Блог GunSmoker-a

...when altering one's mind becomes as easy as programming a computer, what does it mean to be human?..

11 сентября 2011 г.

Сериализация - налоги

Оглавление серии.

Заголовок статьи использует понятие "налог на ПО" - это термин, обозначающий вещи, которые вы делаете не для пользователя или себя, а для поддержания здоровья общей экосистемы компьютера.

Этот пост будет говорить об общих подходах к правильному дизайну работы с файлами и хранилищами данных. В т.н. "книжках по Delphi" об этом обычно не пишут, поскольку такие вещи не имеют отношения к Delphi, а описываются в "книгах по Windows". По какой-то не очень понятной мне причине, "книги по Windows" читаются не так уж многими людьми. Хотя, казалось бы, как можно писать программы под какую-то ОС, не владея знаниями по этой ОС. В любом случае, если вы захотите исправить этот свой недостаток, то вот книги, которые вам стоит прочитать:

- Windows via C/C++. Программирование на языке Visual C++ (старое название: Windows для профессионалов. Создание эффективных Win32-приложений с учетом специфики 64-разрядной версии Windows)
- Защищенный код
- Защищенный код для Windows Vista
- Внутреннее устройство Microsoft Windows: Windows Server 2003, Windows XP, Windows 2000. Мастер-класс

Данные книги являются **обязательным** чтением для любого разработчика программ под Windows.

Тем не менее, в этом посте я лишь кратко упомяну основные принципы. Я не буду их обосновывать - можете поверить мне на слово. Если на слово верить не хочется - придётся читать книги или документацию. Возможно, позже я напишу что-то более развёрнутое, но пока - только так.

Варианты хранения данных

Самая большая проблема с хранением данных программы - вопрос, где хранить данные. Начинающие программисты часто создают кошмарных монстров, которые (к примеру) хранят данные в папке программы или же модифицируют сами себя для хранения конфигурации. Почему они так делают? Потому что они не знают, как надо делать правильно: им об этом никто не сказал.

Поэтому, если вы не знаете, как нужно правильно хранить данные программы - я вам об этом сейчас расскажу ;)

Поясню, что тут я буду говорить, как делать правильно, но не буду говорить, как именно это делается (пишется в реестр, работается с файлами и т.п.). Последнее - тема последующих статей в этой серии (см. в начале статьи ссылку на оглавление).

Итак, основные варианты для хранения данных:

- Файл на диске
- Реестр (registry)

- Ресурсы программы

Давайте кратко опишем для чего пригоден каждый вариант. Начнём с конца.

Ресурсы

Ресурсы программы - это дополнительные внешние данные, которые подключаются к вашему исполняемому файлу (.exe, .dll, .bpl). После запуска программы во время её работы эти данные можно открыть (загрузить) и работать с ними уже в памяти программы. К примеру, в ресурсах программы хранится версионная информация и данные форм (размеры, положения компонентов на форме и т.п. свойства). Поскольку запущенный файл нельзя модифицировать, то ресурсы являются местом хранения данных только для чтения. В них обычно хранят неизменные данные, которые необходимы программе для работы, но эти данные по каким-либо причинам нельзя оформить в виде констант в коде. Причин обычно две: либо данные приходят из внешнего источника (скажем, иконка программы), либо же они имеют большой размер (к примеру, шаблон документа по умолчанию).

Короче говоря, если вам нужны неизменяемые данные, которые вы не можете оформить в коде - то это явный кандидат на размещение в ресурсах программы.

Остальные варианты (диск и реестр) допускают как чтение, так и запись данных.

Реестр

Реестр предназначен для хранения конфигурационных данных программ. Настройки, опции и т.п. вещи - это верный вариант для сохранения в реестр. Обычно использование реестра быстрее, проще и надёжнее хранения конфигурации в файлах настроек. Кроме того, что более важно, следуя простым правилам по хранению данных в реестре, вы подготовите свою программу к работе в ситуациях, про которые вы даже не знаете или не хотите знать (например или вот). А вот мнение разработчика Windows о причинах введения реестра для хранения программной конфигурации.

Что в реестре не нужно хранить:

- Данные большого объёма (это точно не конфигурация)
- Данные, которые должны редактироваться пользователем (вы можете хранить их в реестре, но тогда вам надо сделать GUI по их правке)
- Динамические данные (вроде процента выполненной операции)
- IPC (т.е. взаимодействие между программами), за исключением случая когда эти данные - конфигурация
- Данные, которые являются файлами (скажем, шаблоны документов)
- Наверное, могут быть и другие примеры данных, не подходящих для хранения в реестре, но мне больше в голову ничего не приходит. В общем, помните, что реестр - это база данных конфигурации системы.

Разумеется, не всегда использование реестра возможно (пример: мобильные программы). Понятно, что и из сказанного выше могут быть (и бывают) исключения. Но в общем и целом - если вы можете использовать реестр, то лучше делайте это. Тогда вы будете образцовым гражданином мира Windows. Использование реестра - это зачастую обязательное требование в мире корпораций, потому что они могут использовать групповые политики и производить точечную настройку доступа. Так что если вы хотите, чтобы IBM купила у вас вашу программу на 30'000 рабочих мест - вам лучше бы быть заранее готовым к этому (конечно, это была шутка, но с большой долей правды).

Файлы на диске

Все прочие данные (коих большинство) требуют размещения в отдельных файлах данных.

Размещение данных

Когда вы определились с типом хранилища - это ещё не всё. Нужно ещё определить, где конкретно вы будете хранить данные, в каком именно месте. Если с ресурсами тут всё понятно (потому что и вариантов

никаких нет), то с реестром и файлами не всё так однозначно. Впрочем... вы можете выделить ресурсы в отдельный файл - т.н. "ресурсную DLL": это обычная DLL, но она не содержит экспортируемых функций (не содержит кода), а только ресурсы. Вы можете загрузить её и извлечь из неё ресурсы. Такой трюк иногда может быть полезен. К примеру, чтобы облегчить размер .exe (не нужно заново закачивать неизменные ресурсы при обновлении программы, можно обновить только .exe) или чтобы менять набор ресурсов в run-time (к примеру, язык программы: выгрузили одну DLL и загрузили другую).

