C Sharp — Lurkmore

Правильный заголовок этой статьи — C# ***(см. технические ограничения).***

> **Народ требует ~~хлеба и~~ зрелищ!**
>
> Народ требует иллюстраций к статье!
> В конце концов, если бы мы хотели почитать, мы бы пошли в библиотеку.

> 
>
> **НЯ!**
>
> Эта статья полна любви и обожания.
> Возможно, стоит добавить ещё больше?

C♯ («Си-Шарп» и некоторые его ни разу не верные произношения: Шарфик, Си-сярп, Си-решетка, Решетки, Цэ-решетка, C++++) — мейнстримовый язык для промышленной разработки на платформе .Net, объединивший в себе мощь Java и простоту C++. Наряду с жаббой же является стандартом де-факто энтерпрайза (чем и обуславливаются высококультурные диалоги между адептами противоборствующих сторон). В силу своей молодости ещё достаточно девственен, но уже успел затрахать всех. Люто, бешено ненавидим линуксоидами и яблочниками.

По большому счёту — это Жаба с припаянным сбоку C++ и парой (функциональных => фич).

История

Давным-давно корпорация «Sun» решила разработать язык для стиральных машинок, но ввиду отсутствия в те времена в них интернетов, «изобретение» плавно переехало на серверы. Разработки его велись более 5 лет с привлечением истинных гуру вроде Никлауса Вирта, чья реализация сборщика мусора попала в код первых версий. Java начала свою историю как язык для корпоративного сектора (если пропустить самое начало, когда он был языком для микроволновок), энтырпрайзность зашкаливала за все границы: писали громадные спеки и стандарты и всё ковалось закрыто в Sun Microsystems. Технологически это всё же было лучше чем то, что создавалось до этого — особенно для задач, когда нужно писать бизнес-логику и не думать о том, завалится оно или нет, как завалится и что делать.

В то время набрал обороты Microsoft, вплоть до полного доминирования на рынке почти всего в IT. Жабба им нравилась, или по крайней мере они хотели захватить над ней контроль (применив к ней свою любимую стратегию EEE). После пары неудачных попыток (ака Microsoft Java и J++) они решили сделать свою жабу с указателями и делегатами и назвали её .NET.

Жаба тех лет отличалась невменяемостью стандартов, странными инженерными решениями (доступ к файлам через потоки, загрузка картинок в фоновом режиме и так далее), и исправлять было что. Саму жабу вылечить от этого нельзя, потому что полностью навернётся обратная совместимость и придётся ставить 10 версий одновременно. Кроме того, исправленную Java было бы крайне сложно отличить от .NET. Ну и плюс, она существует под множество платформ и менять её непросто.

.NET сделали за два года, и это было именно то, что надо для большинства Windows VS/Delphi-программистов. Чуждые UNIX-паттерны, которыми пахнет Java, изучать не нужно. Плюс .NET имел много хороших фич вроде (лямбд => и замыканий), которых в жаббе не было до восьмой версии. По сравнению с Java, .NET — это отсутствие мук выбора, «one true way», бóльшая выразительность языка, лёгкий доступ к unsafe, хорошая поддержка десктопного GUI. С другой стороны — зависимость от MS, вера дотнетчиков, что тот опенсорс, который они видят для .NET, это The OpenSource, а не детский сад в песочке с лопатками (как ни кричи на форумах о Mono — твоя платформа винда и тчк). Но в ноябре 2014 года Microsoft перевел .NET Framework под MIT-лицензию и стал неспешно открывать исходный код фреймворка. Злые языки говорят — потому что сами они даже минимально необходимый уровень функциональности и качества для удержания своего положения на этом рынке поддерживать уже не способны. Так что остается надеяться, что через годик-другой приложения .NET на UNIX, станут делом более-менее обыденным, что б там джаверы себе ни думали.

Алсо, во имя исторической справедливости надо вспомнить, что идеологически Java и .NET — полные противоположности: Java — множество платформ для единого языка, .NET — единая платформа для множества языков. Но на самом деле, что одно, что второе — множество языков для множества платформ.

Изначально C# казался неоднозначным: почти полный клон Java без всякой кросплатформенности, но с парой новых фреймворков: WinForms и ASP.NET WebForms, который превращал веб-программирование в подобие десктопного. Сейчас они выглядят нелепо, но тогда этот подход обернулся вином. Во второй версии Мелкомягкие добавили в язык ещё синтаксического сахара и довели до ума ASP.NET. И уже с .NET второй версии началось отставание Java как языка. Затем в .NET 3.5 появился LINQ (основной

аргумент шарпистов в срачах с жаббаистами) и три фреймворка: десктопный WPF, сервисный WCF и вебовский MVC. Наконец, .NET 4.0 кардинально упростил параллельное программирование, а .NET 4.5 — асинхронное.

