C# 7.0

C# версии 7.0 был выпущен вместе с Visual Studio 2017.

Что появилось:

* Переменные Out
* Кортежи и деконструкция
* Сопоставление шаблонов
* Локальные функции
* Расширенные элементы, воплощающие выражение
* Ссылка на локальные параметры
* Возврат ссылок

Другие возможности:

* Пустые переменные
* Двоичные литералы и цифровые разделители
* Выражения throw

C# версии 7.1

Выпущено август 2017 г.

C# начали выпускать доработанные выпуски с C# 7.1. В этой версии добавлен элемент конфигурации выбора версии языка, три новых возможности языка и новое поведение компилятора.

Новые языковые функции в этом выпуске

* метод asyncMain.

Точка входа для приложения может иметь модификатор async.

* Литеральные выражения default

Литеральные выражения по умолчанию можно использовать в выражениях значения по умолчанию, если можно вывести тип целевого объекта.

* Выводимые имена элементов кортежа

Имена элементов кортежа часто можно вывести из инициализации кортежа.

* Сопоставление шаблонов для параметров универсального типа

Выражения сопоставления шаблонов можно использовать с переменными, тип которых является параметром универсального типа.

Наконец, у компилятора есть два параметра -refout и -refonly, которые управляют созданием базовой сборки.

C# версии 7.2

*Выпущено в ноябре 2017 г.*

В C# 7.2 добавлено несколько небольших языковых функций:

* возможность использовать инициализаторы для массивов stackalloc;
* возможность использовать инструкции fixed с любым типом, который поддерживает шаблон;
* доступ к полям фиксированной ширины без закрепления;
* возможность переназначать локальные переменные ref;
* объявление типов readonly struct, указывающее, что структура является неизменяемой и должна передаваться в методы члена как параметр in;
* добавление модификатора in для параметров, указывающего, что аргумент передается по ссылке, но не изменяется вызываемым методом;
* использование модификатора ref readonly для возвращаемого значения метода, указывающего, что метод возвращает значение по ссылке, но не допускает операции записи в соответствующий объект;
* объявление типов ref struct, указывающее, что тип структуры обращается напрямую к управляемой памяти и всегда должен обрабатываться с выделением стека;
* возможность использовать дополнительные универсальные ограничения.
* [Неконечные именованные аргументы](https://learn.microsoft.com/ru-ru/dotnet/csharp/programming-guide/classes-and-structs/named-and-optional-arguments)
  + После именованных аргументов могут следовать позиционные аргументы.
* Ведущие символы подчеркивания в числовых литералах
  + Перед любыми печатными знаками в числовых литералах теперь могут использоваться начальные знаки подчеркивания.
* [Модификатор доступа private protected](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/keywords/access-modifiers)
  + Модификатор доступа private protected разрешает доступ для производных классов в одной сборке.
* Условные ref выражения
  + Результат условного выражения ? : теперь может быть ссылкой.

C# версии 7.3

*Выпущено май 2018 г.*

Новые возможности в выпуске C# 7.3 можно разделить на две основные группы. Одна из них — набор функций для повышения эффективности безопасного кода до уровня небезопасного кода. Вторая — постепенные улучшения существующих функций. В этом выпуске добавлены новые параметры компилятора.

В ту группу, которая отвечает за повышение производительности безопасного кода, входят следующие новые возможности:

* доступ к полям фиксированной ширины без закрепления;
* возможность переназначать локальные переменные ref;
* возможность использовать инициализаторы для массивов stackalloc;
* возможность использовать инструкции fixed с любым типом, который поддерживает шаблон;
* Можно использовать более универсальные ограничения.

Для существующих функций предоставлены следующие улучшения:

* возможность проверить == и != с типами кортежа;
* больше мест для использования переменных выражений;
* возможность подключить атрибуты к резервному полю автоматически реализуемых свойств;
* улучшенное разрешение методов, аргументы которых отличаются модификатором in;
* стало меньше неоднозначных вариантов при разрешении перегрузок.