Локальные и глобальные данные

Прежде всего следует сказать, что программа должна разделять данные на локальные (действующие на текущего пользователя) и глобальные (действующие на всю машину, на всех пользователей). К примеру, показывать или нет иконку в трее - это, очевидно, настройка пользователя. Я могу включить эту опцию, а моя гипотетическая сестра (которая сидит за этой же машиной) - выключить. И у нас будут разные настройки. То, в какой программе играть .mp3 файлы - тоже настройка пользователя. А вот вещи, которые существуют только в одном экземпляре - это глобальные данные. Пример - куда установлена программа (в предположении, что программа на машину ставиться один раз, глобально, установщиком, а не распаковывается в папку каждым пользователем). Ещё пример: настройки файла подкачки. Поскольку файл подкачки - один, он глобален в системе, то два пользователя не могут иметь разные его настройки.

Понятно, что в типичной программе - 99% всех её данных - локальны. И лишь данные вроде пути к установленной программе являются глобальными.

Что ещё тут надо понимать: глобальные данные - это всегда только чтение. Почему? Потому что в противном случае это будет дыра в безопасности. Основной принцип тут достаточно прост и прозрачен: на других пользователей могут влиять только администраторы системы. Простые пользователи могут влиять только на себя, но не на других пользователей.

Поэтому глобальные данные или настройки обычно записываются/создаются установщиком программы - который работает глобально и запускается под администратором системы. После записи они не меняются, а служат "мастер-шаблоном". В процессе же работы самой программы она по желанию может заместить некоторые настройки локально.

Итак, когда вы (надеюсь) почувствовали разницу между двумя типами данных, настало время рассказать про то, где их, собственно, надо хранить. Начнём с самого простого: реестр.

Реестр

Тут всё просто: свои данные программа должна хранить в HKEY_CURRENT_USER\Software*Имя-компании**Имя-программы**Версия*\ (а если данные глобальны, то вместо HKEY_CURRENT_USER используется HKEY_LOCAL_MACHINE). Курсивом выделены части, которые вы должны заменить на свои данные (формат и содержание этих частей не фиксированы, на ваше усмотрение). К примеру, Delphi хранит свои настройки в HKEY_CURRENT_USER\Software\Embarcadero\Delphi\8.0\ (точный путь, конечно, зависит от вашей версии Delphi). Если вы пишете программу как энтузиаст, то часть "Имя-компании" можно опустить (или использовать ваше имя). Если вы не планируете разделять настройки между версиями, то часть "Версия" тоже можно опустить.

Частная ситуация - ассоциации с файлами. Их полагается создавать при первом запуске программы в HKEY_CURRENT_USER\Software\Classes\. К примеру: HKEY_CURRENT_USER\Software\Classes\.txt=MyTextFileType. Если программа устанавливается глобально для всех пользователей, то ассоциацию файлов с программой должен делать установщик программы, используя для этого ключ HKEY_LOCAL_MACHINE\Software\Classes\.

Если у вас на руках есть код, который говорит про ассоциации файлов и оперирует HKEY_CLASSES_ROOT, то вы должны знать, что этот код - старый. Он написан в прошлом веке (а если он написан в этом веке, то автор кода не потрудился привести свои знания в актуальное состояние). Современный код вместо HKEY_CLASSES_ROOT должен использовать HKEY_CURRENT_USER\Software\Classes\. Подробнее - тут (пример 3).

Вот, собственно и всё. Локальные даные - в HKCU\Software*ДальшеВашПуть*\, глобальные данные - в HKLM\Software*ДальшеВашПуть*\, ассоциации файлов в установщике программы - HKLM\Software\Classes\, а в самой программе - HKCU\Software\Classes\.

Конкретная структура данных для ваших данных в реестре не фиксируется и остаётся на ваше усмотрение. Единственный совет от меня - если вы используете какой-то путь вида HKEY_CURRENT_USER\Software\Имя-компании\Имя-программы\Версия\ (не важно, сколько и какие компоненты в нём), то я бы рекомендовал использовать ровно такую же структуру пути и на файловой системе: C:\Program Files\Имя-компании\Имя-программы\Версия\.

Теперь - более сложная часть: файлы на диске. Почему более сложная? Потому что в реестре мы хранили только конфигурацию, а в файлах мы можем хранить много всего, не только настройки программ.

Файлы на диске

Во-первых, сами программы должны устанавливаться в папку Program Files. Под программами я понимаю .exe, DLL и .bpl - т.е. исполняемые файлы. Внутри папки рекомендуется использовать ту же структуру, что и в реестре: *Имя-компании**Имя-программы**Версия*\ - так что полный путь к файлу может выглядеть так (к примеру): C:\Program Files\Embarcadero\BDS\8.0\readme.htm (ну, вообще-то Delphi использует "RAD Studio" вместо "BDS").

Что касается данных программ, то начну я с самой частой ошибки: хранение данных в папке с самой программой. Это - самая большая ошибка новичков. Хотя при некоторых специальных условиях это может быть нормальным (скажем, данные только для чтения или же вы меняете файловые ACL), но когда это делает новичок для хранения всех подряд данных - это неправильно. Почему? Простая причина - вы мешаете разнородные данные в одном месте: код (программа) и данные (не программа). Чем это плохо? Да очень просто: ваша программа же должна менять данные, да? Что означает, что если она может менять данные, то она может менять и программы. А кто у нас может менять программу? Вирусы, конечно же.

Иными словами, если данные и программы хранятся одинаково - то здесь нет никакой защиты от вирусов. Вот почему вам нужен антивирус. Если же данные хранятся в другом месте - вы можете сделать программы только для чтения, а данные - для чтения записи. И тогда вирус просто не сможет заражать программы (и не будет нужен антивирус).

Это грубое и краткое изложение - подробнее см. ссылки в конце статьи или читайте книги по ссылкам в начале. Сейчас же вам важно понять, что (изменяемые) данные в папке с программой - это табу в современном мире.

Но если данные нельзя хранить в папке с программой, то где их хранить?

Тут всё зависит от того, что это за данные. Если это данные, которые нужны вашей программе - то хранить их надо в т.н. папке Application Data (её ещё называют %AppData%). Вас не должно волновать её местоположение, потому что путь к ней меняется на каждой системе и узнавать его надо через специальную функцию (я покажу это ниже), но я дам вам примеры путей, чтобы вы представляли о чём идёт речь:

- Windows XP: C:\Documents and Settings\Александр\Application Data\
- Windows 7: C:\Users\Александр\AppData\Roaming\

(для глобальных данных имя пользователя в пути заменяется на "All Users")

Дело усложняется и тем, что кроме локального и глобального варианта у папки есть третий: локальный вариант делится на Roaming (%AppData%) и собственно Local (%LocalAppData%). Об отличиях можно узнать по этой ссылке. Кратко: если вы не уверены: выбирайте Roaming - это стандартный вариант, и обычно именно его подразумевают, когда говорят об Application Data. В Local же вариант идут данные вроде кэша. Которые не являются твёрдо необходимыми и могут быть пересозданы, либо же зависят от машины (и не имеют смысла при переносе на другую машину).