Анти-История

Существует и альтернативная версия, что создатель C# (Андерс Хейлсберг, он же создатель Turbo Pascal и Delphi) придумал .NET, пока работал на Borland. Но там его не поняли, и он ушел в Microsoft писать J++, а потом умело пропихнул свою идею и всё заверте... И что были даже какие-то разборки между Borland и Microsoft на тему авторства.

Но в любом случае, последующая история — заслуга Большой Конторы, а не одного автора лично.

Инкарнации

C# имеет ни больше, ни меньше три разных синтаксиса. Ну или, если хотите, существует три языка с абсолютно идентичными возможностями и структурой, но разным синтаксисом. Этот майндфак стал реальностью благодаря очередной попытке Мелкософта решить важную мировую проблему: дефицит программистов. Цимес в том, что программистом может стать далеко не каждый (речь, разумеется, не о быдлокодерах). Нужно совмещение в человеке взаимоисключающих параметров, вроде технического склада ума и чувства прекрасного (подобный склад мышления должен быть свойственен, например, архитекторам, но на практике...). Но кроме того, что программеров мало, они ещё и пишут на разных языках. То есть жаббаист нихуя не способен педалить PHP-сайт, обратное верно. Конечно, можно заставить программера выучить новый язык под проект, но во-первых, эти гады, пользуясь дефицитом на рынке труда, могут спокойно послать работодателя в жопу и перейти к конкуренту через дорогу, а во-вторых, учить язык он будет недели 2-4, причём за счёт компании. И вот какого-то сумрачного мелкософтовского гения осенило: а пусть программеры пишут на любом языке, всё равно всё скомпилируется в один и тот же байткод. Профит очевиден: нужно искать не жаббаиста, или PHPетушка, или обгвидка, а тупо программиста. Это и было (если верить мелкософт-овцам) смыслом дотнета — этакая прослойка между языками. IRL схему ждал былинный отказ: если одна часть проекта на C#, а вторая на IronPython, то приходится держать в команде минимум одного шарписта и минимум одного обгвидка, потому что код друг друга они не понимают и при необходимости малейшего баг-фикса или модернизации, в лучшем случае, придётся перепедалить всё заново (в худшем пофиксят так, что потом всё равно придётся).

- C# — ну собственно основной язык под .net с C-подобным синтаксисом. Большинство дотнетчиков кодят именно на нём.
- VB.net — тот же C#, но с бейсико-подобным синтаксисом. С VB 6, разумеется, несовместим, хотя есть возможность, не без танцев с бубном, перекомпилить программу на VB6 под .NET, однако работоспособность получившегося франкенштейна никто не гарантирует. Жив.
- J# — попытка мелкософта приучить жаббаистов к .net. Fail, ибо в синтаксисе Java и C# на тот момент и так было полтора различия, а порядочно отличающуюся стандартную библиотеку жаббы имитировать, естесно, не стали. Выпилен.
- C++/CLI — C++ под .net. Nuff said.
- F# — это ни разу ни очередной C#-подобный язык, а ещё один oCaml, но не упомянуть о нём нельзя. Запилен обратно в дань моде на функциональные языки. Цимес в том, что его разработчики попытались и рыбку съесть, и косточки сожрать, в результате чего язык совмещает массу похожих возможностей из .net и oCaml, что неиллюзорно напоминает раздвоение личности. Впрочем, F# отлично подходит для изучения функциональщины и на нём даже можно написать что-то полезное. Майкрософтовцы признались, что сначала хотели функциональный язык на основе Haskell, но видимо решили, что целевая аудитория до такого не доросла.
- Хуева туча эмуляций всех остальных языков мира, включая Delphi и PHP — .NET глубоко пофиг, как выглядит синтаксис языка для программиста, главное, чтобы был правильно написан транслятор в единый язык .NET.

Особенности языка

- Компиляция в промежуточный язык.
- НИКАКАЯ защита кода. Если программа на C++ поддаётся декомпиляции только при глубоких познаниях в (диз-)ассемблере и Дао, то в C# исходный код программы легко и просто получается программками типа Reflector в доли секунды и декомпилируется без ошибок вплоть до названия переменных[1]. Полученная декомпиляция отличается от авторской разве что отсутствием комментариев — такой вот Open Source поневоле. Единственная защита — сторонние программы-обфускаторы, после которых декомпилированный код абсолютно нечитабелен. Впрочем, это верно и для Жабы, и вообще для всех языков, которые компилирует в байт-код. Но всем похуй, ибо если ты кодишь систему грамотно, то Еве от прочтения кода легче не станет; а если ты кладёшь болт на безопасность, то тебя никакой обфускатор не спасёт.

