

radio-mir

4 • 2004

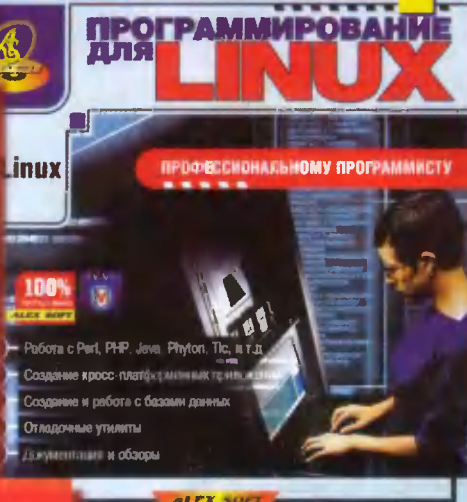
Ваш компьютер



+100 СЛОВА

PROMT X
OFFICE

БИБЛИОТЕКА
3D МОДЕЛИ ДЛЯ 3D MAX
ИНФОРМАЦИЯ
& STOCK LABS



ПРОГРАММИРОВАНИЕ
ДЛЯ LINUX

ПРОФЕССИОНАЛЬНОМУ ПРОГРАММИСТУ

100% АЛЕКС СФОРТ

Работа с Perl, PHP, Java, Python, Tcl, и т.д.
Создание кросс-платформных приложений
Создание и работа с базами данных
Отладочные утилиты
Документация и обзоры

ALEX SOFT

Uroman 2004
БОЛЬШОЕ ПОЛИЧЕСКОЕ ГОЛОВОЛОМКА
НА РУССКОМ ЯЗЫКЕ

91 000 РЕФЕРАТОВ

САМАЯ ПОЛНАЯ ПОДБОРКА

Дипломы,
курсовые,
сочинения,
рефераты
и многое
другое...



Pought us Cross
On 6 U2.0
Pba Taba De lunc
Addyned Gemoni
Gomok U1.0
3D Lings U1.2
S Sign Free Lines
AlMa tenn Lines U
Christmas Edition
Gong
Girl Builder U1.5
Mozina 32

26 РУССКАЯ + АНГЛИЙСКАЯ ВЕРСИИ

autodesk

AutoCAD 2004

26 РУССКАЯ
ЛИНЕЙКА

eXtremator
Мультизагрузочный **Графика**

ACD Systems ACDSee 5.0 PowerPack
русская версия
Adobe Illustrator 10
русская и английская версии
Adobe Photoshop 7.0
русская и английская версии
Corel Draw Graphics Suite 11
русская и английская версии
PM Painter 2.002
MACROMEDIA FIREWORKS MX 6.0
Macromedia Freehand MX 11.0
русская и английская версии



Помощь в экстремальных ситуациях

10.000
мелодий
для сотовых
телефонов
с инструкцией

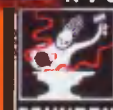


СЭМПЛЫ
ACID и SONAR

КОПИИ ФИРМЕННЫХ ДИСКОВ

Syntonic
Generator

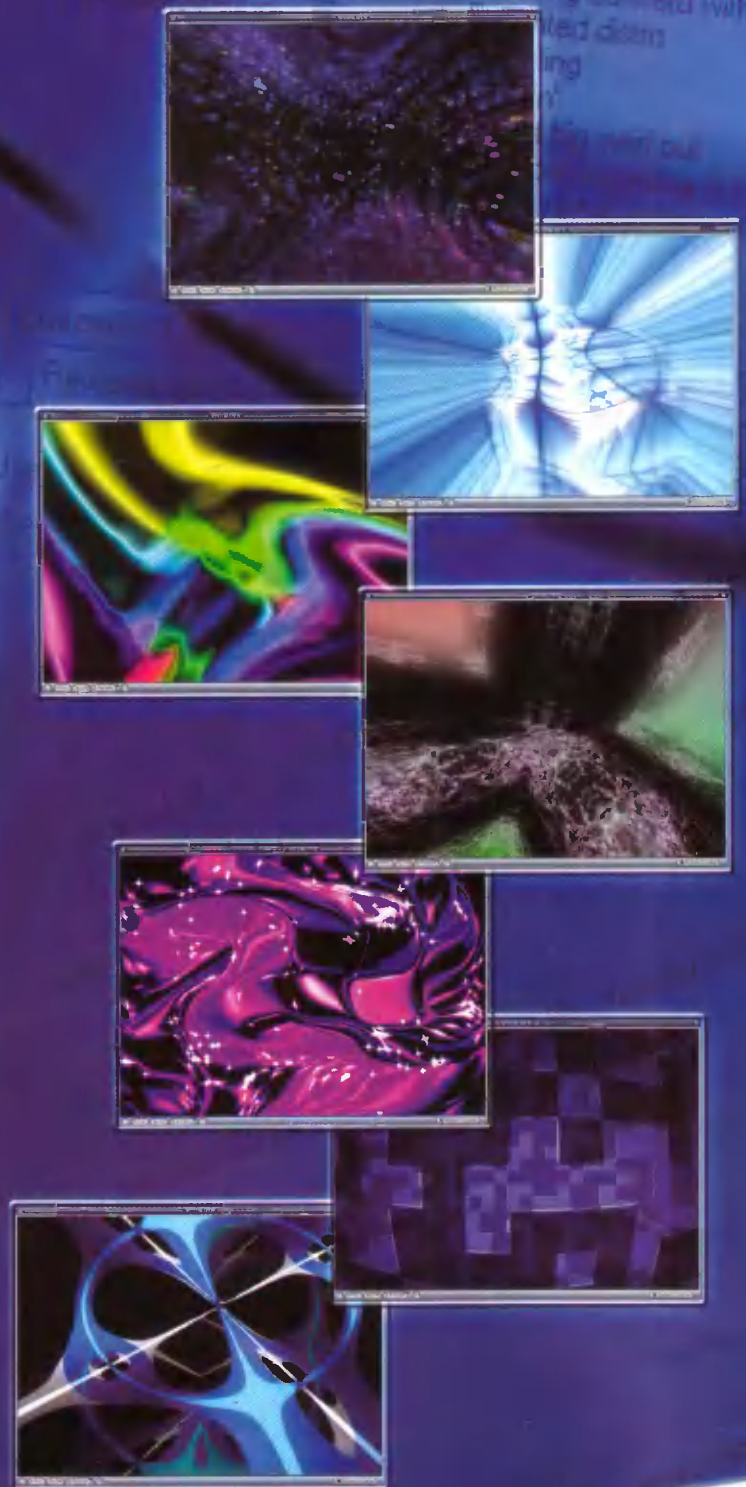
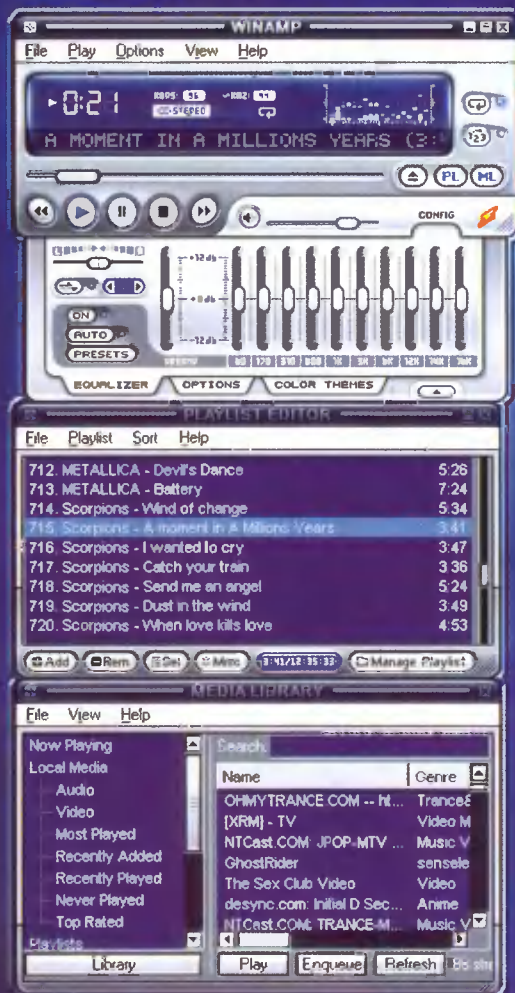
NIC



3

И.Сошин.

Визуализация в Winamp Advanced Visualization Studio



EN

zt=zt+0.05 ; sz=sin[zt]*0.8 ; xt=xt+xts+xp ; yt=yt+yts+yp ; xp=x

Учредитель и издатель:
ООО "Радиомир Пресс"
Свидетельство о регистрации
ПИ №77-9841 от 17.09.2001

Главный редактор
Ольга СТРУЖАНКОВА

Зам. гл. редактора
Иван БЕЛЬСКИЙ

Редколлегия:
Сергей ДРОЗДОВСКИЙ,
Алексей КОНДРАШОВ,
Анатолий КОРБИТ,
Владимир КУЦЕНКО,
Елена ЛЕВИТМАН,
Николай МЕДВЕДЬ,
Геннадий ПЕЧЕНЬ,
Геннадий ШУЛЬГИН

Контактные телефоны:
в Москве (095) 105-99-89,
в Минске (017) 249-41-47

Компьютерная верстка —
Янина БЕЛЬСКАЯ

Техническая графика —
Наталья БЕЛЬСКАЯ,
Александр ГОРАЮТИН,
Мария КАЛАБУХОВА

Оформление обложки —
Наталья БЕЛЬСКАЯ,
Надежда БОГОМОЛОВА

Адреса для писем:
119454, РФ, г.Москва, а/я 37,
факс (095) 432-62-04;
220095, РБ, г.Минск-95, а/я 199

E-mail: rl@radiopage.by
rm@radio-mir.com
WWW: <http://radio-mir.com>

Наши платежные реквизиты:
получатель: ООО "Радиомир Пресс",
ИНН 7729404741, КПП 772901001,
р/с 40702810900010000084
в ООО КБ "Агропромкредит"
ф-л Центральный, г.Москва,
корр. счет 30101810500000000109
БИК 044525109

Адрес банка: 125315, г.Москва,
Ленинградский пр., дом 76, корп. 4

Материалы для публикации принимаются
в рукописном, печатном и электронном
вариантах

Требования к графическим материалам
рекламного характера в электронном виде:
CorelDRAW до 10.0, все шрифты в крияхх;
bitmaps 300 dpi; TIFF 300 dpi; CMYK.
Приложить печатную копию.

За достоверность рекламной и другой
публикуемой информации несут ответ-
ственность рекламодатели и авторы.
Мнение редакции не всегда совпадает
с мнениями авторов

Адрес редакции: 119454, РФ, г.Москва,
ул. Юштыянка, 6 — 233, тел. (095) 105-99-89

Подписано к печати 24.02.2004 г.
Формат 60 x 84 1/8.
Печать офсетная. 6 печ. л.
Цена свободная.

Отпечатано в типографии
ЗАО "Красногорская типография".
Заказ № 639.

радиомир Ваш компьютер

ЕЖЕМЕСЯЧНЫЙ МАССОВЫЙ ЖУРНАЛ
С 1995 г. по 6/2001 г. издавался под названием
"Радиолобитель. Ваш компьютер"

№4/апрель/2004

ЧИТАЙТЕ В НОМЕРЕ

ВОКРУГ ПК

МУЛЬТИМЕДИА

И.СОШИН. Визуализация в Winamp Advanced Visualization Studio 2

НЕ ТОЛЬКО НОВИЧКУ

КЛУБ "1 АПРЕЛЯ" 5

Р.ГАЛЕЕВ /HI-TECH. Saga о кодировках 7

А.ГОРЯЧКИН. Операционная система Menuet OS 8

А.ГРИНЧУК. Анимация с Macromedia Flash MX 10

Анимация формы 10

В.КУЦ. "Упорядочиватели" дискового пространства,

или немного о современных дефрагментаторах 12

С.РЮМИК. Интернет-калейдоскоп, freeware #5 16

ДАЙДЖЕСТ 18

ПРОГРАММЫ И АЛГОРИТМЫ

УРОКИ ПРОГРАММИРОВАНИЯ

А.ИВАНЧИКОВ. COM — это не сон 22

А.КОРБИТ, А.ЩЕРБАКОВ. Программирование в среде Visual C++ 6.0.

Использование динамически загружаемых библиотек 25

ДИАЛОГ ПРОГРАММИСТОВ

CyberManiac /HI-TECH. Теоретические основы крэкинга 27

А.ТЕРЁШКИН/HI-TECH. Нестандартное использование

таблицы релокаций PE-файла 29

Е.ЛЕБЕДЕВ. История одной системы 32

И.ШИРКО. Microsoft Agent

Часть II 34

В.ПОНОМАРЕВ. Верификация и спецификация

в программировании 36

О.ВАЛЬПА. Настройка COM-порта 37

РЕЦЕПТЫ

Р.КАРПАЧ. Записки пользователя Win 3.11 38

МИР 8 БИТ

И.РОЩИН. Построение таблицы громкости в плеере Pro Tracker 3 43

ИГРОТЕКА

А.ВЕНДИЛОВСКИЙ. Call of Duty 46

ДОСКА ОБЪЯВЛЕНИЙ

Куплю, продам, обменяю 48

Полные листинги некоторых программ расположены по адресу
<http://radio-mir.com>



Визуализация в Winamp Advanced Visualization Studio

И. СОШИН,

г. Лысьва, Пермской области,
E-mail: s1w17@permonline.ru

Музыкальный проигрыватель WinAMP предоставляет пользователю очень большой набор функций, самой интересной из которых, на мой взгляд, является возможность визуализации звукового потока. Начиная с версии 2.6, в комплект проигрывателя входит очень мощный инструмент визуализации, называемый Winamp Advanced Visualization Studio (WVS). Этот модуль позволяет строить графическое изображение звука в различных вариантах в оконном или полноэкранном режиме. Кроме готовых визуализаций, в WVS входит редактор, позволяющий пользователю самому создать свой вариант.

Работа с WVS очень проста и может быть освоена буквально за несколько минут. Запуск WVS осуществляется из окна Visualization Plug-ins, вызываемого комбинацией клавиш CTRL-K. Здесь необходимо выбрать из списка плагинов Winamp Advanced Visualization Studio и нажать кнопку Start. В результате этого действия появятся два окна — экран визуализации WVS и окно редактора Winamp AVS Editor. Если окно редактора не появилось, его можно вызвать двойным нажатием левой кнопки мыши на экране визуализации.

Окно редактора разделено на две части (рис. 1). В левой части выводится список используемых эффектов, в правой — настройки каждого эффекта, если они есть. Внизу окна WVS отображается количество кадров в секунду, выводимых на дисплей, и разрешение экрана на визуализации. Главное меню Winamp AVS Editor состоит из двух пунктов — Preset и Setting. Первый пункт отвечает за загрузку и сохранение визуализаций, второй — за настройки плагина.

Разработка визуализации в WVS сводится к комбинации и настройке нескольких стандартных эффектов. При этом у многих из них есть настройки, которые могут задаваться в виде сложных формул, что позволяет создавать новые необычные визуализации звукового потока.

Для создания своего эффекта визуализации в WVS из меню Preset выбирается пункт New. После этого в списке эффектов остается только пункт Main, имеющий одну настройку — Clear every

frame (очищать каждый кадр). Если она используется, то каждый кадр будет рисоваться на чистой поверхности, если данная опция не активна — каждый последующий кадр будет накладываться на предыдущий.

Для добавления эффектов используется кнопка "+". Кнопка "-" удаляет выделенный эффект, кнопка "x2" создает копию выделенного эффекта. При нажатии "+" из выпадающего меню можно выбрать один из следующих пунктов:

- Presets — загрузка готовых визуализаций. Загрузив готовую визуализацию, можно посмотреть и изменить ее структуру, что значительно упрощает процесс освоения WVS;

- RenderAPE — подключаемые эффекты;

- Misc — разные вспомогательные модули;

- Trans — эффекты преобразования изображения;

- Render — эффекты, генерирующие изображение;

- Effect list — список эффектов, позволяющий объединить несколько модулей в один.

Эффекты можно свободно перемещать по списку в нужное место, так как расположение эффекта имеет важное значение.

Основу любой визуализации составляют эффекты, генерирующие изображение. Большинство из них имеют некоторые одинаковые элементы настройки. Прежде всего, это флажок включения — Enable, который очень удобно использовать при отладке ви-

зуализации. Часто используются и настройки цвета. В WVS они задаются в виде цветной полосы, при нажатии на которую появляется окно выбора цвета. Во многих эффектах присутствует управление способом вывода изображения — Blending. Здесь возможны следующие варианты:

- Replace — текущее изображение полностью перекрывает другое;

- Additive blend — цвета изображения складываются;

- Blend 50/50 — старое и новое изображение смешиваются по яркости в пропорции 50:50;

- Default render blend mode — режим, включающий способ вывода, заданный по умолчанию.

В стандартную поставку WVS включен целый ряд эффектов вывода изображения, расположенных в пункте Render.

AVI — видео. Данный модуль выводит видеопоток, записанный в формате AVI. Необходимый видеофайл должен находиться в подкаталоге PLUGIN\WVS каталога WINAMP. Движком Speed задается скорость вывода видеопотока. При выборе способа вывода Blend 50/50 + OnBeat Additive яркость видеоизображения привязывается к импульсам ударных. Движком Beat persistence задается количество импульсов ударных инструментов, через которое происходит изменение яркости;

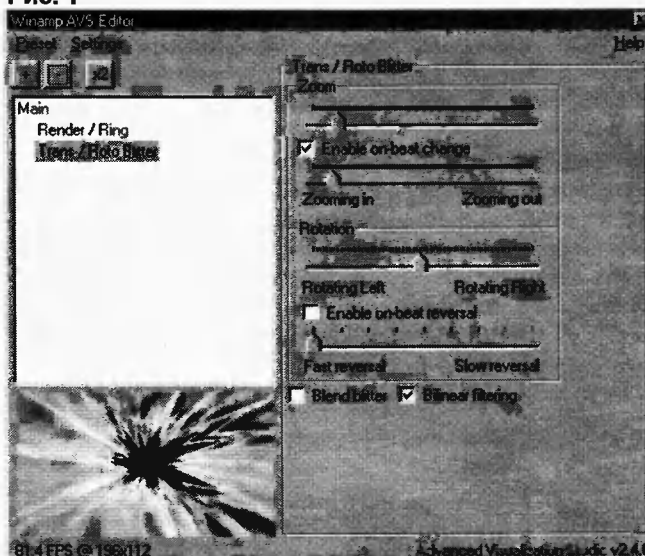
Bass Spin — две вращающиеся фигуры. Можно включить или выключить каждую фигуру и настроить их цвет. Позиции Filled triangles и Lines позволяют задать вид фигур в виде заполненных треугольников или линий.

Clear screen — очистка изображения, позволяющая задавать варианты очистки экрана и цвет фона. Данный модуль должен находиться в начале списка, так как он очищает экран от результатов действия эффектов, стоящих до него.

Dot Fountain — точечный фонтан. В пункте Rotation задается направление и скорость вращения частиц, в пункте Angle — угол зрения на фонтан. Через панель Dot Colors задается цвет частиц от самых верхних (Top) до самых нижних (Bottom).

Dot Grid — сетка из точек. На панели Colors можно задать цвет точек. Также можно установить циклическое изменение

Рис. 1





цвета точек. Для этого в поле **Cycle through** устанавливается число (больше 1), равное количеству цветов в цикле. Нажатием на соответствующую часть цветовой полосы можно задать последовательность цветов в цикле. Поле **Grid movement** задает направление и скорость движения сетки из точек. В поле **Dot spacing** задается частота точек в сети.

Dot Plane — плоскость из точек. Настройки аналогичны настройкам **Dot Fountain**.

Moving Particle — движущаяся частица. Выводит точку, движущуюся около центра экрана. На панели **Distance from center** задается расстояние, на которое точка может удаляться от центра экрана. Движок **Particle size** задает размер точки. При установке флажка **Onbeat sizechange enabled** на панели **Particle size (onbeat)** частица будет изменять свой размер в момент действия ударных. Интересные эффекты получаются при использовании нескольких копий данного модуля.

OnBeat Clear — очистка экрана, привязанная к импульсам ударных инструментов. В поле **Clear every N beats** задается количество импульсов, через которое будет происходить очистка.

Oscilloscope Star — звезда. Выводит звезду, лучи которой имеют форму осциллограммы сигнала. Движком **Star size** задаются размеры звезды. Через панель **Rotation** можно задать направление и скорость вращения лучей. Панель **Star source and position** позволяет задать привязку к каналу звука и положение звезды на экране.

Picture — выводит на экран графическое изображение, записанное в формате BMP. Настройки в целом аналогичны модулю AVI. Изображение автоматически масштабируется под размер экрана. **Keep aspect ratio** привязывает масштаб изображения к одной из осей.

Ring — кольцо. Настройки в целом аналогичны **Oscilloscope Star**. На панели **Ring source and position** можно задать привязку формы кольца к осциллограмме (**Oscilloscope**) или анализатору спектра (**Spectrum**).

Rotating Stars — вращающиеся звезды. Две звезды вращаются вокруг центра экрана визуализации. Размер звезд зависит от уровня сигнала. Из настроек доступно только циклическое изменение цвета.

Simple — графическое отображение сигнала. Выводит осциллограмму (**Oscilloscope**) или анализатор спектра (**Spectrum**) в виде линий (**Lines**), заполненных прямоугольниками (**Solid**) или точек (**Dot**). Можно изменять привязку к каналу звука и границам окна. Доступно также циклическое изменение цвета.

Starfield — звездное небо. Данный модуль позволяет создать эффект полета сквозь звезды. Можно изменять скорость движения, количество звезд, метод отображения и цвет.

SuperScope — универсальный визуализатор. Данный модуль — очень мощное средство визуализации, позволяющее получать самые разнообразные эффекты, вид которых задается математическими формулами. Для этого используются следующие предопределенные переменные:

- **n** — **numpoints** — количество точек, составляющих кривую;

- **x, y** — координаты точки на экране. Центр координат находится в центре экрана визуализации. Значения этих переменных должны находиться в диапазоне от -1 до 1;

- **i** — **pointpos** — номер позиции точки сигнала. Перебирается от 0 до 1 на каждом кадре. Число шагов перебора равно **n**;

- **v** — **value** — значение уровня сигнала в точке **i**, лежащее в диапазоне от -1 до 1;

- **b** — **beat** — ударные. В случае наличия импульса ударных, переменная принимает значение 1, в остальных случаях — 0;

- **w** — ширина экрана в пикселах;

- **h** — высота экрана в пикселах;

- **Red, Green, Blue** — яркость красного, зеленого и синего цвета точки изображения соответственно.

Кроме предопределенных переменных, можно использовать любые произвольные.

Список математических функций, используемых WVS, приведен в таблице.

Для ввода формул используются четыре поля. В поле **Init** задаются начальные значения переменных; в поле **Point** описываются уравнения, по которым вычисляются координаты точек; в поле **Frame** — уравнения изменения переменных на каждом кадре; в поле **Beat** — изменения переменных при наличии импульсов ударных. Поле **Point** в обязательном порядке должно содержать уравнения расчета координат точек изображения **x** и **y**. Также обязательно должно быть задано количество точек кривой **n**. Разделителем между формулами служит точка с запятой.

Например, строка **x=i-0.5;y=v** в поле **Point** и **n=10** в поле **Init** выводят на экран кривую сигнала, количество вершин в которой равно 10.

Данный модуль позволяет использовать данные осциллографа или анализатора спектра и устанавливать вид кривой в виде точек (**Dots**) или линий (**Lines**), а также циклическое изменение цвета.

Text — произвольный текст. Позволяет выводить на экран текстовую строку, заданную в поле **Text to Display**. Через точку с запятой можно задать несколько частей строки, которые будут выводиться последовательно. При установке параметра **Random** вывод частей строки осуществляется в случайном порядке. Модуль позволяет задать привязку текста на экране по горизонтали (**Horizontal**) и вертикали (**Vertical**), либо выводить его в случайном месте экрана (**Random pos**). Также можно задать скорость перемещения текста, его цвет и способ взаимодействия с изображением.

abs (var)	Возвращает абсолютное значение var
sin (var)	Возвращает синус угла var (величина угла задается в радианах)
cos (var)	Возвращает косинус угла var
tan (var)	Возвращает тангенс угла var
asin (var)	Возвращает арксинус var
acos (var)	Возвращает арккосинус var
atan (var)	Возвращает арктангенс var
sqr (var)	Возвращает квадрат var
sqrt (var)	Возвращает квадратный корень var
pow (var, var2)	Возвращает var, возведенную в степень var2
log (var)	Возвращает логарифм var
log10 (var)	Возвращает логарифм по основанию 10 из var
sign (var)	Возвращает знак var или 0
min (var, var2)	Возвращает минимальную из переменных
max (var, var2)	Возвращает максимальную из переменных
rand (var)	Возвращает случайное число между 0 и var
band(var, var2)	Логическое И
bor (var, var2)	Логическое ИЛИ
bnot (var)	Логическое НЕ
if(cond, vartrue, varfalse)	Условие. Если переменная cond не равна нулю, возвращается vartrue, иначе — возвращается varfalse
equal (var, var2)	Возвращает 1, если var = var2, иначе — 0
above (var, var2)	Возвращает 1, если var > var2, иначе — 0
below (var, var2)	Возвращает 1, если var < var2, иначе — 0

TimeScope — временной сигнал. Этот эффект отображает изменение сигнала во времени. Для его нормальной работы нельзя задавать очистку экрана на каждом кадре. Кроме настроек цвета, взаимодействия с изображением и привязки к каналу звука, модуль содержит движок Draw, с помощью которого задается количество полос сигнала, которые будут рисоваться.

