

## ФИЛОСОФИЯ НОВОЙ ВЕРСИИ ПРОГРАММЫ VKACL

Кон В. Г. <http://kohnvict.narod.ru> [kohnvict@yandex.ru](mailto:kohnvict@yandex.ru)

У моей программы было много версий и вариантов. Новая версия опять другая, но я надеюсь, что уже надолго. И я решил объяснить в чем сложность выбора того – как и какую версию стоит развивать. Стоит начать с самого начала. Мне так повезло в жизни, что я начинал работать на компьютере именно с самого начала. В самом начале компьютер был просто более продвинутым арифмометром. Он умел только считать, то есть выполнять элементарные математические операции. Это было очень нужно и важно, так как до этого численные расчеты представляли собой очень утомительную и медленную работу.

Информацию о том, что делать, он получал с перфолент или перфокарт, а результаты выдавал на разного рода пишущие машинки текстом. Сами программы писались в коде компьютера и были примитивно простые. Они состояли из набора команд, например, таких – 4 целых числа: первое – номер операции, второе – адрес в памяти первого числа, третье – то же для второго, четвертое – для результата операции. Команды тоже имели свои адреса и среди операций был переход на произвольный номер команды в зависимости от знака результата предыдущей операции. И все. И этого было достаточно. Языков программирования (ЯП) не было совсем. Даже ассемблера.

Потом ЯП появились, самым массовым был фортран. Его текст тоже записывался на перфокарты. Компьютер прочитывал текст, компилировал программу в код, выполнял этот код и выдавал результат. Промежуточный результат в виде программы в коде компьютера не сохранялся. Через много лет такой режим работы снова стали практиковать, но не сразу. Когда появились мониторы и магнитная запись, а с ними и файлы, стали различать файлы с исходным кодом на ЯП и исполняемые файлы в коде компьютера. Вместе с файлами появились и первые операционные системы. Одной из таких систем была ДОС (дискровая операционная система), которую Майкрософт развивал для персоналок фирмы IBM.

И всем понравилась эта ДОС. Люди стали писать маленькие программы на одну примитивную задачу. Компилировали ЯП в код компьютера. А ДОС запускала одну готовую программу за другой и весь фокус состоял в том, чтобы удачно комбинировать набор программ. Для таких комбинаций были придуманы команды операционной системы из которых получился свой ЯП для этих команд. В ДОС такие командные файлы получили расширение bat от слова batch, которое переводится как пакет, группа, пачка. Главное назначение таких файлов состояло в том, чтобы запускать в работу одну программу за другой. Такие bat-файлы продолжают работать во всех версиях операционной системы Виндовс до сих пор, и число команд системы постоянно увеличивается. Есть такие командные файлы и в других системах (Юникс, Линукс и так далее).

Но развитие философии компьютерных вычислений на этом не остановилось. Появился интернет -- эта огромная газета для всего мира. И был развит язык форматирования этой газеты HTML (Hyper Text Markup Language). Он должен был работать во всех операционных системах сразу. Командные файлы тут не могли решить проблему. Стали делать интерпретаторы этого ЯП во всех системах, то есть без компиляции, сразу на исполнение. А затем появился ЯП Java как полноценная возможность делать на компьютерах любую работу, то есть вычисления, картинки, средства общения с пользователем и так далее. И снова без компиляции, сразу на исполнение, причем в разных системах один и тот же код. Правда компиляция все же осталась, но не в код компьютера а в более компактную запись, которую называли байт-код. В самом языке операторы пишутся длинными словами, чтобы было удобно читать американцам (на их языке). А байт код – то же самое, но читать сложно, да никто и не читает.

А в каждой операционной системе свои интерпретаторы исполняют один и тот же байт-код без всяких проблем. Не надо ничего переделывать и переписывать. Но ЯП Java все же оказался уже весьма сложным, так как он был рассчитан на написание очень больших программ и сделан объектно-ориентированным. То есть в нем есть куски кода, которые как бы сами по себе программы в программе. У них все свое и если они написаны, то надо знать только что на входе и что на выходе. Остальное можно не смотреть. Это позволяет людям работать в команде, где каждый делает свой кусок независимо от других и никто никому не мешает. Но для простых программ такая организация кода выглядит сложной. В ней много сложных правил, которые надо неукоснительно соблюдать. И все таки есть компиляция, значит нужен компилятор, как отдельная программа для перевода исходного кода в байт-код.

