020 |

👁 10592

🔖 6

Как с огромным количеством различных версий языка C++ (C++98, C++03, [C++11](#), [C++14](#), [C++17](#), C++20) компилятор понимает, какую из них ему следует использовать? Как правило, компилятор выбирает стандарт языка по умолчанию (часто не самый последний языковой стандарт). Если вы хотите использовать другой стандарт, то вам придется внести изменения в настройки вашей IDE/компилятора. Эти настройки применяются только к текущему проекту. При создании нового проекта вам придется всё делать заново.

Оглавление:

1. [Кодовые имена для версий языка C++](#)
2. [Установка стандарта языка C++ в Visual Studio](#)
3. [Установка стандарта языка C++ в Code::Blocks](#)
4. [Установка стандарта языка C++ в GCC/G++](#)
5. [Тестирование вашего компилятора](#)

Кодовые имена для версий языка C++

Обратите внимание на то, что каждый языковой стандарт имеет название, указывающее на год его принятия/утверждения (например, C++17 был принят/утвержден в 2017 году).

Однако, когда согласовывается новый языковой стандарт, неясно, в каком году удастся его принять, поэтому действующим языковым стандартам присваиваются кодовые имена, которые затем заменяются фактическими именами при доработке стандарта. Например, C++11 назывался `c++1x`, пока над ним вели работу. Вы можете по-прежнему видеть на просторах Интернета подобные кодовые имена (особенно, когда речь заходит о будущей версии языкового стандарта, у которого еще нет окончательного названия).

Вот сопоставление кодовых имен версий C++ с их окончательными названиями:

- ➔ `c++1x` = C++11
- ➔ `c++1y` = C++14
- ➔ `c++1z` = C++17
- ➔ `c++2a` = C++20

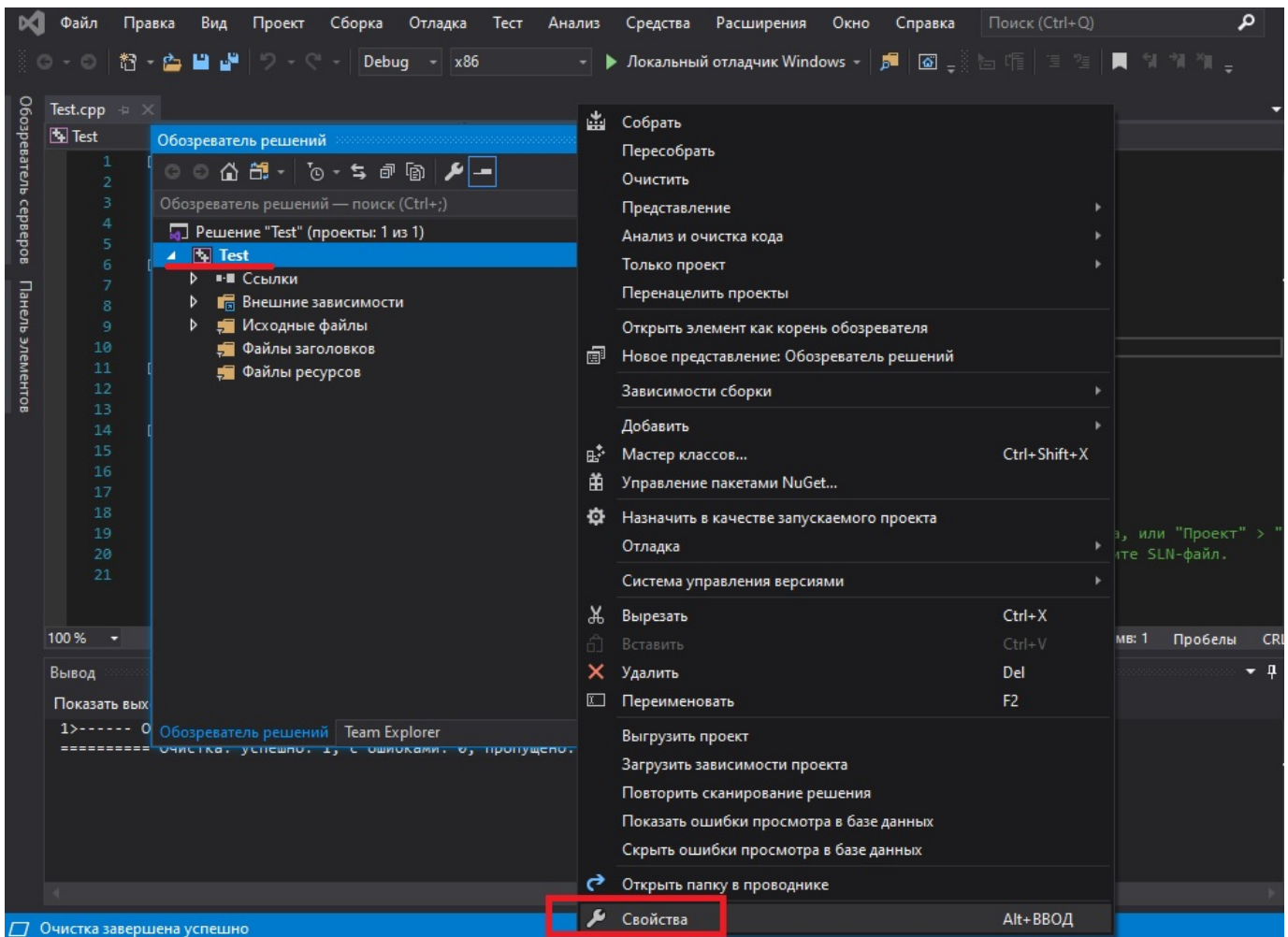
Например, если вы видите `c++1z`, то знайте, что речь идет о стандарте C++17.