Примечание: на самом деле у Application Data есть ещё и четвёртая модификация, но я не буду про неё говорить. Если она вам понадобится - вы уже давно выросли из уровня этой статьи (а если нет, то вы - безумец).

В любом случае, Application Data - стандартное место для хранения данных вашей программы. В ней (не глобальном варианте) программа может хранить и использовать любые свои данные. В глобальный вариант этой папки стоит записывать глобальные данные - скажем, мастер-копии шаблонов, файлов, БД и т.п. файлы только для чтения (напомню: локальные мы менять можем, глобальные - нет).

Внутри папки рекомендуется использовать ту же структуру, что и в реестре: *Имя-компании**Имя-программы**Версия*\ - так что полный путь к файлу может выглядеть так (к примеру): C:\Users\Александр \AppData\Roaming\Embarcadero\BDS\8.0\Default Layout.dst (это файл рабочего стола Delphi с именем "Default Layout").

В самой подпапке структура не фиксируется и остаётся на ваше усмотрение.

Кроме Program Files и Application Data для типичной программы играют важную роль ещё две папки. Первая из них - папка TEMP. Это папка для временных файлов. Её расположение, опять же, не фиксируется, а получается специальной функцией. В эту папку следует помещать временные файлы - т.е. файлы, которые вы создаёте во время работы для временных служебных нужд и удаляете до выхода из программы. Традиционно, для именования файлов в папках следует применять уникальные имена файлов. Как их получить, я покажу ниже. В общем, если в вашей программе попадаются файлы из категории "я не замечу, если это кто-то удалит" - это хороший кандидат на помещение файла в папку TEMP. Предупрежу только, что вам действительно должно быть наплевать, что файл вдруг исчезнет. К примеру, если ваша программа ведёт кэш мини-изображений, то если файл удалят, программа его просто пересоздаст. Но это не называется "наплевать": ведь вы теряете кучу выполненной работы по созданию предспросмотров. Так что такую БД лучше хранить в Application Data. Но если вы создаёте её только на время запуска программы и удаляете при выходе - вот это точно пациент папки TEMP.

Многие новички задают вопросы "как сделать X, не выгружая данные в файл?/не создавая файл?". Хотя иногда решение может быть и найдено, но чаще всего правильный ответ на подобные вопросы - создайте временный файл в папке TEMP (кстати, см. также).

Вторая папка - это "Мои документы" (My Documents). Не редкая ошибка - помещать в эту папку данные программ. К примеру, многие игры любят размещать в этой папке сейвы. Это - большая ошибка. Самое важное отличие между папками Мои документы и Application Data в том, что в Моих документах хранят файлы пользователи, а в Application Data хранят данные программы.

Другими словами, если вы разместите что-то в Моих документах, то вам следует ожидать, что пользователь переименует это, переместит, удалит, отправит по почте друзьям, изменит - т.е. сделает что угодно из того, что обычно пользователи делают со своими файлами. И вас это не должно волновать. Такие изменения не должны затронуть функциональность вашей программы. Вот поэтому, в Мои документы должны идти вещи, которые пользователи распознают как "их вещи". Документы, которые они создали, музыка, которую они скачали - подобного рода вещи.

С другой стороны, если вы размещаете что-то в Application Data, то пользователю не следует это трогать. Это то место, где ваша программа может хранить свои данные, которые не должны меняться пользователем, но которые всё ещё будут ассоциированы с ним. Настройки программ, предпочтения пользователя, исключения в проверке грамматики...

Простое правило, чтобы определить, что файл нужно поместить в Мои документы - его выбрал пользователь диалогом сохранения. Т.е. ваша программа вообще не должна ссылаться на Мои документы - она должна писать туда файлы только если это указал ей пользователь, только если такой путь к файлу пришёл извне, но не по собственному желанию. Единственный случай, когда вашей программе может потребоваться получение пути к папке Мои документы, Мои рисунки и т.п. - указать этот путь по умолчанию в диалоге сохранения/открытия файлов.

В системе есть ещё много специальных папок, которые могут быть интересны вашей программе в специальных случаях, но для типичной программы достаточно вышеуказанных. Хочу лишь отметить только ещё одну папку, которая может быть интересна создателям игр: папка для сейвов (сохранённых игр и профилей игры). Как её получить, я тоже покажу в примерах ниже.

Обращение к данным

Заканчивает эту статью практический раздел - как работать с размещениями файлов. В Delphi и операционной системе есть несколько функций по работе с файлами и путями к ним, которые будут вам полезны. К их краткому обзору я и приступаю.

Пути файлов

Начать здесь хочется с такого момента: относительные пути к файлам. Здесь я снова сделаю голословное утверждение, а вам следует мне поверить на слово: относительные пути **не предназначены для использования их в программах**. Они предназначены для пользователя: чтобы пользователь мог легко указывать размещение файла. Но единственное, что может сделать программа с относительным именем файла - конвертировать его в абсолютное (тут "может" употреблено в смысле "следует").

Иными словами, когда вы обращаетесь к файлу - никаких 'MyFile.txt'! Имя файла должно указываться полностью, начиная с диска (ну или имени сервера для UNC-путей). Тут сразу же возникает вопрос: а откуда его, это имя, взять?