Плюсы языка

- Средний порог практического вхождения для знакомых с каким-нибудь языком, похожим на C *(спойлер:* C/C++/BCPL/B/CPL/D/Java/J++/J#/AS3/PHP/Limbo/Go/Vala/Alef — тысячи их*)*, и соответственно нехилый для тех, кто с такими языками вообще незнаком. Впрочем, количество незнакомых с Це-синтаксисом кодеров исчезающе мало — это либо окончательно задеревеневшие мозгом дельфикодеры, либо веобу-функциональное небыдло с крайней степенью ФГМ.
- Полностью ООП, даже элементарные типы данных.
- Тонны синтаксического сахара начиная с третьей версии.
- JIT-компиляция производится сразу в команды целевой архитектуры (как бэ на то она и JIT). То есть конечная программа собирается специально под ту машину, на которой запускается. Разруливание битности архитектуры берёт на себя как раз JIT-компиляция.
- Огромное количество уже готовых классов на все случаи жизни, только и ждущих, чтобы ими воспользовались. Не надо ничего придумывать. Всё уже есть, осталось только скопипастить примеры из документации.
- Для языка есть собственная среда разработки.
- В разработке языка участвовали отцы Хаскелля и Delphi.
- Наличие event-ов и, соответственно, встроенного обработчика событий.
- Присутствует возможность работать с памятью напрямую (что критикуется тру-программистами).
- Есть служба «Microsoft .NET Framework NGEN», которая сразу компилит в нативный бинарь и кэширует этот бинарник для его последующих запусков.
- Интеграция с неуправляемыми языками одной командой (в частности с C и C++). Нужны функции кернела? Легко! Некоторые умудряются скомутировать даже с Фортраном.
- Есть настоящие (а не только вложенные) многомерные массивы и опционально — проверка переполнения.
- Функциональщина. Полноценно применять ФП нельзя: отсутствует нативное каррирование. Но даже то, что есть может здорово облегчить код и логику.
- В ноябре 2014 внезапно выпустили VS Комьюнити Эдишн — бесплатный вариант VS Professional. Бесплатное использование для индивидуальных разработчиков в любых целях, для организаций с (числом ПК < 250 || годовым доходом < 1кк$) — до 5 штук на организацию, для open source, обучения или академических исследований — неограниченное количество. По слухам, Столлман таки добрался до верхов M$ и покусал там всех. Шарписты в экстазе.
- В 2016 выпустили опенсорсный .NET Core, на котором можно писать приложения для Линукса. К сожалению, GUI'вые фреймворки (WinForms и WPF) пока не портировали.

Минусы языка

- Чтобы вести нормальную коммерческую деятельность в этой стране, придётся затариться лицензиями Windows и Visual Studio (впрочем, существует и достаточно функциональная бесплатная версия). Кроме того, существует открытый аналог VS — SharpDevelop, написанный, вы не поверите, на шарпе. Вполне успешно мимикрирует под VS, реализуя большую часть ее функционала, позволяя невозбранно быдлокодить на решетках и VB.NET.

В текущей современности Visual Studio Community бесплатен для небольших команд и некоммерческого использования, а Visual Studio Code так и вообще бесплатен абсолютно.

- В синтаксическом сахаре зарыты неочевидные для индусов способы выстрелить себе в ногу. Но так как сахар индусы не любят — всем как всегда.
- Из-за огромной любви Билла Гейтса к базику сохранён эпический оператор GOTO. Что странно, потому как другие сомнительные операторы и конструкции из языка выпилили. Впрочем, для тех, кто «Дисциплине программирования» Дейкстры, предпочитает «Искусство программирования» Дональда Кнута — это не минус, а плюс, потому что Кнут в отличие от Дейкстры, GOTO ненавидевшего, умел его использовать с умом и толком.
- Присутствует возможность работать с памятью напрямую, что убивает зайчатки надёжности, но позволяет делать эпичнейшие по своей убойности костыли. Сделано, как это обычно бывает у Microsoft, из маркетинговых соображений[2].
- Видимо, по той же причине были оставлены беззнаковые целые — штука довольно полезная, когда приходится воевать за каждый байтик, ну, или хотя бы за килобайтик, но абсолютно ненужная в ынтырпрайз-приложухе, которая уже на старте отжирает несколько десятков гигабайт, и при этом позволяющая заработать весьма неочевидные баги, на поиске которых поседел не один десяток программистов.
- В отличии от жабы и прочих мейнстримовых языков с виртуальными машинами методы классов по дефолту не виртуальные. Нафига это сделано — непонятно: добавляется лишний синтаксический сахар, увеличивается вероятность пусть и легко обнаруживаемых, но все же ошибок, при этом прирост по производительности весьма сомнительный.
- Таки проигрывает пресловутой жабе и ее потомкам по скорости работы, хотя нужно признать, что в данный момент процветает куча языков, уступающих по этому параметру решеткам. Как всегда, все познается в сравнении.
- Для использования инструментов «из коробки» зачастую нужно изрядно потрахаться с конфигами.
- Язык создан Империей добра, в отличие от альтернативы, порабощенной компанией зла.
- В разработке языка участвовали отцы Хаскелля и Delphi.
- Как и всё от майкрософта, встроенные библиотеки работают только с другой продукцией майкрософта. Например, реализация клиента SMTP.