Установка стандарта языка C++ в Visual Studio

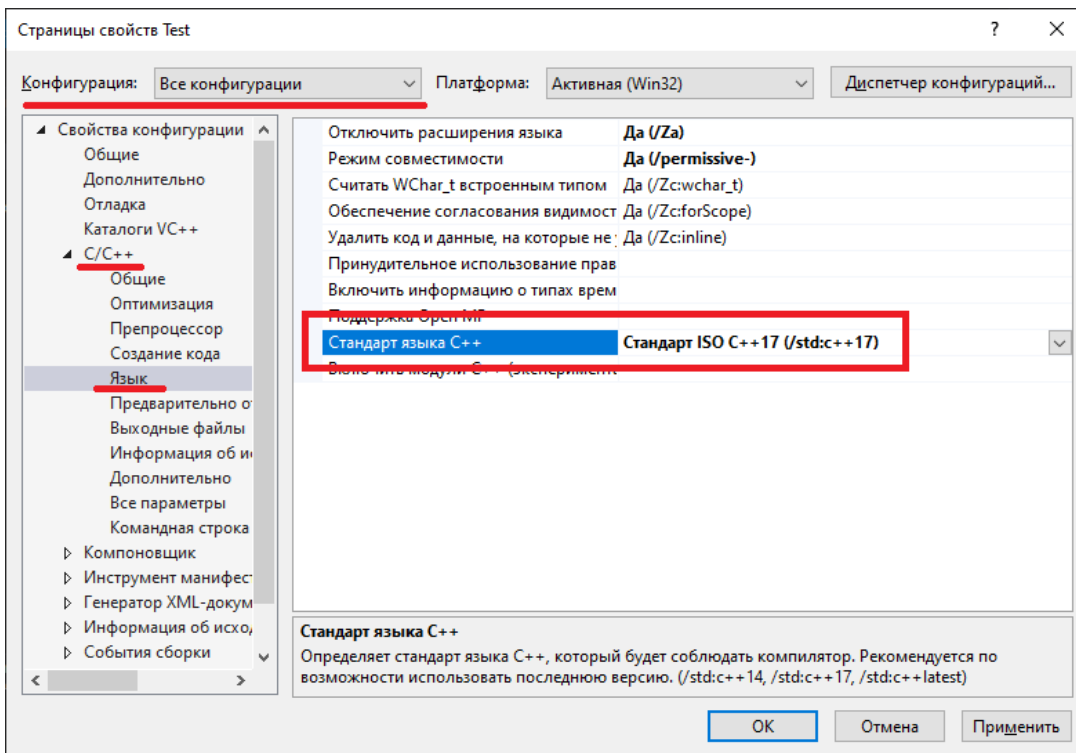
На момент написания данной статьи, Visual Studio 2019 по умолчанию использует возможности C++14, что не позволяет использовать более новые [фичи](#), представленные в C++17 и в C++20.

Чтобы использовать новый функционал, вам необходимо подключить новый языковой стандарт. К сожалению, сейчас нет способа сделать это глобально — вы должны делать это к каждому проекту индивидуально.

