История Си шарп

C# живет по принципу «всякая сущность есть объект». Его причисляют к объектно-ориентированным, а точнее объектным, языкам программирования. «Язык основан на строгой компонентной архитектуре и реализует передовые механизмы обеспечения безопасности кода» – так принято характеризовать его. Однако скептики сомневаются как минимум в его безопасности.

Сторонники C# называют его самым мультипарадигменным, универсальным, продвинутым и удобным в использовании языком программирования. Учитывая тот факт, что за ним стоит платформа Microsoft .NET, число таких сторонников достаточно велико.

Предки

Далекие предки C# появились еще в 60-х годах. Все началось с появления языка B, который в 1969 году был создан коллективом разработчиков из Технологического института Массачусетса (MIT). Главным автором B является Кен Томпсон. Тогда команда работала над операционной системой UNIX. Уже существовавший язык PL/I, применявшийся в то время для мэйнфреймов производства компании IBM, был достаточно громоздким и меньше подходил для поставленной задачи. Поэтому ученые решили создать новый язык, который и получил название B. Он является типичным представителем ранних императивных языков программирования.

После B, как это ни странно, последовал C, который был изобретен в 1972 году. Основой для нового языка послужил сам B.

Создателями C были Кен Томпсон и Денис Ритчи, которые работали в исследовательской лаборатории компании AT&T (AT&T Bell Telephone Laboratories). В 1971 году Ритчи начал создавать расширенную версию B. Сначала он назвал её NB (New B), но когда язык стал сильно отличаться от B, название сменили на C. B расширился за счет явного использования типов, структур и ряда новых операций.

По поводу возникновения языка Си Питер Мойлан в своей книге «The case against C» писал: «Нужен был язык, способный обойти некоторые жесткие правила, встроенные в большинство языков высокого уровня и обеспечивающие их надежность. Нужен был такой язык, который позволил бы делать то, что до него можно было реализовать только на ассемблере или на уровне машинного кода».

Изображение из книги «Язык Си»: М. Уэйт, С. Прата, Д. Мартин

В 1984 году Бьярне Страуструп (Bell Labs) выступил с проектом языка C++. Когда Страуструп занимался исследованиями в фирме, ему потребовалось написать несколько имитационных программ для моделирования распределенных вычислений. SIMULA-67 –

объектно-ориентированный язык – мог бы стать идеальным инструментом для решения подобных задач, если бы не его сравнительно низкая скорость выполнения программ.
Если вам приходится выбирать между написанием «хорошего» и «быстрого» кода, значит, здесь что-то не так. Потому что «хороший» код должен быть «быстрым», – объясняет Страуструп свою позицию в интервью.

Так был создан язык программирования C++, первоначально получивший название «Си с классами» (C with classes). Название «C++» придумал Рик Мэсчитти. "++" — это оператор инкремента в C, который как бы намекает на то, что язык C++, нечто больше, чем просто C.

C#

Microsoft решила отметить Миллениум выпуском новых программных продуктов. К 2000 году компания подготовила промышленные версии новых компонентных технологий и решений в области обмена сообщениями и данными, а также создания Internet-приложений (COM+, ASP+, ADO+, SOAP, Biztalk Framework). В поддержку этих новшеств Microsoft выпустила инструментарий для разработки приложений – платформу .NET. Она также объединяла «под одной крышей» несколько языков программирования, что было в новинку для того времени.

Еще одним новшеством платформы .NET была технология активных серверных страниц ASP.NET (Active Server Page). С её помощью можно было относительно быстро разработать веб-приложения, взаимодействующие с базами данных.

Язык программирования C# был создан специально для ASP.NET. На C# полностью была написана и сама ASP.NET.

Название «Си шарп» (от англ. sharp — диез) несет «сакральный» смысл. Знак «#» (в музыкальной нотации читается как «диез») означает повышение высоты звука на полтона. С другой стороны, название «C#» получается путем следующей «эволюционной цепочки»: C → C++ → C++++(C#), так как символ «#» можно составить из 4-х знаков «+».