FyrewurX (фейерверк) — рисует на экране салют. Из настроек есть только цвет частиц. Модуль находится в пункте RenderAPE.

В дополнение к эффектам, рисующим изображение, WVS содержит большое количество модулей, осуществляющих преобразование готового изображения. Данные модули расположены в пункте Trans. Важно расположение этих модулей в списке эффектов, так как действуют они на предыдущие эффекты рендеринга. Ниже описаны некоторые из эффектов преобразования.

Blitter Feedback — эхо. Модуль отрисовывает копию экрана с масштабом, задаваемым движком Blitter Direction. Изменение масштаба можно привязать к импульсам ударных, установив флажок Blitter Direction (OnBeat).

Blur — размытие. Уменьшает резкость изображения.

Brightness — яркость. Этот модуль позволяет изменять яркость изображения в целом и каждого из трех основных цветов — красного (Red), зеленого (Green) и голубого (Blue) — отдельно при установленном флажке Dissociate RGB Values.

Dynamic Distance Modifier — изменение координат точек изображения в динамическом режиме. Модуль изменяет расстояние от центра экрана до точки изображения согласно заданным формулам в поле Pixel. Значение расстояния задается переменной d. Работа с Dynamic Distance Modifier в целом аналогична работе с SuperScope.

Dynamic Shift — динамический сдвиг. Этот модуль осуществляет сдвиг изображения на расстояние, задаваемое переменными x и y. Значения переменных рассчитываются по произвольным формулам.

Fadeout — обесцвечивание изображения. Осуществляет преобразование цвета элементов изображения к заданному в поле Fade to color. Скорость преобразования задается движком Fade Velocity.

Grain — зерно. Фильтр, осуществляющий вывод изображения в зернистом виде.

Interferences — интерференция. Создает эффект блика и разложения по цветам.

Interleave — решетка. Накладывает на изображение решетку.

Invert — инвертирование цветов изображения предыдущих модулей.

Mirror — зеркальное отображение. Осуществляет зеркальное отображение половины изображения по следующим направлениям:

- Top to Bottom — сверху вниз;
- Bottom to Top — снизу вверх;
- Left to Right — слева направо;
- Right to Left — справа налево.

Параметр Static задает постоянное отображение, OnBeat Random привязывает отображение к ударным, Smooth Transitions сглаживает переходы при изменении отображения.

Mosaic — разбивает изображение на квадратные элементы заданного размера, образующие мозаику.

Movement — искажение. Предлагает большой список возможных искажений элементов изображения. Здесь есть размытие, закручивание, смещение центра и др. Можно задавать свой вариант изменения изображения.

Roto Blitter — масштабирование и поворот. Движок Zoom задает масштаб рисунка. При малых масштабах рисунок циклически повторяется на экране. Движок Rotation задает значение угла поворота изображения вокруг центра экрана. Оба параметра можно привязать к импульсам ударных инструментов.

Scatter — рассеивание. Разбивает изображение на пиксели и смещает их друг относительно друга.

Unique tone — единый цвет. Заменяет цвет всех предыдущих элементов на заданный.

Water — вода. Еще один эффект размытия изображения.

Water Bump — волна на воде. Изменяет изображение в виде концентрических волн на воде. Волна возникает в момент импульса ударных. Drop position задает центр окружности волны.

Пункт меню Misc содержит вспомогательные модули.

Comment — комментарий. Позволяет записать произвольный текст.

Custom BPM — ударные. Позволяет производить настройку ударных для использования в WVS. В частности, можно установить длительность импульса (Arbitrary), количество пропускаемых импульсов (Skip).

Set render mode — установка режима вывода. Позволяет задавать способ вывода изображения для последующих модулей, а также толщину линий элементов.

Кроме различных эффектов, WVS имеет значительное число настроек. Они расположены в меню Settings.

Display — настройки экрана. Здесь следует отметить флажок No minimize on WinAmp minimize, запрещающий сворачивать экран визуализации при сворачивании проигрывателя.

Fullscreen — настройки полноэкранного режима.

Presets/Hotkeys — настройка "горячих" клавиш, которые могут запускать тот или иной готовый модуль визуализации. При установке флажка Randomly switch presets модули визуализации будут случайным образом меняться через заданное время.

Beat Detection — настройки детектора импульсов ударных инструментов.

В качестве примера создадим очень простую визуализацию. Для ее создания из меню Render выберем эффект Ring. В настройках эффекта в поле Ring source and position установим привязку вида кольца к анализатору спектра — Spectrum. Затем из меню Trans выберем эффект Roto Blitter. В

настройках этого эффекта установим флажок Enable OnBeat changed и несколько уменьшим значение соответствующего движка. Остальные настройки можно оставить без изменения. Результат работы данной визуализации приведен на рис.2.

На второй странице обложки представлено несколько примеров визуализации.

Дополнительную информацию о Winamp Advanced Visualization Studio можно найти на сайтах www.winamp.com и www.nullsoft.com.

Рис. 2





Дорогие друзья!
Сегодня на страницах "Вашего компьютера" проводится очередное, и надеемся, не последнее заседание "Клуба 1 апреля". Давайте вместе отпразднуем День смеха так, чтобы заряд бодрости и веселья помог преодолеть все трудности и достичь новых высот в творчестве!
Редакция

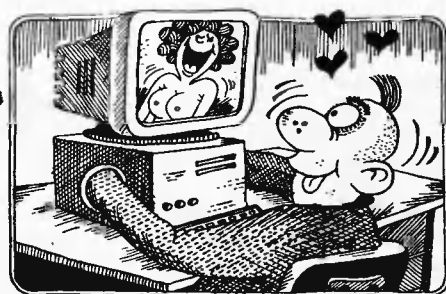


Иллюстрация О. Косачева

Теория ошибок

Основной закон

Ошибки так же неисчерпаемы, как и атом.

Аксиома

В любой программе есть ошибки.

Закон пропорциональности

Чем более программа необходима, тем больше в ней ошибок.

Следствие

Ошибок не содержит лишь совершенно ненужная программа.

Фундаментальный закон теории ошибок

На ошибках учатся.

Следствие 1

Программист, написавший программу, становится ученым.

Следствие 2

Чем больше программист делает ошибок, тем быстрее он становится ученым.

Следствие 3

Крупный ученый-программист никогда не пишет правильные программы.

Замечание

На то он и ученый.

Указание начинающему программисту

Если вы с первого раза сумели написать программу, в которой транслятор не обнаружил ни одной ошибки, сообщите об этом системному программисту, он исправит ошибки в трансляторе.

Закон находимости ошибок

Программист может найти ошибку только в чужой программе.

Следствие

Ошибке не все равно, кто ее обнаружит.

Совет начинающему программисту

Никогда не исправляйте найденные ошибки, ибо это повлечет за собой появление неизвестного числа ненайденных ошибок. Лучше опишите найденные ошибки в сопроводительной документации как особенность программы.

Определение языка ошибок

Языком ошибок называют правила, в обход которых пишутся программы.

Язык ошибок

Ошибки могут следовать друг за другом.

От перестановки двух эквивалентных ошибок результат не меняется (коммутативность эквивалентных ошибок).

Две последовательные ошибки можно объединить в одну, более сильную.

Одинаковые ошибки не обязательно делать каждый раз, достаточно сделать одну, а затем обращаться к ней по мере необходимости из любого места программы.

Ошибки могут образовывать циклы. Наиболее устойчивый из них — бесконечный.

Ошибки могут вызывать друг друга и сами себя (рекурсивность ошибок).

Ошибки допускают многократное еложение друг в друга.

Две одинаковые вложенные ошибки называются четной ошибкой и ошибкой не являются.

Свойство четности ошибок

Если написанная программа работает правильно, то это значит, что во время ее работы выполнялось четное число ошибок, или программист не понял задание.

Взаимодействие ошибок

с базовой операционной системой

Во время исполнения программы ошибки

имеют наивысший приоритет.

Прервать исполнение ошибки может только другая, более активная ошибка.

Запросы операционной системы к ошибкам ошибками могут игнорироваться.

Запросы ошибок к операционной системе игнорироваться не могут.

При работе с файлами ошибки могут пользоваться файловой системой базовой ОС и ее ошибками.

На ЭВМ с параллельной архитектурой может выполняться несколько ошибок одновременно.

Системные программы

Системные программы облегчают процесс написания прикладных программ и их ошибок.

Тестирование — это процесс нахождения ошибок в тесте.

Хороший тест должен содержать ошибки, компенсирующие их нехватку в тестируемой программе.

Языковой редактор, призванный уберечь программиста от синтаксических ошибок, позволяет вносить в программу весьма хитроумные ошибки, которые не удается обнаружить ни транслятором, ни отладчиком. Обычный текстовый редактор таких возможностей не предоставляет.

Программа-транслятор, предназначенная для перевода программ с языка высокого уровня на машинный язык, при переводе порождает ошибки. Ошибки, которые содержались в исходном описании, переводятся безошибочно.

До начала работы над проектом следует тщательно продумать все необходимые ошибки и связи между ними. Это значительно упростит работу над ошибками в самом проекте.

Машинные стихи

(читать вполголоса)

714, 15, 2247,
13, 318, 140, 327.
028, 220, 170, 045,
615, 3, 020 —
715, 012,
2 000 035.

"За науку", №12(607), 1 апреля 1977 г.

Из архива службы технической поддержки

— У меня не печатает ваш принтер.

— А в чем вы работаете?

— В хозуправлении.

— Нет, откуда вы печатаете?

— Из своего кабинета.

— Какой программой вы пользуетесь?

— Я не знаю, я — завхоз.

— Уберите у меня магнитную программу.

— Не понял?

— У меня бумага к монитору липнет.

— Ну так статика, экран намагничивается.

— А почему при нажатии ESC она отмагничивается?

— А можно CD-ROM восстановить?

— Попробовать можно.

Клиент достает компакт-диск, разломанный на две части...



В гарантийном отделе одной из компьютерных фирм к окошку подходит молодой человек и говорит:

- Здравствуйте! Я вчера отдал вам память.
- А кому именно Вы ее отдавали?
- Не помню.

Для того чтобы получить доступ в Интернет, достаточно зайти в наш интернет-магазин, выбрать одну из предлагаемых интернет-карт, оплатить ее удобным для вас способом, получить по электронной почте PIN-код, и можно подключаться.

Звонок в компьютерную контору:

- С моим компьютером проблемы, он не включается...
- Как не включается?
- Да пишет все время: "Вставьте дискету, вставьте дискету..."
- А вы вставляли?
- Нет... Как-то не догадался.

Звонок в службу технической поддержки:

- У меня компьютер не работает!
- После чего это произошло?

— Я его включил — загрузился Нортон. Смотрю — у меня слева диск С: и справа диск С:. Я подумал — зачем мне два диска С:? И стер правый.

Звонок в фирму по продаже компьютеров и комплектующих:

- Здравствуйте! У вас жидкие мониторы есть?
- Продавец:
- Вам на сколько литров: 15, 17, 19...?

Звонок в фирму, которая занимается ремонтом компьютеров:

- Мой принтер начал плохо печатать!
- Вероятно, его надо просто почистить. Это стоит \$50. Но для вас будет лучше, если вы прочтете инструкцию и выполните эту работу сами.

Клиент, удивленный такой откровенностью, спрашивает:

- А ваш босс знает, что вы таким образом препятствуете бизнесу?
- На самом деле, это его идея. Мы получаем куда больше прибыли, когда позволяем нашим клиентам сначала самим попытаться что-то отремонтировать.

- У вас память есть?
- Да, причем фирменная.
- Почему же вы меня не помните?

Мыши и джойстики — удобные, точные и надежные манипуляторы для правой, левой и большой руки.

Объявление на фирме ремонта компьютеров: "Перепрошьем ваш BIOS так, что мать родная не узнает".

- Винчестер крутили??? — Нет, он же сам крутится...
- Я его вставил, он там зашелкнулся, не вылезает и шебушит! (это про диск в дисковом).
- Пиликает, пиликает, а как кино, так ваще писец...

- А сколько можно CD-ROMов поставить?
- А сколько у вас дырок?



Программистские приметы

Если к вам перестали поступать жалобы на вашу программу, значит, ею уже никто не пользуется.

Если при объяснении вашей программы слушатель начинает кивать головой, его пора бить.

Если двое программистов пишут в точности одну и ту же программу, нужно преобразовать каждую из них в микрокод, и тогда они, конечно, не будут одинаковыми.

Если новая программа с первого раза компилируется без ошибок, значит, она написана принципиально неправильно.

Если программа заработала сразу, то искать ошибку будет значительно труднее.

Если в вашей процедуре 10 параметров, ве-

роятно, какой-нибудь пропущен.

Если тебе что-то не нравится, то не торопись это исправлять, если не помнишь, для чего это сделали.

Если вы хотите узнать, как работает программа, а описания и демо-ролика нет, то просто посадите на клавиатуру кошку — вы узнаете все документированные и undocumented возможности.

Не стоит изучать язык, который не меняет вашего представления о программировании.

Нет более живучих программ, чем заплатки, сделанные на скорую руку.

Ошибки легче всего делаются и труднее всего обнаруживаются в самых простых местах программы.

Оптимизация препятствует эволюции.

Проще написать неправильную программу, чем понять правильную.

Чем чаще программист жалуется на чужой софт, тем хуже он делает свой.

Чем универсальнее написана программа, тем меньше найдется мест, где можно было бы ее применить.

Чем точнее программист выполняет требования заказчика, тем бестолковее получается программа.

Чем больше заказчик понимает в программировании, тем больше он мешает работе.

Чтобы понять программу, необходимо отождествить себя с машиной, и с программой.

Мысли и фразы

Вечный вопрос — что было раньше: компилятор или исходники компилятора.

Зачем юзеру две руки? Левая — чтобы нажимать Esc, правая — Enter.

Насколько проще была бы жизнь, если бы она была в исходниках.

Нетлок, а недокументированная возможность!

Людам свойственно ошибаться, но окончательно все запутать может только компьютер.

Жизнь слишком коротка, чтобы писать на ассемблере.

...Идет настройка самонастраивающихся устройств...

"Жизнь на кончиках пальцев" — повествование авиадиспетчера о неудобных мышке и клавиатуре.

... Раз в месяц — контрольное форматирование винчестера. История Win95.

Великий грех всех программ перед программистами — они делают то, что программист написал, а не то, что он имел в виду.

...среди деловых приложений для Windows лидирует Uninstaller.

Ахиллесовой пятой чаще всего бывает голова.

Бесполезно придумывать защиту от дурака — ведь дураки так гениальны.

Быстро поднятый сервер не считается упавшим.

В "кривых" руках и калькулятор зависает.

Компьютер позволяет сделать из мухи споним легким движением мыши.

Мой компьютер постоянно обыгрывает меня в шахматы. Зато я всегда побеждаю его в боксерском поединке!





Р.ГАЛЕЕВ /HI-TECH,
Башкортостан, г.Туймазы.

Сага о кодировках

(Окончание. Начало в №№1-3/2004)

UTF-8

Использование UCS-2 (или UCS-4) с традиционными ОС, такими как MS DOS или UNIX, рассчитанными на работу с ASCII-кодами, вело к очень серьезным проблемам. Строки в таких кодировках содержали в качестве составных частей многобайтных символов байты со значениями, соответствующими символам "0", "1" и т.д., имеющим особое значение в именах файлов или в параметрах стандартных функций Си. Поэтому для символов в кодировке Unicode стали разрабатывать специальные формы представления, получившие обозначение UTF (UCF Transformation Format).

Табл. 14. Кодировка UTF-8

Диапазон значений Unicode	Кодируется байтами
U-00000000...U-0000007F	0xxxxxxx
U-00000080...U-000007FF	110xxxxx 10xxxxxx
U-00000800...U-0000FFFF	1110xxxx 10xxxxxx 10xxxxxx
U-00010000...U-001FFFFF	11110xxx 10xxxxxx 10xxxxxx 10xxxxxx
U-00200000...U-03FFFFFF	111110xx 10xxxxxx 10xxxxxx 10xxxxxx 10xxxxxx
U-04000000...U-7FFFFFFF	1111110x 10xxxxxx 10xxxxxx 10xxxxxx 10xxxxxx 10xxxxxx

Первоначальная UTF-1 уже устарела и в настоящее время не используется. Вслед за ней была разработана FSS-UTF (File System Safe UTF), которая после усовершенствования получила название UTF-2, но потом была переименована в UTF-8 (табл.14). Вот ее характеристики:

- для совместимости с ASCII символы UCS с U+0000 по U+007F кодируются просто как байты в диапазоне от 0 до 7Fh (т.е. строки, содержащие лишь ASCII-символы, в ASCII и UTF-8 кодируются одинаково);

- все символы UCS со значениями больше U+007F кодируются последовательностью нескольких байтов, у каждого из которых установлен старший бит. Следовательно, ни один байт в диапазоне ASCII-символов не может появиться в качестве части многобайтного символа;

- первый байт в многобайтной последовательности может быть в диапазоне от 0C0h до 0FDh. Число единиц до первого 0 в старших разрядах этого байта равно количеству байтов, которыми закодирован символ. Все остальные байты последовательности находятся в диапазоне от 80h до 0BFh (начиная с двоичных "10"). Это позволяет легко определять начало многобайтной последовательности в общем потоке байтов;

- таким способом можно закодировать все возможные 2^{31} кодов UCS;

- максимальное число байтов для кодировки одного символа UCS равно 6, однако для 16-разрядных символов в основной плоскости (BMP) достаточно 3 байтов, а для запланированного к использованию 21-разрядного диапазона — 4;

- байты 0FEh и 0FFh никогда не встречаются в кодировке UTF-8.

В табл.14 символы "xxx" обозначают биты кодируемого значения, причем крайние справа биты являющиеся младшими. Допустимой считается лишь самая короткая из всех возможных записей кода. Это значит, что символ U+0000 можно закодировать единственным способом — "0", но, например, не как последовательность "0C0h 80h".

UTF-16 и UTF-32

Когда было принято решение расширить диапазон значений кодов Unicode до 21 разряда, для представления новых значений была разработана новая кодировка UTF-16. Она

позволяет записывать коды в виде либо одного, либо двух 16-разрядных беззнаковых целых чисел. Символы со значением менее 10000h представляются в виде одного 16-разрядного числа. Для кодирования символов в диапазоне от 10000h до 10FFFFh в области BMP была зарезервирована так называемая суррогатная зона от U+D800 до U+DFFF размером 2x1024 позиций, которой не были назначены символы. Символы за пределами области BMP кодируются двумя 16-разрядными значениями, причем первое лежит в области от U+D800 до U+BFFF, второе — от U+C000 до U+DFFF, что дает общее число возможных сочетаний $1024 \times 1024 = 2^{20}$ (в сумме с символами BMP — 2^{21}).

С появлением 16-разрядных чисел возникает проблема взаимного расположения старшего и младшего байтов: "big-endian" — первым идет старший байт, или "little-endian" — первым идет младший байт (как на процессорах i80x86). Для явного указания порядка следования байтов кодировка может обозначаться как UTF-16BE в первом случае, и UTF-16LE во втором. Если порядок следования байтов в названии кодировки явно не указан (UTF-16), в начале файлов Unicode (особенно на платформе Win32) стали помещать отметку порядка байтов (Byte-Order Mark — BOM) — символ U+FEFF. Поскольку символа Unicode U+FFFE не существует, последовательность байтов 0FFh...0FEh в начале файла указывает, что порядок следования байтов "little-endian"; если же первыми идут байты 0Feh...0FFh, то это "big-endian". Если в файле с кодировкой UTF-16 BOM отсутствует, по умолчанию предполагается порядок "big-endian".

Термин UTF-32 был введен в Unicode для обозначения 4-байтной кодировки "расширенного" 21-разрядного Unicode и, в сущности, является тем же самым, что и UCS-4. Исключением является лишь диапазон представляемых символов — по определению, UTF-32 никогда не используется для представления символов свыше U-0010FFFF, тогда как UCS-4 охватывает все возможные 2^{31} позиции вплоть до U-7FFFFFFF. К UTF-32 применяются те же рассмотренные ранее соображения по поводу порядка следования байтов (то же, кстати, относится и к UCS-2 и UCS-4).

UTF-7

Чтобы завершить рассказ об UTF, необходимо рассмотреть еще одну кодировку — UTF-7. Она предназначена для разрешения проблем с протоколом SMTP, о котором уже упоминалось в разделе о кодировке KOI8-R. UTF-7 формально не относится ни к OSI 10646, ни к Unicode; она была определена службой поддержки Интернета IETF и описана в RFC2152.

Символы в UTF-7 кодируются следующим образом.

ASCII-символы от A до Z и от a до z, цифры от 0 до 9, а также следующие 9 символов:

' () , - . / : ?

остаются без изменений. Для всех остальных символов используется код Unicode, к которому применяется дополнительное преобразование по модифицированному методу Base64. Последовательность закодированных таким образом символов начинается с ASCII-символа "+", а завершаться может любым символом, не входящим в алфавит Base64. Специальным символом для завершения



последовательности является ASCII-символ “—”. Последовательность может завершаться также символом конца строки <CRLF>, т.е. закодированная последовательность не может переноситься на новую строку, и в каждой строке должна заново начинаться с символа “+”. При декодировании символы “+” в начале и “—” в конце удаляются.

Преобразование Base64 состоит в разбиении входящего потока байтов (в нашем случае — значений символов Unicode) на тройки, с последующим разбиением полученных 24-битных отрезков на 4 “куска” по 6 битов. Каждый такой “кусок” кодируется одним символом из алфавита A...Z, a...z, 0...9, +.../ (все символы в “безопасном” диапазоне ASCII). Модификация метода Base64 заключается в том, что для дополнения последнего отрезка до 24 битов (padding) используются нулевые биты, а не символ “=”, как в основной версии Base64.

Поддержка Unicode

Языки программирования, появившиеся после 1993 г., имеют специальные типы данных для символов Unicode.

К ним относятся Ada95, TCL, Perl, Python, C# и др., а также языки разметки HTML и XML. Java представляет строковые и символьные данные исключительно в Unicode.

В Windows 95 Unicode использовался в длинных именах файлов (VFAT) и для хранения Win32-ресурсов. Популярный офисный пакет MS Office с 1997 г. также перешел на Unicode. А ОС Windows NT, как и ее “родная” файловая система NTFS, изначально использовали Unicode для внутреннего представления символов. Функции Win32 API для работы со строками имеют две разновидности — для строк в формате Unicode (названия функций оканчиваются на W) и для строк в формате ANSI (названия функций оканчиваются на A), например, MessageBoxW и MessageBoxA. Большинство функций в “версии” Unicode поддерживаются только в Windows NT (и последующих ОС этой линейки), но не в Windows 95/98. Имеются также специальные функции для перекодировки строк, записанных в виде последовательности байтов (включая форматы UTF-8 и UTF-7), в 16-разрядные строки Unicode и обратно: MultiByteToWideChar и WideCharToMultiByte соответственно.

Операционная система Menuet OS

*Что нам стоит ось построить,
Закомпилим — будем жить.*

Горячие финские парни в очередной раз бросают вызов Microsoft и доказывают всему миру, что их операционные системы могут составить серьезную конкуренцию продукции от “мелкомягких”. Сначала в августе 1991 г. 22-летний финский аспирант Линус Торвалдс (Linus Torvalds) заявил о завершении работы над ядром своей будущей операционной системы Linux. Затем 16 мая 2000 г. вышла в свет первая версия операционной системы Menuet OS. Создатель этой принципиально новой операционной системы — финский программист Вилле Турьянмаа (Ville Turenmaa). В последнее время о Menuet OS серьезно заговорили. Не так давно этой операционной системе исполнилось три года, и она стала совсем “взрослой”. На сегодняшний день Menuet OS — это современная многозадачная 32-разрядная операционная система с графическим интерфейсом.

Установка Menuet OS

Для установки операционной системы Menuet OS требуется следующая минимальная конфигурация: компьютер на базе процессора Pentium с оперативной памятью 32 Мб и видеокартой, поддерживающей стандарты VESA 1.2 или VESA 2.0. Menuet OS можно установить на компьютере, на котором уже установлены операционные системы Windows 9x/Me/2000/XP или Linux.