C# версии 8.0

*Выпущено сентября 2019 г.*

C# 8.0 — это первый основной выпуск C#, специально предназначенный для .NET Core. Некоторые функции основаны на новых возможностях среды CLR, другие — на типах библиотек, добавленных только в .NET Core. В C# 8.0 добавлены следующие функции и улучшения языка C#:

* [Члены только для чтения](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/builtin-types/struct#readonly-instance-members)
* [Методы интерфейса по умолчанию](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/keywords/interface#default-interface-members)
* [Улучшения сопоставления шаблонов](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/operators/patterns):
  + Выражения switch
  + Шаблоны свойств
  + Шаблоны кортежей
  + Позиционные шаблоны
* [Объявления using](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/statements/using).
* [Статические локальные функции](https://learn.microsoft.com/ru-ru/dotnet/csharp/programming-guide/classes-and-structs/local-functions).
* [Удаляемые ссылочные структуры](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/builtin-types/ref-struct).
* [Ссылочные типы, допускающие значение NULL](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/builtin-types/nullable-reference-types)
* [Асинхронные потоки](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/statements/iteration-statements#await-foreach).
* [Индексы и диапазоны](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/operators/member-access-operators#range-operator-).
* [Присваивание объединения со значением NULL](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/operators/assignment-operator#null-coalescing-assignment)
* [Неуправляемые сконструированные типы](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/keywords/where-generic-type-constraint)
* [Выражение stackalloc во вложенных выражениях](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/operators/stackalloc)
* [Улучшение интерполированных строк verbatim](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/tokens/interpolated)

Для членов интерфейса по умолчанию требуются улучшения среды CLR. Эти функции были добавлены в среду CLR для .NET Core 3.0. Для диапазонов и индексов, а также для асинхронных потоков требуются новые типы в библиотеках .NET Core 3.0. Ссылочные типы, допускающие значение NULL, реализованные в компиляторе, гораздо удобнее использовать, когда библиотеки снабжаются заметками для предоставления семантической информации о состоянии NULL аргументов и возвращаемых значений. Эти заметки сейчас добавляются в библиотеки .NET Core.

**C# версии 9**

*Выпущено в ноябре 2020 г.*

C# версии 9 был выпущен с .NET 5. Это версия языка используется по умолчанию для любой сборки, предназначенной для выпуска .NET 5. Она содержит следующие новые и улучшенные функции:

* [Записи](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/builtin-types/record)
* [Инициализация только наборов](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/keywords/init)
* [Операторы верхнего уровня](https://learn.microsoft.com/ru-ru/dotnet/csharp/fundamentals/program-structure/top-level-statements)
* Улучшения сопоставления шаблонов[:](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/operators/patterns#relational-patterns)реляционные шаблоны и [логические шаблоны](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/operators/patterns#logical-patterns)
* [Производительность и взаимодействие](https://learn.microsoft.com/ru-ru/dotnet/csharp/whats-new/csharp-version-history#performance-and-interop)
  + [Целые числа собственного размера](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/proposals/csharp-9.0/native-integers)
  + [Указатели функций](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/proposals/csharp-9.0/function-pointers)
  + [Отключение флага localsinit](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/proposals/csharp-9.0/skip-localsinit)
  + [Инициализаторы модулей](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/proposals/csharp-9.0/module-initializers)
  + [Новые функции для частичных методов](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/proposals/csharp-9.0/extending-partial-methods)
* [Функции соответствия и отделки](https://learn.microsoft.com/ru-ru/dotnet/csharp/whats-new/csharp-version-history#fit-and-finish-features)
  + [Выражения new с целевым типом](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/proposals/csharp-9.0/target-typed-new)
  + [Анонимные функции static](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/proposals/csharp-9.0/static-anonymous-functions)
  + [Целевые условные выражения с типом](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/proposals/csharp-9.0/target-typed-conditional-expression)
  + [Типы возвращаемых ковариантных значений](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/proposals/csharp-9.0/covariant-returns)
  + [Поддержка расширения GetEnumerator для циклов foreach](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/proposals/csharp-9.0/extension-getenumerator)
  + [Параметры лямбда-дис карта](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/proposals/csharp-9.0/lambda-discard-parameters)
  + [Атрибуты для локальных функций](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/proposals/csharp-9.0/local-function-attributes)

В C# 9 получили развитие три направления предыдущих выпусков: устранение формальностей, отделение данных от алгоритмов и доступность большего количества шаблонов в большем количестве мест.