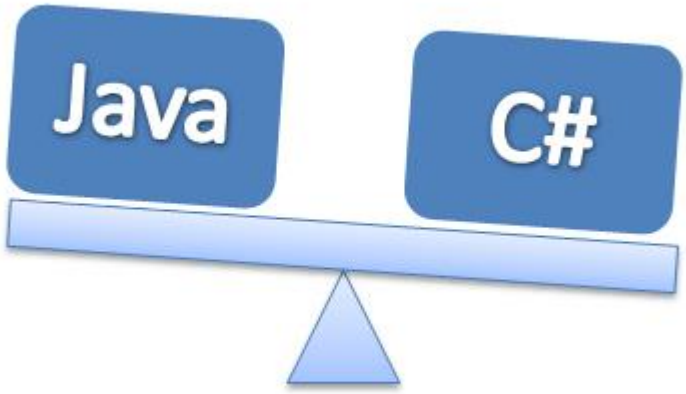
Вследствие технических ограничений на отображение (стандартные шрифты, браузеры и т. д.) и того, что знак диез ♯ не представлен на стандартной клавиатуре, знак # был выбран для представления знака диез при записи имени языка программирования. Это соглашение отражено в Спецификации Языка C# ECMA-334. Названия языков программирования не принято переводить, поэтому язык следует называть по-английски «Си шарп».

Авторами этого языка программирования стали Скотт Вилтамут и Андерс Хейльсберг — создатель Турбо Паскаля и Дельфи, перешедший в 1996 году в Microsoft.

По одной из версий, он вынашивал замысел нового языка и даже новой платформы (которая сейчас носит название .NET), еще работая в компании Borland.
C# поддерживает все три «столпа» объектно-ориентированного программирования: инкапсуляцию, наследование и полиморфизм. Кроме того, в нем была реализована автоматическая «сборка мусора», обработки исключений, динамическое связывание.

Сравнение с Java

Как и Java, C# изначально предназначался для веб-разработки, и примерно 75% его синтаксических возможностей аналогичны языку программирования Java. Его также называют «очищенной версией Java». 10% позаимствовано из C++, а 5% – из Visual Basic. И около 10% C# – это реализация собственных идей разработчиков.



Несмотря на весьма существенные различия между компонентной объектной моделью COM (основного стандарта Microsoft для компонентного проектирования и реализации программного обеспечения) и моделью Java, языки программирования имеет довольно много общего.

Единая среда выполнения программ основана на использовании промежуточного языка IL (Intermediate Language — промежуточный язык), исполняющего почти ту же роль, что и байт-код виртуальной машины Java. Используемые в рамках технологии .NET компиляторы с различных языков транслируют программы в IL-код. Так же, как и байт-код Java, IL-код представляет собой команды гипотетической стековой вычислительной машины. Но есть и разница в устройстве и использовании IL.

Во-первых, в отличие от JVM, IL не привязан к одному языку программирования. В составе, предварительных версий Microsoft.NET имеются компиляторы языков Си++, C#, Visual Basic. Независимые разработчики могут добавлять другие языки, создавая свои компиляторы в IL-код.

Во-вторых, IL предназначен не для программной интерпретации, а для последующей компиляции в машинный код. Это позволяет достичь существенно большего быстродействия программ. Содержащие IL-код файлы несут достаточно информации для работы оптимизирующего компилятора.

Кик Рэдек из Microsoft считает C# более сложным языком, чем Java. По его мнению, «язык Java был построен таким образом, чтобы уберечь разработчика от выстрела в собственную ногу» (англ. «Java was built to keep a developer from shooting himself in the foot»), а «C# был построен так, чтобы дать разработчику пистолет, но оставить его на предохранителе» (англ. «C# was built to give the developer a gun but leave the safety turned on»).

Версии

Работа над C# началась в декабре 1998 года. Проект получил кодовое название COOL (C-style Object Oriented Language).

Первая бета-версия C# 1.0 увидела свет летом 2000 года, а в феврале 2002 вместе с Microsoft Visual Studio вышла окончательная версия языка. Так как C# сочетает лучшие стороны предшествующих ей популярных языков программирования в виде C, Java и C++, программистам легко осуществить переход на C#, основываясь на знаниях любого из перечисленных языков.

Главным отличием от предшественников стала возможность написания компонентов. В C# появились такие средства, как события, методы и свойства.