Вот нужные вам функции:

- `ParamStr(0)`, `GetModuleName(0)` или `Application.ExeName` вернут вам имя текущей программы. Т.е. путь вида 'C:\Program Files\MyProduct\MyApp.exe'.
- Если же вы пишете DLL и вам нужен путь к вашей DLL, а не к .exe файлу, который вас использует, то вам нужно использовать `GetModuleName(HInstance)` - это возвратит путь вида 'C:\Program Files\MyProduct\MyComponent.dll'.
- Далее, вам пригодятся функции модификации файловых путей. К примеру:
  - Извлечение каталога (с ведомым разделителем): ExtractFilePath('C:\Path\Folder\File.txt') = 'C:\Path\Folder\'
  - Извлечение каталога (без ведомого разделителя): ExtractFileDir('C:\Path\Folder\File.txt') = 'C:\Path\Folder'
  - Вообще, это общий принцип именования функций в Delphi: если в имени функции используется "Dir", то функция работает без ведомого разделителя. Если "Path" - то функция подразумевает наличие ведомого разделителя. Одно в другое можно превратить ExcludeTrailingPathDelimiter('C:\Path\Folder\') = 'C:\Path\Folder' и IncludeTrailingPathDelimiter('C:\Path\Folder') = 'C:\Path\Folder\'
  - Извлечение имени файла из пути: ExtractFileName('C:\Path\Folder\MyFile.txt') = 'MyFile.txt'
  - Смена расширения файла: ChangeFileExt('C:\MyFile.exe', '.chm') = 'C:\MyFile.chm' и ChangeFileExt('C:\MyFile.exe', '') = 'C:\MyFile'
  - Смена пути файла (удобно для копирования файлов): ChangeFilePath('C:\Path\Folder\MyFile.txt', 'D:\MyFolder\') = 'D:\MyFolder\MyFile.txt'
  - Есть ещё несколько функций по работе с файлами, но они вам пригодятся реже. Список их можно увидеть тут. Чаще всего вы будете пользоваться комбинацией вышеуказанных функций. К примеру: `HelpFile := ChangeFileExt(ParamStr(0), '.chm');` или `HelpFile := ExtractFilePath(ParamStr(0)) + 'Help.chm';`

Помимо функций манипуляции именами, вам пригодятся ещё и такие функции:

- Сравнение файлов: SameFileName('C:\myfile.txt', 'C:\MyFile.txt') = True
- Создание дерева папок: ForceDirectories(AppData + 'MyCompany\MyProduct\MyVersion');
- Если вам повезло использовать новые версии Delphi, то вы можете воспользоваться новыми

функциями (вот тут есть их описание в трёх частях). Если же у вас старая версия Delphi - рекомендую использовать модуль JclFileUtils из JCL. В частности, отмечу такие функции:

- Дополнение относительного пути до абсолютного: TPath.GetFullPath('MyFolder\MyFile.txt') = 'C:\Path\MyFolder\MyFile.txt' (аналог в JCL: PathIsAbsolute для определения абсолютности пути и добавление GetCurrentDir в начало относительных путей)
- Генерация случайного имени: TPath.GetRandomFileName (аналог в JCL: нет, можно использовать генератор случайных чисел)
- Генерация уникального имени: TPath.GetGUIDFileName (аналог в JCL: нет, можно создать GUID и сконвертировать его в строку)
- Генерация имени временного файла: TPath.GetTempFileName (аналог в JCL: нет, можно использовать GetTempFileName)

- Ну а полный список функций можно увидеть тут.

Одна из возможных задач - фильтрация ввода пользователя для ввода допустимого имени файла. Этот вопрос в деталях обсуждается тут.

Системные папки

В заключение остаётся показать, как можно получить пути к специальным папкам:

```
uses
  ActiveX, ShlObj;

function GetSpecialFolderLocation(const Folder: Integer; const FolderNew: TGUID): Str
const
  KF_FLAG_DONT_VERIFY         = $00004000;
var
  FolderPath: PWideChar;
  SHGetFolderPath: function(hwnd: HWND; csidl: Integer; hToken: THandle; dwFlags: DWO
  SHGetKnownFolderPath: function(const rfid: TIID; dwFlags: DWORD; hToken: THandle; v
begin
  Result := '';

  if not CompareMem(@FolderNew, @GUID_NULL, SizeOf(TGUID)) then
  begin
    SHGetKnownFolderPath := GetProcAddress(GetModuleHandle('Shell32.dll'), 'SHGetKnow
    if Assigned(SHGetKnownFolderPath) then
    begin
      FolderPath := nil;
      SetLastError(Cardinal(SHGetKnownFolderPath(FolderNew, KF_FLAG_DONT_VERIFY, 0, F
      if Succeeded(HRESULT(GetLastError)) then
      begin
        Result := FolderPath;
        CoTaskMemFree(FolderPath);
      end;
    end;
  end;

  if (Result = '') and (Folder >= 0) then
  begin
    SHGetFolderPath := GetProcAddress(GetModuleHandle('Shell32.dll'), 'SHGetFolderPat
    if Assigned(SHGetFolderPath) then
    begin
      FolderPath := AllocMem((MAX_PATH + 1) * SizeOf(WideChar));
      SetLastError(Cardinal(SHGetFolderPath(0, Folder, 0, 0, FolderPath)));
      if Succeeded(HRESULT(GetLastError)) then
        Result := FolderPath;
      FreeMem(FolderPath);
    end;
  end;

  if Result <> '' then
    Result := IncludeTrailingPathDelimiter(Result);
end;
```