- Это всё-таки язык с парадигмой «чтобы легче писалось», а не «чтобы лучше работало». Как, впрочем, и жаба, и почти всё, появившееся после Си. Отличается сабж тем, что умело маскируется под нормальный язык (дьявол скрыт в таких детальках, что надо быть вторым Дональдом Кнутом, чтобы заметить западло вовремя), в результате чего быдломенеджеры принудительно пихают его в такие разработки, в которым ни ему, ни самим этим менеджерам не место, примерно как не место, скажем, ардуине в проекте «Вояджер».

Mono/Xamarin

Свободная кроссплатформенная реализация данного C# и .NET. Главный разработчик — Мигель де Иказа, до этого прославившийся как один из разработчиков Gnome. Сейчас уже реализована поддержка .NET 4.5 (кроме, разве что, OLE).

Эта среда породила немало специальных олимпиад, в частности, на ЛОРе. Срачи обострились после призыва одного полоумного бомжа отказаться от C# для разработки свободных приложений, поскольку патенты на данный язык принадлежат Microsoft (что конечно же не совсем так, ибо C# и CLR являются стандартами ECMA-334/335, что и позволяет свободно запилить свой фреймворк (как например тот же Mono) с сами знаете чем). Устные обещания не предъявлять по ним претензии бомжа также не устроили, так как, по его, бомжа, мнению, они — филькина грамота.

Аргументы сторонников моно и Мигелюшки следующие:

- Во всех странах, кроме пиндостана, патенты на ПО не действуют.
- В тех же, в которых действуют, они распространяются только на коммерческое использование патентованных технологий. Аргумент не совсем состоятельный, поскольку *свободное* ПО по определению может использоваться и в коммерческих целях. А ПО, которое обладает всеми критериями свободности, за исключением коммерческого использования, в терминологии Столлмана называется *полусвободным* и относится к несвободному.
- Для некоторых задач более свободной альтернативы нет, а изобретать велосипед смысла не имеет, по вышесказанным причинам. Тоже не очевидно, противники Mono предлагают, например, Python. Раньше предлагалось использовать жабу, но после недавних событий она тоже стала не нужна (с тех пор тучи рассеялись, потом снова сгустились… но кто знает, что будет завтра?).
- Бóльшая часть .NET изложена в стандарте ECMA, а эту часть Microsoft сам выпустил под свободной патентной (не софтварной) лицензией. Остальное для работы линуксового монософта не нужно (только для совместимости с виндовым). Это мнение указано в FAQ проекта Mono.

Позже Столлман назвал Мигеля предателем сообщества за то, что тот интегрирует Mono в Gnome, а также за другие не очень хорошие дела, например, за сотрудничество с MS с целью переноса свободных программ на Windows. В ответ Мигель заметил, что Господь любит всех живых существ, даже Столлмана. Вообще весь .NET является причиной попоболи всех красноглазых, поскольку уже сам факт его существования часто становится доказательством того что линукс не нужен не только на десктопе, но и на сервере (никто ведь не будет запускать серьезный проект под Mono). Множество успешных проектов на ASP.NET и вебсервисов тому доказательство.

Переименовались в Xamarin и коммерциализировалось: инди-лицензия на Xamarin.Android (который раньше был MonoDroid) или Xamarin.iOS (MonoTouch) стоит $300 в год. За всё вместе — $540. Для компаний каждая обойдётся в целый килобакс. Вот вам и свободная реализация. Впрочем остальные реализации по-прежнему бесплатные, что позволяет невозбранно писать на C# под Gnome или Windows. Правда студенты могут бесплатно получить Xamarin.Android и Xamarin.iOS тут

24 февраля 2016 года Xamarin был с потрохами куплен мелкомягкими и стал бесплатным для разработчиков небольших компаний пруф.

Unity3D

Мультиплатформенный игровой движок, который работает везде. То есть вообще везде: совместим с Windows, Linux, OS X, Android, iOS, Wii, PlayStation 3, Xbox 360 и даже в браузере запускается (с помощью трансляции в JavaScript/WebGL). Разрабатывался тремя школьниками, которые хотели написать игру, но не могли договориться какую. В итоге решили сначала написать движок с инструментарием и всё заверте… В общем, пацаны пришли к успеху. На данный момент — один из основных конкурентов Unreal Engine и наиболее перспективное направление для начинающего игродела.

Хорош он не только мультиплатформенностью, но и мощной средой разработки, которая позволяет запускать игру прямо в редакторе, а также широкой поддержкой различных форматов. Но самое главное — язык скриптов, которым и является сабж. В то время, как на большинстве коммерческих движков приходится использовать или C++, или встроенные языки, типа богомерзкого UnrealScript, Unity3D предлагает кодить на тёплом ламповом Шарпе. В серьёзном бизнесе пока что непопулярен (ибо всё ещё выдаёт картинку похуже конкурентов), но большинство перспективных кикстартеских проектов делают именно на нём (в том числе Wasteland 2 и Torments Tides of Numenera).

Примечательно, что сам движок пишется на C++.