В 2005 году вышел окончательный релиз версии C# 2.0. Это укрепило его позиции на рынке. Добавление новых возможностей, таких как анонимные методы, обобщения, частичные и параметризованные типы значительно расширили возможности применения C#.

Во второй версии была добавлена поддержка 64-х разрядных вычислений, что открыло возможность увеличения адресного пространства. Также было реализовано создание триггеров, хранимых процедур и типов данных на .NET языках.

Версия 3.0, вышедшая в 2008 году, позволила C# вырваться вперед в «гонке вооружений» языков программирования. Среди нововведений C# 3.0 – язык интегрированных запросов (LINQ); не явно типизированные переменные и методы расширения; Lambda-выражения, которые назначают реализацию кода делегатам с помощью нового, более простого синтаксиса. Особенно «отличились» Lambda-выражения вместе с языком интегрированных запросов.

Анонимные типы переменных позволили избежать громоздкости и откровенного неудобства при описании переменных, дав возможность объявлять новый тип непосредственно при ее создании. Новинкой в C# 3.0 также стали так называемые «ленивые вычисления», которые производят необходимые вычисления только при запросе нужных соответствующих данных.

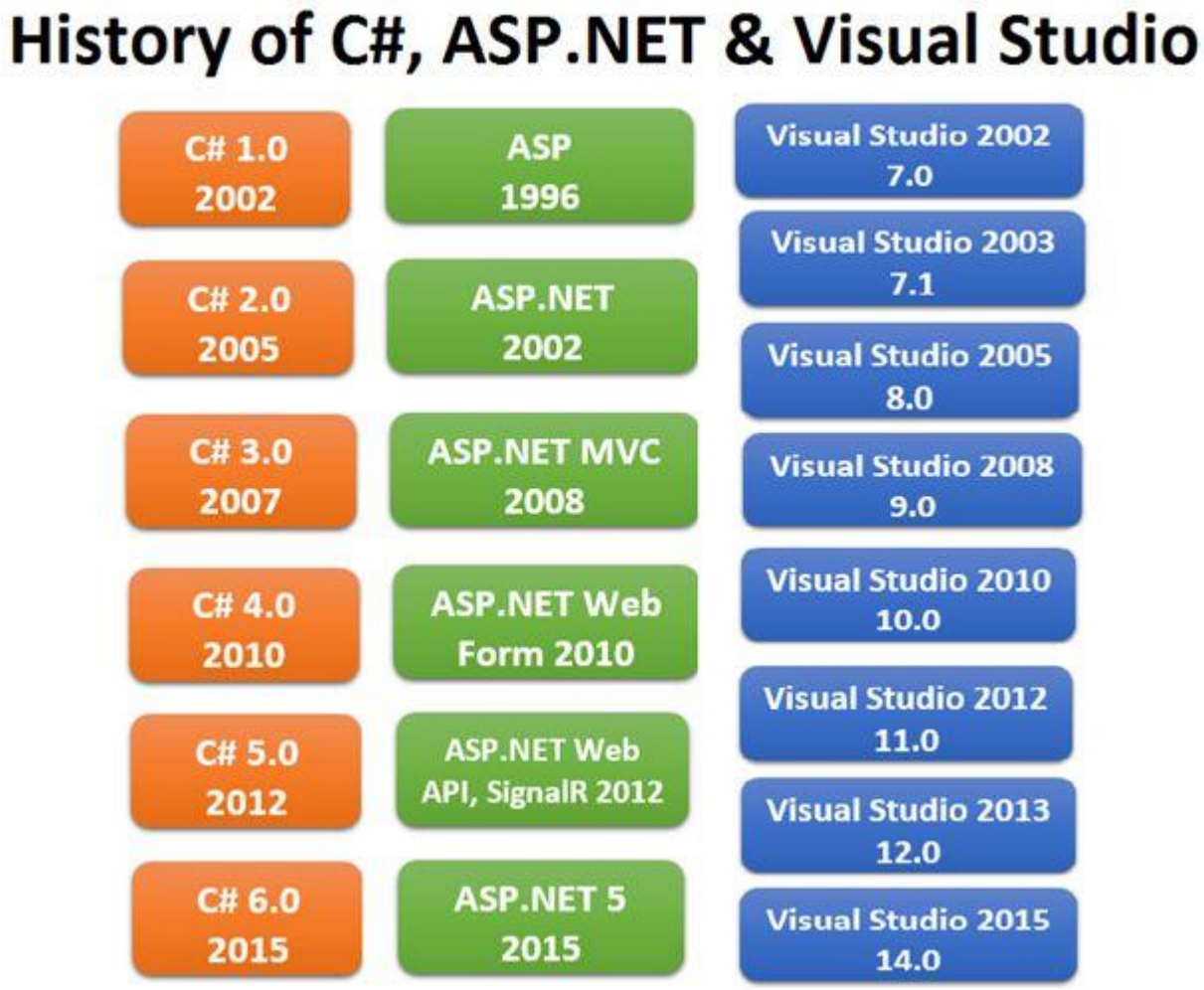
В 2010 году была выпущена версия C# 4.0. Ее главным дополнением к предыдущим версиям стали именованные и необязательные аргументы. Первые дают возможность привязки аргумента и параметра по имени, а вторые позволяют указать аргумент, который используется по умолчанию, для каждого параметра. Не менее важное новшество – тип dynamic. Он позволяет осуществлять проверку на соответствие типов объектов не на этапе компиляции, а непосредственно во время выполнения программы.