Функция GetSpecialFolderLocation позволяет получить специальную папку по CSIDL-значению (Windows XP и ранее) или FOLDERID (Windows Vista и новее). Если в вашей версии Delphi отсутствуют какие-то идентификаторы, то вот полный список:

```
const
  CSIDL_DESKTOP                 = $0000;
  CSIDL_INTERNET                = $0001;
  CSIDL_PROGRAMS                = $0002;
  CSIDL_CONTROLS                = $0003;
  CSIDL_PRINTERS                = $0004;
  CSIDL_PERSONAL                = $0005;
  CSIDL_FAVORITES               = $0006;
  CSIDL_STARTUP                 = $0007;
  CSIDL_RECENT                  = $0008;
  CSIDL_SENDTO                  = $0009;
  CSIDL_BITBUCKET               = $000a;
  CSIDL_STARTMENU               = $000b;
  CSIDL_MYDOCUMENTS             = CSIDL_PERSONAL;
  CSIDL_MYMUSIC                 = $000d;
  CSIDL_MYVIDEO                 = $000e;
  CSIDL_DESKTOPDIRECTORY        = $0010;
  CSIDL_DRIVES                  = $0011;
  CSIDL_NETWORK                 = $0012;
  CSIDL_NETHOOD                 = $0013;
  CSIDL_FONTS                   = $0014;
  CSIDL_TEMPLATES               = $0015;
  CSIDL_COMMON_STARTMENU        = $0016;
  CSIDL_COMMON_PROGRAMS         = $0017;
  CSIDL_COMMON_STARTUP          = $0018;
  CSIDL_COMMON_DESKTOPDIRECTORY = $0019;
  CSIDL_APPDATA                 = $001a;
  CSIDL_PRINTHOOD               = $001b;
  CSIDL_LOCAL_APPDATA           = $001c;
  CSIDL_ALTSTARTUP              = $001d;
  CSIDL_COMMON_ALTSTARTUP       = $001e;
  CSIDL_COMMON_FAVORITES        = $001f;
  CSIDL_INTERNET_CACHE          = $0020;
  CSIDL_COOKIES                 = $0021;
  CSIDL_HISTORY                 = $0022;
  CSIDL_COMMON_APPDATA          = $0023;
  CSIDL_WINDOWS                 = $0024;
  CSIDL_SYSTEM                  = $0025;
  CSIDL_PROGRAM_FILES           = $0026;
  CSIDL_MYPICTURES              = $0027;
  CSIDL_PROFILE                 = $0028;
  CSIDL_SYSTEMX86               = $0029;
  CSIDL_PROGRAM_FILESX86        = $002a;
  CSIDL_PROGRAM_FILES_COMMON    = $002b;
  CSIDL_PROGRAM_FILES_COMMONX86 = $002c;
  CSIDL_COMMON_TEMPLATES        = $002d;
  CSIDL_COMMON_DOCUMENTS        = $002e;
  CSIDL_COMMON_ADMINTOOLS       = $002f;
  CSIDL_ADMINTOOLS              = $0030;
  CSIDL_CONNECTIONS             = $0031;
  CSIDL_COMMON_MUSIC            = $0035;
  CSIDL_COMMON_PICTURES         = $0036;
  CSIDL_COMMON_VIDEO            = $0037;
  CSIDL_RESOURCES               = $0038;
  CSIDL_RESOURCES_LOCALIZED     = $0039;
  CSIDL_COMMON_OEM_LINKS        = $003a;
  CSIDL_CDBURN_AREA             = $003b;
  CSIDL_COMPUTERSNEARME         = $003d;
  CSIDL_PROFILES                = $003e;

  FOLDERID_AddNewPrograms: TGUID                = '{DE61D971-5EBC-4F02-A3A9-6C82895E
  FOLDERID_AdminTools: TGUID                    = '{724EF170-A42D-4FEF-9F26-B60E846F
  FOLDERID_AppUpdates: TGUID                    = '{A305CE99-F527-492B-8B1A-7E76FA98
  FOLDERID_CDBurning: TGUID                     = '{9E52AB10-F80D-49DF-ACB8-4330F568
  FOLDERID_ChangeRemovePrograms: TGUID          = '{DF7266AC-9274-4867-8D55-3BD661DE
  FOLDERID_CommonAdminTools: TGUID              = '{D0384E7D-BAC3-4797-8F14-CBA229B3
  FOLDERID_CommonOEMLinks: TGUID                = '{C1BAE2D0-10DF-4334-BEDD-7AA20B22
```

```
68    FOLDERID_CommonPrograms: TGUID                 = '{0139D44E-6AFE-49F2-8690-3DAFCAE6
69    FOLDERID_CommonStartMenu: TGUID                = '{A4115719-D62E-491D-AA7C-E74B8BE3
70    FOLDERID_CommonStartup: TGUID                  = '{82A5EA35-D9CD-47C5-9629-E15D2F71
71    FOLDERID_CommonTemplates: TGUID                = '{B94237E7-57AC-4347-9151-B08C6C32
72    FOLDERID_ComputerFolder: TGUID                 = '{0AC0837C-BBF8-452A-850D-79D08E66
73    FOLDERID_ConflictFolder: TGUID                 = '{4BFEFB45-347D-4006-A5BE-AC0CB056
74    FOLDERID_ConnectionsFolder: TGUID              = '{6F0CD92B-2E97-45D1-88FF-B0D186B8
75    FOLDERID_Contacts: TGUID                       = '{56784854-C6CB-462B-8169-88E350AC
76    FOLDERID_ControlPanelFolder: TGUID             = '{82A74AEB-AEB4-465C-A014-D097EE34
77    FOLDERID_Cookies: TGUID                        = '{2B0F765D-C0E9-4171-908E-08A611B8
78    FOLDERID_Desktop: TGUID                        = '{B4BFCC3A-DB2C-424C-B029-7FE99A87
79    FOLDERID_DeviceMetadataStore: TGUID            = '{5CE4A5E9-E4EB-479D-B89F-130C0288
80    FOLDERID_Documents: TGUID                      = '{FDD39AD0-238F-46AF-ADB4-6C854803
81    FOLDERID_DocumentsLibrary: TGUID               = '{7B0DB17D-9CD2-4A93-9733-46CC8902
82    FOLDERID_Downloads: TGUID                      = '{374DE290-123F-4565-9164-39C4925E
83    FOLDERID_Favorites: TGUID                      = '{1777F761-68AD-4D8A-87BD-30B759FA
84    FOLDERID_Fonts: TGUID                          = '{FD228CB7-AE11-4AE3-864C-16F3910A
85    FOLDERID_Games: TGUID                          = '{CAC52C1A-B53D-4EDC-92D7-6B2E8AC1
86    FOLDERID_GameTasks: TGUID                      = '{054FAE61-4DD8-4787-80B6-090220C4
87    FOLDERID_History: TGUID                        = '{D9DC8A3B-B784-432E-A781-5A1130A7
88    FOLDERID_HomeGroup: TGUID                      = '{52528A6B-B9E3-4ADD-B60D-588C2DBA
89    FOLDERID_ImplicitAppShortcuts: TGUID           = '{BCB5256F-79F6-4CEE-B725-DC34E402
90    FOLDERID_InternetCache: TGUID                  = '{352481E8-33BE-4251-BA85-6007CAED
91    FOLDERID_InternetFolder: TGUID                 = '{4D9F7874-4E0C-4904-967B-40B0D20C
92    FOLDERID_Libraries: TGUID                      = '{1B3EA5DC-B587-4786-B4EF-BD1DC332
93    FOLDERID_Links: TGUID                          = '{BFB9D5E0-C6A9-404C-B2B2-AE6DB6AF
94    FOLDERID_LocalAppData: TGUID                   = '{F1B32785-6FBA-4FCF-9D55-7B8E7F15
95    FOLDERID_LocalAppDataLow: TGUID                = '{A520A1A4-1780-4FF6-BD18-167343C5
96    FOLDERID_LocalizedResourcesDir: TGUID          = '{2A00375E-224C-49DE-B8D1-440DF7EF