Холивары

C# vs Java

Одна из основных дисциплин специальной олимпиады быдлокодеров (потому что для настоящего программиста язык вторичен). Цимес в том, что Жаба и Шарп — аналоги, да ещё со схожим синтаксисом. Казалось бы, живи и радуйся, но нет, сотни быдлокодеров тратят сотни рабочих человеко-часов, доказывая друг другу, чья прелесть прелестнее.

Когда Шарп только вышел, это была покоцанная Жаба с несколькими C++ фичами и парой фреймворков, которые вчистую сливали аналогам. Кросплатформенность оказалась весьма условной (то есть весь код — Windows only). Жабаисты и дельфяры напророчили шарпокапец и успокоились. А зря, ибо вторая версия Шарпа нагнула Жабу по возможностям, и даже ASP.NET наконец-то стал генерить веб-странички вменяемых размеров. Но настоящая драма развернулась после выхода третьей версии. Внезапно появился LINQ, который многократно упростил обработку данных, а также WPF/WCF/MVC. Пощипывание пониже спины выросло в жгучую попаболь, и быдлокодеры кинулись на баррикады: жабаисты доказывать (прежде всего самим себе), что шарп не нужен, а шарписты — наоборот. Ситуация ещё больше усугубилась с выходом 4 версии Шарпа, в которой многопоточное программирование стало детской забавой, а в 5-й версии async и await заставили брызгать испаряющейся кислотой.

Плюсы Java:

- Относительно вменяемый (по меркам ООП) синтаксис с минимумом ключевых слов. Язык проще в освоении благодаря более последовательному развитию.
- Отсутствие переподвывертов родом из Pascal вроде ref и out. И уж тем более наркоманских "bool?", "long?" (bool + null).
- Принудительная типобезопасность (а не типа-безопасность) без unsafe и прочих б-гомерзких goto.
- Хотя бы условная кроссплатформенность большинства сопутствующих технологий (тот же javafx).
- Разнообразие бесплатных сред разработки. Причём они, в отличие от *платной* VS, не зависают на 3-5 секунд при использовании Ctrl-X. Впрочем, все равно большая часть этих сред выглядит выкидышами свиборга по функциональностью и дизайну. А единственная вменяемая - Intellij Idea таки стоит денег и немалых, причем последние ее версии, написанные на Kotlin, нещадно тупят, периодически отказываются собирать проект из-за неправильно закешировавшегося пакета и вообще classpath exception 206 (под виндой джава порождает совершенно невменямой длины имена классов, которые не влезают в виндовую же консоль и из-за этого отказываются запускаться).
- Вменяемая структура наследования: несмотря на присутствие всякой хреноты из времён зарождения ООП (вроде abstract class), Java оставляет заметно меньше способов прострелить себе ногу.
- Есть хоть какая-то оптимизация: Java работает заметно быстрее C#. Спорное утверждение: апологеты жабы любят приводить сравнения скорости работы разных программ в разных средах, а так же обожают сравнивать один-в-один переписанный код, что не совсем корректно, так как среды разные и способы оптимизации тоже. В таких условиях и C++ иногда медленнее Java. В любом случае, свежий .NET Core сильно быстрее своего Windows-only предшественника .NET Framework, так что холивар вспыхнул с новой силой.
- Обратная совместимость и поддержка. В отличие от Microsoft, продукт не приходится переписывать каждые 3-4 года из-за того, что ваша версия JVM объявлена deprecated.
- Вывод функций не зависит от локали и прочих настроек системы.
- Хороший, годный StreamAPI начиная с Java 8.

Основной аргумент джаваистов: А у нас полная кроссплатформенность, а у вас?

- А у нас Моно. Моно всё-таки имеет ограничения, да и выходит с опозданиями. Хотя в последнее время его начинают довольно активно использовать в Ынтерпрайзе.
  - Mono не поддерживается Мелкософтом. В этом случае можно сделать троллфейс и объяснить оппоненту, чем славен Мигелюшка. Если до этого звучали упрёки в анальном рабстве у Мелкософта, то следует объявить, что жабаисты в анальном рабстве у Оракла, раз уж мусье не признаёт сторонних реализаций JVM.
- Далеко не всем программам нужна кросплатформенность, особенно в энтерпрайзе. Да и любое кроссплатформенное решение страдает именно оттого, что вынуждено быть чересчур универсальным, вследствие чего многие выгодные особенности конкретной платформы часто приходится либо игнорировать, либо реализовывать дополнительные слои абстракций, чтобы эти особенности были доступны и на других платформах.
  - Mono не доверяют и использовать не будут. Любой .NET-кун, который не раз и не два попадал на моно проекты, сразу же скажет, что это хуита, но проверять никто не будет и доказать ничего нельзя
- Будет очень скоро, надо только подождать. И в 2016 дождались! Server Core теперь открытый и кроссплатформенный. Можно писать серверные приложения на C# под что угодно. Насчёт кроссплатформенных окошечек — это то ещё мучение даже на Java со свежайшим JavaFX.
  - Карается баном.