Для установки Menuet OS понадобится архив инсталлятора **msetup.zip**, который можно скачать по адресу [1]. В этом архиве находится файл **msetup.exe** (DOS-программа, инсталлятор Menuet OS), который нужно поместить в корневую каталог жесткого диска. Этот файл не надо пытаться копировать на дискету, тем более, он все равно туда не поместится. Затем нужно вставить в дисковод чистую отформатированную дискету 1,44 Мб и после этого запустить с жесткого диска файл **msetup.exe**. Начнется процесс инсталляции системных файлов на дискету. Все эти операции можно проделать, находясь непосредственно в Windows. После окончания инсталляции производят перезагрузку, и в Setup BIOS устанавливают загрузку с дискеты. В начале загрузки Menuet OS устанавливают требуемый режим видеосистемы и порт мыши.

А.ГОРЯЧКИН,

г.Кыштым, Челябинской области.
E-mail: anatoliy_goryach@mail.ru

При выборе режима видеосистемы можно ориентироваться на сообщение, какую версию VESA поддерживает видеокарта — **Display Bios: Vesa x.x**.

Существуют три варианта загрузки Menuet OS.

Первый вариант — производить загрузку только с дискеты. Это самый медленный способ загрузки, но он позволяет загружать Menuet OS практически на любом компьютере, так как при этом не требуется размещать файлы системы на жестком диске.

Значительно сократить время загрузки Menuet OS позволит второй вариант. В этом случае стартуют, как обычно, с дискеты, а завершают загрузку с жесткого диска. Для этого нужно во время загрузки при выводе сообщения “Load ramdisk from ...” выбрать пункт 2, при условии, что файл **msetup.exe** находится в корневом каталоге жесткого диска.

Третий вариант — загружать Menuet OS только с жесткого диска. Этот способ предполагает наличие первичного FAT32-раздела на жестком диске, который подключен к Primary IDE-контроллеру. Также вам понадобится загрузчик, который необходимо скачать со страницы [2]. Этот загрузчик должен быть размещен в корневом каталоге вместе с файлом **msetup.exe**. В файл **autoexec.bat** вносят строку **meosload** (по имени файла-загрузчика). Во всех случаях в процессе загрузки в оперативной памяти создается виртуальный RAM-диск, куда и копируются файлы операционной системы Menuet OS.

Интерфейс

Эффект после окончания загрузки Menuet OS просто ошеломляющий. Рабочий стол, главное меню, окна, панели с иконками — налицо все атрибуты настоящей операционной системы. Графический интерфейс с обилием цветовых гамм понравится многим пользователям (рис.1). Окна программ можно открывать, закрывать, перетаскивать по рабочему столу и разворачивать на весь экран. Сворачивать окна, увы, пока возможности нет.

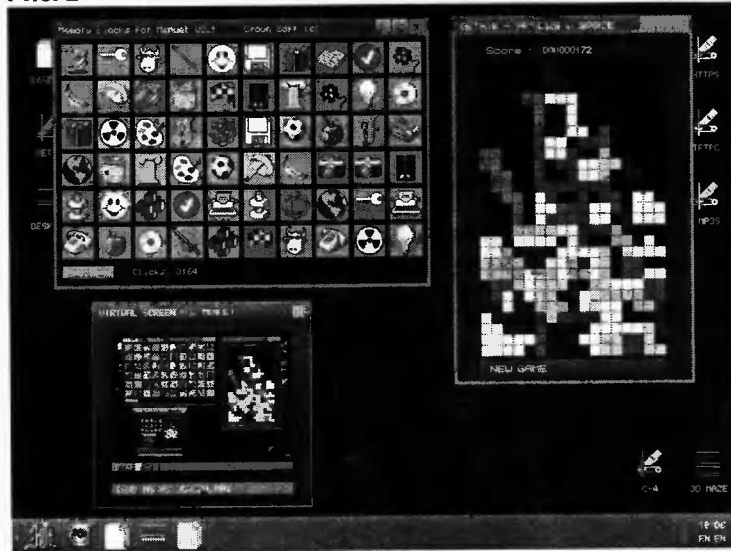
После того как угаснет эйфория после первого знакомства с Menuet OS, неплохо подумать о практическом применении этой микроскопической операционной системы (вот уж кому надо заслуженно носить имя Microsoft). Кроме ядра и графической оболочки, в состав Menuet OS включен базовый комплект си-

Рис. 1



стемных и прикладных программ и приложений. Не обойдены вниманием и геймеры, для которых написано несколько игр: Memory Block, Pong, Tetris, C4 и Caner (рис.2).

Рис. 2



Если говорить о более серьезных вещах, то в Menuet OS имеется файловый менеджер System Tree (не путать с System Tray в Windows). В System Tree частично включена поддержка обработки файлов по расширению. Например, если щелкнуть по файлу с расширением TXT или ASM, загрузится текстовый редактор. С помощью файлового менеджера можно просматривать содержимое жесткого диска и RAM-диска. Для копирования файлов используют программу File Copier.

Для определения текущей частоты центрального процессора используют утилиту CPU Speed. Tityrad — это простой текстовый редактор. Из графических приложений в состав операционной системы включены две программы — просмотрщик BMP Viewer и редактор иконок Icon Editor. Мультимедийный набор в Menuet OS представлен проигрывателями аудио компакт-дисков, MIDI- и WAV-файлов. Для регулировки громкости звучания используют микшер. Имеется в Menuet OS и хранитель экрана — Menu/Applications/Draw/Game Demos/Screen Saver Demo. Но, в отличие от Windows, в Menuet OS пока отсутствует автоматический запуск заставки при простое компьютера.

Если читатель обратил внимание на скриншоты, то заметил, что на них присутствует окно программы Virtual Screen. Эта программа позволяет сохранить экранную область в файле **screen.raw**, который после сохранения размещается в корневом каталоге жесткого диска. Загрузившись в Windows, файл **screen.raw** преобразуют в графический формат BMP, используя утилиту **raw2bmp**, которую также можно скачать со страницы [2]. Затем загружают полученный BMP-файл в графический редактор, например, в штатный Paint, и переворачивают изображение по вертикали. Таким образом получают скриншоты Menuet OS.

Выход из системы стандартный — из главного меню кнопкой Menu/Quit. В меню завершения работы будет предложено четыре пункта. Команда **Save Ramdisk to Floppy** позволит сохранить содержимое на дискету. Следующая команда **APM — PowerOff** выключит компьютер, **Reboot** — перезагрузит компьютер. Команда **Restart Kernel** позволит произвести "горячий" рестарт ядра системы. В общем, как говорят в таких случаях, интерфейс Menuet OS интуитивно понятен, и проблем в освоении операционной системы не возникает.

Настройка Menuet OS

Для настройки системных параметров, "железа" и клавиатуры применяют программу Menuet Setup. Настройку иконок, размещаемых на рабочем столе, производят с помощью программы Desktop (Icon manager). Запуск программы Stack позволит установить требуемые параметры сетевого протокола TCP/IP для подключения к Интернету. Эти иконки находятся на рабочем столе. Настроить внешний вид панели меню можно с помощью Panel Setup. Если вам захочется сменить фон рабочего стола, то для этого надо воспользоваться программой Background II, или с помощью BMP Viewer поместить на рабочий стол рисунок в формате BMP. Сменить системные цвета можно с помощью программы Desktop Colours. Созданная сторонним разработчиком утилита VRR (Vertical Refresh Rater) способна изменить частоту обновления экрана. Утилита VRR размещена на странице [2].

Достоинства...

Теперь поговорим о плюсах Menuet OS. Надо признать, что разработчику удалось добиться главного — система вполне работоспособна при рекордно малом объеме. Значительно упрощает установку и испытание Menuet OS то обстоятельство, что не требуется "сносить" Windows. Обе операционные системы мирно соседствуют и не мешают при работе друг другу. Любой пользователь при желании может испытать Menuet OS на своем компьютере. Привлекает и компактность операционной системы — ее дистрибутив умещается на одной дискете 1,44 Мб. Реализована поддержка протокола TCP/IP. Организована в Menuet OS и поддержка русского языка. К тому же, Menuet OS не столь требовательна, как Windows, к аппаратным ресурсам. Кроме того, Menuet OS — это свободно распространяемое программное обеспечение, поэтому на дискете вместе с исполняемыми файлами вы обнаружите исходные коды, написанные полностью на 32-битном ассемблере для x86. Поскольку исходные коды Menuet OS открыты и доступны для всех желающих, то в разработке программного обеспечения для этой операционной системы приняли участие многие программисты из различных стран, в том числе и российские. Для любителей программировать на ассемблере прилагается компилятор FASM.

Menuet OS постоянно обновляется. На русскоязычном сайте этой операционной системы [3] периодически выкладываются свежие новости о Menuet OS. Например, есть сведения о том, что в последних версиях Menuet OS разработаны средства для проигрывания MP3-файлов, появилась возможность работать с почтой, есть E-mail-клиент и Web-браузер. Правда, графика в Web-браузере отображаться не будет, и во время навигации по Интернету придется довольствоваться пока только текстовой информацией. Есть информация и о том, что в Интернете заработал первый Web-сервер под управлением Menuet OS [4]. Если развитие этой операционной системы будет продолжаться подобными темпами, то вскоре о Menuet OS можно будет говорить как об альтернативе Microsoft Windows.

... и недостатки Menuet OS

И о минусах. Единственный существенный недостаток Menuet OS — не сохраняются настройки. Поэтому приходится каждый раз после загрузки системы настраивать все системные параметры заново. В отличие от Windows, операционная система Menuet OS не способна пока тестировать оборудование и не распознает, как подключен винчестер и CD-ROM, хотя эта проблема вполне решаема. Чтобы каждый раз после загрузки не заниматься заново настройкой системы, рекомендуется при выходе из Menuet OS сохранить содержимое RAM-диска на дискету. Правда, в сохранении содержимого RAM-диска на дискете есть один "подводный камень" — загрузить Menuet OS с этой дискеты будет невозможно на другом компьютере, так как конфигурации компьютеров будут не идентичны. К тому же, функция сохранения RAM-диска срабатывает почему-то не на всех компьютерах (испытывалась версия Menuet 0.73 Beta).

В некоторых случаях происходит следующее — обращение к дисководу есть, светодиод светится, но запись файлов не производится, и через непродолжительное время начинается процесс перезагрузки Menuet OS. Возможно, этот "глюк" будет устранен в последующих версиях. И еще. К сожалению, Menuet OS позволяет пока работать только с файловой системой FAT32.

Заключение

Знакомство с Menuet OS расширит представление пользователей об альтернативных операционных системах, позволит сравнить возможности Menuet OS с Windows и с другими операционными системами. Хотелось бы, чтобы финский "осознатель" Вилле Турьянмаа не забросил свое "детище", и Menuet OS продолжала бы свое стремительное развитие и совершенствование. Думается, что этот проект определенно имеет большое будущее.

Литература

1. <http://www.menuetos.org/MSETUP.ZIP> — архив инсталлятора операционной системы Menuet OS.
2. <http://menuet.narod.ru/down.htm> — страница для скачивания программ Menuet OS.
3. <http://menuet.narod.ru> — русскоязычный сайт Menuet OS.
4. <http://213.243.134.211> — первый Web-сервер под управлением Menuet OS.
5. villem@silmu.jyu.fi — адрес электронной почты разработчика Menuet OS.
6. В.Холмогоров. Маленький гигант. — Chip, 2003, № 2, С.108.
7. Дайджест. — Радиомир. Ваш компьютер, 2003, № 6, С.10.

А.ГРИНЧУК,

E-mail: gav@nihe.niks.by

Анимация с Macromedia Flash MX

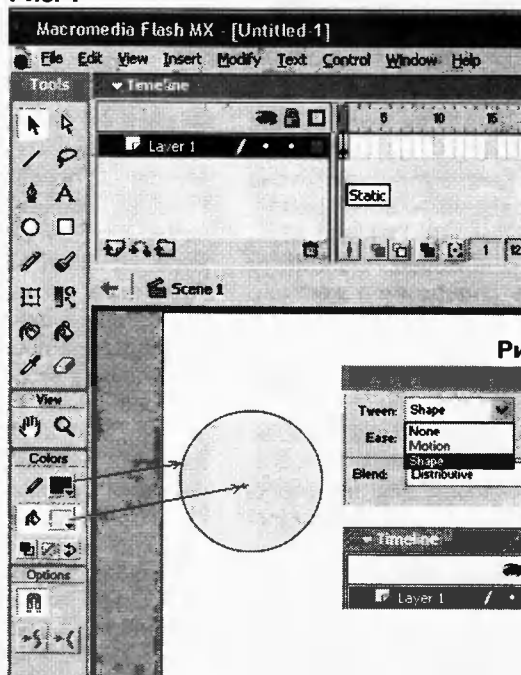
Анимация формы

После того как мы заставили двигаться фигуры, наша задача — добиться большей эффектности. И в этом нам поможет другой тип анимации — анимация формы, или морфинг. Здесь по-прежнему работает правило размещения каждой отдельной анимации в своем слое. Но для того чтобы фигурка не только перемещалась, но и изменяла свой внешний вид, логично предположить, что группированные объекты для этих целей непригодны. Действительно, нам придется работать именно с негруппированными объектами. В связи с этим немного изменятся последовательность действий, а заодно придется изучить несколько новых команд.

Начнем, как обычно, с самого простого. В качестве примера заставим желтый круг превращаться в синий прямоугольник. В первом кадре рисуем (с помощью панели инструментов, Tools) желтый круг с синей границей (рис.1). Затем делаем ключевым 30-й кадр (выделить кадр и F6). А вот в последнем кадре нам придется уже удалить круг и нарисовать синий квадрат с желтой гра-

ницей. Цвета, естественно, могут выбираться произвольно, просто будет создаваться плавный переход не только

Рис. 1

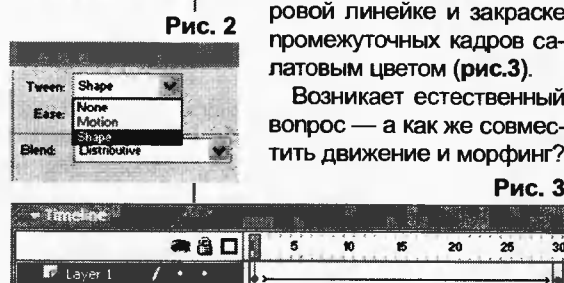


формы, но и цветов. Увы, красоту оттенков в промежуточных кадрах гарантировать нельзя, здесь придется при необ-

ходимости поэкспериментировать. После прорисовки последнего кадра необходимо вернуться в первый и задать тип анимации. Первый кадр должен быть выделен, а на панели свойств (Properties) в качестве типа автозаполнения укажем форму (Tween: Shape, рис.2). О правильности выполнения всех действий можно судить по появлению сплошной стрелки на кадровой линейке и закраске промежуточных кадров салатовым цветом (рис.3).

Возникает естественный вопрос — а как же совместить движение и морфинг?

Рис. 3



Да нет ничего проще, достаточно нарисовать в пер-

Рис. 4

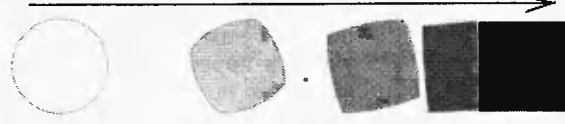


Рис. 5



Ease: -70 (замедление на входе)

Рис. 6



Ease: 70 (замедление на выходе)

Рис. 7



вом и последнем кадрах фигуры в разных областях рабочего поля, и все произойдет автоматически (рис.4). Более того, некоторые из трюков анимации движения по-прежнему остаются в силе. Можно задавать замедление на входе (рис.5) или на выходе (рис.6) с помощью параметра **Ease** (рис.2). Также допускается и масштабирование фигур. Например, на рис.7 показано, как изменится наш ролик, если в последнем кадре выделить квадрат вместе с границей, выполнить команду **Modify/Transform/Scale** и за угловые маркеры немного растянуть его. Только приходится всегда иметь в виду, что анимация формы применяется к набору точек, а не к единому сгруппированному объекту. Поэтому вращать фигуры уже не получится, настройка вращения после выбора **Tween: Shape** вообще не появляется на панели.

Таким образом, два типа анимации довольно сильно отличаются, и это приводит к появлению ряда дополнительных опций. Для их освоения рассмотрим достаточно эффектный прием — трансформацию текста. Для "разминки" заставим цифру 1 превратиться в цифру 2. С помощью инструмента **Текст** и панели **Properties** напечатаем единицу 96-м шрифтом. Затем сделаем ключевым 30-й кадр, сделаем двойной щелчок по цифре и заменим ее на двойку. Самым неприятным обстоятельством является то, что текст автоматически создается как сгруппированный объект, поэтому ра-

нее нам легко удавалось заставить его двигаться. А вот для морфинга придется повозиться. И в начальном, и в конечном кадрах необходимо проделать следующее: выделить текстовую надпись и командой **Modify/Break Apart** (или **Ctrl + B**) разбить ее на точки. Заметим, что операция группировки не позволит воссоздать из точек текстовый символ, так что стоит лишний раз проверить надписи перед разбиением. Но после этой операции смело можно настраивать анимацию формы. Правда, пока она выглядит не самым лучшим образом (рис.8). Можно немного улучшить ситуацию, если в первом кадре после выбора типа анимации задать и тип перехода (**Blend** — рис.9).

Рис. 8: Последовательность кадров, иллюстрирующая делительный переход (**Distributive**) в большей степени подходит для плавных фигур и резких и угловатых очертаний предпочтительнее угловой переход (**Angular**, рис.10).

Рис. 8



Рис. 9

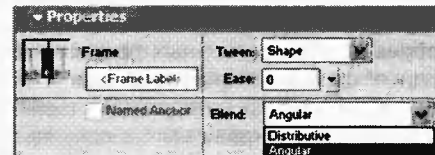


Рис. 10



Подводим промежуточный итог — немного лучше, но что-то не то. Все равно возникает непреодолимое желание весь процесс трансформации держать под контролем. Для этих целей были разработаны указатели формы (**Shape Hints**). Видны эти указатели только в первом и последнем кадрах, маркируются буквами латинского алфавита и выглядят как кружочки с маркером внутри. Создаются указатели только в начальном кадре отрезка анимации с помощью команды **Modify/Shape/Add Shape Hint** (или **Ctrl + Shift + H**). Вначале заставим

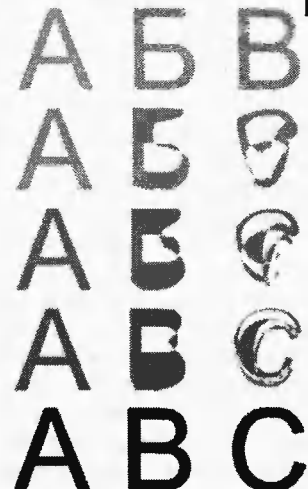
"носик" цифры 1 переходить в "носик" двойки. Для этого создаем указатель формы в начальном кадре, перемещаем его в нужную позицию с помощью мыши, а затем переходим в последний кадр и перемещаем указатель и там. Если мало одного указателя, то процесс можно продолжить. Только не рекомендуется сразу же в первом кадре выставлять несколько указателей. Лучше делать все не спеша — сначала создается и устанавливается в ключевых кадрах один указатель, затем следует возврат в первый кадр, создается следующий и т.д. Проблема заключается в том, что для перерасчета фигур требуются достаточно большие вычислительные ресурсы и используются сложные алгоритмы. Спешка здесь может привести к "глюкам программы" и выходу процесса анимации из-под контроля. Лишние указатели формы удаляются щелчком правой клавишей мыши и выбором команды **Remove Hint**, а все сразу — **Remove All Hints**. Количество точек приходится подбирать в зависимости от сложности фигуры, в нашем случае хватило трех (рис.11).

Рис. 11



Обычно неплохо выглядит трансформация русского текста в английский или наоборот (рис.12). Здесь можно дать несколько общих рекомендаций. Чтобы анимация осуществлялась без хаотического слипания букв в одно пятно, лучше текст набирать вразрядку, хотя бы даже с пробелами между буквами. Команду разбиения (**Modify/Break Apart** или **Ctrl + B**) во Flash MX придется повторить дважды, т.к. после первого применения текст всего лишь разделится на отдельные буквы.

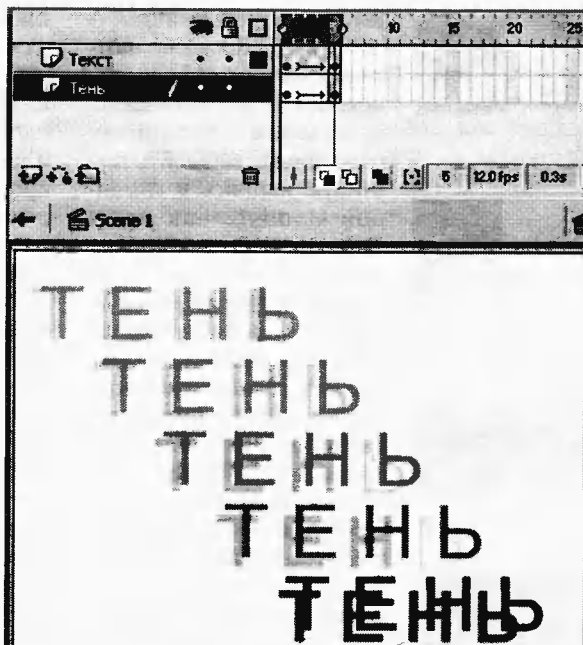
Рис. 12



А вот установка указателей формы в таких сложных объектах мало поможет. Чтобы нормально использовать **Shape Hints**, придется каждую букву разместить в отдельном слое (**Modify/Break Apart** и **Modify/Distribute to Layers**) и работать с ними в индивидуальном порядке. Кроме увеличения размеров файла и усложнения работы, морфинг обладает еще одним недостатком — неспособностью корректно обработать импортированный рисунок. Немного лучше получается с векторной графикой (WMF, EMF, EPS), но предпочтительнее для этих целей создавать рисунки средствами Flash.

Зато ничто не мешает комбинировать в одном ролике два типа анимации. Так, на рис. 13

Рис. 13



показано, как примерно будут выглядеть отдельные кадры анимации, если в одном слое будет двигаться просто текст, а в другом — производиться морфинг. Для большей реалистичности в последнем кадре слоя **Тень** задано дополнительное смещение влево и вниз, а также сглажены углы тени: **Modify/Shape/Soften Fill Edges**.

Итак, наш арсенал понемногу расширяется. Но и это еще не все. Для большего правдоподобия нам необходимо научиться перемещать фигуры не только по прямым линиям, но еще и выйти за рамки синхронного исполнения морфинга и движения. Этим мы и займемся в следующей статье, где рассмотрим специальные виды анимации.

“Упорядочиватели” дискового пространства,

или немного о современных дефрагментаторах

Не знаю, кому как, а лично мне очень часто приходится устанавливать самые различные программы, удалять их, опять устанавливать, но уже другие — и так до бесконечности. Поэтому примерно через месяц-другой после очередной переустановки Windows большая часть софта, установленного на моей машине, обновляется, а многое из него и не единожды. Одним из самых неприятных последствий такой суматошной жизни является сильная фрагментация файлов, расположенных на жестком диске. Он, как известно, разбит на маленькие ячейки памяти, называемые кластерами, размер которых колеблется от одного до нескольких десятков, а то и сотен килобайт, в зависимости от размера раздела (или, правильнее, тома) на диске. Длина же файлов чаще всего превышает величину кластера, поэтому при записи файла на диск его “куски” попадают в разные кластеры, которые могут быть расположены где угодно, по всей поверхности диска. Получается, что для того чтобы прочитать один файл, разные части которого разбросаны по всему разделу, головкам винчестера приходится неоднократно перемещаться, затрачивая на каждое из таких перемещений определенное вре-

мя. Следовательно, чем больше разрозненных “фрагментов” файлов, тем медленнее компьютер выполняет операции ввода/вывода, чему в особой мере подвержены старые файловые системы типа FAT. Еще более неприятно то, что в результате программных (ох уж эти винды...) или аппаратных сбоев цепочка различных фрагментов одного файла может разорваться, и на диске появляются потерянные кластеры, которые откровенно захламляют дисковое пространство. Выходом из создавшегося положения может служить регулярная дефрагментация дисков. Тема эта отнюдь не новая, но, тем не менее, никогда не теряющая своей актуальности.

В состав большинства версий Windows входят штатные утилиты, довольно сносно выполняющие эти функции, но не более того. Каждый, кто желает иметь по-настоящему качественный дефрагментатор, должен обратиться к продуктам третьих фирм. И таких программ, особенно ориентированных на поддерживаемые Windows 9x файловые системы FAT и FAT32, существует большое количество. Но сейчас, в связи с высокой популярностью Windows 2000 в корпоративной среде и, ничуть не меньшей, Windows XP на домашних ком-

пьютерах, основной файловой системой которых является NTFS версий 5.0 и 5.1 соответственно, гораздо интереснее будет рассмотреть программы-дефрагментаторы именно для NTFS. Файловая система NTFS, в отличие от FAT, обладает усложненной логической структурой, позволяющей без особых проблем работать с достаточно сильно фрагментированными файлами, хотя от физического последствия такой фрагментации — лишних перемещений считывающих головок — она, конечно, не спасает.