97    FOLDERID_Music: TGUID                          = '{4BD8D571-6D19-48D3-BE97-42222008
98    FOLDERID_MusicLibrary: TGUID                   = '{2112AB0A-C86A-4FFE-A368-0DE96E47
99    FOLDERID_NetHood: TGUID                        = '{C5ABBF53-E17F-4121-8900-86626FC2
100   FOLDERID_NetworkFolder: TGUID                  = '{D20BEEC4-5CA8-4905-AE3B-BF251EA0
101   FOLDERID_OriginalImages: TGUID                 = '{2C36C0AA-5812-4B87-BFD0-4CD0DFB1
102   FOLDERID_PhotoAlbums: TGUID                    = '{69D2CF90-FC33-4FB7-9A0C-EBB0F0FC
103   FOLDERID_Pictures: TGUID                       = '{33E28130-4E1E-4676-835A-98395C3B
104   FOLDERID_PicturesLibrary: TGUID                = '{A990AE9F-A03B-4E80-94BC-9912D750
105   FOLDERID_Playlists: TGUID                      = '{DE92C1C7-837F-4F69-A3BB-86E63120
106   FOLDERID_PrintersFolder: TGUID                 = '{76FC4E2D-D6AD-4519-A663-37BD5606
107   FOLDERID_PrintHood: TGUID                      = '{9274BD8D-CFD1-41C3-B35E-B13F55A7
108   FOLDERID_Profile: TGUID                        = '{5E6C858F-0E22-4760-9AFE-EA3317B6
109   FOLDERID_ProgramData: TGUID                    = '{62AB5D82-FDC1-4DC3-A9DD-070D1D49
110   FOLDERID_ProgramFiles: TGUID                   = '{905E63B6-C1BF-494E-B29C-65B732D3
111   FOLDERID_ProgramFilesCommon: TGUID             = '{F7F1ED05-9F6D-47A2-AAAE-29D317C6
112   FOLDERID_ProgramFilesCommonX64: TGUID          = '{6365D5A7-0F0D-45E5-87F6-0DA56B6A
113   FOLDERID_ProgramFilesCommonX86: TGUID          = '{DE974D24-D9C6-4D3E-BF91-F4455120
114   FOLDERID_ProgramFilesX64: TGUID                = '{6D809377-6AF0-444B-8957-A3773F02
115   FOLDERID_ProgramFilesX86: TGUID                = '{7C5A40EF-A0FB-4BFC-874A-C0F2E0B9
116   FOLDERID_Programs: TGUID                       = '{A77F5D77-2E2B-44C3-A6A2-ABA60105
117   FOLDERID_Public: TGUID                         = '{DFDF76A2-C82A-4D63-906A-5644AC45
118   FOLDERID_PublicDesktop: TGUID                  = '{C4AA340D-F20F-4863-AFEF-F87EF2E6
119   FOLDERID_PublicDocuments: TGUID                = '{ED4824AF-DCE4-45A8-81E2-FC796508
120   FOLDERID_PublicDownloads: TGUID                = '{3D644C9B-1FB8-4F30-9B45-F670235F
121   FOLDERID_PublicGameTasks: TGUID                = '{DEBF2536-E1A8-4C59-B6A2-41458647
122   FOLDERID_PublicLibraries: TGUID                = '{48daf80b-e6cf-4f4e-b800-0e69d84e
123   FOLDERID_PublicMusic: TGUID                    = '{3214FAB5-9757-4298-BB61-92A9DEAA
124   FOLDERID_PublicPictures: TGUID                 = '{B6EBFB86-6907-413C-9AF7-4FC2ABF0
125   FOLDERID_PublicRingtones: TGUID                = '{E555AB60-153B-4D17-9F04-A5FE99FC
126   FOLDERID_PublicVideos: TGUID                   = '{2400183A-6185-49FB-A2D8-4A392A60
127   FOLDERID_QuickLaunch: TGUID                    = '{52A4F021-7B75-48A9-9F6B-4B87A210
128   FOLDERID_Recent: TGUID                         = '{AE50C081-EBD2-438A-8655-8A092E34
129   FOLDERID_RecordedTVLibrary: TGUID              = '{1A6FDBA2-F42D-4358-A798-B74D7459
130   FOLDERID_RecycleBinFolder: TGUID               = '{B7534046-3ECB-4C18-BE4E-64CD4CB7
131   FOLDERID_ResourceDir: TGUID                    = '{8AD10C31-2ADB-4296-A8F7-E4701232
132   FOLDERID_Ringtones: TGUID                      = '{C870044B-F49E-4126-A9C3-B52A1FF4
133   FOLDERID_RoamingAppData: TGUID                 = '{3EB685DB-65F9-4CF6-A03A-E3EF6572
134   FOLDERID_SampleMusic: TGUID                    = '{B250C668-F57D-4EE1-A63C-290EE7D1
135   FOLDERID_SamplePictures: TGUID                 = '{C4900540-2379-4C75-844B-64E6FAF8
136   FOLDERID_SamplePlaylists: TGUID                = '{15CA69B3-30EE-49C1-ACE1-6B5EC372
137   FOLDERID_SampleVideos: TGUID                   = '{859EAD94-2E85-48AD-A71A-0969CB56
138   FOLDERID_SavedGames: TGUID                     = '{4C5C32FF-BB9D-43B0-B5B4-2D72E54E
139   FOLDERID_SavedSearches: TGUID                  = '{7D1D3A04-DEBB-4115-95CF-2F29DA29
```

```
  FOLDERID_SEARCH_CSC: TGUID                    = '{EE32E446-31CA-4ABA-814F-A5EBD2FD
  FOLDERID_SEARCH_MAPI: TGUID                   = '{98EC0E18-2098-4D44-8644-66979315
  FOLDERID_SearchHome: TGUID                    = '{190337D1-B8CA-4121-A639-6D472D16
  FOLDERID_SendTo: TGUID                        = '{8983036C-27C0-404B-8F08-102D10DC
  FOLDERID_SidebarDefaultParts: TGUID           = '{7B396E54-9EC5-4300-BE0A-2482EBAE
  FOLDERID_SidebarParts: TGUID                  = '{A75D362E-50FC-4FB7-AC2C-A8BEAA31
  FOLDERID_StartMenu: TGUID                     = '{625B53C3-AB48-4EC1-BA1F-A1EF4146
  FOLDERID_Startup: TGUID                       = '{B97D20BB-F46A-4C97-BA10-5E360843
  FOLDERID_SyncManagerFolder: TGUID             = '{43668BF8-C14E-49B2-97C9-747784D7
  FOLDERID_SyncResultsFolder: TGUID             = '{289A9A43-BE44-4057-A41B-587A76D7
  FOLDERID_SyncSetupFolder: TGUID               = '{0F214138-B1D3-4A90-BBA9-27CBC0C5
  FOLDERID_System: TGUID                        = '{1AC14E77-02E7-4E5D-B744-2EB1AE51
  FOLDERID_SystemX86: TGUID                     = '{D65231B0-B2F1-4857-A4CE-A8E7C6EA
  FOLDERID_Templates: TGUID                     = '{A63293E8-664E-48DB-A079-DF759E05
  FOLDERID_UserPinned: TGUID                    = '{9E3995AB-1F9C-4F13-B827-48B24B6C
  FOLDERID_UserProfiles: TGUID                  = '{0762D272-C50A-4BB0-A382-697DCD72
  FOLDERID_UserProgramFiles: TGUID              = '{5CD7AEE2-2219-4A67-B85D-6C9CE156
  FOLDERID_UserProgramFilesCommon: TGUID        = '{BCBD3057-CA5C-4622-B42D-BC56DB0A
  FOLDERID_UsersFiles: TGUID                    = '{F3CE0F7C-4901-4ACC-8648-D5D44B04
  FOLDERID_UsersLibraries: TGUID                = '{A302545D-DEFF-464B-ABE8-61C8648D
  FOLDERID_Videos: TGUID                        = '{18989B1D-99B5-455B-841C-AB7C74E4
  FOLDERID_VideosLibrary: TGUID                 = '{491E922F-5643-4AF4-A7EB-4E7A138D
  FOLDERID_Windows: TGUID                       = '{F38BF404-1D43-42F2-9305-67DE0B28
```