[Операторы верхнего уровня](https://learn.microsoft.com/ru-ru/dotnet/csharp/fundamentals/program-structure/top-level-statements) упрощают чтение основной программы. Сократилась потребность в формальностях: больше не нужны ни пространства имен, Program, ни классы, ни static void Main().

Введение [records](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/builtin-types/record) содержит краткий синтаксис для ссылочных типов, которые соответствуют семантике значений для равенства. Эти типы используются для определения контейнеров данных, которые обычно определяют минимальное поведение. [Методы задания только инициализации](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/keywords/init) предоставляют возможность для обратимого изменения (выражений with) в записях. В C# 9 также добавлены [ковариантные возвращаемые типы](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/proposals/csharp-9.0/covariant-returns), благодаря чему производные записи могут переопределять виртуальные методы и возвращать тип, производный от возвращаемого типа базового метода.

Возможности [сопоставления шаблонов](https://learn.microsoft.com/ru-ru/dotnet/csharp/fundamentals/functional/pattern-matching) расширены в нескольких направлениях. Числовые типы теперь поддерживают *шаблоны диапазонов*. Шаблоны можно комбинировать с помощью шаблонов and, or и not. Скобки можно добавить для уточнения более сложных шаблонов:

C# 9 включает новые улучшения сопоставления шаблонов.

* ***Шаблоны*** типов соответствуют объекту определенного типа
* ***Шаблоны в круглых скобках*** усиливают или подчеркивают приоритет сочетаний шаблонов.
* ***В шаблонах конъюнкций and*** требуется соответствие обоих шаблонов.
* ***В шаблонах дизъюнкций or*** требуется соответствие хотя бы одного из шаблонов.
* В ***шаблонах not с отрицанием*** требуется несоответствие данного шаблона.
* ***В шаблонах сравнения*** требуется, чтобы входные данные были меньше, больше, меньше или равны, больше или равны данной константе.

Высокопроизводительные вычисления на C# поддерживает еще один набор функций.

* Типы nint и nuint моделируют целочисленные типы собственного размера в целевом ЦП.
* [Указатели функций](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/unsafe-code#function-pointers) предоставляют функциональные возможности, подобные возможностям делегатов, но при этом не требуют выделения, необходимого для создания объекта-делегата.
* Инструкцию localsinit можно опустить для сохранения инструкций.

**Производительность и взаимодействие**

Еще один набор улучшений поддерживает сценарии, в которых *генераторы кода* добавляют функциональные возможности.

* [Инициализаторы модулей](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/attributes/general#moduleinitializer-attribute) — это методы, которые среда выполнения вызывает при загрузке сборки.
* [Разделяемые методы](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/keywords/partial-method) поддерживают новые модификаторы доступа и возвращаемые типы, отличные от void. В таких случаях необходимо предоставить реализацию.

**Функции подбора и завершения**

В C# 9 добавлено множество других небольших функций, улучшающих производительность разработчиков, а также написание и чтение кода:

* Выражения new целевого типа
* Анонимные функции static
* Условные выражения целевого типа
* Поддержка расширения GetEnumerator() для циклов foreach
* Лямбда-выражения могут объявлять параметры отмены
* Атрибуты могут применяться к локальным функциям

Выпуск C# 9 — это следующий этап развития, благодаря которому C# остается современным языком программирования общего назначения. Его возможности обеспечат дальнейшую поддержку современных рабочих нагрузок и типов приложений.