Чтобы использовать новый языковой стандарт в Visual Studio, откройте ваш проект, затем щелкните правой кнопкой мышки по названию вашего проекта в меню "Обозреватель решений" > "Свойства":



В диалоговом окне вашего проекта убедитесь, что в пункте "Конфигурация" установлено значение "Все конфигурации". Затем перейдите на вкладку "C/C++" > "Язык" и в пункте "Стандарт языка C++" выберите ту версию языка C++, которую хотели бы использовать:

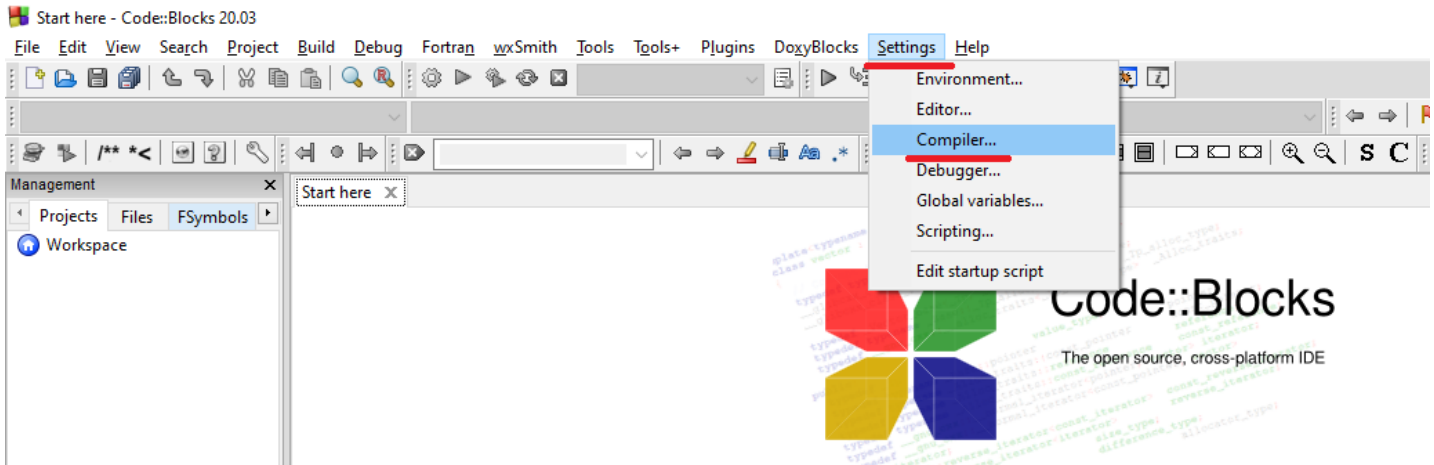


На момент написания данной статьи, я рекомендую выбрать "Стандарт ISO C++17 (/std:c++17)", который является последним стабильным стандартом.

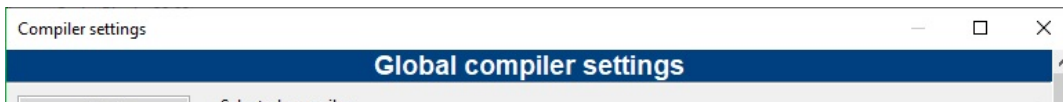
Если вы хотите поэкспериментировать с возможностями грядущего стандарта языка C++ — C++20, то вы можете выбрать пункт "Предварительная версия ... (/std:c++latest)". Просто помните, что его поддержка может иметь [баги](#).

Установка стандарта языка C++ в Code::Blocks

Code::Blocks по умолчанию может использовать стандарт C++11. Хорошей новостью является то, что Code::Blocks позволяет устанавливать ваш стандарт языка C++ глобально, поэтому вы можете установить его один раз и сразу на все проекты (а не для каждого проекта в индивидуальном порядке). Для этого перейдите в меню "Settings" > "Compiler":



Затем на вкладке "Compiler flags" найдите следующие пункты меню:



Отметьте тот пункт, у которого число обозначает ближайший (к текущему) год утверждения стабильной версии и нажмите "OK" (на вышеприведенном скриншоте этим пунктом является "Have g++ follow the C++17 ISO C++ language standard [-std=c++17]").

Примечание: Если вы не нашли в ваших настройках опцию с C++17, то вам следует обновить вашу версию Code::Blocks.

