На чем писать программы?

(краткий обзор возможностей языков программирования)

бЕЛОРУКОВА ЕЛИЗАВЕТА игоревна

2017

Оглавление

[***Пункт 1.*** 3](#_Toc497211555)

[***Пункт 2.*** 4](#_Toc497211556)

[***Пункт 3.*** 4](#_Toc497211557)

[***Пункт 4.*** 4](#_Toc497211558)

[***Пункт 5.*** 4](#_Toc497211559)

[***Пункт 6.*** 4](#_Toc497211560)

Сегодня мы продолжаем разговор на тему какой язык программирования выбрать, начатый с статье[*“Delphi или Visual C++ - вот в чем вопрос”*](http://progcpp.narod.ru/doc/articles/delphi.htm)*.* Тема оказаласьглубже*,* чем казалось на первый взгляд, и я провел некоторые исследования по этому вопросу. Результат я хочу представить на ваш суд.

Подвигла меня к этому статья “Лучшие продукты” в журнале “Мир ПК” за январь 2001. Где опубликованы результаты опроса среди читателей журнала на тему лучших инструментов разработки, лучших персональных СУБДи корпоративных СУБД.

Так, среди персональных СУБД лучшей считают:   
Microsoft Access 2000 – 49%  
голосовавших за другие версии того же Microsoft Access – 23%   
голосов на долю FoxPro – 19%  
и только 9% на долю остальных СУБД.

Нужно ли говорить, что Access программируется, в основном, при помощи Visual Basic.

Если рассматривать корпоративные СУБД, то уверенно лидирует MS SQL server, 21% за Oracle и только один из опрошенных назвал Borland IB Database

Из средств разработки лучшим назвали:  
Delphi от Borland – 45% голосов  
Borland С++ Builder – 14%  
Microsoft Visual Basic – 28%  
Visual C++ - 8%

Цифры меня заинтересовали, так как я считал, что Microsoft лидирует по всем направлениям, и я провел небольшое сравнение этих средств разработки. Результаты представлены в Таблице 1:

Я не рассматривал отдельно Delphi и Builder только по тому, что все возможности Delphi доступны в Builder.

Таблица 1 Сравнительная характеристика языков программирования.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Характеристики** | **Средства** | | | |
|  |  | **Visual Foxpro** | **Access (VisualBasic)** | **VC++MFC** | **BC++ Builder** |
| 1. | Принцип обработки кода | Интерп.(псевдо Компилятор) | Интерп.(псевдо- компилятор.) | Компил. | Компил. |
| 2. | Язык | DBASE c  с объектами | Basic c Объектами | С++ | С++, Pascal |
| 3. | Система | Закрытая | Закрытая | Открытая | Открытая |
| 4. | Встроенные базы данных | DBF, DBC, ODBC | MDB, ODBC | Классы обработки MDB, SQL черезDB-библиотеку,  ODBC | Объекты обработки dBASE,Paradox, InterBase, MDB, Informix, FoxPro, ODBC |
| 5. | Создание пользовательских мастеров | - | - | + | + |
| 6. | Динамическое создание форм ввода, обработки сообщений | + | + | - | + |
| 7. | Модель создания приложения | - | - | каркасная (мастер) | компонентная (мастер) |
| 8. | Технология | Построители экранов, меню, отчетов (drag-and-drop), классов | Построители экранов, меню, отчетов (drag-and-drop), классов | Редактор классов, Редактор ресурсов (drag-and-drop) | Редактор объектов (drag-and-drop) |
| 9. | Вывод из баз данных на печать | Встроенный Report | Встроенный Report | Внешний | Объект : Report |
| 10 | Обработка исключений | Процедура | Процедура | Объект | Объект |
| 11 | Поддержка CASE Rational Rose | - | + | + | - |

Естественно, таблица нуждается в некотором пояснении.

***Пункт 1.***

FoxPro, Basic являются интерпретаторами, С++ - компилятором. Что из этого следует - понятно. При помощи интерпретаторов можно быстро собрать работающее приложение и изменять его по ходу работы программы, вплоть до того, что создавать программы “на лету”(строка 6 таблицы). Но за это приходится расплачиваться скоростью работы конечной программы. Правда, при таком развитии компьютерной техники, когда уже 1000 Мгц никого не удивишь, на это можно слегка прикрыть глаза.

Но вот размер готовой программы для интерпретатора принципиально не может быть маленьким, ведь необходимо тащить за собой всю оболочку, авось понадобиться какая-нибудь функция. Компиляторы на эту тему намного скромнее.

Последний Visual Basic уже заявлен, как и компилятор тоже, но что-то слабо вериться, что можно сделать из Basic нормальный компилятор, но кто знает…

# ***Пункт 2.***

Не нуждается в комментариях

# ***Пункт 3.***

Почему я называю первые два инструмента закрытыми? Потому что это действительно так. Если какая-то функция языка работает не так, как нужно, то тут уже ничего не поделаешь. Если оболочка выпадает по Fatal Error, то в С++ можно всегда посмотреть код, пройтись по трассировке и проверить, что сделано неправильно. В VC++ вся библиотека MFC поставляется с открытым кодом. В интерпретаторах остается только гадать, как обойти ошибку. Например, в Foxpro команда

rele menu \_msysmenu (удалить системное меню)

а затем щелчок мыши по остаткам меню, которые почему-то не удаляются, приведет к фатальному завершению программы и ничего с этим не поделаешь.