Может быть поэтому в Сети до сих пор циркулируют легенды, что, мол, NTFS вообще не подвержена фрагментации, и, естественно, дефрагментаторы для нее нужны как рыбка зонтик. К сожалению, NTFS подвержена фрагментации, хотя, конечно, не так быстро и не так сильно, как старушка FAT. Но сложность логической структуры NTFS, в свою очередь, значительно усложняет дефрагментацию дисков, поэтому для облегчения этого процесса в состав операционных систем Windows 2000 и XP включено стандартное API дефрагментации, разработанное Microsoft, которое и использует большинство описываемых ниже программ-дефрагментаторов. Итак, к делу.

В.КУЦ,

E-mail: victor_kootz@mail.ru

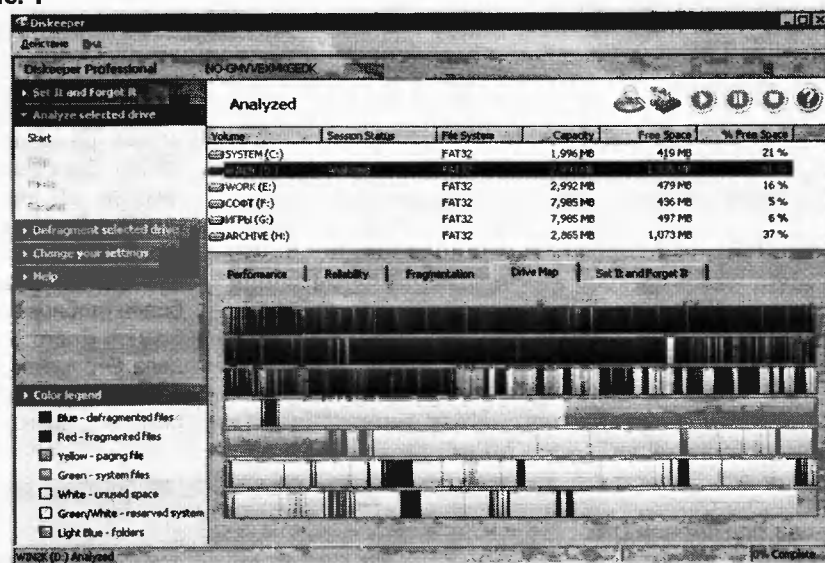
Diskeeper Professional 8.0.459

Без сомнения, самым популярным дефрагментатором для NTFS является разработанная компанией Executive Software International (<http://www.execsoft.com/>) программа Diskeeper. Эта программа в настоящее время представлена двумя версиями для настольных компьютеров (самая простая — Home, не умеющая дефрагментировать тома по сети, и Professional), поддерживающими как Windows 9x, так и одноименные версии Windows NT/2000/XP, а также целыми четырьмя вариантами серверных версий: Diskeeper Server Standard, Web, Enterprise и Datacenter Edition (возможности — по нарастающей). Системные администраторы могут устанавливать программу через сеть и после этого производить дефрагментацию и задавать график автоматической дефрагментации для всех компьютеров в сети с единой консоли Diskeeper Administrator Edition.

Интерфейс программы Diskeeper построен на базе технологий DCOM (Distributed Component Object Model) и MMC (Microsoft Management Console), и представляет собой стандартную MMC-консоль, которую при необходимости можно легко модернизировать путем добавления в нее необходимых пользовательских элементов (рис.1). Все основные команды управления программой сосредоточились в специальной командной панели, занимающей левую часть рабочего окна консоли. Ее нижнюю часть занимают информационные панели, предоставляющие пользователю самую разнообразную информацию о работе программы.

Diskeeper поддерживает два режима дефрагментации: ручной и автоматический. С первым, по-моему, все ясно, и в дополнительных комментариях он не нуждается. Другой же режим дефрагментации — автоматический, когда Diskeeper сам определяет время и условия для своего запуска. Работа в фоновом режиме позволяет осуществлять обработку одновременно нескольких дисков, не вызывая никаких конфликтов с доступом пользователей к файлам на таких дисках. Предварительная настройка параметров этого режима осуществляется при помощи соответствующего меню, в котором можно задать скорость дефрагментации и приоритет процесса. По умолчанию, процесс дефрагментации в автоматическом режиме имеет наименьший приоритет, тогда как в ручном — нормальный. Естественно, можно выставить и самый высокий приоритет, но тогда, хотя процесс дефрагментации значительно ускорится, практически невозможно будет работать в это время с каким-нибудь

Рис. 1



другим приложением — до такой степени все будет заторможено. Хотя возможны и промежуточные, компромиссные варианты. Помимо обычной дефрагментации по расписанию, существует и достаточно интеллектуальный режим с многозначительным названием Set It and Forget It (Установи и Забудь), в основе которого лежит функция интеллектуального планирования (Smart Scheduling), которая динамически корректирует расписание процесса дефрагментации на основании текущего состояния диска. Diskeeper принимает во внимание уровни фрагментации файлов и настраивает период выполнения задания дефрагментации с интервалом от одного часа до одной недели. При этом, в зависимости от загрузки системы, программа самостоятельно выбирает еще и приоритет выполнения. От пользователя или администратора сети не требуется учитывать загрузку сети — Diskeeper автоматически подстраивается под нее и обеспечивает ей максимальную производительность.

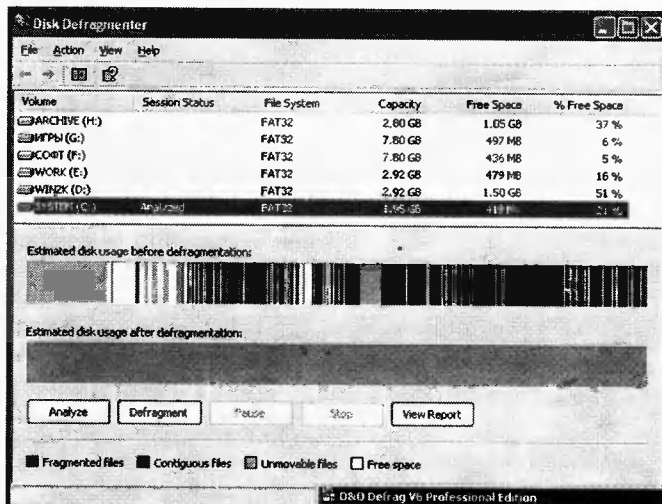
Среди сугубо специфических для NTFS функций можно выделить загрузочный вариант (Boot-Time Defragmentation), позволяющий дефрагментировать MFT и файл подкачки. MFT (Master File Table) является специальным файлом, в котором содержится информация обо всех других файлах, хранящихся в данном NTFS-томе. Эта информация включает в себя имя каждого файла, его физическое месторасположение на диске и некоторую другую служебную информацию. Грубо говоря, MFT является аналогом FAT (File Allocation Table). Дефрагментация тома производится в момент запуска компьютера, когда MFT и файл подкачки еще не контролируются операционной системой. Загрузочный вариант дефрагментации используется не столь часто, как ос-

тальные варианты. Им лучше всего воспользоваться при первом запуске впервые установленного Diskeeper'a, а также в тех случаях, когда необходимо сгруппировать большое количество новых директорий, появившихся, к примеру, в результате установки какого-либо нового приложения. Кроме того, довольно интересна функция FragGuard, позволяющая определить количество фрагментов файла подкачки и MFT, при котором во время следующей загрузки автоматически будет осуществлена дефрагментация.

Многие пользователи Diskeeper отмечают, наряду с достаточно хорошим качеством работы программы, ее высокую скорость. Не меньшее значение имеет и надежность, когда данные на обрабатываемом разделе сохраняются независимо от любых "потрясений" системы в процессе дефрагментации. Такой "букет" достоинств, собранных в одном месте, делает Diskeeper общепризнанным стандартом "де-факто" для всего семейства ОС Windows NT/2000/XP. Но, наряду с неоспоримыми достоинствами, Diskeeper имеет и ряд недостатков. Так, например, для корректной работы программы необходимо иметь никак не менее 20% свободного места на диске, иначе на разделах с большим количеством файлов дефрагментация может просто не осуществиться. Впрочем, в той или иной мере такой недостаток свойственен всем дефрагментаторам, использующим в своей работе стандартное API дефрагментации от Microsoft. Кроме того, уже давно, буквально с первых версий Diskeeper'a, всем хорошо известны ее довольно "натянутые отношения" с разделами, созданными программой Partition Magic.

С облегченным вариантом программы Diskeeper все пользователи Windows 2000/XP хорошо знакомы —

Рис. 2



ведь она, под именем Disk Defragmenter, входит в состав этих ОС в качестве штатного дефрагментатора (рис.2). Лишенный всех «наворотов» своего прототипа, Disk Defragmenter не производит дефрагментацию ни таблиц расположения файлов (FAT или MFT), ни файла подкачки (который в NTFS называется пейджфайл), ни реестра. Способен он только на самую примитивную дефрагментацию томов на локальном диске и только в ручном режиме.

Но зато эта программа предельно проста в использовании, что делает операцию дефрагментации доступной даже начинающим «дефрагментаторам». Кроме того, нельзя не отметить довольно высокую скорость ее работы. Disk Defragmenter работает только с локальным диском, однако в домашних условиях такое ограничение не является большим недостатком.

O&O Defrag Professional 6.0.609

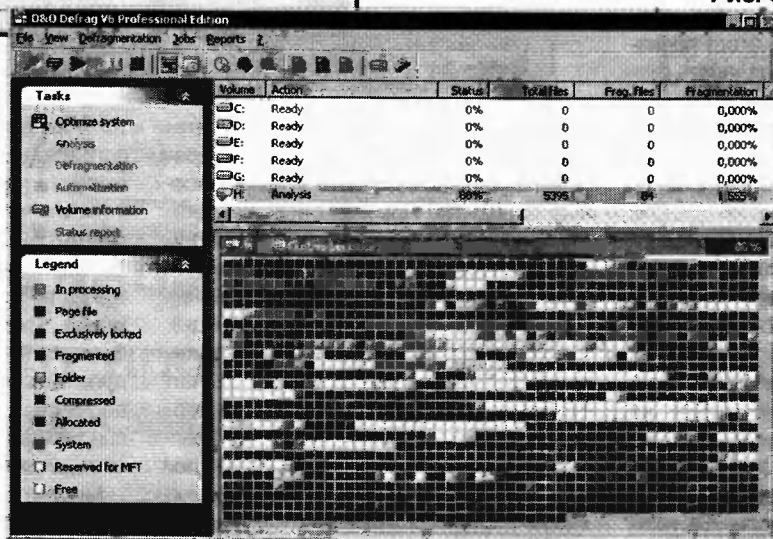
Очень похож на Diskkeeper дефрагментатор O&O Defrag Professional (рис.3), разработанный немецкой компанией O&O Software (<http://www.oosoft.de>). И в этом нет ничего удивительного — оба эти продукта используют стандартный API и базируются на базе интерфейса, предоставляемого Microsoft Management Console (MMC). Правда, работает описываемая версия программы только с NT-подобными ОС Windows — NT 4.0/2000/XP, однако при этом обеспечивает дефрагментацию томов всех типов, включая RAID-системы и зашифрованные (EFS). От всех аналогичных программ O&O Defrag отличается, в первую очередь, исполь-

зованием целых трех методов не столько дефрагментации, сколько оптимизации дисковых разделов. Самый быстрый из них, SPACE, организует перенос фрагментированных файлов в наиболее удобное свободное место раздела. Этот метод не дает особого выигрыша в производи-

ющих дефрагментациях, поскольку дефрагментировать придется только некоторые изменившиеся файлы. Естественно, все три «подметода» COMPLETE требуют гораздо большего времени, да и системных ресурсов, чем SPACE, но и прирост производительности дисковых операций они обеспечивают существенно больший. Ну а если в процессе COMPLETE-дефрагментации возникнут какие-либо проблемы, обычно возникающие при дефиците свободного места на диске, то программа автоматически переключается на работу в облегченном режиме SPACE.

Третий метод дефрагментации —

Рис. 3



STEALTH — функционально схож с SPACE, но осуществляется скрытно, не требуя при этом ни большого количества системных ресурсов, ни свободного места на диске (вполне хватает и скромных 5%). Обеспечивается это фирменной технологией O&O Activity Guard, которая управляет процессом дефрагментации в зависимости от текущей загрузки процессора. Режим STEALTH в наибольшей степени подходит для таких серьезных устройств как не-

прерывно функционирующие серверы, имеющие на своих дисках тома очень большого размера и содержащие большое количество (от 250 тысяч и больше) самых разнообразных файлов. Стандартными возможностями программ такого плана стали возможность запуска по расписанию, дефрагментация MFT, файла подкачки и реестра в режиме Boot-Time, т.е. во время перезагрузки системы, просмотр содержимого блоков и развернутая статистика фрагментации (побочно). O&O Defrag всем этим располагает в полной мере. Удобство использования O&O Defrag в мобильных компьютерах повышает функция по отслеживанию состояния системы управления питанием компьютера и приостановки процесса дефрагментации при переходе ноутбука в режим питания от батарей. Однако творение фирмы O&O не без недостатков — несмотря на то что, по сравнению с предыдущими версиями, скорость работы программы значительно повысилась, тем не менее, она работает гораздо медленнее, чем ее ближайшие конкуренты Diskkeeper и Perfect Disk, не говоря уж о просто «реактивном» VoptXP.

Третий метод дефрагментации —

STEALTH — функционально схож с SPACE, но осуществляется скрытно, не требуя при этом ни большого количества системных ресурсов, ни свободного места на диске (вполне хватает и скромных 5%). Обеспечивается это фирменной технологией O&O Activity Guard, которая управляет процессом дефрагментации в зависимости от текущей загрузки процессора. Режим STEALTH в наибольшей степени подходит для таких серьезных устройств как не-

прерывно функционирующие серверы, имеющие на своих дисках тома очень большого размера и содержащие большое количество (от 250 тысяч и больше) самых разнообразных файлов. Стандартными возможностями программ такого плана стали возможность запуска по расписанию, дефрагментация MFT, файла подкачки и реестра в режиме Boot-Time, т.е. во время перезагрузки системы, просмотр содержимого блоков и развернутая статистика фрагментации (побочно). O&O Defrag всем этим располагает в полной мере. Удобство использования O&O Defrag в мобильных компьютерах повышает функция по отслеживанию состояния системы управления питанием компьютера и приостановки процесса дефрагментации при переходе ноутбука в режим питания от батарей. Однако творение фирмы O&O не без недостатков — несмотря на то что, по сравнению с предыдущими версиями, скорость работы программы значительно повысилась, тем не менее, она работает гораздо медленнее, чем ее ближайшие конкуренты Diskkeeper и Perfect Disk, не говоря уж о просто «реактивном» VoptXP.

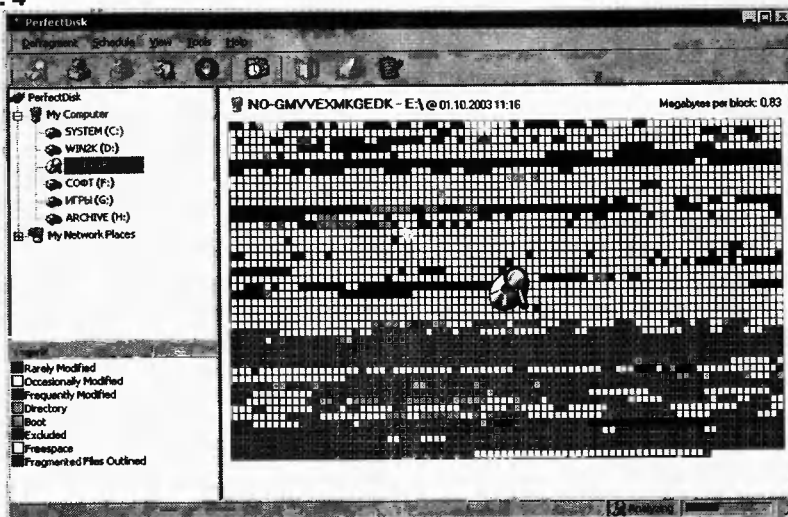
Perfect Disk 6.0.26

Утилита Perfect Disk разработана компанией Raxco Software (<http://www.raxco.com/>). Программа использует стандартное API от Microsoft, но имеет собственную графическую оболочку, работает только с операционными системами Windows 2000 (с установленным ServicePack 3 и выше) и Windows XP/2003, и одинаково успешно обрабатывает как FAT/FAT 32-разделы, так и NTFS. Среди дополнительных плюсов программы отмечается небольшое "отъедание" системных ресурсов при работе и возможность одновременной дефрагментации нескольких дисков.

Главное окно программы имеет интуитивно понятный графический интерфейс (рис.4), из которого доступны все команды управления работой Perfect Disk. С помощью кнопок верхнего меню или правым кликом мыши по значку соответствующего диска можно запустить анализ или сразу дефрагментацию. По завершении процедуры анализа тома программа выводит информацию о его состоянии и выдает рекомендации по использованию того или иного способа дефрагментации хранящихся на нем данных. При этом можно получить дополнительные сведения о количестве файлов на диске, наиболее фрагментированных файлах и так называемых "исключенных" файлах, т.е. тех файлах, доступ к которым при работе ОС класса Windows NT, как правило, запрещен.

Кроме обычной дефрагментации, Perfect Disk имеет еще и дополнительный режим оптимизации, использующий патентованную технологию Smart Placement, когда в процессе дефрагментации файлы сортируются по частоте использования, причем места их расположения можно определять вручную. Второй способ предполагает перемещение часто используемых данных в самое начало диска. PerfectDisk подвергает оптимизации и директории, убирая удаленные записи. Эта программа — одна из немногих, которые в оперативном режиме умеют дефрагментировать каталоги в NTFS-разделах (а в Windows XP/2003 — еще и MFT). Критические системные файлы могут приводиться в порядок, как обычно, во время загрузки системы. Но кроме того, существует еще и особый режим работы — Offline, когда программа на время дефрагментации блокирует эти файлы от любого постороннего воздействия, чем обеспечивается их дефрагментация без перезагрузки системы. Системным администраторам очень понравится возможность дефрагментировать сетевые диски, а также развитая система документирования всей проделанной Perfect Disk'ом работы.

Рис. 4



Как и в обеих вышеописанных программах, для автоматизации работы имеется не только запуск дефрагментатора по расписанию, но и интеллектуальный режим Perfect-Time, при включении которого программа автоматически запускается при достижении определенного уровня фрагментации файлов. Кроме того, предусмотрены возможность управления программой из командной строки и выполнение пакетных заданий, что делает ее просто чемпионом обзора в плане автоматизации управления программой. А для любителей досконально отслеживать все и вся, Perfect Disk может оказаться интересным в том плане, что он записывает в лог-файл время начала и окончания дефрагментации. Оригинальными также являются функции настройки через профили и поддержки RAID-массивов большого объема.

Личные впечатления от использования данного продукта, скорее, положительные, чем отрицательные. Работает Perfect Disk достаточно шустро, хотя

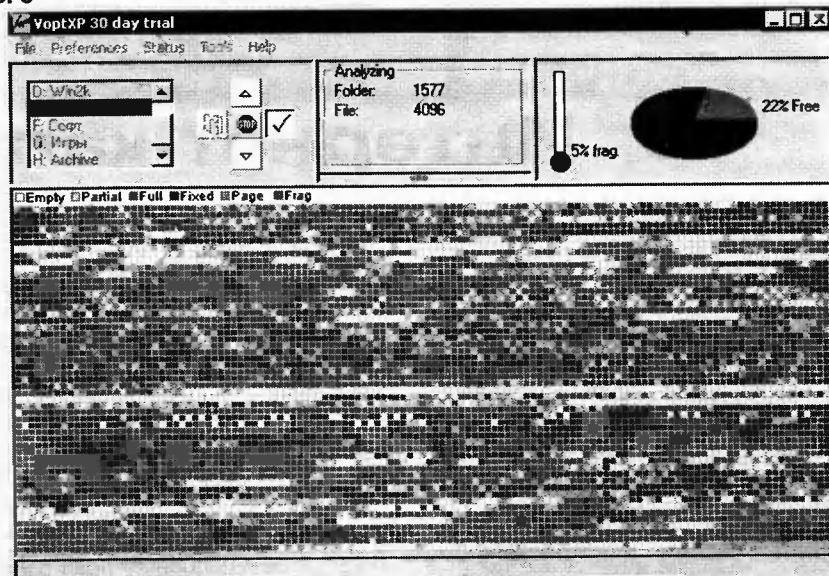
Diskeeper'у все-таки немного уступает. На нехватку свободного места особо не ругается, но, если его меньше 10...15%, то большие файлы остаются фрагментированными.

VoptXP 7.21

Последний дефрагментатор из рассмотренных в сегодняшнем обзоре — VoptXP, созданный компанией Golden Bow Systems (www.goldenbow.com). От подобных программ он отличается небольшими размерами и самой высокой скоростью работы. Программа также может проверить диск на наличие ошибок, обрабатывать тома в пакетном режиме, в том числе и по заданному пользователем расписанию; автоматически закрывает те работающие приложения, которые препятствуют процессу дефрагментации; защищает диск от потери данных, которая может произойти из-за различных сбоев во время дефрагментации.

Выбивается VoptXP из общего строя

Рис. 5





прочих дефрагментаторов, авторы которых почему-то обожают делать их интерфейс на базе стандартной, но очень уж примитивной консоли MMC, и внешне (рис.5). Причем, надо заметить, отличается от них в лучшую сторону. Небольшие, но толково продуманные рабочие окна программы позволяют разместить в них все необходимые элементы контроля и управления программой.

Хотя программа VoptXP не может похвастать наличием каких-либо “хитрых” алгоритмов дефрагментирования, тем не менее, одним из наиболее весомых ее достоинств является высокая эффективность при работе под управлением любой из версий Windows. Так, в Windows 9x/Me она дефрагментирует все файлы и папки, включая и файл подкачки. Такая же картина наблюдается и в Windows XP, как под FAT32, так и под NTFS (включая MFT). И лишь в Windows 2000 (FAT и NTFS) она дефрагментирует только файлы, а вот папки и MFT отказывается.

Программа VoptXP собирает и выдает пользователю достаточно подробную статистику по фрагментации файлов, возможен просмотр содержимого блоков. Кроме того, она предлагает некоторые дополнительные инструменты, которые могут заинтересовать некоторых пользователей. В их число входят чистильщик системы, мониторинг системы и даже простейший оптимизатор используемых ресурсов, позволяющий изменять размеры файла подкачки и кеша, а также настраивать (в определенных рамках, конечно) параметры интернет-соединения и даже дисплея. Но особое умиление вызывает наличие встроенного бенчмарка скорости чтения жесткого диска, который, по задумкам разработчиков, должен наглядно продемонстрировать пользователю немалый прирост производительности дисковой подсистемы после дефрагментации. Но его затрапезный вид и не внушающая большого доверия верность измерений — все

это лишний раз подчеркивает, что сапоги должен тачать сапожник, а вот дефрагментатор должен аккуратно собирать в единое целое разрозненные сегменты файлов, а не пытаться чистить диски или измерять их производительность. Все равно у специализированных программ это будет получаться гораздо лучше. Уж лучше бы разработчики программы больше внимания уделили планированию работ, который, кроме самого минимального набора базовых функций, похвастать ничем не может.

Подводим итоги

Итак, рассмотрев ряд программ-дефрагментаторов, можно констатировать отсутствие в этой области четко выраженного лидера — определенное “безвластие”. Без сомнения, наилучшим решением для однократной (или, по крайней мере, нерегулярной) дефрагментации, особенно для тех, у кого слабые машины с устаревшей ОС Windows 9x/Me, является VoptXP — быстрая, легкая, да и качество ее работы находится вполне на уровне конкурентов. А вот для всех тех, кто действительно хочет содержать свои разделы NTFS в полном порядке без всяких хлопот и головной боли, я бы все-таки рекомендовал поставить себе такого “тяжеловеса” как восьмая Diskkeeper, после установки запустить разок Boot-Time Defragmentation, а все остальное поручить умнице Smart Scheduling’у. И все будет, как говорится, “Set and forget”. Останется только одна забота — время от времени заглядывать в программу, чтобы полюбоваться, какие у вас разделы красивые и ухоженные, совсем как... (эпитеты можете вставить сами).

Что ж до двух других программ, O&O Defrag Pro и PerfectDisk, то обе они, при всех их неоспоримых достоинствах, все-таки проигрывают как Diskkeeper (по широте функциональных возможностей и удобству автоматизации процесса деф-

рагментации), так и VoptXP (в простоте использования и скорости работы).