C# версии 10

*Выпущено в ноябре 2021 г.*

В C# 10 добавлены следующие функции и улучшения языка C#:

* [Структуры записей](https://learn.microsoft.com/ru-ru/dotnet/csharp/whats-new/csharp-10#record-structs)
* [Улучшения типов структуры](https://learn.microsoft.com/ru-ru/dotnet/csharp/whats-new/csharp-10#improvements-of-structure-types)
* [Обработчики интерполированных строк](https://learn.microsoft.com/ru-ru/dotnet/csharp/whats-new/csharp-10#interpolated-string-handler)
* [Директивы global using](https://learn.microsoft.com/ru-ru/dotnet/csharp/whats-new/csharp-10#global-using-directives)
* [Объявление пространства имен в пределах файла](https://learn.microsoft.com/ru-ru/dotnet/csharp/whats-new/csharp-10#file-scoped-namespace-declaration)
* [Расширенные шаблоны свойств](https://learn.microsoft.com/ru-ru/dotnet/csharp/whats-new/csharp-10#extended-property-patterns)
* [Усовершенствования в лямбда-выражениях](https://learn.microsoft.com/ru-ru/dotnet/csharp/whats-new/csharp-10#lambda-expression-improvements)
* [Разрешены интерполированные строки const](https://learn.microsoft.com/ru-ru/dotnet/csharp/whats-new/csharp-10#constant-interpolated-strings)
* [Типы записей могут запечатывать ToString()](https://learn.microsoft.com/ru-ru/dotnet/csharp/whats-new/csharp-10#record-types-can-seal-tostring)
* [Улучшенное определенное назначение](https://learn.microsoft.com/ru-ru/dotnet/csharp/whats-new/csharp-10#improved-definite-assignment)
* [Разрешено присваивание и объявление в одном и том же деконструировании](https://learn.microsoft.com/ru-ru/dotnet/csharp/whats-new/csharp-10#assignment-and-declaration-in-same-deconstruction)
* [Разрешен атрибут AsyncMethodBuilder в методах](https://learn.microsoft.com/ru-ru/dotnet/csharp/whats-new/csharp-10#allow-asyncmethodbuilder-attribute-on-methods)
* [Атрибут CallerArgumentExpression](https://learn.microsoft.com/ru-ru/dotnet/csharp/whats-new/csharp-10#callerargumentexpression-attribute-diagnostics)
* [Улучшенная прагма #line](https://learn.microsoft.com/ru-ru/dotnet/csharp/whats-new/csharp-10#enhanced-line-pragma)

Дополнительные возможности доступны в *режиме предварительной версии* . Чтобы использовать эти функции, необходимо [задать <LangVersion> значение Preview](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/compiler-options/language#langversion) в проекте:

* [Универсальные атрибуты](https://learn.microsoft.com/ru-ru/dotnet/csharp/whats-new/csharp-11#generic-attributes) далее в этой статье.
* [статические абстрактные члены в интерфейсах](https://learn.microsoft.com/ru-ru/dotnet/csharp/whats-new/csharp-11#generic-math-support)

C# 10 продолжает работу над темами удаления церемонии, разделения данных от алгоритмов и повышения производительности среды выполнения .NET.

Многие функции означают, что вы вводите меньше кода, чтобы выразить те же понятия. *Структуры записей* синтезируют многие из одинаковых методов, которые *записывают классы* . Структуры и анонимные типы поддерживаются *выражениями*. *Глобальные директивыи объявления* пространства имен область файла означают, что вы выражаете зависимости и организацию пространства имен более четко. *Лямбда-улучшения* упрощают объявление лямбда-выражений, где они используются. Новые шаблоны свойств и улучшения деконструкции создают более краткий код.

Новые интерполированные обработчики строк и AsyncMethodBuilder поведение могут повысить производительность. Эти языковые функции были применены в среде выполнения .NET для повышения производительности в .NET 6.

C# 10 также отмечает большую часть смены ежегодной каденции для выпусков .NET. Так как не каждая функция может быть завершена в течение ежегодного периода времени, вы можете попробовать несколько функций предварительной версии в C# 10. Можно использовать как универсальные атрибуты, так *и статические абстрактные члены в интерфейсах*, но эти предварительные версии могут изменяться до их окончательного выпуска.