Уверен, что не всем это нравится, вот и мне не понравилось. Я попробовал сделать ЯП, который сразу исполняется и был бы удобен для написания относительно простых программ. Раз ЯП сразу исполняется, то нужен интерпретатор – одна единственная программа для какой-либо операционной системы. Но в то время, когда я начинал второй этап создания интерпретатора, был очень популярен ЯП Java и я решил писать интерпретатор на нем. Первый этап создания интерпретатора был выполнен на фортране для ДОС. Но потом он перестал работать, так как Виндовс блокировала в нем многие операции. А ДОС уже не годился для полноценной работы.

В то время мне приходилось работать и в Виндовс и в Юникс, поэтому возможность иметь интерпретатор, который работает во всех операционных системах мне была интересной. Для создания собственного интерпретатора я выбрал ЯП Java. Кроме самого ЯП сразу встал вопрос – каким делать внешний вид программы. Я решил делать 3 варианта. Первый – стандартное окно как средство разработки программ на МЛЯПе. Сокращение МЛЯП (Мой Личный Язык Программирования) появилось много лет спустя. В этом окне есть стандартное меню для разработки программ и их исполнения и можно добавить новые разделы меню. Второй – вариант без окна для готовых программ. Такие программы могут работать в командных файлах и у них больше свободы манипулирования. Но такой вариант годится только для готовой программы, которых может быть много. Третий – программа на одну задачу с собственным окном и меню и тоже для готовых программ.

Но практика показала, что размножать интерпретатор на каждую задачу и глупо, и не удобно для разработчика. Дело в том, что МЛЯП не закончился, он все еще развивается и интерпретатор получает новые версии. И получается ерунда – у разных программ разные интерпретаторы, где старые, где новые. В конце концов я пришел к выводу, что программа должна быть одна, то есть один интерпретатор. И он должен решать все задачи, которые программируются на МЛЯПе. А внешних видов можно оставить два. Первый, как был. Второй – без окна для исполнения всех готовых программ. Оба вида запускаются из разных bat файлов, но все находится в одной папке.

А как запускать разные программы? Тут тоже есть варианты, можно делать разные формы типа тех, что существуют на сайтах в интернете. Это могут быть таблицы кнопок разных форм и видов. Есть еще одна проблема -- не все программы нужны всем. По этой причине дистрибутив программы разумно разделить на части. Первая часть – интерпретатор ЯП Java. Можно использовать тот, который я предлагаю, но если у вас есть другой и лучше, но тоже будет работать. Вторая часть – интерпретатор МЛЯП и набор базовых программ. Третья часть – набор файлов для демонстрации базовых программ. Четвертая часть – депозитарий, из которого можно скачать специальные и большие программы. Их список вместе с аннотацией представлен на сайте. Ну а документация есть как в самих программах, так и в интернете на моем сайте.

Документации уже много, она есть как в виде текстов, так и в виде видео-роликов. Некоторые старые тексты частично устаревают, но не полностью. Их все равно можно использовать. Некоторые программы уже имеют свои собственные сайты, но работать они будут как часть общей программы. Почему удобно так, а не по другому. Причина простая – реклама в современном мире не только двигатель торговли. Это уже оружие в битве за умы

масс, за насаждение собственной культуры и идеалов, как сейчас модно говорит – собственных ценностей. И когда человек выбирает свою программу из общего списка, он видит, что есть и другие программы, возможно в какой-то момент они его заинтересуют. Это как в ресторане часто требуют сразу огласить полное меню банкета.

Надеюсь, что такая форма программы продержится достаточно долго. Новые версии программы буду появляться, но с ними уже легче будет работать, так как общая основа будет той же. Интересно, что хотя МЛЯП первоначально создавался для написания простых программ, но я уже давно пишу на нем сложные и большие программы и пока все получается. И даже более удобно. Хотя, конечно, за все надо платить. За удобство в одном приходится терпеть неудобство в другом. Я потому и назвал свой ЯП личным, потому что он сделан для меня как первого пользователя. Но я знаю других людей, которые использовали МЛЯП и им нравилось. Но это было давно и для других компьютеров.

Сейчас очень модным и полезным считается Питон. Моя программа в чем-то похожа на Питон, но есть и очень существенные различия. Питон заставляют учить студентов и даже школьников, он всюду. Это случилось не сразу. А я не занимался рекламой и я вообще не программист, у меня другая работа. А МЛЯП я сознательно делал только для себя. Однако познакомившись с Питоном я понял как активно он мне не нравится. Есть много других ЯП и среди них даже есть такие, в которых отдельные элементы совпадают с моими. Но не полностью. МЛЯП все равно уникален.