Над первой из них можно немного похихикать, вспомнив, что своим появлением на свет она обязана религиозной нетерпимости немецкого правительства, которое почему-то совсем не любит сайентологов и все то, что они делают, включая Diskkeeper. Но, даже признавая за программой O&O Defrag отдельные, но от этого не менее весомые достоинства, не могу не отметить, что, предоставляя пользователю выбирать один из пяти мало отличающихся друг от друга режимов функционирования программы, ее создатели приложили максимум усилий для того, чтобы запустить любого непрофессионала. И они этого вполне добились. Тогда как вторая... PerfectDisk, несмотря на всю свою “сверхнавороченность” (причем большинство из этих “наворотов” никогда совершенно не понадобятся в повседневном обиходе), а также благодаря излишне яркому, грубоватому интерфейсу, почему-то оставляет впечатление программы просто “никакой”. Знаете, есть такая категория программ, которые не хочется ни ругать, ни хвалить. Поставил, посмотрел, удалил и забыл. Все.

И в качестве последней придирки ко всем трем “тяжелым” дефрагментаторам — Diskkeeper, O&O Defrag Pro и PerfectDisk — можно отметить тот факт, что все они после установки безо всяких предупреждений вносят свои службы в число системных сервисов. Причем по умолчанию все они запускаются автоматически, “отъедая” изрядный кусок и без этого не безграничных системных ресурсов, даже если дефрагментаторы будут использоваться исключительно в ручном режиме. Ну, а Disk Defragmenter — штатный “виндовский” дефрагментатор, он и есть штатный. Слаб, хил, но дело свое худо-бедно делает, притом бесплатно и всегда под рукой. Многих и такой вполне устроит.

С.РЮМИК,
г.Чернигов.

Интернет-калейдоскоп, freeware #5

*В улыбающееся лицо стрелу не пускают.
Японская пословица*

Хорошее настроение — это источник положительных эмоций, а добрая улыбка — генератор творческого настроения. Видимо, поэтому у программистов, как у людей творческой профессии, свое, оригинальное отношение к юмору. Выражается оно в трех ипостасях: графика, текст, звук.

“Здесь был Вася...”

Если представить себе Интернет в

виде большого города, то на его улицах будут встречаться здания с нарисованными на стенах разноцветными граффити. Слово “граффити” (от итальянского graffiti — нацарапанные) — древнеримского происхождения. Так называли бытовые, магические и обрядовые надписи, нацарапанные на стенах зданий и поверхностях домашней посуды. Как видно, стремление современных подростков к подобного рода самовы-

ражению — это не “ноу-хау” культуры “Хип-Хоп”, а наследственное...

В Интернете есть специальный сайт (<http://artcontext.org/glyphiti>), на котором любой желающий может “отвести душу” и оставить потомкам свои рисунки и надписи. “Царапанье” производится не тюбиком несмываемой краски по экрану монитора, а “мышью” в режиме реального времени. Каждому посетителю предлагается на одном

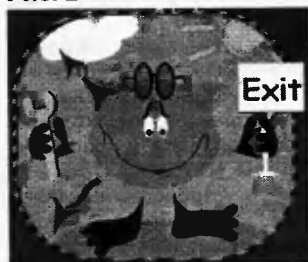
или нескольких холстах размером 32x32 элемента нарисовать любое изображение, зачерняя его квадратные клетки нажатием кнопки "мыши". Здесь можно симитировать надписи в шрифтах Bomb, FX, Wild Style, обычно применяемых профессиональными "граффитчиками". Для сохранения рисунка нажимают кнопку ADD в правом нижнем углу экрана.

Холсты разных авторов размещаются на одном большом поле и каждый месяц архивируются в свободно распространяемых zip-архивах. Пример интернационального творения за ноябрь 2003 г. приведен на **рис.1** (<http://www.artcontext.com/glyphiti/anim>).

Рис. 1



Рис. 2



Автор проекта Andy Desk (США) — дизайнер по интернет-арту. Получаемые рисунки он называет "глиффити" (glyphiti, от древнегреческого glyph — вырезающий). Такую технику, прообраз граффити, применяли еще майя и египтяне. Черно-белая раскраска глиффити, по мнению автора, олицетворяет борьбу двух стихий.

К сфере граффити можно отнести и назойливые баннеры на сайтах. По философскому смыслу многие из них мало чем отличаются от наскальных рисунков древней эпохи.

Необычные замысловатые фигуры, напоминающие граффити, можно найти в компьютерных программах. На **рис.2** показан внешний вид начального меню голосового ежедневника "ReadyVox" (<http://www.readyvox.pp.ru>).

Оригинальная улыбающаяся зверушка — не что иное как указатель выбора опций. Наведение курсора на рот зверька — функция "PLAY", на хвост — "Icon", на очки — "Database", на облако — "Exit" и т.д. Красивая и запоминающаяся идея!

Ключевые слова для поиска: graffiti, все о граффити (<http://www.to75.nm.ru>), glyphiti.

Афоризмы капитана Мерфи

Технический юмор отличается от бытового хотя бы тем, что его не все понимают. Программисты часто пользуются афоризмами (от греческого arhorismos — краткое изречение). По традиции к наиболее популярным из них относятся "Законы Мерфи". Появившиеся в 1949 г. с легкой руки (точнее, языка) американского инженера-исследователя Эдварда А.Мерфи (капитана военно-воздушных сил США), а затем литературно обработанные и дополненные Артуром Блохом, они и по сей день являются актуальными.

На **рис.3** приведен образец страницы иллюстрированного собрания "Все законы Мерфи" из файла-справки <http://merphy.kominet.ru/merphy.zip>, 1,8 Мб.

Из русскоязычных творений наиболее близки по стилю афоризмы Козьмы Пруtkова. В отличие от Мерфи, Козьма Пруtkов — вымышленный персонаж, к тому же, не пессимист, а

его нет. "Кому бы подарить нашего Гошу?" — спрашивают те, у кого он уже есть. Решить нелегкую проблему поможет виртуальный попугай, удобно устроившийся на жердочке внутри компьютерного монитора (**рис.4**). Программа "Vina the Digital Talking Parrot" ф.Аvnet Ltd (<http://www.voicechanger.biz/download/dtparrot.exe>, 1,8 Мб) позволяет научить виртуального попугая с именем Vina "человеческим" словечкам и коротким фразам. Обучать его будет сам хозяин, пользуясь микрофоном, подключенным к звуковой карте. Попугай исправно повторяет и запоминает услышанные фразы, а затем в случайном порядке их выкрикивает.

С физической точки зрения, слова, сказанные человеком перед микрофоном, преобразуются в цифровую форму, а затем преднамеренно искажаются по трем параметрам: Pitch level, Vibration period, Vibration level. Степень искажения можно регулировать вручную, задавая тот или иной уровень "скрипучести" голоса.

Минимальные требования к компьютеру: Pentium 200, Windows 9x/2000, DirectX 7.0. Хорошее качество звука напрямую связано с техническими параметрами микрофона и степенью экранирования его проводов. Виртуальный попугай может использоваться не только на вечеринках в качестве забавной программы, но и как хранитель экрана (screen saver).

В Интернете программу "Digital Talking Parrot" иногда называют "эмулятором попугая", что по смыслу неверно, ведь эмулировать можно однородные (а не разнородные) по материалам системы [1].

Ключевые слова для поиска: виртуальный попугай, синтез речи, Digital Talking Parrot.

Литература

1. Рюмик С. Интернет-калейдоскоп, freeware #4. — Радиомир. Ваш компьютер, 2004, №3, С.18.

Рис. 4



оптимист. Это коллективный псевдоним поэта А.К.Толстого и братьев Жемчужниковых середины XIX века. С полным сборником трудов можно ознакомиться в файле <http://lib.ru/LITRA/PRUTKOW/litografik.txt>, куда вошли знаменитые высказывания: "Смотри в корень!", "Никто не обнимет необъятного", "Принимаясь за дело, соберись с духом" и др.

Ключевые слова для поиска: законы Мерфи, компьютерные анекдоты (<http://416.412.ru/humor>).

Имитатор голоса попугая

"Хорошо бы завести в доме говорящего попугая" — мечтают те, у кого



ЛИСТАЯ СТРАНИЦЫ

Перспективам использования 64-разрядных процессоров в настольных ПК и результатам их тестирования посвящена статья "64 бит AMD против 32 бит Intel" (CHIP, №12/2003, С.28...39).

Долгие годы AMD разрабатывала свой Hammer — 64-битный процессор. В основе его лежала мысль создать серверный процессор, способный, а отличие от интеловского Itanium, функционировать на 32-битных десктопах. В результате сегодня мы имеем Athlon 64, ранее известный под "псевдонимом" Clawhammer. Согласно обещаниям AMD, Athlon 64 должен был еще год назад оказаться на прилавках магазинов. Однако его выход протормозился из-за неплохих продаж процессоров Athlon XP, а также из-за сырого состояния 64-битной операционной системы. Теперь, когда Athlon XP на базе ядра Barton приблизился к пределу своих возможностей, самое время выпустить в бой 64-разрядный процессор.

Если сравнивать старый 16-битный центральный процессор с граммофоном, то 32-битный процессор — это современный проигрыватель виниловых пластинок, а 64-битный — уже CD-плеер. Правда, тут есть маленькая неувязка: процессор в состоянии использовать свою 64-битную мощь лишь в том случае, если вся система в целом поддерживает 64-разрядную шину. В противном случае толку от магической цифры 64 будет мало.

Что же получит пользователь, когда 64-битная поддержка будет реализована в полном объеме? Первое преимущество заключается в увеличенной емкости адресуемой памяти. Это значит, что если 32-битная система может работать с оперативной памятью объемом 4 Гб, то 64-разрядный компьютер теоретически в состоянии оперировать 16 эксабайтами оперативной памяти (примерно 16 800 000 000 Гб). Второе преимущество компьютера с 64-битным процессором заключается в потрясающей точности производимых вычислений. Он примерно в 4 млрд. раз точнее, чем 32-разрядный процессор, производит вычисления. По словам представителей AMD, такую точность оценят прежде всего программисты, создающие игры. Она позволит им со стопроцентной достоверностью воссоздавать абсолютно все детали окружающей обстановки, повышать уровень интеллекта врагов. При работе с мультимедийными программами 64 бита значительно увеличат скорость обработки данных. Ближайшее будущее передовой технологии заключается в разделении 64-разрядного ядра на два 32-разрядных, что позволит осуществлять одновременно два разных вычисления.

А что же Intel? Когда-нибудь и она явится на выводе на рынок десктопных процессоров 64-битной технологии, пока же в ее официальных планах подобный продукт отсутствует. Технология

HyperThreading и внутренняя структура процессоров Pentium 4, позволяющая безболезненно наращивать тактовую частоту, дают Intel возможность спокойно взирать на суету конкурентов вокруг 64-разрядных процессоров десктопной версии.

Однако и AMD, и Apple, недавно представившая свой новый ПК Power Mac G5 на основе 64-битного процессора Power4 от IBM, настолько предусмотрительны, что продолжают и в новых процессорах осуществлять поддержку 32-битных программ. При этом учитывается естественное желание пользователей, выкладывающих немалые деньги за новинку, сразу же получить заметный прирост производительности а тех же 32-битных программах. Для этого AMD и Apple дополнительно используют хорошо зарекомендовавшие себя методы ускорения процессоров — повышение тактовой частоты с 1250 до 1600...2000 МГц (Apple) и удаение кэша L2 с 512 до 1024 кб (AMD). Кроме того, 64-битный Athlon унаследовал интеловское расширение инструкций SSE2.

Этот шаг AMD очень много значит как для всего Windows-сообщества, так и для самой AMD. Уже много месяцев особо пессимистичные знатоки компьютерного мира в один голос твердят, что Hammer является для нее последним шансом в конкурентной борьбе с Intel. Прежняя плеяда процессоров AMD выигрывала в основном за счет отличного соотношения цены и производительности, а вот рекорды по-прежнему бьет Pentium 4 с тактовой частотой от 3 ГГц. Последней в ряду процессоров семейства K7 была модель с ядром Barton. С технологической точки зрения она полностью исчерпала себя.

Это значит, что наступило время для ввода в бой восьмого поколения процессоров под индексом K8. Так AMD обозначает целое семейство процессоров: это серверный процессор Opteron, а также процессоры Athlon 64 и 64 FX для настольных ПК. Но и в этих процессорах увеличение тактовой частоты ожидается лишь в том случае, если в 2004 г. произойдет переход технологического процесса со 130 нм на 90 нм. Даже новый Athlon FX будет иметь поначалу рабочую частоту 2200 МГц. Новые рубежи производительности AMD намеревается завоевывать иными путями. Во-первых, как уже отмечалось, увеличением объема кэша L2 и, во-вторых, благодаря интеграции контроллера памяти непосредственно в процессор. Это позволило уменьшить время обмена данными между процессором и памятью, и, как следствие, повысилась пропускная способность памяти. Другим положительным результатом данной меры стало повышение совместимости с чипсетами сторонних производителей и, соответственно, стабильности работы системы.

Еще одна "изюминка" нового процессора заключается в использовании технологии HyperTransport с пропускной способностью 6,4 Гб/с, позволившей расширить "игольное ушко" канала обмена данными между процессором и всей осталь-

ной системой. Она уже применялась в процессорах Athlon XP, но функционировала только при использовании чипсетов NVIDIA, так как только NVIDIA интегрировала в северный мост своих чипсетов поддержку этой шины.

И все же самое главное, по мнению автора, в переходе AMD на 64-битную технологию — это ее постепенное внедрение. Это означает, что новые процессоры Athlon 64 прекрасно работают и на привычной 32-битной платформе. Сейчас 64-битные процессоры не являются чем-то принципиально новым, просто до сегодняшнего дня они применялись исключительно в серверах. Такие процессоры как, например, интеловский Itanium работают исключительно с 64-битными программами. Это не позволяет надеяться на возможность его использования в домашних условиях. Совместимость с "нижестоящими" программами является непременным условием успеха на рынке массового использования, поскольку резкий переход на 64-битные приложения увеличит расходы потенциальных покупателей.

Некоторые производители уже высказались в поддержку 64-битных систем, однако следует заметить, что они не торопятся с выпуском для них соответствующего софта. Да и операционные системы пока что нельзя назвать готовыми к работе в новых условиях. К моменту появления процессоров Athlon 64 была готова лишь Suse-Linux версии 8.3, 64-битная Linux от Red Hat ожидается несколько позднее. Компания Microsoft, как ожидалось, планировала завершить работу над Windows XP Professional 64-Bit Edition к концу 2003 года.

Однако дело не ограничивается одной лишь операционной системой. Чтобы использовать потенциал нового процессора на все сто процентов, необходимо, чтобы с ним "сотрудничали" и программные интерфейсы (DirectX, OpenGL, TWIN), драйверы устройств и прикладные программы. Таким образом, воплощение мечты о всеобщей "64-битизации" компьютерного мира пока остается в отдаленном будущем.

AMD надеется, что эру процессоров Athlon 64 поддержат а первую очередь игроки, а потом и производители программ. Это придаст существенный импульс процессу завоевания рынка, который должен начаться с 2005 года. Вспомним, однако, недавнее прошлое персональных компьютеров. В 1985-м Intel выбросила на рынок, где царили 16-битные процессоры, свой первый 32-разрядный процессор i386. После этого прошло долгих 10 лет, пока Windows 95 и OS/2 не сошлись до того уровня, чтобы выйти на прилавки магазинов. Разумеется, в этот раз не придется ждать так долго. Специалисты из AMD полагают, что было бы неправильным считать выход процессоров Athlon 64 на пользовательский рынок преждевременным. Рано или поздно этот шаг должен был кто-то сделать. А пока новому Athlon предстоит зарекомендовать себя на 32-битном "фронте".



Кроме того, еще одна цель создания семейства Athlon 64 заключается в нападении на позиции Intel, закрепившейся на новом «оборонительном рубеже» под названием Prescott и технологии Hyper-Threading. Атака должна будет вестись по двум направлениям: цена и производительность.

В редакции журнала «CHIP» был опробован компьютер на базе процессора Athlon 64 FX-51. В отличие от младшей модели, процессор с индексом FX работает с двухканальной DDR-памятью на частотах до 400 МГц. Кроме того, Athlon 64 FX не располагает пока собственными материнскими платами, что заставляет использовать платы для Opteron с разъемом Socket 940. Недостаток очевиден: эти материнские платы, изготавливаемые специально для серверных станций, имеют более дорогую шестислойную топологию, в то время как обычные платы обходятся четырьмя слоями. Кроме того, им требуется регистровая память (или память с буферизацией). Она позволяет повысить стабильность системы и количество применяемых модулей оперативной памяти. При этом следует упомянуть о том, что регистровые модули DDR400 пока еще не имеют официальной спецификации. Все это не способствует удешевлению системы.

Примерно через полгода Athlon 64 FX должен получить собственную платформу с 939-контактным исполнением, четырехслойной материнской платой и небуферизованной памятью. В качестве дополнителной «конфетки», привлекающей пользователей, она будет иметь функцию экономии энергии под названием Cool'n'Quiet. Как и в мобильных процессорах, тактовая частота автоматически уменьшается, если процессор не загру-

функциональности от 64-битного Athlon. И для G5 64 бита означают высокую точность вычислений, более быструю обработку чисел с плавающей точкой, а также более высокий объем адресуемой оперативной памяти. Современные модели Apple, напоминает автор, в состоянии работать только с 8 Гб ОЗУ. В них работа выполняется двумя процессорами, соединенными с периферийными устройствами системными шинами, работающими на частотах 1 ГГц.

Дизайн нового Mac потребовался и новый дизайн корпуса, который изготовлен из алюминия и имеет в общей сложности девять вентиляторов. Семь из них вращаются очень медленно и поэтому практически не слышны. Кроме того, компьютер разделен на четыре тепловые зоны, которые контролируются операционной системой по отдельности.

Прозрачное окно из акрила под правой боковой крышкой даже при открытом корпусе обеспечивает регулировку движения воздушного потока. Громкость работы компьютера Power Mac на «холостом ходу» составила 1,4 сон. Это значит, что в этом режиме он работает почти неслышно. Однако когда процессор загружен до предела, ситуация может резко перемениться. Если в этот момент снять пластиковую крышку, компьютер начинает реветь как пылесос, производя шум 8,5 сон. По всей вероятности, это послужило причиной задержки выхода G5 на рынок. По некоторым данным, проблема регулировки теплоотвода доставила инженерам Apple немало хлопот.

Чтобы корректно сравнивать компьютеры с разными операционными системами, на них запускались программы, являющиеся стандартом для обеих платформ. В целом Athlon 64 FX-51 показал хорошие результаты. В тестах,

связанных с проверкой работы оперативной памяти (табл.1), он обошел своих конкурентов во главе с Pentium 4C 3200 МГц (FSB 800 МГц) со счетом 1665:1188 Мб/сек и произвел хорошее впечатление практически во всех игровых тестах. Athlon 64 FX-51 опередил почти на 9% «коллег» из Intel и почти на 15% — Athlon XP 3200+, который работал на той же тактовой частоте. Между тем Pentium 4 опередил соперника «на своем поле» — в Cinebench 2003, в котором он, благодаря Hyper-Threading, выигрывает от поддержки программами мультипроцессорного режима. Вместе с тем, типично мультимедийные расчеты, такие как перекодирование в MP3 или DivX, Pentium 4 производил достаточно быстро. И хотя «мотор» Athlon с его кэшем L2 объемом в 1024 кб выглядит достаточно внушительно, 2,2 ГГц тактовой частоты — это очень малая мощность по сравнению с Pentium 4. Однако в Adobe Photoshop 7.01 Athlon 64 FX опять выглядел весьма достойно на фоне Pentium 4 (табл.2), который по всем прогнозам должен был продемонстрировать явное преимущество.

При работе с Photoshop, G5, несмотря на специально для него выпущенное обновление от Adobe, выглядел не совсем убедительно. А вот при преобразовании файла изображения размером 217 Мб из CMYK в RGB он значительно опередил Athlon (табл.3). Однако при обработке изображения с помощью фильтра Гаусса они показали одинаковый результат. При использовании фильтров искажений Twirl и Chrome процессор Athlon был значительно быстрее. Возможно, еще не все фильтры оптимизированы к возможностям G5.

Для обработки видеоизображения (конвертирование сцены фильма в фор-

Табл. 1

Модель	Тест				
	Пропускная способность памяти (CHIP-Bench32), Мб/сек	Игровая производительность, %	Cinebench 2003 CPU-Rendering, сек	Конвертирование формата WAV в MP3 (альбом 78 мин), мин:сек	Кодирование DVD в DivX (фильм 90 мин), мин:сек
Athlon XP 3200+	760	100	95,2	5:26	46:52
Pentium 4C 3200	1188	106	69,6	5:42	43:03
Athlon 64 FX-51	1665	115	85,6	6:01	45:52

жен полностью. Это позволяет ему в обычном режиме работы меньше нагреваться, а вентилятору — работать тише. Эта идея назрела давно, и наконец-то хоть один крупный производитель осуществил ее на практике. Теперь дело за малым: поддержка функции Cool'n'Quiet должна быть реализована в BIOS материнских плат.

В тестировании также участвовали Apple G5 1600 МГц, Apple G4, имеющий два 32-разрядных процессора на 1 ГГц, и компьютеры на базе 32-разрядных Pentium 4C 3200 МГц и Athlon XP 3200+. Что касается компьютеров Apple, то G5 представляет собой новое поколение компьютеров этой фирмы. Процессор этого ПК не очень отличается по своей

Табл. 2

Модель	Тест		
	Unreal Tournament 2003, кадров/сек	Cinebench 2003 CPU-Rendering, сек	Photoshop, сек
Athlon 64 FX-51	93	85,6	53,6
Pentium 4C 3200	77	69,6	67,2
Apple G5 1600	25	154	95,9

Табл. 3

Модель	Тест				
	Конвертирование CMYK в RGB, сек	Gaussian Blur, сек	Twirl, сек	Chrome, сек	Photoshop (всего), сек
Apple G4	11	33	7	48	99
Apple G5	9	23	32	32	96
Athlon 64 FX-51	13	23	5	12	53



Табл. 4

Модель	Тест			
	Cinebench 2003 CPU-Rendering, сек	Конвертирование формата WAV в MP3 (альбом 70 мин), мин:сек	Кодирование AVI в DivX (файл 50 Мб), мин:сек	Unreal 2003 Botmatch, кадров/сек
Apple G4	149	4:00	5:00	20
Apple G5	155	4:36	4:00	23
Athlon 64 FX-51	86	5:40	1:12	77

Следует признать, что и новый процессор AMD способен пробить заметную брешь в семейном бюджете. Успехи AMD в игровых тестах не повод для радости, так как весьма рискованно позиционировать процессор стоимостью 730 USD в качестве только игрового. Сегодня AMD самоуверенно заявляет, что Athlon 64 превзойдет по своей производительности интеловский Prescott. С нынешними результатами в это как-то слабо верится. Тем не менее, AMD имеет в своем распоряжении до выхода в свет Prescott около 6 месяцев форы, которые должны быть использованы для перехода с 0,13 на 0,09 мкм техпроцесс.

мат DivX, табл.4) Athlon понадобилось 1 мин 12 с, в то время как G5 был почти на 3 минуты медленнее, а G4 — еще на 1 минуту медленнее. Игровая производительность была оценена с помощью демо из Unreal Tournament 2003. Athlon показал результат 77 кадров/с, а G5 — всего 23. Светлым пятном для Apple явился результат кодирования 70-минутного WAV-файла в MP3, на которое G5 затратил 4 мин 36 с, а G4 — 4 мин. Athlon работал 5 мин 40 с.

Подводя итоги, автор отмечает, что звание самого быстрого G5 пока не заслужил, хотя цена такой игрушки сравнима со стоимостью автомобиля. Справедливее было бы сказать "временами более быстрый, чем G4". Конечно, это "первенец" Apple, и ждать от него слишком много было бы наивным. Тем более, что для достижения наилучших результатов ПО должно быть оптимизировано.

Для охлаждения самого горячего элемента — процессора — здесь используется бесшумный кулер фирмы Zalman. Его основа выполнена целиком из меди, обладающей вдвое более высокой теплопроводностью, чем традиционный алюминий. Над радиатором расположен аэрилятор большого диаметра, который, в отличие от традиционной схемы, работает на отток воздуха от процессора. Через специальный пластиковый короб воздушный поток подается к корпусному вентилятору, который и завершает вынос нагретого воздуха из корпуса ПК. Обороты корпусного вентилятора (диаметром 110 мм против обычных 80) регулируются трехпозиционным переключателем, способным поднимать звуковую тревогу в случае останова.



По мере совершенствования вычислительной техники все больше внимания уделяется повышению комфортности работы с ПК, и в частности, снижению его шумности. Сильно шумящий компьютер быстро утомляет пользователя. Поэтому разработчики стали уделять снижению шума от ПК достаточно много внимания, примером чего является компьютер Team Office b712:SE, о котором рассказывает А.Дмитриев в заметке "Тишина от Team Computers" (Мир ПК, №11/2003, С.35).