Примечание: не все специальные папки являются физическими (на диске). Соответственно, не ко всем папкам можно получить путь на диске. К примеру, Мои документы на диске найти можно, но Мой компьютер - нет.

Пояснения по CSIDL можно узнать здесь. Пояснения по FOLDERID можно узнать здесь.

Функция GetSpecialFolderLocation вернёт имя запрошенной папки с ведомым разделителем пути или пустую строку при ошибке (при этом ошибку можно получить через GetLastError/RaiseLastOSError).

Примеры использования (без обработки ошибок):

```
AppData       := GetSpecialFolderLocation(CSIDL_APPDATA,          FOLDERID_RoamingAppl
LocalAppData  := GetSpecialFolderLocation(CSIDL_LOCAL_APPDATA,    FOLDERID_LocalAppDa
CommonAppData := GetSpecialFolderLocation(CSIDL_COMMON_APPDATA,   FOLDERID_ProgramDat
Autorun       := GetSpecialFolderLocation(CSIDL_STARTUP,          FOLDERID_Startup);
ProgramFiles  := GetSpecialFolderLocation(CSIDL_PROGRAM_FILES,    FOLDERID_ProgramFil
MyDocuments   := GetSpecialFolderLocation(CSIDL_PERSONAL,         FOLDERID_Documents)
Desktop       := GetSpecialFolderLocation(CSIDL_DESKTOPDIRECTORY, FOLDERID_Desktop);
```

Заметьте, что не все значения доступны в любых версиях Windows. Возможно, вам нужно будет делать проверку версии Windows, например:

```
SavedGames := GetSpecialFolderLocation(-1, FOLDERID_SavedGames); // папка для сейвов
SavedGames := GetSpecialFolderLocation(CSIDL_APPDATA, FOLDERID_SavedGames) + 'Saved G
// Улучшенный вариант:
if CheckWin32Version(6, 0) then
  SavedGames := GetSpecialFolderLocation(-1, FOLDERID_SavedGames) // Vista +
else
  SavedGames := GetSpecialFolderLocation(CSIDL_APPDATA, FOLDERID_RoamingAppData) + 'S
```

Для уточнения деталей по наличию/доступности конкретной папки в версиях Windows - см. документацию по ссылкам выше (ссылки на CSIDL и FOLDERID).

Примечание: в редких случаях запрашиваемая папка может не существовать. К примеру, если её до вас никто не запрашивал. Поэтому после получения пути вам надо её создать - вызовом ForceDirectory(Folder).

Дополнительно

Дополнительные ссылки для чтения:

- Длинная и печальная история ключа Shell Folders
- В чём разница между папками Мои документы и Application Data?
- Остановите это безумие: подпапки в Моих документах
- Почему папка "Мои рисунки" возвращается, после того, как я её удалил?
- Несколько слов о UAC в Vista
- Несколько слов о виртуализации в Vista

Тэги Delphi, начинающим, Статья

11 комментариев:

Аноним 16 сентября 2011 г., 20:52

что-то не очень хорошо работает плагин "раскраски" с исходниками длиннее 100 строк

по 4 вариант AppData - что это все-таки?))

Ответить

GunSmoker 17 сентября 2011 г., 4:18

>>> *что-то не очень хорошо работает плагин "раскраски" с исходниками длиннее 100 строк*

Какой браузер?

>>> *По 4 вариант AppData - что это все-таки?))*

Это FOLDERID_LocalAppDataLow. Что это такое и зачем нужно - я оставлю в качестве домашнего упражнения ;)

Ответить

Анонимный 28 сентября 2011 г., 23:08

FOLDERID_LocalAppDataLow = %USERPROFILE%\AppData\LocalLow, используется для хранения настроек, применяемых в защищенном режиме работы приложения, когда Windows при запуске процесса помещает его в защищенное адресное пространство озу, не позволяя каким-либо сторонним процессам вмешиваться в это адресное пространство с какими то ни было целями, исключения только для процессов уровня ядра системы, например драйверов. ИМХО, как-то так.

Ответить

Анонимный 28 сентября 2011 г., 23:15

Странно, содержание данной статьи прописная азбука, встречается во множестве литературы по Delphi/BSD/RAD Studio, правда, большей частью в урезанном виде, в статье все это обобщено и разжевано (чего в книгах встречается редко), за что автору респект и прочие блага сей жизни, но именно на этой азбуке я встречал большую часть ошибок всего рассмотренного мною чьего-либо кода, иногда даже своего (здесь респект рефакторингу).

Ответить

Виталий 15 октября 2011 г., 12:05

Шикарная статья, спасибо! Многие вещи знал, а некоторых догадывался, но когда все в одном месте, четко и с

примерами - просто здорово!

Ответить

Анонимный 19 октября 2011 г., 20:51

Сейчас же вам важно понять, что (изменяемые) данные в папке с программой - это табу в современном мире.
Portable-программы не в счет уже? Они на то и портабл, чтоб можно было с собой таскать, вместе с настройками. Тут, по-моему, сохранение настроек "рядом" с программой - самый верный вариант.