Плюсы C#:

- Язык быстро развивается и постоянно внедряет фичи первым. Из свежего (версия 8), например, Валидация.
- Есть структуры (как объекты, но передаются по значению) и для них не нужно писать отдельные контейнеры (IntStream, DoubleStream и т.п.).
- Обилие замечательных способов прострелить себе ногу: ref, out, var, goto, адресная арифметика (с unsafe, требует включения компиляторной фичи и не используется почти никогда). Для гурманов есть is и as с приведением подтипов во время исполнения. Ваш быдлокодер купил бы себе Visual Studio!
- Nullable-типы значений: int? bool? и тп. В C# 8.0 ввели обязательное указание nullable и для ссылок, которые сейчас по умолчанию всегда могут быть null. Да здравствует строгая типизация!
- JetBrains Rider. IDE прямиком из стана идеологических противников, фактически Idea только для .NET. По сравнению с VS работает шустрее, однако иногда тупит над индексацией больших проектов.
- Есть инфиксные операции: =, +, -, *, / и т.п.
- Язык заточен под винду, есть поддержка всех основных продуктов некрософта: MS Office (уже не нужен, после 2019 будет только Office 365), LINQ to (Microsoft) SQL, WinForms, WPF (WinForms для аутистов, с векторной графикой и XAML), UWP под десятку, Винду (сервисы например), а так же их облако Azure.
- Синтаксический сахар: LINQ, лямбды (с неканоничными стрелочками: => вместо ->), свойства, вменяемые модификаторы доступа по умолчанию, var, интеграция IEnumerable не только с foreach, но и с функциями произвольного числа аргументов, ссылки на функции.
- Microsoft делает ставку на CLR: в отличие от Java, которая использует .jar (архивный формат), байт-код CLR гордо именуется exe-шником и поддерживается системой "из коробки" (системой, в которой "из коробки" нет ни драйверов, ни кодеков, ни нормального браузера).
- Microsoft не стесняется ломать обратную совместимость раз в несколько лет, выкидывая накопившееся legacy и вынуждая разработчиков заниматься тем же, что положительно сказывается на качестве разработки в целом. Там, где на Java вас заставят ковырять нечто, что было написано в середине нулевых и запускается под IE6 (кроссплатформенность, ага), на .NET вам чаще придется переписывать примерно 40% кода 5-летней давности, потому что хостинг Azure больше не поддерживает ваш уютненький .NET 4.2 родом из 2012го. А при переписывании кода каким-никаким рефакторингом все равно озаботишься. Даже если это делается по-принципу "хуяк-хуяк и в продакшн", разгребать свежее местечковое говно куда проще, чем окаменевшее и вросшее в фундамент говно мамонта. Отдельной статьи заслуживает удовольствие переписывание с обычного .NET под .NET CORE.
- Благодаря предыдущему пункту - ПОЛНОЦЕННЫЕ Generic Types. В Java, например, любой List<T> не более чем синтаксический сахар и на этапе компиляции превращается в ArrayList Object'ов, которые в нужном месте тайпкастятся к T. Так, в принципе, может оказаться, что вы List<Foo> передали в переменную типа List<Bar> и получили InvalidTypeException при попытке вызвать метод класса Bar. И да, никаких способов получить информацию о типе в рантайме через Reflection тоже не существует, если не городить костыли вроде сохранения мета-информации в самом базовом классе (так поступают в Scala, например). В дотнете же, List<string> и List<int> - разные типы, как при компиляции, так и в рантайме, соответственно, легко вытащить информацию о типе через reflection или даже просто через typeof.
- Практически отсутствует так любимая джавистами с их зоопарком систем сборки (maven, gradle, ant, что там еще придумали?) ебля с зависимостями. Все зависимости проекта видны через интерфейс Nuget, все обновляется/удаляется/ставится в один клик. Правда со временем эта штука превращается в помойку, но кого это волнует?
- В C# стандарт строже.

Основной аргумент шарпистов: А у нас ~~в квартире газ~~ LINQ, TPL, свойства, лямбды, замыкания и ещё куча ништяков, а у вас?
Ответ: не нужно. Весьма примечательно, что джаваисты объявляют ненужным абсолютно всё ровно до того момента, пока в Джаве это таки не появляется. После этого фича внезапно становится нужной и полезной. При этом, у Джаваистов принято люто фапать на Scala, которая состоит из вышеперечисленных ништяков чуть более, чем на половину.

Пример холивара

C# vs VB.NET

Внутрений .netовский холивар. В отличие от предыдущего, ведётся даже не быдлокодерами, а конченными хеллоуворлдщиками. Дело в том, что C# и VB.NET — это одно и то же. Да-да, мой юный падаван, возможности этих языков абсолютно одинаковы, и единственное различие между ними — синтаксис. В C# он C-подобный, а в VB.NET — Бэйсико-подобный. И всё. Таким образом, участники данной специальной олимпиады спорят о том, что круче: скобочки или энды.