Team Office b712:SE построен на базе процессора Pentium 4 с тактовой частотой 2,66 ГГц, оснащен видеоадаптером ATI Radeon 9700, ОЗУ емкостью 512 Мб типа DDR и жестким диском 60 Гб. Измерения показали, что уровень шума этой модели составляет всего 27 дБ при фоне 24 дБ. А это означает, что даже в момент включения ПК работает настолько тихо, что услышать его практически невозможно.

Известно, что основной источник шума — системы охлаждения компонентов ПК.



О новом формфакторе материнских плат, предложенном ф. Intel, рассказывает С.Пахомов в статье "Новый формфактор ВТХ" (КомпьютерПресс, №12/2003, С.160...161).

Понятие формфактора материнской платы появилось на заре развития персональных компьютеров, когда в 1981 г. компания IBM выпустила первый ПК, заложив фундамент для развития IBM PC-совместимых компьютеров. Все IBM PC-совместимые компьютеры имеют открытую архитектуру, то есть предполагают использование комплектующих различных производителей с полной гарантией их совместимости. Но для обеспечения такой совместимости необходимо, чтобы все производители следовали одним и тем же правилам при разработке комплектующих. Именно эти правила и формируют понятие формфактора материнской платы.

Первый формфактор, как стандарт, разработанный в 1983 г. компанией IBM,

получил наименование XT (eXtended Technology) и подразумевал некоторые конструктивные изменения материнских плат (по сравнению с первым дизайном платы). В частности, увеличилось количество слотов, изменилось расстояние между ними и т.д.

В 1984 г. на смену формфактору XT пришел стандарт AT (Advanced Technology), где определялись размеры материнских плат 30,5х33 см, которые устанавливались в корпус типа Desktop и Tower.

Продолжением AT стал стандарт Baby AT, принятый в 1990 г. Этот формфактор предусматривал уменьшение размеров плат (22,5х33 см) и обладал обратной совместимостью с AT.

В середине 1995 г. компания Intel предложила новый формфактор — ATX (Advanced Technology eXtended). В новом стандарте, в частности, определялось, что все разъемы портов ввода-вывода должны быть расположены в левом верхнем

углу платы. Изменилось и расположение процессорного гнезда и слотов памяти. Кроме того, в платах ATX изменился разъем для подключения блока питания. Размеры плат формфактора ATX составляли 30,5х24,4 см.

В 1997 г. ATX получил свое логическое развитие в виде формфактора microATX с размером платы 24,4х24,4 см, а в 1999-м появился стандарт FlexATX с размером платы 22,9х20,3 см.

И вот теперь пришло время нового формфактора — ВТХ (Balanced Technology eXtended), ранее известного под кодовым названием Big Water. Необходимость перехода к новому стандарту вызвана прежде всего появлением новых шин (USB 2.0, SATA, PCI Express), а также изменившимися требованиями к энергопотреблению ПК (а следовательно, и к теплоотводу) и к акустическим характеристикам ПК. Причем формфактор ВТХ обратно совместим с ATX, то есть плату с формфактором ВТХ можно



будет устанавливать и в старый корпус стандарта ATX, хотя обратное невозможно.

Среди основных преимуществ нового формфактора можно назвать поддержку низкопрофильных устройств для создания компактных ПК, расположение компонентов платы с учетом обеспечения термобаланса, продуманный дизайн для максимально эффективного теплоотвода и оптимизированную структуру крепления плат.

Стандарт BTX существует в трех вариантах — собственно BTX, microBTX и picoBTX. Платы формфактора BTX имеют размер по глубине 325,12 мм, допускают наличие семи слотов и десяти отверстий для монтажа платы к шасси. Такие платы используются в корпусах объемом 15 л и более.

Формфактор microBTX характеризует платы, которые обладают размером по глубине 264,16 мм, допускают наличие четырех слотов и семи отверстий для монтажа платы к шасси. Указанные платы применяются в корпусах объемом от 10 до 15 л.

Платы формфактора picoBTX имеют размер по глубине 203,20 мм, они допускают наличие всего одного слота и четы-

рех отверстий для монтажа платы к шасси и используются в корпусах объемом от 6 до 10 л.

Платы стандартов BTX, microBTX и picoBTX имеют иное, по сравнению с платами стандарта ATX, расположение основных компонентов — слотов для установки модулей памяти, процессорного гнезда, VRM-модуля, северного и южного мостов чипсета и т.д. Такие серьезные изменения были необходимы для создания термобаланса. С помощью всего одного вентилятора удается осуществлять теплоотвод и от VRM-модуля, и от процессора, и от графической карты, и от модулей памяти.

Как известно, чем больше вентиляторов устанавливается внутри корпуса, тем более "шумную" систему мы получаем. Для плат с формфактором BTX предполагается использование всего одного вентилятора. Конструкция самого вентилятора (то есть количество лопастей и кривизна их изгиба) претерпела существенные изменения, а кроме того, стала иной и конструкция радиатора процессора, который теперь установлен в специальный направляющий патрубок. В

спецификации BTX система охлаждения называется модулем термического баланса (Thermal Module). Под этим термином подразумевается комплект из системы охлаждения процессора и направляющего патрубка для оптимизации потоков воздуха внутри корпуса. В настоящий момент разработаны два типа таких модулей — полноразмерный и низкопрофильный.

Платы стандартов BTX, microBTX и picoBTX крепятся к корпусу несколько иначе, чем платы стандарта ATX. Во-первых, если формфактором ATX предусмотрено минимальное расстояние между шасси корпуса и материнской платой 0,25 дюйма, то в стандарте BTX это расстояние увеличено до 0,4 дюйма. Крепление плат формфактора BTX к шасси производится с помощью модуля SRM (Support and Retention Module). Сам SRM-модуль обеспечивает поддержку оптимального режима охлаждения компонентов системы, особенно процессора, а кроме того, предотвращает возможный изгиб материнской платы. Переход к формфактору BTX произойдет в течение 2004 года.

Одним из основных недостатков струйных принтеров всегда была плохая стойкость отпечатков. Однако в **новых принтерах Stylus C84 и C63 ф. Epson**, о которых в статье "Несмазливый чернила" (**Мир ПК**, №12/2003, С.36) рассказывает В.Дмитриев, используются свето- и влагостойкие чернила DURABrite, для которых гарантируется светостойкость 80 лет!

Принтер C84 (цена 149 USD) продемонстрировал весьма высокое качество фотывывода, и это не удивительно при решении 5760x1440 dpi. По качеству печати фотографий C84 не уступает лучшим цветным лазерным принтерам, вплотную приближаясь к фотопринтерам. Точная цветопередача и отсутствие заметных изъянов при передаче естественных оттенков на портрете и пейзаже, особенно при печати на специальной фотобумаге

DURABrite, позволяют рекомендовать эту модель достаточно требовательным пользователям.

Цветной документ формата A4 принтер напечатал за 59 с, а фотографию размером 10x15 см в режиме наилучшего качества — за 1 мин 56 с. По спецификации скорость печати составляет до 22 стр./мин в черно-белом режиме и до 12 стр./мин в цвете. Стоимость одного черно-белого текстового отпечатка на листе формата A4 составляет 4 цента, если за основу взять рекомендованные в России цены.

Из недостатков изделия можно отметить сильную вибрацию при печати в черновом режиме и появление розоватого оттенка при выводе градиций серого. В целом Stylus C84 произвел очень благоприятное впечатление, и может стать отличным выбором в качестве офисной модели, если покупателя не смутит цена в 150 USD.

ки системы и файл-менеджер DOS Navigator, разработанный в свое время хорошо известной компанией Rittlabs, а ныне развиваемый как проект с открытым исходным кодом.

По мнению автора, у этой разработки есть будущее, и многим, возможно, еще предстоит воспользоваться этой ОС. Преимуществами PTS-DOS перед другими подобными системами, в первую очередь перед MS-DOS, являются меньшая загруженность системных ресурсов и, следовательно, несколько большая скорость выполнения DOS-приложений. Поскольку и сейчас используется много вычислительных программ для DOS, которые требуют серьезных аппаратных ресурсов, то система PTS-DOS 32, поддерживающая работу с большими объемами оперативной памяти, может найти свое место во многих научных лабораториях. Другой аргумент в пользу PTS-DOS — ее невысокая цена: OEM-версия стоит около 5 USD.

Вторая модель, Stylus C63, существенно отличается от C84 по скорости и качеству печати. Цветной отпечаток текста A4 был выдан через 1 мин 48 с, а фото 10x15 см — через 3 мин 26 с. По спецификации производитель обещает до 17 стр./мин в черно-белом режиме и до 9 стр./мин в цвете.

Изображение в черновом режиме у Stylus C63 выглядело более бледным, чем хотелось бы. Кроме того, на отпечатках фотографий, полученных на C63, был заметнее растр, а цветовая гамма — менее насыщенная. Зато градиент серого в режиме "фото" у него оказался идеально равномерным.

Оба принтера подключаются к компьютеру как через LPT-порт, так и через USB, и имеют драйверы со стандартным набором настроек для данной категории.

Кто бы мог подумать, но оказывается, **ДОС еще жива и даже развивается**. Об этом рассказывает С.Полтеев в статье "PTS-DOS 32: есть ли будущее у прошлого?" (**Мир ПК**, №12/2003, С.69).

Компания "Физтех-софт" выпустила новый релиз хорошо известной в свое время системы PTS-DOS. Как и прежде, все файлы PTS-DOS помещаются на одной дискете, но поставляются они уже на компакт-диске вместе с несколькими дополнительными программами. Установка системы, как и раньше, не вызывает никаких проблем. Не произошло никаких существенных изменений в интерфейсе — это командная строка. По словам разработчиков, изменения в ядре системы также минимальны. Главные отличия PTS-DOS 32 от более ранних версий — поддержка жестких дисков большой емкости (до 100 Гб) и больших объемов оперативной памяти (до 4 Гб). Кроме того, в комплект входят набор утилит для диагности-

При покупке готового системного блока мы так или иначе оплачиваем стоимость предустановленной на нем ОС; как правило, это бывает какая-либо из версий MS Windows. Таким образом, приобретение компьютера с предустановленной PTS-DOS позволяет сэкономить определенную сумму и в дальнейшем предоставляет свободу при выборе наиболее подходящей ОС. Кроме того, загрузочный диск с PTS-DOS вместе с набором утилит для диагностики может пригодиться и как средство для аварийной загрузки при возникновении каких-либо сбоев. Впрочем, на этот случай лучше иметь образ диска, с которого можно довольно быстро восстановить информацию.

Конечно же, PTS-DOS 32 вряд ли войдет в число лидеров продаж, подчеркивает автор, но и исчезновение с рынка этой системы не грозит. А в будущем разработчики PTS-DOS, возможно, порадуют нас новыми интересными решениями.



COM — это не сон

А.ИВАНЧИКОВ,
E-mail: iaa@rbcmail.ru

Данная статья посвящена основам создания программ в соответствии с технологией COM. На конкретном примере — разработке интерфейса и реализующего его на языке C++ компонента, обеспечивающих необходимую функциональность для программирования карточных игр, подробно описаны основные проблемы компонентной архитектуры. Цель публикации — познакомить читателей с современной технологией разработки приложений, определить, реализовать и пояснить основные этапы работы в рамках создания конкретного двоичного компонента. Сам создаваемый компонент может быть полезен для разработки широкого спектра приложений игрового характера.

Компоненты

Традиционно приложение состоит из отдельных файлов, модулей или классов, которые компилируются и компоноуются как единое целое. После того как приложение собрано, оно остается неизменным — пока не будет собрана и поставлена пользователю новая версия. При современных темпах развития индустрии программирования приложениям нельзя оставаться застывшими. Решение состоит в том, чтобы разбить монолитное приложение на отдельные части — *компоненты*. Приложения, построенные по такому принципу, постепенно эволюционируют с заменой старых компонентов новыми. Кроме того, из отдельных существующих уже компонентов легко создавать и абсолютно новые приложения.

Преимущества использования компонентов непосредственно вытекают из способности последних подключаться к приложению и отключаться от него. Для этого компоненты должны удовлетворять двум требованиям. Во-первых, они должны удовлетворять динамически. Во-вторых, они должны скрывать детали своей реализации [1]. Действительно, чтобы обеспечить функционирование конечного приложения, использующего компоненты (такие приложения называются *клиентами*), необходимо иметь возможность замены компонентов во время выполнения (без перекомпиляции) клиента, что и требует динамической компоновки. С другой стороны, новый компонент должен подсоединяться к клиенту (компонент подсоединяется к клиенту посредством *интерфейса*) тем же способом, что и старый. Для этого компоненты и клиенты должны стараться не изменять свои интерфейсы. Это возможно только в том случае, когда детали реализации клиента и компонента не отражаются в интерфейсах, т.е. интерфейсы являются инкапсулирующими.

Необходимость изоляции клиента от деталей реализации накладывает на компоненты ряд важных ограничений:

- компонент должен скрывать используемый язык программирования. Любой клиент, выполненный на любом языке, должен иметь возможность использовать компонент, независимо от языка программирования, на котором данный компонент реализован;

- компоненты должны распространяться в двоичном формате. Действительно, поскольку они должны скрывать язык реализации, их необходимо поставлять уже скомпилированными и скомпонованными;

- должна быть возможность модернизировать компоненты, не затрагивая уже существующих пользователей. Новые версии компонента должны работать как с новыми, так и со старыми клиентами;

- необходимо, чтобы компонент и использующая его программа могли выполняться внутри одного процесса, в разных процессах или на разных машинах. Клиент должен рассматривать удаленный компонент так же, как локальный.

Все эти ограничения удовлетворяют компоненты, выполненные по спецификации COM.

COM как аббревиатура

COM (Component Object Model — компонентная объектная модель) — это спецификация создания компонентов и построения из них приложений. Спецификация описывает метод разработки программных компонентов (двоичных исполняемых файлов), которые предоставляют необходимые сервисы приложениям, операционным системам и другим компонентам. COM является основой, на которой построены такие технологии Microsoft как ActiveX, DirectX и OLE. Хотя COM — это двоичный стандарт интерфейса объектов в Windows, можно использовать подход COM в любой операционной системе. Разумеется, Windows предоставляет код, который облегчает программирование в духе COM, однако основную часть этого кода нетрудно реализовать под любой операционной системой.

Компоненты COM состоят из исполняемого кода, распространяемого в виде динамически компокуемых библиотек (DLL) или EXE-файлов Win32.

Описание проектируемого компонента

Обсуждение компонентной архитектуры построения приложений будем проводить параллельно с разработкой некоторого компонента. Данный компонент будет представлять собой удобную в использовании оболочку системной динамической библиотеки cards.dll.

В семействе ОС Windows 95/98/ME cards.dll является 16-разрядной, а начиная с Windows NT cards.dll — это 32-разрядная библиотека. Проектируемый компонент будет работать только с 32-разрядной библиотекой. Успешная работа компонента под управлением операционных систем Windows 95/98/ME может быть обеспечена либо установкой 32-разрядной версии библиотеки cards32.dll (на ее распространение, однако, необходима соответствующая лицензия от Microsoft), либо использованием технологии Flat Thunking по совмещению 32- и 16-разрядного кода [2, 3].

Библиотека cards.dll экспортирует пять функций.

1. Функция

BOOL WINAPI cdtInit (int *width, int *height)

Инициализирует библиотеку. По адресам width и height сохраняются ширина и высота карт, которые будут изображаться в окне приложения.

2. Функция

BOOL WINAPI cdtDraw (HDC hdc, int x, int y,
int card, int type, DWORD color)

Отображает карту card в позиции x, y на устройстве с контекстом hdc. Параметр type управляет выводом карты либо лицевой стороной, либо рубашкой, либо инвертированным изображением лицевой стороны. Параметр color определяет цвет рубашки для типа рубашки с номером 53 ("косая сетка"). Все остальные рубашки являются растровыми рисунками и не могут изменить свой цвет. Параметр color также влияет на цвет инвертированного изображения карты.

3. Функция

BOOL WINAPI cdtDrawExt (HDC hdc, int x, int y,
int dx, int dy, int card, int suit,
DWORD color)

Подобна cdtDraw, однако параметрами dx и dy можно



```
// завершение работы с библиотекой
```

```
typedef BOOL (WINAPI* LPCDTERM)();
```

```
// имена библиотек с "карточными" функциями:
// для 32-разрядных Win
#define CARDS_DLL_NAME "cards.dll"
// для 16-разрядных Win
#define CARDS32_DLL_NAME "cards32.dll"
```

```
// имена экспортируемых функций
#define CDTINIT_NAME      "cdtInit"
#define CDTDRAW_NAME      "cdtDraw"
#define CDTDRAWEX_NAME    "cdtDrawExt"
#define CDTANIMATE_NAME    "cdtAnimate"
#define CDTTERM_NAME      "cdtTerm"
```

```
#endif // CARDSINSIDE H
```

Для обеспечения удобной работы клиента с множественными константами, которые используются функциями карточной библиотеки, определим все константы в заголовочном файле `CardsFlags.h`, представленном ниже.

```
//~\\~//~\\~//~\\~//~\\~//~\\~//~\\~//~\\~//~\\~//  
// CardsFlags.h  
//~\\~//~\\~//~\\~//~\\~//~\\~//~\\~//~\\~//~\\~//  
//-----
```

```
#ifndef CARDSFLAGS_H_
#define CARDSFLAGS_H
```

```
// определитель масти
typedef enum SUIT TYPE
```

```

S_CLUBS,      // крести
S_DIAMONDS,   // бубны
S_HEARTS,     // червы
S_SPADES      // пики
) SUIT_TYPE;

```

```
// ранг карты
typedef enum _RANG_TYPE
```

```

R_ ACE,      // туз
R_ TWO,      // 2
R_ THREE,    // 3
R_ FOUR,     // 4
R_ FIVE,     // 5
R_ SIX,      // 6
R_ SEVEN,    // 7
R_ EIGHT,    // 8
R_ NINE,     // 9
R_ TEN,      // 10
R_ JACK,     // валет
R_ QUEEN,    // дама
R_ KING      // король
} RANG TYPE;

```

```
// определитель рубашки
typedef enum BACK TYPE
```

```

1 MAX CARD SIZE = 52, // количество карт
  B_CROSSHATCH, // косые линии
  B_WEAVE1, // плетенка косая
  B_WEAVE2, // плетенка прямая
  B_ROBOT, // робот
  B_FLOWERS, // розы
  B_VINE1, // узор
  B_VINE2, // узор
  B_FISH1, // рыбы
  B_FISH2, // рыбы
  B_SHELLS, // ракушка
  B_CASTLE, // замок
  B_ISLAND, // тропический остров
  B_CARDHAND, // рука с картами
  B_UNUSED, // пусто
  B_THE_X, // знак "X"
  B_THE_O, // знак "O"
1 BACK TYPE:

```

```
// отображение эффектов на рубашке карт
typedef BOOL (WINAPI* LPCDTPANIMATE) (
    // описатель контекста граф. устройства
    HDC hDC,
    int nBackT, // тип рубашки карты
    // координата X левой стороны карты
    int nXCoord,
    // координата Y верхней стороны карты
    int nYCoord,
    int nState); // режим отображения
```

Интерфейс DLL — это набор функций, экспортируемых ею. Интерфейс класса C++ — это набор членов данного класса. В отличие от перечисленного, в COM интерфейсом является определенная структура в памяти, содержащая массив указателей на функции. Каждый элемент массива содержит адрес функции, реализуемой компонентом. Таким образом, интерфейс в COM — упомянутая структура в памяти; все остальное — это детали реализации, которые для COM не имеют значения.

Поскольку подавляющее большинство компиляторов C++ автоматически генерируют в памяти верную структуру при использовании абстрактных классов, на C++ интерфейс реализуется с помощью абстрактных классов. Так как каждый компонент COM может поддерживать сколь угодно много интерфейсов, для реализации компонента с несколькими интерфейсами используется множественное наследование абстрактных классов.

Исходя из целей создания компонента, определим его интерфейс, который поместим в файл `ICardsService.h`.

[illegible]

```
#ifndef __ICARDSSERVICE_H__
#define __ICARDSSERVICE_H__
```

```
#include <objbase.h>
```

```
//-----  
interface ICardsService : IUnknown  
{
```

```
virtual HRESULT __stdcall ShowCards(  
    // описатель контекста граф. устройства  
    HDC hDC,  
    // координата X левой стороны карты  
    int nXCoord,  
    // координата Y верхней стороны карты  
    int nYCoord,  
    int nCardV,    // значение карты  
    int nDrawT,    // признак отображения карты  
    int nColor,    // цвет карты  
    int nXSize = 0, // размер карты по ширине  
    int nYSize = 0) // размер карты по высоте  
= 0;
```

```
virtual HRESULT __stdcall AnimateCards(  
    // описатель контекста граф. устройства  
    HDC hDC,  
    int nBackT, // тип рубашки карты  
    // координата X левой стороны карты  
    int nXCoord,  
    // координата Y верхней стороны карты  
    int nYCoord,  
    int nState) = 0; // режим отображения
```

```
virtual HRESULT __stdcall GetNormalCardsSize(
    int* pnXSize,    // размер карты по ширине
    int* pnYSize) = 0; // размер по высоте
```

```
virtual HRESULT __stdcall ShuffleCards(  
    int* pnCards, // массив значений карт  
    // размерность массива  
    int nCountCards) = 0;  
  
virtual HRESULT __stdcall GetNumCards(  
    int nCardsSuit, // масть карты  
    int nCardsRang, // ранг карты  
    int* pnCardsNumber) = 0; // номер карты  
  
virtual HRESULT __stdcall GetSuitRangCards(  
    int nCardsNumber, // номер карты  
    int* pnCardsRang, // масть карты  
    int* pnCardsSuit) = 0; // ранг карты  
};
```

```
// {F345C351-28FD-4ee3-ABFA-9DBF9BE9A38A}
DEFINE_GUID(IID_CardsService,
0xf345c351, 0x28fd, 0x4ee3, 0xab, 0xfa, 0x9d,
0xbf, 0x9b, 0xe9, 0xa3, 0x8a);
```

```
// {100D2CC6-16F0-455b-A5DD-A0C563765AB4}
DEFINE_GUID(CLSID_CardsService,
0x100d2cc6, 0x16f0, 0x455b, 0xa5, 0xdd, 0xa0,
0xc5, 0x63, 0x76, 0x5a, 0xb4);
```

```
#endif // __ICARDSSERVICE H
```

Описание интерфейса использует ключевое слово `interface`, которое определено в файле `objbase.h` следующим образом:

```
#define interface struct
```

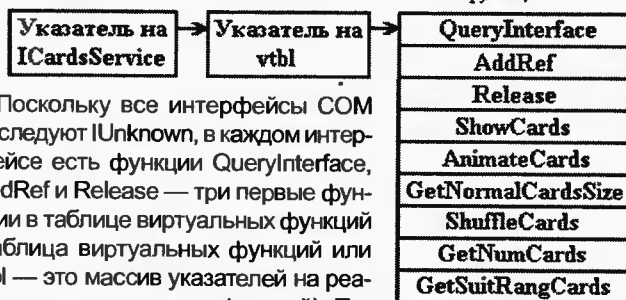
Как известно, структура отличается от класса исключительно областью видимости по умолчанию — для структуры это `public`, что и является причиной использования именно такого определения.

Все интерфейсы COM должны наследовать интерфейс IUnknown. Определение IUnknown, содержащееся в заголовочном файле `unknwn.h`, если опустить некоторые подробности, выглядит примерно так:

```
interface IUnknown
{
    virtual HRESULT STDMETHODCALLTYPE
        QueryInterface(REFIID riid,
            void __RPC_FAR * __RPC_FAR *ppvObject) = 0;
    virtual ULONG STDMETHODCALLTYPE
        AddRef(void) = 0;
    virtual ULONG STDMETHODCALLTYPE
        Release(void) = 0;
};
```

Клиент

Таблица виртуальных функций



Поскольку все интерфейсы COM наследуют IUnknown, в каждом интерфейсе есть функции QueryInterface, AddRef и Release — три первые функции в таблице виртуальных функций (таблица виртуальных функций или vtbl — это массив указателей на реализации виртуальных функций). Так как определение абстрактного класса фактически определяет структуру некоторого блока памяти, то все реализации абстрактных классов являются блоками памяти однотипной структуры. На рис. 1 показана структура памяти (таблица виртуальных функций) для интерфейса ICardService.

Все интерфейсы COM можно полиморфно трактовать как интерфейсы IUnknown. Если в первых трех элементах vtbl интерфейса не содержатся указатели на три перечисленные выше функции, то это не интерфейс COM.

(Продолжение следует)



А.КОРБИТ, А.ЩЕРБАКОВ,
г.Минск, БГУИР.

Программирование в среде Visual C++ 6.0

Использование динамически загружаемых библиотек

Цель этой статьи — изучить возможности и методику использования динамически загружаемых библиотек (DLL), а также статическое и динамическое подключение DLL.