# ***Пункт 4.***

Самая продвинутая система в обработке баз данных – это Borland, поддерживает все на свете. Единственное, что можно сказать, что через ODBC всегда можно получить доступ к любому формату данных, а постоянно работать можно только с тем форматом, обработка которого встроена и, следовательно, имеет более быстрый доступ, чем ODBC (убирается лишнее звено). Таким образом, большое количество поддерживаемых напрямую форматов – это неплохо, но не может быть главным критерием выбора.

# ***Пункт 5.***

Создание Мастеров пользователя позволяет создавать на основе мастеров шаблоны наиболее распространенных приложений. Это просто удобно для создания большого количества одинаковых по форме, но немного разных по сути приложений, например, обработки классификаторов, но в большинстве случаев программисты вместо мастера используют свои старые наработки и все довольны.

# ***Пункт 6.***

Только Microsoft не позволяет динамически создавать формы это вытекает из принципиальной разницы между моделями создания приложения Пункт 7. Интерпретаторы, понятно, могут строить все, что угодно при работе приложения, а компиляторы только то, что заложено изначально в пределах, в которых это изначальное позволяет. У Borland более продвинутые объекты, которые позволяют устанавливать кучу свойств. Конечно, VC++ может включить в форму дополнительные кнопки и поля при работе приложения, при большом желании можно сделать и целую форму, но проще ее создать в редакторе ресурсов при создании приложения, на эту тему в Borland более продвинутая система, которая уже близка к интерпретатору, поэтому я и поставил VC++ минус. Хотя на С++ можно написать все что угодно, но если в инструменте нет специально предусмотренных для указанной задачи средств, то решение этой задачи будет трудоемко, а в интерпретаторах и вовсе невозможно.

Пункт 7. На этом я уже останавливался в статье [“Delphi или Visual C++ - вот в чем вопрос”](http://progcpp.narod.ru/doc/articles/delphi.htm). Модель от Microsoft более открыта, так как дает полностью открытый шаблон кода, тогда как Borland не пускает программиста дальше, чем позволяют свойства объектов. Это удобно для начинающих, но может привести к ситуации Пункта 3. Именно по причине разной модели построения приложения требуется разные затраты на изучение этих оболочек. В VC++ необходимо изучить шаблон приложения и только потом писать его. Хотя можно иметь только общее представление куда добавить необходимый код и не погружаться в детали. Таким образом, я бы сказал, что такая модель создания приложения более легкая для изучения в Borland, но дающая большие возможности для профессионала в Microsoft уже после изучения.

Пункт 8. Здесь дело привычки, ко всему привыкаешь, сравнивать нечего.

Пункт 9 Здесь Microsoft VC++ предлагает воспользоваться внешним Crystal Report, который специализирован для построения отчетов, тогда как в других инструментах используются встроенные генераторы (в Borland -объект отчета). Хорошо это или плохо – трудно сказать, часто специализированный инструмент лучше универсального, но требует дополнительного изучения.

Пункт 10. Показывает, что С++ просто более продвинутые системы для обработки исключительных ситуаций.

Пункт 11. Здесь Visual Basic и VC++ на высоте. Только за одно — это можно выбрать продукты Microsoft. CASE средства позволяют значительно сократить долю ручного кодирования и, что самое главное, облегчить последующее сопровождение. Когда через пару месяцев в код готовой программы необходимо внести изменения, то вспомнить иерархию и взаимодействие классов поможет именно Rational Rose, который в графическом виде представит все эти данные уже по готовому коду приложения.

Какой вывод можно сделать из этого обзора? Если необходимо быстро написать одно-два приложения, то я бы выбрал MS Access, что подтверждается абсолютным большинством опрошенных, или FoxPro, что тоже неплохо.

Но для больших систем они не годятся по причине неповоротливости и трудности в сопровождении полученных программ (по себе знаю). При написании системы есть большая вероятность, что понадобиться нечто, что не было предусмотрено изначально в языке. В С++ это решается всегда.

Если нужно написать серьезную программу, быструю и занимающую мало места, нужно выбирать между VC++ и Builder. Если при этом вы любите pascal то ваш путь к Builder(Delphi), если необходимо обрабатывать несколько типов баз данных на скорости большей, чем предоставляет драйвер ODBC, то опять Borland и, если необходима работа с базами данных Borland, такими как Informix, то выбор предопределен.

Однако, для серьезных приложений или систем, где без CASE средств не обойтись, для работы с MS SQL Server необходимо выбрать VC++.

Если вы серьезно собираетесь заниматься программированием, то выясните с какими базами и программами предстоит работать. А Microsoft во всех своих проявлениях распространен значительно больше… Вот и весь вывод.