Параллельно появились новшества в .NET Framework 4.0 – библиотека распараллеливания задач (TPL) и параллельный вариант языка интегрированных запросов (PLNQ). Их поддержка позволяет осуществить параллельность выполнения кода в компьютерах с многоядерными процессорами или несколькими одноядерными.

Версия C# 5.0 появилась в 2012 году. В ней было реализовано совсем немного нововведений:

Асинхронные методы

Операция получения сведений о вызывающем объекте



Версия C# 6.0

Эта версия была выпущена через три года после выхода пятой версии — в 2015. Ее основные новшества:

• Инициализация свойств со значениями

В C# 6.0 можно инициализировать свойства со значениями. Это поможет избежать ошибки с null и пустыми значениями свойства.

• Интерполяция строк

Каждый день нам приходится сталкиваться с конкатенацией строк. Кто-то в основном использует оператор “+”, кто-то — метод string.Format(). Но проблемы с ним всем известны: при слишком большом количестве параметров тяжело понимать, что означают каждое число – {1}, {2}, {3}. В C# 6.0 придумали новую возможность, которая должна объединить достоинства обоих методов.

• Использование лямбда-выражений

В C# 6.0 свойства и методы можно определять через лямбда-выражения. Это сильно уменьшает количество кода.

• Импорт статических классов

Все статические члены класса могут быть определены с помощью другого статического класса. Но нам приходится постоянно повторять имя данного статического класса. При большом количестве свойств приходится много раз повторять одно и то же.

В C# 6.0 появилась возможность импортировать с помощью ключевого слова using статические классы.

• Null-условный оператор

C# 6.0 вводит новый так называемый Null-условный оператор (?.), который будет работать поверх условного оператора (?:). Он призван облегчить проверку на NULL значения.

Он возвращает значение null, если объект класса, к которому применен оператор, равен null.

• nameof оператор

В C# 6.0 оператор nameof будет использоваться, чтобы избежать появления в коде строковых литералов свойств. Этот оператор возвращает строковый литерал передаваемого в него элемента. В качестве параметра можно передать любой член класса или сам класс.

• Await в catch и finally блоках

До C# 6.0 нельзя было использовать в блоках catch и final оператор await. Ее можно применять для освобождения ресурсов или для ведения логов ошибок.