1. Особенности использования DLL

DLL — это наборы функций, собранные в библиотеки. Однако, в отличие от статических библиотек, библиотеки DLL не присоединены непосредственно к исполняемому файлу с помощью редактора связей. В исполняемый файл занесена только информация об их местонахождении. В момент выполнения программы загружается вся библиотека целиком. Благодаря этому разные процессы могут пользоваться совместно одними и теми же библиотеками, находящимися в памяти. Такой подход позволяет сократить объем памяти, необходимый для нескольких приложений, использующих много общих библиотек, а также контролировать размеры EXE-файлов. Так, например, все функции Win32 API находятся в динамически загружаемых библиотеках.

Обычно DLL подключается статически на этапе компоновки, в этом случае загрузкой DLL при выполнении программы управляет операционная система. Однако DLL можно загрузить и явно или динамически, по мере использования ее экспортируемых функций, что увеличивает гибкость использования оперативной памяти.

При своем запуске каждое приложение пытается найти все файлы DLL, неявно подключенные к приложению, и поместить их в область оперативной памяти, занимаемую данным процессом. Поиск файлов DLL операционной системой осуществляется в следующей последовательности.

- каталог, в котором находится EXE-файл;
- текущий каталог процесса;
- системный каталог Windows.

Если библиотека DLL не обнаружена, приложение выводит диалоговое окно с сообщением о ее отсутствии и путях, по которым осуществлялся поиск. Затем процесс отключается.

Если нужная библиотека найдена, она помещается в оперативную память процесса, где и остается до его окончания. Теперь приложение может обращаться к функциям, содержащимся в DLL.

2. Пример создания DLL

Пусть требуется создать динамически загружаемую библиотеку, которая экспортирует функцию для вычисления формулы

$$R=x^2+y^2+z^2.$$

Для выполнения задачи следует создать проект и в качестве типа выбрать Win 32 Dynamic-Link Library. На втором шаге в качестве типа DLL выбрать An Empty DLL project, после чего будет создан пустой проект. К созданному проекту требуется добавить заголовочный файл (например, lab_dll.h), файл программы (lab_dll.cpp) и файл определения модуля с расширением *.def (например, lab_dll.def). Для

подключения заголовочного файла и файла программы требуется выбрать пункты меню Project — Add to Project — New и далее соответственно пункты C/C++ Header File или C++ Source File. Для создания файла определения модуля следует выбрать Project — Add to Project — New — Text File. При этом в поле File Name необходимо явно указать расширение *.def (lab_dll.def).

После создания всех необходимых файлов можно приступить к написанию программы. Исходный код заголовочного файла lab_dll.h:

```
#include <windows.h>
#ifdef FUNC_API
#define EXPORT __declspec (dllexport)
#else
#define EXPORT __declspec (dllimport)
#endif
EXPORT double CALLBACK func_r(double,double,double);
```

Сначала в заголовочном файле определяется макроконстанта **EXPORT**. Использование этого ключевого слова при определении некоторой функции динамически подключаемой библиотеки позволяет сообщить компоновщику, что эта функция доступна для использования другими программами, в результате чего он заносит ее в библиотеку импорта. Далее идет объявление самой экспортируемой функции.

Файл программы будет содержать следующий код:

```
#define FUNC_API
#include "lab_dll.h"
#include <math.h>

EXPORT double func_r(double x,double y,double z)
{
    double r;
    r=pow(x,2)+pow(y,2)+pow(z,2);
    return r;
}
```

Здесь сначала подключаются необходимые заголовочные файлы, и далее идет обычное определение функции, но с модификатором EXPORT.

Файл определения модуля (lab_dll.def) будет содержать следующий код:

```
LIBRARY "lab_dll"
DESCRIPTION "пример DLL-библиотеки"
EXPORTS
    func_r @1
```

В первой строке указывается имя библиотеки, далее идет ее краткое описание, и в конце — список экспортируемых функций, причем перед символом @ ставится ее порядковый номер.

После компиляции проекта в директории Debug появятся: файл lab_dll.dll — двоичный файл, содержащий исполняемый код библиотеки, и файл lab_dll.lib — так называемая библиотека импорта, в которой содержатся ссылки, экспортируемые DLL.

3. Пример программы со статической загрузкой DLL

Будем использовать динамически загружаемую библиотеку, созданную на предыдущем шаге.

Для использования DLL создадим новый проект на основе диалогового окна с интерфейсом, представленным на рис.1.

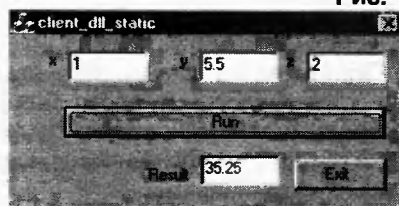


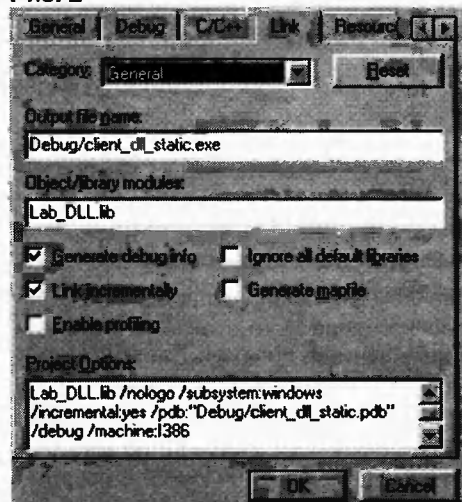
Рис. 1



При нажатии кнопки Run будет вызываться функция `func_r` из динамической библиотеки, и результат вычислений отобразится на форме.

Для использования DLL следует скопировать заголовочный файл динамической библиотеки, двоичный файл с расширением *.dll и lib-файл из предыдущего проекта в каталог программы-клиента.

Рис. 2



закладку Link, установить в поле Object/Library modules требуемый lib-файл (рис.2).

Обработчик нажатия кнопки Run, в котором вызывается функция динамической библиотеки, будет следующим:

```
#include "lab_dll.h"
void CClient_dll_staticDlg::OnButton1()
{
    UpdateData();
    m_res=func_r(m_x,m_y,m_z);
    UpdateData(0);
}
```

Как видно, при статическом импорте вызов функций из DLL ничем не отличается от вызова обычных функций.

4. Пример программы с динамической загрузкой DLL

Для создания программы с динамической загрузкой DLL создадим новый проект с интерфейсом, аналогичным представленному на рис.2. При использовании динамической загрузки не требуется использовать lib-файл, поэтому необходимо скопировать только файл библиотеки *.dll.

Первое, что необходимо сделать при динамической загрузке DLL — поместить модуль библиотеки в память процесса. Данная операция выполняется с помощью функции `::LoadLibrary`, имеющей единственный аргумент — имя загружаемого модуля. Соответствующий фрагмент программы должен выглядеть так:

```
HINSTANCE hMyDll;
if ((hMyDll=LoadLibrary("dll_lab.dll"))==NULL) return ;
```

Перед тем как использовать функции библиотеки, необходимо получить их адрес. Для этого сначала следует воспользоваться директивой `typedef` для определения типа указателя на функцию и определить переменную этого нового типа. Например:

```
typedef double (WINAPI *PFN_MyFunction)(double x,double y,double z);
PFN_MyFunction pfnMyFunction;
```

Адрес функции определяется при помощи функции `::GetProcAddress`, ей следует передать имя библиотеки и имя функции:

```
pfnMyFunction=(PFN_MyFunction)GetProcAddress(hMyDll,"func_r");
```

После получения указателя на функцию можно осуществить ее вызов:

```
m_res=(*pfnMyFunction)(m_x,m_y,m_z);
```

Полный текст обработчика кнопки Run примет вид:

```
void CDll_lab_dynamikDlg::OnButton1()
{
    HINSTANCE hMyDll;
    if ((hMyDll=LoadLibrary("dll_lab.dll"))==NULL) return ;
    typedef double (WINAPI *PFN_MyFunction)(double x,double y,double z);
    PFN_MyFunction pfnMyFunction;

    pfnMyFunction=(PFN_MyFunction)GetProcAddress(hMyDll,"func_r");
    UpdateData();
    m_res=(*pfnMyFunction)(m_x,m_y,m_z);
    UpdateData(0);
    FreeLibrary(hMyDll);
}
```

5. Индивидуальные задания

Требуется создать DLL-библиотеку, экспортирующую функцию для вычисления формулы

$$h = \frac{x^{y+1} + e^{y-1}}{1+x \cdot |y-tgz|} \cdot (1+|y-x|) + \frac{|y-x|^2}{2} - \frac{|y-x|^3}{3}.$$

При $x=2,444$; $y=0,0869$; $z=-130,0$ значение h должно быть равно $-0,49871$.

Необходимо написать две программы-клиента:

- со статическим вызовом DLL;
- с динамическим вызовом DLL.

Литература

1. С.Холзнер. Visual C++ 6: учебный курс. — СПб.: Питер, 1999. — 576 с.
2. Н.Ю.Секунов. Самоучитель Visual C++ 6 (+ дискета). — СПб.: BHV-Санкт-Петербург, 1999. — 960 с.
3. К.Грегори. Использование Visual C++ 6. Спец.издание. — СПб.: Вильямс, 1999. — 864 с.
4. Ю.В.Тихомиров. Visual C++ 6 (+ дискета). — СПб.: BHV-Санкт-Петербург, 1999. — 496 с.
5. Майкл Дж. Янг. Visual C++ 6. Полное руководство (+ CD-ROM). — BHV-Киев, 2000. — 1056 с.
6. С.Гилберт, Б.Маккарти. Самоучитель Visual C++ 6 (+ CD-ROM). — М.: ДиаСофт, 2000. — 496 с.
7. К.Паппас, У.Мюррей. Visual C++ 6 для пользователя. — BHV-Киев, 2000. — 336 с.
8. К.Паппас, У.Мюррей. Visual C++ 6: руководство разработчика. — BHV-Киев, 2000. — 624 с.
9. И.Баженова. Visual C++ 6. 0. VISUAL STUDIO 98: Уроки программирования. — М.: Диалог-МИФИ, 2001. — 416 с.
10. Лэйси Дж. Visual C++ 6 Distributed. Сертификационный экзамен — экстерном. — СПб.: Питер, 2001. — 624 с.
11. А.Черносвитов. Visual C++ 7: учебный курс (с дискетой). — СПб.: Питер, 2001. — 528 с.
12. А.Корбит, А.Щербаков, А.Соловей. Программирование в среде Visual C++ 6.0. — Радиомир. Ваш компьютер, 2003, №4-12, 2004, №3.



ТЕОРЕТИЧЕСКИЕ ОСНОВЫ КРЭКИНГА

(Продолжение. Начало в №№8-12/2003, 1-3/2004)

С технической точки зрения изучать данные в памяти гораздо менее удобно, чем "препарировать" исполняемый файл на винчестере, да и инструментов для интеллектуального поиска данных в чужом адресном пространстве не так уж много. Кроме того, если информация загружается в динамически выделяемые области памяти, исследуемые данные будут от запуска к запуску "плавать" по адресному пространству программы, каждый раз располагаясь по новым адресам. А это будет совсем уж нехорошо — проделывать массу рутинной работы только из-за того, что программу угораздило выделить очередной кусок памяти на сотню байтов выше, чем в предыдущий сеанс. Поэтому позаботимся о создании элементарных удобств для работы. Прежде всего нам понадобится перенести наше поле деятельности из чрезвычайно "нежной" оперативной памяти на гораздо более "жесткий" диск, где можно будет проводить любые эксперименты не опасаясь, что случайное нажатие "не той" клавиши приведет к потере результатов длительных исследований. Вы наверняка уже догадались, что самым простым решением было бы снятие дампа с нужных областей памяти и сохранение этого дампа на "винчестере". Увы, все далеко не так просто, как хотелось бы.

Во-первых, вам придется озадачиться поиском подходящего инструмента. Классические дамперы из крэкерского арсенала вам не помогут, поскольку они предназначены для снятия дампа **программы**, а не данных, с которыми эта программа работает. Более того, случайный захват данных для классического дампера — явление крайне нежелательное, поскольку основное назначение дамперов — распаковка программ и получение работоспособного EXE-файла с минимальным количеством избыточной информации внутри, а не анализ "мусора", перемалываемого программой в процессе ее работы. А вот нам нужен именно этот "мусор". Поэтому придется либо писать программу для "правильного" дампинга своими силами, либо извлекать нужные данные вручную, при помощи отладчика, позволяющего сохранять содержимое кусков памяти в файлах.

Во-вторых, данные могут быть разбросаны по адресному пространству программы, но сдампить их нужно за один сеанс отладки. Не так уж редки программы, в которых часть информации хранится, к примеру, в секции инициализированных данных, а другая часть — в динамически выделяемых блоках памяти, и если сдампить содержимое динамической памяти программы, но забыть про инициализированные данные, такой дамп, скорее всего, можно выбросить. Нет никакой гарантии, что при следующем запуске, когда вы выясните, что указатели на динамические блоки находятся в секции инициализированных данных, и что эту секцию тоже нужно дампить, эти указатели будут указывать на те же адреса, что и в прошлый сеанс работы. Поэтому я настоятельно не рекомендую экономить место на диске, а делать полный снимок всех секций программы, поскольку лишнее всегда можно выки-

нуть, а вот недостающие данные взять будет неоткуда.

И в-третьих, снимая дамп, никогда не забывайте записывать базовые адреса тех кусков памяти, которые вы дампите — в самом ближайшем будущем они вам определенно понадобятся.

Но допустим, что мы аккуратно сделали полный снимок подопытной программы, и теперь все ее секции аккуратно разложены на нашем винчестере в идеальном порядке. Что дальше? Возьмите хороший шестнадцатеричный редактор и загрузите в него какую-либо из секций. Теперь настройте этот редактор так, чтобы вместо смещений в файле он показывал смещения относительно базового адреса этой секции в памяти. Т.е. если вы сбрасывали на винчестер кусок памяти с адреса 401000h по 402000h, после соответствующей настройки смещение первого байта файла должно отображаться именно как 401000h, а не как 0. В частности, такую операцию умеет выполнять HIEW — для этого необходимо нажать Ctrl-F5 и ввести новую базу. Если ваш шестнадцатеричный редактор делать такие фокусы не умеет, значит, вы выбрали недостаточно хороший шестнадцатеричный редактор, и вам будет заметно сложнее постигать развернувшиеся перед вами глубины программы. Возможно даже, что несовершенство вашего инструментария подвигнет вас на написание нового, уникального шестнадцатеричного редактора с доселе невиданными возможностями — великие дела совершались по куда менее значительным поводам, чем отсутствие подходящего инструмента для копания в кодах. В принципе, можно обойтись даже без автоматического пересчета смещения, но тогда вам придется проделывать необходимые вычисления в уме, и за всей этой шестнадцатерично-арифметической рутинной вы можете не прочувствовать до конца всю силу и эффективность предлагаемого метода.

После того как вы проделаете все, о чем я говорил, внимательно посмотрите на экран монитора, включите на полную мощность свое воображение и представьте, что вы разглядываете не кучу байтов, тонким слоем рассыпанных по поверхностям жесткого диска, а мгновение из жизни программы, которое вы вольны сделать сколь угодно долгим. И то, что вы видите в окне шестнадцатеричного редактора, по сути, ничем не отличается от того, что вы бы увидели в окне отладчика, разглядывая память "живой" программы. Вы точно так же можете, следуя указателям, бродить по адресному пространству, дизассемблировать куски кода, искать константы и переменные (поскольку "замороженная" переменная есть не что иное как константа) по их значению — в общем, делать с программой все то, о чем я говорил в двух предыдущих главах.

Однако помимо "честных" методов поиска, требующих хотя бы минимальных сведений о структуре и типизации искомых данных, есть еще один нехитрый прием, не требующий ничего, кроме терпения. Суть метода проста —



вы вырываете из файла на жестком диске небольшой кусок и пытаетесь найти точно такой же кусок в памяти программы. При некотором везении, идентичный кусок обнаружится в памяти, а его местоположение укажет вам, куда программа загрузила соответствующие байтики из файла. После этого вы можете попытаться логически проанализировать наблюдаемую картину либо просто вклеить аппаратную точку останова на чтение всей прилегающей памяти (это будет очень, очень большая "точка") и посмотреть, что будет делать с данными подопытная программа. Несмотря на то что такой поиск внешне сильно напоминает пресловутый "метод научного тыка", в его основе лежит вполне логичная идея — если информация из файла переносится в память без потерь и существенных изменений, соответствующие элементы структур в файлах и в памяти будут идентичны. Проще говоря, если где-то в файле хранилось двухбайтное число 12345, есть вероятность, что оно и в памяти будет выглядеть двухбайтным числом 12345. Хотя, конечно, вполне возможны программы, загружающие числа типа "байт", но обрабатывающие их как 10-байтные с плавающей точкой. Разумеется, этот метод можно усовершенствовать, заметно повысив его эффективность. Например, брать не случайные куски, а содержащие осмысленный текст — тогда вы будете знать, что ищете текстовое поле, на которое почти наверняка будет существовать указатель, а сам этот указатель, скорее всего, будет входить в структуру, а структуры будут организованы в массив или список... Как видите, хотя набор базовых приемов не так уж велик, и каждый из них отнюдь не свободен от недостатков и ограничений, но комбинируя и адаптируя их под особенности конкретных программ, можно весьма многого добиться.

Рассмотренные выше техники добывания данных из недр "живой" программы имели одно общее свойство — по отношению к программе их можно было охарактеризовать как "принуждение". Посудите сами — программа спокойно себе работает, никого не обижает, но тут в ее спокойную и размеренную жизнь врывается крэкер с отладчиком наголо и начинает направо и налево дампить секции и разбрасывать точки останова. Такой подход, конечно, приносит свои плоды — но в некоторых случаях проблемы извлечения данных проще решать не "грубой силой", даже если это сила интеллекта, но хитростью, использованием всевозможных лазеек в коде программы или даже через нетривиальное использование стандартных средств ОС или самой программы. Практика показала, что если программу вежливо и в изысканной форме попросить, она вполне может поделиться с вами нужными вам данными.

В Windows роль вежливых просьб играют системные сообщения — традиционное средство, используемое для огромного количества всевозможных действий — от элементарного закрытия окна до рассылки программам уведомлений о выходе операционной системы из "спячки", иначе именуемой Hibernate. Сила сообщений в Windows весьма велика, и, овладев и правильно распорядившись ею, можно получать весьма интересные результаты. Например, при помощи сообщений можно вытащить все строки из выпадающего списка (ComboBox) или таблицы (ListView), если автор программы забыл предусмотреть в своем детище более традиционный способ сохранения данных. Что для этого нужно? Только документация и некоторые навыки в программировании с исполь-

зованием WinAPI. А теперь мы плавно перейдем от теории к практике и рассмотрим пример того, как можно применить эту технику для решения конкретной задачи. Но для начала — немного истории.

Как-то раз у меня возникла необходимость получить полный список имен всех сообщений Windows и числовых значений, которые за этими именами скрываются. Задача, надо сказать, была совсем не праздная — этот список был мне жизненно необходим, чтобы включить его в состав моих программ. Но вот незадача — в заголовочном файле из комплекта поставки MASM32 эти имена были разбросаны по всему windows.inc в совершенном беспорядке, и меня совершенно не радовала перспектива проявлять чудеса трудолюбия, вручную выискивая и обрабатывая несколько сотен строк. Полный список, разумеется, можно было бы извлечь из заголовочных файлов последней версии Visual Studio, но кроме того, что я вообще не являюсь поклонником данного продукта, в частности, у меня не было никакого желания искать где-то дистрибутив оной "студии" и устанавливать его ради одного-единственного файла. Однако так уж исторически сложилось, что у меня все-таки была одна небольшая часть Visual Studio — утилита, именуемая Spy++. Одна из функций этой утилиты заключалась в том, чтобы отслеживать сообщения, которые пользователь указывал в специальном окне, по-научному называемом ListBox. В этом окне как раз и отображался полный список сообщений, среди которых можно было "мышкой" отметить те сообщения, которые требовалось отлавливать. Иными словами, было совершенно очевидно, что Spy++ содержал всю необходимую мне информацию, и требовалось лишь найти способ эту информацию извлечь.

Первое, что пришло мне в голову, это пропустить файл spyxx.exe через утилиту, вычлняющую текстовые строки, и затем выбрать из всех найденных строк имена сообщений. Однако после некоторых размышлений я отверг этот путь: во-первых, мне хотелось получить список сообщений отсортированным в том же порядке, что и в Spy++; а во-вторых, у меня не было желания разбирать ту кучу мала, которую обычно вываливают утилиты поиска текстовых строк. Поэтому я решил поступить проще — написал программку, которая при помощи сообщения LB_GETCOUNT определяла количество строк в нужном мне ListBox'e, а потом построчно считывала содержимое ListBox'a, в цикле отправляя ему сообщение LB_GETTEXT. Через считанные минуты у меня на винчестере покоился в виде текстового файла полный список сообщений из Spy++. После этого оставалось только извлечь из исполняемого файла числовые значения, соответствующие именам сообщений, что я и сделал при помощи методов, о которых я говорил в предыдущей главе. Если у вас есть желание попрактиковаться в применении этих методов — можете самостоятельно попробовать извлечь эти данные, особой сложности это не представляет.

Нередко для обработки и отображения данных программисты под ОС Windows используют ActiveX-компоненты, одним из полезных свойств которых является возможность получить доступ к интерфейсам такого компонента без всякой документации и заголовочных файлов. Например, импортировав нужный ActiveX-компонент в Delphi, вы сразу же сможете увидеть свойства и методы, присущие этому компоненту. Запрашивая значение нужных свойств и вызывая соответствующие методы, вы, скорее всего, сможете научиться извлекать данные, которые этот ActiveX отображает. Бо-



лее того, вы получите возможность экспериментировать с этим компонентом в "лабораторных условиях" собственных тестовых примеров, имитирующих работу программы, из которой вы собираетесь вытащить данные, а не непосредственно на "поле битвы" с чужим кодом. Вы можете подумать: ну и какая польза от этих экспериментов — ведь нужные данные находятся в **другой** программе. Но не спешите с выводами. Представьте себе, что вам удалось внедрить свой код в исследуемую программу и получить доступ к интерфейсам нужного ActiveX... А впрочем, почему только "представьте"? Внедряйте, получайте доступ — и считывайте возжеленную информацию!

И, наконец, не бойтесь пользоваться простейшими мето-

дами. Может случиться так, что один из множества инструментов автоматизации, умеющий листать страницы в указанном окне и делать скриншоты, объединенный с программой распознавания текста, поможет вам получить распечатку защищенного от копирования телефонного справочника быстрее, чем если извлекать ту же информацию из глубин адресного пространства при помощи отладчика. В конце концов, если информация где-то отображается, значит, ее можно оттуда извлечь и сохранить в желаемом виде — нужно лишь изобрести подходящий метод. Но это уже совсем другая, далекая от крэкинга история.

(Продолжение следует)

Нестандартное использование таблицы релокаций PE-файла

А.ТЕРЁШКИН/НИ-ТЕСН.

Введение

Таблица релокаций (relocation table) — это информация, необходимая загрузчику PE-файла в том случае, если модуль не может быть загружен по предпочитаемой базе. Загрузчик должен исправить все абсолютные указатели в модуле, так как их старые значения, указанные линкером, основаны на предпочитаемой базе, и, следовательно, уже не являются верными. Таблица релокаций содержит данные о расположении абсолютных указателей в модуле.

Если такая информация имеется в наличии, то код, содержащийся в модуле, можно легко перебазировать, причем не обязательно на число, кратное 0x10000 (как известно, база win32-модуля обязательно должна быть кратна 0x10000). По сути, информация о расположении абсолютных адресов делает код, их содержащий, базонезависимым.

Рассмотрим структуру таблицы релокаций.

Таблица является последовательностью блоков, каждый из которых содержит данные об абсолютных указателях 4-килобайтного участка модуля. Блок начинается со структуры IMAGE_BASE_RELOCATION, за которой следует массив слов (IMAGE_FIXUP_ENTRY). Начало блока выровнено на 4 байта:

```
typedef struct _IMAGE_BASE_RELOCATION {
    DWORD      VirtualAddress;
    DWORD      SizeOfBlock;
} IMAGE_BASE_RELOCATION;
typedef IMAGE_BASE_RELOCATION;

typedef struct {
    WORD      offset:12;
    WORD      type:4;
} IMAGE_FIXUP_ENTRY, *PIMAGE_FIXUP_ENTRY;
```

Здесь:

- VirtualAddress — адрес начала 4-килобайтного участка, к которому применяется информация о релокации;
- SizeOfBlock — размер блока информации, включая размер структуры IMAGE_BASE_RELOCATION;
- offset — смещение абсолютного указателя относительно начала 4-килобайтного участка. Размер этого поля — 12 битов, т.е. это значение лежит в промежутке [0, 4095];
- type — тип элемента релокации. На практике (PE-фай-

лы для процессоров Intel) встречаются два типа: IMAGE_REL_BASED_ABSOLUTE ("пропустить элемент", используется для выравнивания блоков) и IMAGE_REL_BASED_HIGHLOW ("прибавить разницу между физической и предпочитаемой базой").