Луговский подобные кадры охарактеризовал следующим образом:

> Я осознаю, что существуют безмозглые пидоры, у которых в список значимых критериев может входить, к примеру, синтаксис языка. Ну так накласть мне на них. Меня интересуют объективные критерии, а не цацки всякие

О так-то. Если вы узнали, что ваш коллега участвует в подобном холиваре — бейте его талмудом

Макконнелла по голове до просветления.

Исключение: в C# все-таки возможно напрямую работать с памятью (unsafe code) в обход CLR, что невозможно в VB.NET, хотя подобные свойства в языке используются чуть реже, чем никогда. Ну а ещё VB.NET case-insensitive по умолчанию — дань предкам, однако.

Исключение № 2: В C# по-умолчанию нет некоторых функций, которые есть в VB.Net. Например, IsNumeric(). Которая, однако, может быть вызвана добавлением в проект референса Microsoft.VisualBasic.

Исключение № 3: Синтаксис (и местами поведение — привет, VB.NET-ный автотайпкастинг!) того же LINQ отличаются, внезапно, не в пользу C#. Однако небыдлокодерам похуй — они давно пишут LINQ-запросы соответствующими методами, предсказуемо смотря на литеральный LINQ, как на сами знаете что.

Суть: и VB.NET, и C# в итоге компилируются в IL, и виртуальная машина оперирует IL инструкциями, CLR знать не знает ничего, ни о C#, ни о VB.NET (о них знают только компиляторы), и из этого вытекают отличия языков — какую фишку CLR добавили в язык (все .NET языки должны поддерживать CLS, но возможности CLR шире, можно вообще используя грязные хаки и Mono.Cecil нагенерить сборку с тем, что ни в одном языке не разрешено, но разрешено в CLR). Обычно все новое вначале добавляют в C#, а позже в VB.NET, но иногда, бывает и наоборот. Пример: конструкция catch when в VB.NET, которую таки добавят в C# 6

C# vs PHP

Один из самых невменяемых быдлокодерских холиваров. Дело в том, что C# популярен в виде бэкэнда для веба. И хотя он, в отличие от PHP, может использоваться и для десктопа (WinForms, WPF), и для мобайла (Modern, Xamarin) и даже для геймдева (XNA, Unity3D), бóльшая часть .net вакансий — ASP.net. Ну и как следствие, ASP.net занимает второе место среди серверных технологий. И это только в общедоступном интернете, а в Ынтерпрайзе доля стремится процентам к сорока (остальное зохавано Жабой). Когда PHPетушки видят, что какие-то корпоративные шарпобляди делают сайты не хуже, за бóльшую зарплату, да ещё и без привязки к домену, они испытывают лютейший баттхёрт и открывают на каком-нибудь SQL.ru или PHPClub.ru очередную тему «ASP.net не нужен». Туда обязательно набигает пара .net-кунов, и всё заверте…

Бóльшая часть участников холивара отличается крайней некомпетентностью в обсираемой технологии. Ну, а хуле: если бы PHP-кодерок освоил ASP.net, он бы на нём и работал, а среднему дотнетчику не нужен PHP. В результате холивар часто мутирует в Windows vs Linux или Apache vs IIS. Разумеется, в целевой системе/сервере оппонента холиварщик тоже ничего не понимает. Иногда поциэнты даже не ленятся нагуглить какой-нибудь майндфак и потом используют его как главный аргумент в споре. Например, PHPришники очень любят рассказывать об ужасах ViewState, хотя в нормальном сайте он занимает от силы 10-20 кб, что компенсируется отсутствием повторных запросов к базе, причём он отключаемый, причём в ASP.net MVC или том же NancyFx его просто нет.

А на самом деле кодерки сравнивают МПХ с пальцем: PHP — динамически слабо и неявно типизированный перлоподобный язык для создания маленьких сайтов (о чём намекает расшифровка его названия: «Personal Home Page»[Уже давно нет.][2]), а C# — статически сильно и явно типизированный сиподобный язык для тырпрайза (о чём неоднократно говорили сами мелкософт-овцы). Не конкуренты они ни разу и общего у них ничего нет. Но всем как всегда.

C# и штангисты

Пришло время воровать фичи! ККОКОКОКОКОКОКО! ИЧСХ, каждый раз умудряются поломать украденное…