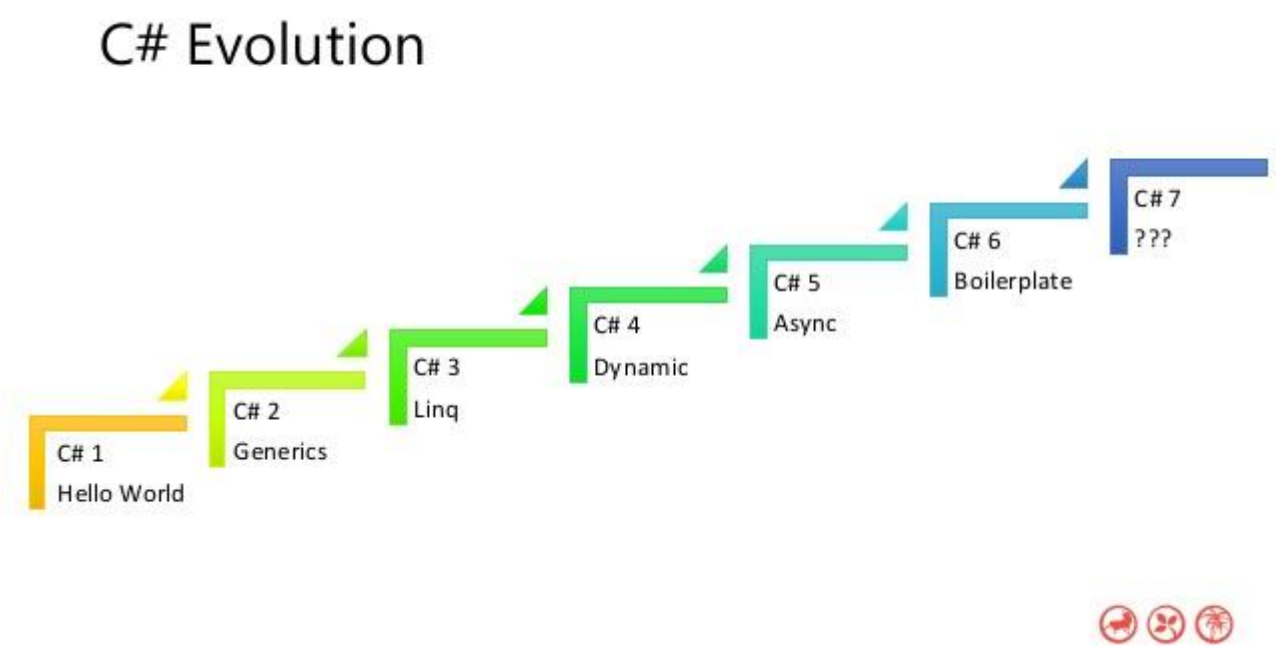
• Фильтры исключений

Фильтры исключений были в CLR, но только в VB. Теперь данная возможность появилась в C#, и можно накладывать дополнительный фильтр на исключения.

• Инициализация Dictionary

В C# 6.0 добавлена возможность инициализации Dictionary по ключу значения. Это должно упростить инициализацию словарей.

Также Microsoft улучшила новый компилятор в плане производительности.



C# 7.0

В 2016 году стало известно о новшествах готовящейся версии C# 7.0:

• Бинарные литералы

• Локальные функции

Они позволяют структурировать код, например, в стиле JavaScript.

• Сопоставление с образцом (Pattern matching)

Теперь можно использовать разнообразные средства для сопоставления.

• Условия и использование объектов в переключателях

Маленькая революция для разработчиков. Теперь switch почти ничем не ограничен. Можно использовать сопоставления.

• Записи

Автоматическое создание простых классов с произвольными полями.

• Кортежи

Иногда хочется вернуть несколько значений из метода. Для упрощения этой задачи в C# 7 были добавлены кортежи и литералы кортежей.

Перспективы C#

«Все больше компаний для крупных проектов выбирает платформу Microsoft .NET, пишет в своем блоге Михаил Флёнов. – И тут все просто, Java потерял свободу, когда ушел под крыло Oracle. До этого перехода Sun тратил огромные ресурсы на этот язык и много делалось ради свободы, открытого кода и просто Java. С переходом под крыло Oracle язык все же потерял, стал развиваться медленнее.

В случае с .NET, Microsoft делает намного больше усилий. Язык развивается все ещё достаточно серьёзно, компания выпустила бесплатный редактор кода под все платформы, и как раз начинает вкладываться в открытость, делает все, что освобождает .NET.

Трудно сейчас сказать, что будет с Windows в будущем, но пока что эта платформа останется самой популярной. А тут C# набирает популярность. Майкрософт продолжает вкладываться и в свою мобильную платформу и если у них эти действия принесут плоды, то C# может ещё более серьёзно выстрелить», заключает он.