Ответить

GunSmoker 20 октября 2011 г., 3:36

Разумеется.

Об этом я тоже говорю - "не всегда это возможно (пример: мобильные программы). Понятно, что и из сказанного выше могут быть (и бывают) исключения."

Или вы хотите, чтобы я эту фразу к каждому предложению приписывал? :)

Ответить

Fr0sT 28 декабря 2011 г., 11:18

Об этом я тоже говорю - "не всегда это возможно (пример: мобильные программы). Понятно, что и из сказанного выше могут быть (и бывают) исключения."
Просто начитавшись столь категоричных заявлений (*хранение данных в папке с самой программой. Это - самая большая ошибка новичков, (изменяемые) данные в папке с программой - это табу в современном мире*), народ примет это как аксиому. А мелкие утилитки, которые гадят в реестр и в app data - вот самая большая ошибка, причем не только новичков, а вообще огромной кучи разрабов, больных реестризмом. Если система полетела, то папку с программами легко скопировать. App data с грехом пополам тоже можно достать (если из-под аварийной среды сумеешь разрулить с правами на ФС). А вот с реестром при этом хлебнешь геморроя так, что мало не покажется.
А перенести свои настройки с одного компа на другой? Лишь малая часть софта имеет средства экспорта/импорта настроек, остальные предлагают юзеру самому ковыряться в проблеме. И в этом случае перенос файлов (даже засунутых в app data) куда лучше, чем лазание в реестре.
Еще один тонкий момент: если нужно иметь две копии программы с разными настройками? Лишь единицы программ дают возможность работать с профилями. Тут и реестр, и app data дадут несчастному юзеру полный отлуп. И что делать? Заводить отдельный аккаунт, ставить виртуалку? Фиг, юзер пойдет и скачает портабилированную версию, с которой нет никаких проблем.
Так что здесь мой манифест таков:
1) Хранение исключительно в файлах (в app data, так уж и быть)
2) Встроенное средство сделать программу portable
и/либо, если так уж охота засрать юзеру реестр, - то **в обязательном порядке**
3) Встроенные средства экспорта/импорта настроек.

относительные пути не предназначены для использования их в программах. Они предназначены для пользователя: чтобы пользователь мог легко указывать размещение файла. Но единственное, что может сделать программа с относительным именем файла - конвертировать его в абсолютное (тут "может" употреблено в смысле "следует").
Угу. К примеру, при запуске таким образом
C:\>d:\test\testproject\project.exe
GetCurrentDir будет возвращать C:\, соответственно, относительные пути будут отсчитываться от неё. Также при использовании ярлыков свойство "текущая директория" может быть написана неверно или затёрта, в итоге значение CurrentDir также непредсказуемо.

Ответить

Александр Алексеев 28 декабря 2011 г., 11:22

>>> *Просто начитавшись столь категоричных заявлений, народ примет это как аксиому.*

А это и есть аксиома программирования под Windows.

Софт, хранящий конфиги в C:\Program Files\ и C:\Windows - это, по-твоему, нормально?

Ответить

Fr0sT 28 декабря 2011 г., 15:01

Софт, хранящий конфиги в C:\Program Files\ и C:\Windows - это, по-твоему, нормально?
А что, кроме как в Program Files, софтину нельзя никуда поставить (вспоминаем дурным словом MSVS)? Я, к примеру, имею для этого отдельную папку C:\Soft, в которой, по крайней мере, нет пробелов, - ну и дальше разделение по назначению программы. Иерархия по производителю нахрен никому не нужна, кроме самого производителя (чтобы юзеры не забывали, чей софт юзают).
Но когда открываешь Program files, видишь кучу странных названий и начинаешь чесать репу "Я этого не ставил!" - вот это как раз результат подобного дурацкого расположения.

А это и есть аксиома программирования под Windows.
Я считаю, это дурная аксиома. По крайней мере, в таком категоричном ключе. Примеры неудобств с насаждаемыми мелкософтом принципами я уже привел выше, поэтому я за то, чтобы юзеру давали выбор. Пускай эта опция будет запрятана глубоко в диалог настроек, или же будет реализовываться странным образом (напр., созданием пустого сигнального файла рядом с программой), но возможность добиться портабельности **должна быть**! Потому что ИТ должны делать человека свободным, вместо этого его заставляют наживать огромный гемор, когда требуется выйти за рамки действий инсталл-запуск-анинсталл.

Ответить

Анонимный 9 августа 2012 г., 0:18

"Я считаю, это дурная аксиома. По крайней мере, в таком категоричном ключе." - не, это правильная аксиома. вас видимо не задалбывали программы написанные "програмистами" которые об правилах поведения в системе даже не слышали. поэтому пусть будет аксиома. а ведь такие "студенты" зачастую пишут софт в серьёзных организациях. вот подрастут - сами научатся понимать когда куда чего надо. а до тех пор - аксиома.

Ответить

Введите комментарий...

Подпись комментария: Аккаунт Goog ▼

Публикация **Просмотр**

Можно использовать некоторые HTML-теги, например:

Жирный
Курсив
Ссылка

Вам необязательно регистрироваться для комментирования - для этого просто выберите из списка "Анонимный" (для

анонимного комментария) или "Имя/URL" (для указания вашего имени и (опционально) ссылки на сайт). Все прочие варианты потребуют от вас входа в вашу учётку (поддерживается OpenID).

Пожалуйста, по возможности **используйте "Имя/URL" вместо "Анонимный"**. URL можно просто не указывать.

Ваше сообщение может быть помечено как спам спам-фильтром - не волнуйтесь, оно появится после проверки администратором.

Ссылки

Создать ссылку

Следующее Главная страница Предыдущее

Подписаться на: Комментарии к сообщению (Atom)

Поиск по блогу

Google™ Пользовательский поиск

Поиск

Архив

Архив

Тэги

Delphi (185) Статья (72) задачки (38) ты можешь это сделать (30) начинающим (26) обработка ошибок (26) случайные мысли (26) EurekaLog (22) блог (17) прочее (16) Windows (14) не делай так (11) роботы/киберпанк (9) журнал (6) 7 (4) Tiburon (4) Vista (4) Королевство Delphi (4) TasksEx (3) x64 (2) Коты (2) кроссплатформенность (1) работа (1)

Подписка

Шаблон "Simple". Технологии Blogger.