- Переменные типа (aka **шаблоны**). Украдены Страуструпом когда C++ был ещё препроцессором, сломаны им же: вместо параметрического полиморфизма (одна реализация для потенциально бесконечного множества типов) получилась тупо перегрузка (по одной реализации на каждый используемый в тип), то же самое в C делалось простеньким хаком. Прикол в том, что разработчики C# (как и их предшественники с C++ и Java) тупо скопировали эту хрень, не удосужившись доработать напильником систему типов. Combo! Triple fail!
- Классы типов (aka **интерфейсы**). В стародавние времена ООП-макаки искренне верили, что абстрактные классы — решение всех проблем. Когда тупость этой идеи стала очевидна даже им самим, было принято тактическое решение украсть идею у хаскеллистов. Чтобы не палиться, было принято решение "догнать и перегнать": интерфейсы в C# могут иметь сколько угодно параметров, а в Haskell — только один (хотя это самоограничение и обходится одной прагмой). В отличие от предыдущего случая, это был таки вин (и, одновременно, фейл, т.к. ООП в чистом виде фактически умерло).
- Инфиксная нотация (aka. **операции**, надм. операторы). Допустимые операции (битовые, арифметические и сравнения), типы аргументов и приоритеты захардкожены, наследования нет, стандартных интерфейсов для обобщения этих функций тоже нет. В Haskell же можно объявлять любые операции (<$>, <*>, >>=, !!, да хоть (\^/), а для возведения в степень их, например, целых 3: **, ^ и ^^), а числовые, списочные и строковые литералы и вовсе приводятся к **любому**

подходящему типу автоматически.
- В Haskell есть константа по умолчанию def (для каждого типа можно установить своё значение) и расширение DefaultSignatures, которое позволяет указывать реализации по умолчанию для особо хитровыдуманных функций в классах типов (aka. методов в интерфейсах). Прикол в следующем: в C# есть ключевое слово default (т.е. "подставь какую-нибудь константу", переопределить значение нельзя), а в Java 8 — default-методы в классах типов.
- Списочные выражения (aka. **LINQ**). Сравните: `[ e | e <- toList es, odd e, e > 0 ]` и `from e in es where e % 2 == 0 && e > 0 select e`. Для тех, кто в танке: выбрать из es чётные натуральные числа (es ! i — аналог es[i]). Впрочем, гораздо раньше эту фичу себе отхватил Гнидо Ван Россум.
  - Код на Haskell (первый) можно написать ещё короче: `filter (\ e -> odd e && e > 0) $ toList es`.
  - Код на шарпе никто не пишет так, как он указан. В extension-методах он будет выглядеть так: `es.Where(e => e%2==0 && e>0)`
  - По хорошему, пример на C# (второй) пишется в 3 строки, хотя его читаемость это не увеличивает — по количеству синтаксического шума C# вырывается вперёд.
  - В Java 8 сделали Stream API по образу и подобию LINQ. Правда ограничились методами расширения, а укороченную форму копировать побоялись. Double fail!
- Лямбды (aka **лямбда-выражения**) — тут уже поломали всё что можно. От стрелки (=> вместо православной ->) до аппликации и каррирования. Хорошо хоть сделали функции объектами первого порядка (хоть и через б-гомерзкие делегаты с ненужным объявлением типа). Впрочем, в Java и того хуже — там сломали даже β-редукцию (пожалели лямбдам собственное пространство имён).

Интересные факты

- На C# уже написаны кросс-компиляторы, такие как script# для компиляции C# в JavaScript и jsc compiler для компиляции на IL в JavaScript, ActionScript,
- Для помощи начинающим разработчикам MS запилила бесплатный учебник, который можно невозбранно скачать с сайта З. Ы. Учебник являет собой пособие для школьников младшей и средней школы
- «C-Pound» стал локальным мемом сайта The Daily WTF.
- Внезапно Microsoft©®™ решилась открыть исходники компилятора C#. Но кто знает, что будет дальше? И таки пошло еще дальше. Обещается полный переход на опенсорс, портирование на пингвина и бесплатная редакция Visual Studio. Некоторые обещания таки выполнены: Visual Studio Community можно невозбранно скачать и пользоваться, заплатив за это незначительными ограничениями, увидела свет кроссплатформенная реализация .net.

Ссылки

Эпический тред про ненужность VB.NET и нужность C# Копия на WebArchive

Примечания

1. ↑ Между прочим Жаба тоже этим грешит: [1]
2. ↑ От C# ждут относительно безопасной работы с памятью — вот вам ссылки. Но если не дать возможность работать с указателями, все, кто мнят себя труЪ си-кодерами, сразу завопят об убогости и ущербности языка. Так что вот вам unsafe.

Языки программирования
++i + ++i 1C AJAX BrainFuck C Sharp C++ Dummy mode Erlang Forth FUBAR
God is real, unless explicitly declared as integer GOTO Haskell Ifconfig Java JavaScript LISP
My other car Oracle Pascal Perl PHP Prolog Pure C Python RegExp Reverse Engineering
Ruby SAP SICP Tcl TeX Xyzzy Анти-паттерн Ассемблер Быдлокодер
Выстрелить себе в ногу Грязный хак Дискета ЕГГОГ Индусский код Инжалид дежице
Капча КОИ-8 Костыль Лог Метод научного тыка Очередь Помолясь Проблема 2000
Программист Процент эс Рекурсия Свистелки и перделки Спортивное программирование
СУБД Тестировщик Умение разбираться в чужом коде Фаза Луны Фортран Хакер
Языки программирования

w:C Sharp en.w:C Sharp (programming language)