Введение в кроссплатформенное программирование

Кроссплатформенность (межплатформенность) — способность программного обеспечения работать с несколькими аппаратными платформами или операционными системами. Обеспечивается благодаря использованию высокоуровневых языков программирования, сред разработки и выполнения, поддерживающих условную компиляцию, компоновку и выполнение кода для различных платформ.

1 Кроссплатформенные языки программирования
2 Кроссплатформенные среды исполнения
3 Кроссплатформенный пользовательский интерфейс
4 Условная компиляция
5 Прикладные программы
6 Операционные системы
7 Среды разработки
8 Эмуляция

Кроссплатформенные языки программирования

Кроссплатформенными можно назвать большинство современных высокоуровневых языков программирования. Например, Си, C++, Free Pascal, FreeBASIC, PureBasic — кроссплатформенные языки на уровне компиляции, то есть для этих языков есть компиляторы под различные платформы. Это позволяет — при надлежащем качестве кода — не переписывать основной движок программы, меняются только особые системозависимые части.

Не менее важны для кроссплатформенности стандартизованные библиотеки среды выполнения. В частности, стандартом стала библиотека языка Си (POSIX). Из крупных кроссплатформенных библиотек — Qt, GTK+, FLTK, STL, Boost, OpenGL, SDL, OpenAL, OpenCL.

Существуют кросскомпиляторы — компиляторы, генерирующие исполняемый код для платформы, отличной от той, на которой запущен сам компилятор.

Кроссплатформенные среды исполнения

PHP, Perl, Python, Tcl и Ruby — кроссплатформенные интерпретируемые языки, их интерпретаторы существуют для многих платформ.

Среды исполнения ActionScript Virtual Machine, Java Virtual Machine и .NET также кроссплатформенны, однако на их вход подаётся не исходный текст, а промежуточный код. Поэтому программы, написанные на ActionScript, Java и

C#, можно запускать под разными операционными системами без предварительной перекомпиляции.

Кроссплатформенный пользовательский интерфейс

На разных операционных системах — независимо от того, как технически достигнута работа в них — стандартные элементы интерфейса имеют разные размеры. Поэтому простое жёсткое позиционирование элементов интерфейса невозможно — под другой операционной системой они могут «налезать» друг на друга. Существует несколько подходов:

Единый стиль, общий для всех операционных систем, программы выглядят одинаково под всеми системами. Так работают интерфейсные библиотеки Java наподобие Swing.

Плюс: можно жёстко расставлять элементы управления на манер Delphi; оригинальный стиль.

Минус: системе приходится иметь свои экранные шрифты; стиль отличается от стиля ОС.

Самоадаптирующийся интерфейс, подстраивающий сетку под реальные размеры элементов управления. Типичные примеры — Qt, wxWidgets, XUL.

Плюс: стандартный стиль операционной системы, очень быстрый и «скинующийся» под Windows XP, Vista и 7; некоторая автоматизация локализации.

Минус: чтобы собрать самоадаптирующуюся сетку, требуется квалифицированный программист; затруднена плотная компоновка.

Гибридный подход реализован в GTK+.

Плюс: шрифты можно брать из системы, а не «тащить» свои; некоторая автоматизация локализации.

Минус: берёт все недостатки от первых двух подходов. Стиль отличается от стиля операционной системы; затруднена плотная компоновка.

В любом случае, под другими операционными системами требуется хотя бы минимальное тестирование, так как возможны ошибки компоновки.

Условная компиляция

Даже несмотря на широкую, в общем, стандартизацию аппаратного и программного обеспечения, программисту часто приходится налаживать ветви под разные операционные системы, включая ту или другую с помощью условной компиляции.

Например, браузер Mozilla Firefox имеет разные комплекты значков под разные операционные системы.

Прикладные программы

Большое количество прикладных программ также являются кроссплатформенными. Особенно это качество выражено у программ, изначально разработанных для Unix-подобных операционных систем. Важным условием их переносимости на другие платформы является совместимость платформ с рекомендациями POSIX, а также существование компилятора GCC для платформы, на которую осуществляется перенос.

Операционные системы

Современные операционные системы также часто являются кроссплатформенными. Например, операционные системы с открытым исходным кодом, например, NetBSD, Linux, FreeBSD, AROS могут работать на нескольких различных платформах, чаще всего это x86, m68k, PowerPC, Alpha, AMD64, SPARC. Первый выпуск Microsoft Windows NT 4, вышедший в 1996 году, поддерживал четыре платформы (x86, Alpha, MIPS и PowerPC), в дальнейших версиях Windows NT осталась только поддержка платформы x86. Современная Microsoft Windows может работать как на платформе Intel x86, так и на Intel Itanium (точнее, для Itanium есть только версии Windows 2000/XP, Windows 2003 и Windows 2008, после чего поддержка была свёрнута). Операционная система NetBSD является самой переносимой[нейтральность?] [1], она портирована на большинство существующих платформ.

Среды разработки

Ряд IDE, в том числе Free Pascal, Lazarus, Qt Creator, работают на разных операционных системах: Linux, Windows и других[2].

Эмуляция

Если программа не предназначена для исполнения (запуска) на определённой платформе, но для этой платформы существует эмулятор платформы, базовой для данной программы, то программа может быть исполнена в среде эмулятора.

Обычно исполнение программы в среде эмулятора приводит к снижению производительности по сравнению с аналогичными программами, для которых платформа является базовой, так как значительная часть ресурсов системы расходуется на выполнение функций эмулятора.