Установка стандарта языка C++ в GCC/G++

В GCC/G++ вы можете прописать соответствующие флаги `-std=c++11`, `-std=c++14`, `-std=c++17` или `-std=c++2a`, чтобы подключить функционал C++11/14/17/20, соответственно.

Тестирование вашего компилятора

После подключения версии C++17 или выше, вы можете провести тест, который позволит понять, всё ли верно сделано и действительно ли подключена новая версия языка C++. Следующая программа в C++17 должна выполняться без каких-либо предупреждений или ошибок:

```

1 #include <array>
2 #include <iostream>
3 #include <string_view>
4 #include <tuple>
5 #include <type_traits>
6
7 namespace a::b::c
8 {
9     inline constexpr std::string_view str{ "hello" };
10 }
11
12 template <class... T>
13 std::tuple<std::size_t, std::common_type_t<T...>> sum(T... args)
14 {
15     return { sizeof...(T), (args + ...) };
16 }
17
18 int main()
19 {
20     auto [iNumbers, iSum]{ sum(1, 2, 3) };
21     std::cout << a::b::c::str << ' ' << iNumbers << ' ' << iSum << '\n';
22
23     std::array arr{ 1, 2, 3 };
24
25     std::cout << std::size(arr) << '\n';
26
27     return 0;
28 }
```

Если вы не можете скомпилировать этот код, то либо вы не подключили C++17, либо ваш компилятор не полностью поддерживает C++17. В последнем случае обновите версию IDE или компилятора.

Оценить статью:

★★★★★ (80 оценок, среднее: 4,98 из 5)



[← Конфигурация компилятора: Уровни предупреждений и ошибки](#)

[Урок №7. Решения самых распространенных проблем](#)



Комментариев: 6



1. *Rammsnia:*
[15 августа 2020 в 18:04](#)

Здравствуй, у меня выдавало ошибку "No such file or directory" Ничего не помогало, переустанавливал много раз разные версии с оф. сайта и Компилятор отдельно тоже устанавливал, но помогло только то, что я из Other compiler options убрал -Wsign-conversion -Werror. Скажите, это как-то повлияет на дальнейшую работу?

[Ответить](#)



1. *zzz:*
[2 сентября 2020 в 12:32](#)

Проблема кроется в черточке (–) перед Werror. Вы, вероятно, как и я просто скопировали текст из предыдущего урока:
-Wsign-conversion –Werror
Надо удалить черточку и написать минус (-). Должно получиться вот так:
-Wsign-conversion -Werror
И всё работает.

[Ответить](#)

2. *Ronnie:*[16 мая 2020 в 19:36](#)

в codeblocks данный вами для проверки (подключился ли C++17) код не работает, выдает 2 ошибки —
 No such file or directory
 error: no input files

[Ответить](#)1. *Юрий:*[17 мая 2020 в 00:18](#)

Только что всё проверил у себя — работает. Может вы не подключили C++17 или подключили но не то. В качестве решения могу предложить скачать последнюю версию Code::Blocks с оф. сайта и внимательно проделать то, что указано в статье ещё раз)

[Ответить](#)1. *Ronnie:*[17 мая 2020 в 15:58](#)

Переустановила (оба раза скачивала тот же файл с оф. стр. C::B), с C++17 работает, проверила ради интереса с [-std=C++20], выдает ошибку: unrecognized command line option '-std=C++20', did you mean '-std=c++2a'

[Ответить](#)1. *Юрий:*[27 мая 2020 в 17:52](#)

Проблема в использовании '-std=C++20', в статье поправил это, спасибо.

Добавить комментарий

Ваш E-mail не будет опубликован. Обязательные поля помечены *

Имя * Email *

Комментарий






☐ Сохранить моё Имя и E-mail. Видеть комментарии, отправленные на модерацию

☐ Получать уведомления о новых комментариях по электронной почте. Вы можете [подписаться](#) без комментирования.

[TELEGRAM](#)  [КАНАЛ](#)


[ПАБЛИК](#) 

ТОП СТАТЬИ

-  [Словарь программиста. Сленг, который должен знать каждый колдер](#)
-  [Урок №1. Введение в программирование](#)
-  [70+ бесплатных ресурсов для изучения программирования](#)
-  [Урок №1: Введение в создание игры «Same Game»](#)
-  [Урок №4. Установка IDE \(Интегрированной Среды Разработки\)](#)

- [Ravesli](#)
- - [О проекте](#) -
- - [Пользовательское Соглашение](#) -
- - [Все статьи](#) -
- Copyright © 2015 - 2020