Совершив обход таблицы релокаций, можно как угодно изменить абсолютные указатели. Как это можно использовать? Например, перебазирование (смещение) кода — это нестандартный метод получения свободного места в PE-файлах.

Перебазируя код, например, на X байтов вперед, и перестроив виртуальную структуру PE-файла на ALIGN(X, SectionAlignment), можно выделить виртуально свободное место для внедрения нового объекта по освобожденному адресу. Или же, можно сдвинуть содержимое виртуально первой секции, не изменяя ее виртуальный и физический адрес, тем самым получив свободное место в начале этой секции.

Разберемся с вторым вариантом. Для простоты будем выделять место, равное SectionAlignment модуля. Для реализации подобной техники нужно предпринять следующие шаги.

1. Подготовить указатели.
 2. Найти секцию с минимальным VirtualAddress (т.е. виртуально первую секцию).
 3. Прибавить SectionAlignment ко всем абсолютным адресам в модуле.
 4. Прибавить SectionAlignment ко всем RVA структуры PE, большим адреса начала секции с минимальным RVA. В частности, это касается следующих объектов:
 - AddressOfEntryPoint;
 - BaseOfData;
 - SizeOfImage;
 - начала всех директорий;
 - всех RVA директории импорта;
 - всех RVA директории экспорта;
 - всех RVA директории ресурсов.
 5. Сдвинуть на SectionAlignment физически и виртуально все секции, кроме виртуально первой.
 6. Увеличить физический и виртуальный размер виртуально первой секции, ее содержимое сдвинуть.
- Рассмотрим эти пункты более подробно.



Подготовка указателей

```

BOOL GetHeaders(PBYTE ibase,
                PIMAGE_FILE_HEADER *pfh,
                PIMAGE_OPTIONAL_HEADER *poh,
                PIMAGE_SECTION_HEADER *psh)
{
    PIMAGE_DOS_HEADER mzhead=(PIMAGE_DOS_HEADER)ibase;

    if ((mzhead->e_magic!=IMAGE_DOS_SIGNATURE) ||
        (ibaseDD[mzhead->e_lfanew]!=IMAGE_NT_SIGNATURE))
        return FALSE;

    *pfh=(PIMAGE_FILE_HEADER)&ibase[mzhead->e_lfanew];
    if (((PIMAGE_NT_HEADERS)*pfh)->Signature!=IMAGE_NT_SIGNATURE)
        return FALSE;

    *pfh=(PIMAGE_FILE_HEADER)((PBYTE)*pfh+sizeof(IMAGE_NT_SIGNATURE));

    *poh=(PIMAGE_OPTIONAL_HEADER)((PBYTE)*pfh+sizeof(IMAGE_FILE_HEADER));
    if ((*poh)->Magic!=IMAGE_NT_OPTIONAL_HDR32_MAGIC)
        return FALSE;

    *psh=(PIMAGE_SECTION_HEADER)((PBYTE)*poh+sizeof(IMAGE_OPTIONAL_HEADER));
    return TRUE;
}

```

Этот код чрезвычайно прямолинеен. Его задача — проверить, является ли переданный образ образом PE- файла, и в случае успеха установить указатели на соответствующие заголовки этого файла.

```

DWORD RVAtPA(DWORD rva, DWORD base)
{
    DWORD i, rc;

    if (rva<o_poh->SizeOfHeaders) return rva+base;

    for (DWORD i=0; i<o_pfh->NumberOfSections; i++)
        if ((o_psh[i].VirtualAddress <= rva) &&
            (rva<o_psh[i].VirtualAddress + o_psh[i].Misc.VirtualSize)) {
            rc=o_psh[i].PointerToRawData+rva-o_psh[i].VirtualAddress;
            // pure virtual RVA?
            if (rc>o_psh[i].PointerToRawData+o_psh[i].SizeOfRawData) return 0;
            return rc+base;
        }
    return 0;
}

```

Эта функция производит перевод RVA в физический адрес и прибавляет к нему значение base. Здесь используется стандартный алгоритм трансляции относительного виртуального адреса — производится перебор секций и проверка значения RVA на принадлежность к их виртуальным регионам. В случае принадлежности поиск прекращается, и возвращается сумма указанной базы, физического адреса начала найденной секции и смещения RVA относительно начала этой секции.

```

void FixRVA(PDWORD rva)
{
    if (!*rva) return;
    if (*rva>minsectRVA) *rva+=o_poh->SectionAlignment;
}

void FixSize(PDWORD size)
{
    *size+=o_poh->SectionAlignment;
}

```

Как было условлено, размер добавляемой области не

больше, чем число, на которое виртуально выравниваются секции. Функции предназначены для исправления значений полей в заголовке PE и в других областях файла — например, с помощью FixRVA производится исправление всех абсолютных адресов.

Функция FixRVA отличается от FixSize тем, что она проверяет переданный через указатель адрес на превышение или равенство относительному виртуальному адресу самой первой (виртуально) секции. Этим производится дополнительный контроль ошибок — исправление адреса, лежащего до области раздвижения, производиться не должно.

Функция FixSize используется для увеличения размеров подвергающейся раздвижению секции (виртуально и физического). Разумеется, это увеличение должно производиться в любом случае.

Найти секцию с минимальным VirtualAddress (т.е. виртуально первую секцию)

```

maxsectPA=o_psh[0].SizeOfRawData;
minsectRVA=o_psh[0].VirtualAddress;
minsect=0;maxsect=0;

for (DWORD i=0; i<o_pfh->NumberOfSections; i++) {
    if (minsectRVA>o_psh[i].VirtualAddress) {
        minsectRVA=o_psh[i].VirtualAddress;
        minsect=i;
    }
    if (maxsectPA<o_psh[i].PointerToRawData) {
        maxsectPA=o_psh[i].PointerToRawData;
        maxsect=i;
    }
}

DWORD ovroffs=maxsectPA+o_psh[maxsect].SizeOfRawData;
ovrsize=0;

```

Производится перебор секций PE с целью найти секцию с минимальным виртуальным адресом начала и секцию с максимальным физическим адресом. Первый адрес необходим для проверки корректируемых в будущем RVA — если корректируемый адрес меньше, чем минимальный адрес начала секции, то он не нуждается в корректировке. Второй физический адрес необходим для обнаружения оверлея — т.е. информации в файле, начинающейся после конца последней (в файле) секции. При наличии оверлея должна производиться его перезапись после сделанных в файле изменений; если в заголовке файла присутствовала контрольная сумма, она пересчитывается не только по образу файла, но и по его оверлею.

Прибавить SectionAlignment ко всем абсолютным адресам в модуле

```

PIMAGE_BASE_RELOCATION o_pbr;
PIMAGE_FIXUP_ENTRY o_pfe;
DWORD fixups=0;

if ((o_poh->DataDirectory[IMAGE_DIRECTORY_ENTRY_BASERELOC].VirtualAddress) &&
    (!(o_pfh->Characteristics)&IMAGE_FILE_RELOCS_STRIPPED)) {
    o_pbr=(PIMAGE_BASE_RELOCATION)RVAtPA(o_poh->DataDirectory[
        IMAGE_DIRECTORY_ENTRY_BASERELOC ].VirtualAddress, (DWORD)oldibase);

    while (o_pbr->VirtualAddress) {
        o_pfe=(PIMAGE_FIXUP_ENTRY)((DWORD)o_pbr+sizeof(IMAGE_BASE_RELOCATION));

        for (DWORD i=0; i<(o_pbr->SizeOfBlock - sizeof

```



```

(IMAGE_BASE_RELOCATION)/2; i++) {
    if (((* (PWORD) o_pfe)>>12)==IMAGE_REL_BASED_HIGHLOW) {
        fixups++;

        FixRVA((PWORD) (RVAToPA(o_pbr->VirtualAddress,
(DWORD)oldibase+((* (PWORD) o_pfe)&0xFFF))));
    } else
        if (((* (PWORD) o_pfe)>>12) != IMAGE_REL_BASED_ABSOLUTE)
            printf("\trelo type %d found at .%X\n",
(* (PWORD) o_pfe)>>12, o_pbr->VirtualAddress+((* (PWORD) o_pfe)&0xFFF));
            o_pfe++;
        }
        FixRVA(&(o_pbr->VirtualAddress));
        * (PWORD) &o_pbr+=o_pbr->SizeOfBlock;
    }
}

```

Алгоритм поиска и исправления абсолютных адресов можно представить такой последовательностью действий.

1. Вычислить адрес директории `IMAGE_DIRECTORY_BASE_RELOC`. В ней содержится информация об элементах релокации — иными словами, указатели на абсолютные адреса в файле. Если такой директории не существует, или флаги образа говорят об отсутствии таблицы релокации, выйти.

2. Приравнять адрес структуры `IMAGE_BASE_RELOCATION` к адресу таблицы релокаций — с нее будет начат обход абсолютных адресов. Каждая такая структура определяет блок элементов релокации, располагающихся на одной странице.

3. Пока поле `VirtualAddress` структуры `IMAGE_BASE_RELOCATION` текущего блока не равно нулю:

4. Для каждого слова, следующего за структурой-заголовком блока и входящего в этот блок (размер блока элементов релокаций указывается в поле `SizeOfData` структуры `IMAGE_BASE_RELOCATION`), проверяется значение 4-х старших битов — эти биты указывают на тип элемента релокации. Обычно в x86 используются лишь 2 типа: `IMAGE_REL_BASED_HIGHLOW` (указатель вычисляется сложением младших 12 битов элемента релокации с `VirtualAddress`, указанным в структуре-заголовке блока) и `IMAGE_REL_BASED_ABSOLUTE` (никаких действий не производится — этот тип используется для выравнивания блоков релокации).

Прибавить `SectionAlignment` ко всем `RVA` структуры `PE`, большим адреса начала секции с минимальным `RVA`

```

FixRVA(&(o_poh->AddressOfEntryPoint));
FixRVA(&(o_poh->BaseOfData));
FixRVA(&(o_poh->SizeOfImage));

```

```

//перебрать все директории
for (DWORD i=0; i<MIN(IMAGE_NUMBEROF_DIRECTORY_ENTRIES,
o_poh->NumberOfRvaAndSizes); i++) {
    if (o_poh->DataDirectory[i].VirtualAddress) {
        FixRVA(&(o_poh->DataDirectory[i].VirtualAddress));
    }
}

```

Смысл этого пункта — “починить” те `RVA` в структуре `PE`, которые находятся после виртуального адреса внедренного участка. Здесь не приводится код для пересчетов относительных адресов в таблицах импорта, экспорта и ресурсов — это всего лишь обход документированных структур.

Физически сдвинуть все секции, виртуально сдвинуть все секции, кроме первой, а также увеличить

физический и виртуальный размер первой секции

```

FixSize(&(psh[minsect].SizeOfRawData));
FixSize(&(psh[minsect].Misc.VirtualSize));

for (DWORD i=0; i<poh->NumberOfSections; i++) {
    if (i!=minsect) {
        // сдвинуть не первую
        if (psh[minsect].PointerToRawData<psh[i].PointerToRawData)
            psh[i].PointerToRawData+=MAX(poh->FileAlignment,
            poh->SectionAlignment);
        FixRVA(&(psh[i].VirtualAddress));

        // скопировать память по другому смещению
        memcpy(&newibase[psh[i].PointerToRawData],
            &oldibase[o_psh[i].PointerToRawData],
            o_psh[i].SizeOfRawData);
    } else {
        // скопировать содержимое первой, выделить память
        memcpy(&newibase[psh[i].PointerToRawData+
            poh->SectionAlignment],
            &oldibase[o_psh[i].PointerToRawData],
            o_psh[i].SizeOfRawData);
        // заполнить добавленное место байтами 0xCC (int 3)
        // для отладочных целей
        memset(&newibase[psh[i].PointerToRawData], 0xcc, poh
            ->SectionAlignment);
    }
}

```

Первыми двумя строчками производится увеличение виртуального и физического размера виртуально первой секции. Затем в цикле содержимое всех секций сдвигается на `SectionAlignment`, и корректируются виртуальные и физические адреса начала всех секций, кроме виртуально первой, то есть происходит выделение физического и виртуального места размером в `SectionAlignment`.

Заключение

Базонезависимость кода можно использовать для решения более общих проблем. Например, та же вставка свободного места в `PE` может быть осуществлена несколькими способами:

- вставить секции любой длины между любыми двумя секциями (или перед первой — после последней в любых смыслах) — главное, чтобы в заголовке файла хватило места для объявления еще одного объекта;
- не вставляя новую секцию, сместить содержимое любой секции на необходимый размер, не трогая `RVA` ее начала. Секции, `RVA` начала которых больше, сдвинуть;
- действия, аналогичные предыдущему методу, но увеличить виртуальный размер секции, не сдвигая ее содержимое. Свободное место будет находиться в конце выбранной секции. Секции, `RVA` начала которых больше, сдвинуть. Здесь могут возникнуть ошибки при дополнении секции с данными — особенно, если ее `VirtualSize` больше `SizeOfRawData`.

Наличие информации об элементах релокации дает практически неограниченные возможности по изменению структуры файла и резко упрощает манипулирование кодом — чего стоит хотя бы возможность “вручную” загрузить такой файл по любой базе (не обязательно кратной `0x10000`) без потери работоспособности? Единственно, с чем будут проблемы у образов с нестандартной базой — с чтением ресурсов функциями `API`.



История одной системы

Е.ЛЕБЕДЕВ,
г.Череповец.

К написанию этой статьи меня подтолкнуло любопытство... Смогу ли я обойти "устои" системы, и так ли она хорошо защищена...? Обо всех своих исследованиях я честно доложил местному администратору!

Предыстория...

Я хочу поделиться результатами анализа защиты одной компьютерной системы, которая существует и в настоящее время, но сначала я хочу ввести читателя в курс дела и рассказать небольшую предысторию. Однажды в моем учебном заведении мне представилась возможность поработать за одним из компьютеров (при выполнении лабораторной работы по предмету X, название предмета значения не имеет). Когда я в первый раз попал в соответствующую компьютерную аудиторию, то "осмотрелся" в компьютере, и, так сказать, произвел "первичный зрительный" анализ защиты системы, который не дал никаких полезных сведений. В день выполнения лабораторной работы я заметил установленный на диске PowerQuest Partition Magic v6.0. Как известно, Partition Magic зарекомендовал себя как отличный менеджер жесткого диска. Сразу же мелькнула мысль: "Зачем было ставить Partition Magic в учебном заведении? Ведь студенты могут повредить данные на любом из дисков!!!". Я запустил программу, но

денты не удалили операционку или не утащили rwl-файлы, отвечающие за пароли. Итак, Partition Magic (далее — PM) скрыл второй раздел диска, и передо мной стояла задача "залезть" в PM, миновав пароль.

В дополнение к этому, на всех компьютерах стоял Антивирус Касперского v3.55, который из-за значительной нагрузки сети и низкой мощности компьютеров сильно затормаживал систему. В системном трее (system tray) располагалась иконка антивируса, но при попытке отключить программу выскакивало окошко с просьбой ввести пароль (рис.2). Таким образом, передо мной встала проблема обхода двух парольных защит.

В этой статье мы рассмотрим, как получить доступ к жесткому диску компьютера и убрать парольный доступ к настройкам антивируса.

Исследование защиты Partition Magic v6.0

Администратор местной ЛВС не стал делать запретов на флорпу-дисковод, и поэтому у нас есть возможность работать с посторонними файлами и программами. Вначале мне понадобился ехе-файл для подробного изучения его защиты дома, в спокойной обстановке. Но, к сожалению, размер оригинального исполняемого файла — 3,9 Мб. Для дискеты файл никак не подойдет, так что пришлось его сжать с помощью утилитки UPX. После компрессии файл значительно "похудел" и стал "весить" всего лишь 1,22 Мб. Для исследований защиты также понадобится какая-либо версия Partition Magic. Ехе-шник от версии 6 будет работать с файлами от версии 7.

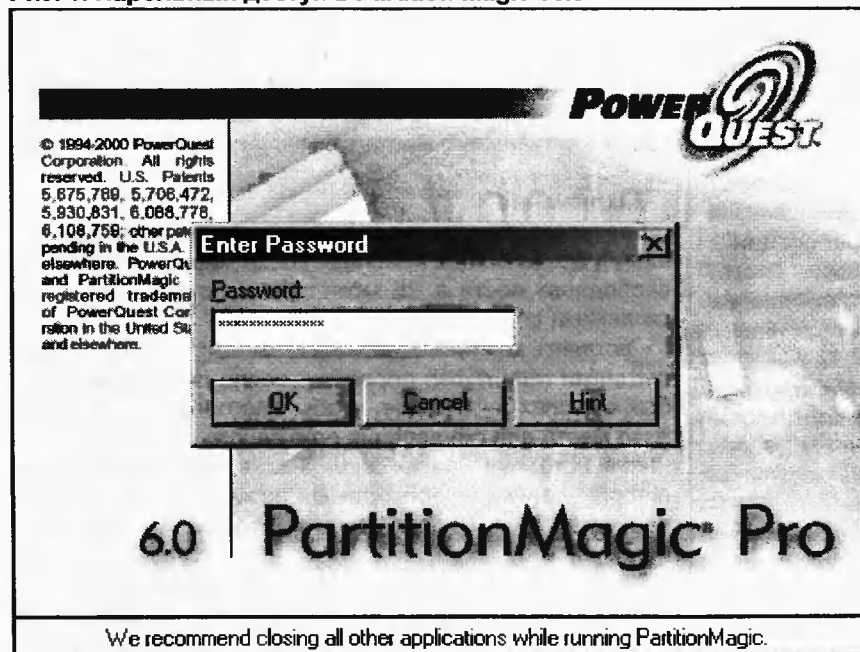
В исследовании защитного алгоритма нам потребуются инструменты: отладчик (TRW2000) и hex-редактор (HexWorkshop v3.0). Запускаем программу, а следом — отладчик. В Partition Magic'e ставим пароль на запуск и перезапускаем программу. В отладчике устанавливаем прерывание на стандартные функции считывания данных из полей ввода:

```
Bpx GetDlgItemText
Bpx GetWindowText
```

К сожалению, точки останова (бряки) не сработают! Тогда ставим бряк на копирование памяти:

```
Bpx hmemcpy
```

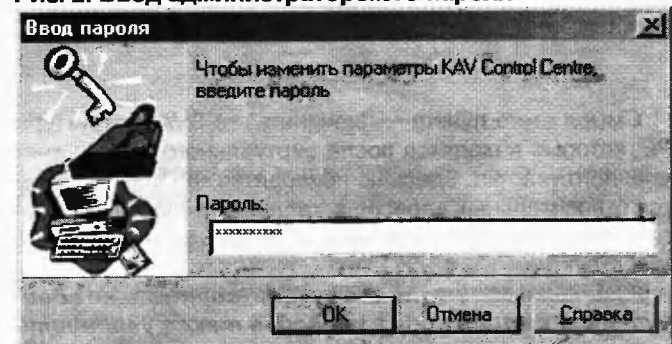
Рис. 1. Парольный доступ в Partition Magic v6.0



администратор местной локальной сети (ЛВС) оказался не дурак и поставил парольный доступ к программе (рис.1).

Зачем же на диске стоял Partition Magic? На винчестере стояли две операционные системы (ОС) — Windows 2000 и Windows ME. Первый раздел был отформатирован как NTFS, а второй, естественно, как FAT32. Оказывается, администратор сети воспользовался данной утилитой, для того чтобы скрыть в Windows 2000 раздел жесткого диска с установленным Windows ME. За сохранность первого раздела админу можно было не беспокоиться, т.к. из Windows ME в NTFS-раздел "нормально" залезть было невозможно (т.к. была включена опция шифрования данных!). А вот в Win'2000 админ решил скрыть раздел Win'ME, чтобы сту-

Рис. 2. Ввод администраторского пароля





Выйдем к программе, нажмем ОК, и вылезет отладчик в модуле Kernel. Нам этот модуль не нужен, так что вернемся к вызову функции, нажав определенное количество раз F12. Как только в отладчике покажется имя модуля Partition Magic, остановимся в следующем участке:

```
:00429A10 8D45FC      lea eax, dword ptr [ebp-04]
;<<курсор в этой строке
:00429A13 8D93E4020000 lea edx, dword ptr [ebx+000002E4]
:00429A19 E80A2D1C00      call 005EC728
```

В этом коде нет ничего полезного, так что идем дальше трассировать программу по F10. После недолгих “скитаний” по программе мы наткнемся на следующий код:

```
:00429A32 0F848D000000 je 00429AC5 ;<<ключевой переход
:00429A38 6A00      push 00000000
:00429A3A 6A00      push 00000000
:00429A3C 6A00      push 00000000
:00429A3E 6A60      push 00000060
:00429A40 68E4030000 push 000003E4
:00429A45 A120A06200 mov eax, dword ptr [0062A020]
:00429A4A 50      push eax
:00429A4B 8B10      mov edx, dword ptr [eax]
:00429A4D FF521C      call [edx+1C] ;<<вызов окна с сообщением о неверном пароле
```

Первый раз мы можем не обратить внимание на “спящий” переход по адресу 00429A32, и выполнится call по адресу 00429A4D, который покажет окошко, сообщающее о неверном пароле. После нажатия ОК выскочит отладчик, в котором ставим бряк на адрес 00429A32. Повторив заново процесс подтверждения пароля, мы окажемся в строке ключевого перехода. Перевод окажется неактивным (“спящим”), так что придется “разбудить” его, поменяв флаг состояния Z на противоположный. Теперь переход осуществится, и программа “подумает”, что пароль верный, а следовательно, загрузится главная часть программы. Отлично — пароль миновали! Теперь списываем ключевые байты: 0F848D0000006A006A006A006A60

Запускаем HexWorkshop, нажимаем “Find” и вводим последовательность. После того как найдутся нужные байты, меняем 0F84 на 0F85. Сохраняем. Запускаем новый (“исправленный”) файл и вводим любой пароль. Программа послушно соглашается с введенными данными, и загружается Partition Magic.

Парольная защита была обезврежена, и теперь нам открылся доступ к дискам “атакуемого” компьютера. Далее можно было бы сделать патч, который менял бы исполняемый файл, но этот вариант нам не подойдет. Почему? Потому что если пропатчить РМ на “атакуемом” компьютере и тем самым “испортить” программу, то админ может заподозрить неладное и вообще удалить Partition Magic. Сделаем программу-загрузчик, которая при запуске выполнит оригинальный ехе-файл Partition Magic’a и изменит в памяти нужные байты (файл же останется полностью неизменным!). Для создания загрузчика воспользуемся программой DZAPatcher.

Создание загрузчика (лодера — loader) не требует больших знаний. Сначала выбирается исполняемый файл программы, в которой будут проводиться изменения. Далее заполняем пункт “Offset” — здесь мы указываем виртуальный адрес (который будет сформирован в памяти). У нас виртуальный адрес: 00429A32. Чуть ниже вводим оригинальный байт (84), который мы будем изменять, а еще ниже — байт, на который мы изменим (85). Нажимаем кнопку создания лодера, и в папке с РМ появляется файл loader.exe. Лодер готов. Теперь в нашей власти ока-

зался жесткий диск, над которым мы можем творить совершенно безумные вещи! :)

Исследование защиты Antivirus Kaspersky v3.5

Partition Magic был покорен, но остался очень тормозящий систему антивирусный пакет г-на Касперского. При попытке закрыть антивирус, он выдает окошко для ввода пароля (рис.2).

Сначала скопируем файл аврсс.ехе из соответствующей папки AVP на дискетку и начнем исследования в “домашней” обстановке (придется отыскать версию 3.5). Запустим аврсс.ехе и поставим пароллизацию на работу с “чувствительными” настройками антивируса. Далее нажимаем выключение антивируса — появится окошко, показанное на рис.2. После ввода неправильного пароля запустим отладчик и поставим бряк на функцию “hmemscr”. Теперь нажимаем ОК, и программа прервется в модуле Kernel. После этого возвращаемся в модуль аврсс. Отладчик покажет следующий код:

```
:00420BB4 C20400      ret 0004
```

В отладчике покажется модуль самой программы (что нам и нужно), теперь пойдем путем простейшей трассировки по F10 до следующего участка кода:

```
:00420EBB 7426      je 00420EE3 ;<<ключевой переход
:00420EBD 51      push ecx
:00420EBE 8BCC      mov ecx, esp
:00420EC0 8965E0      mov dword ptr [ebp-20], esp
:00420EC3 68742E4600 push 00462E74
:00420EC8 E80D310200 call 00443FDA
:00420ECD 8945C0      mov dword ptr [ebp-40], eax
:00420ED0 6A25      push 00000025
:00420ED2 8B45DC      mov eax, dword ptr [ebp-24]
:00420ED5 8B4820      mov ecx, dword ptr [eax+20]
:00420ED8 51      push ecx
:00420ED9 E8AB350000 call 00424489 ;<<пароль не
:00420EDE 83C40C      add esp, 0000000C
:00420EE1 EB08      jmp 00420EEB
:00420EE3 7426      je 00420EE3 ;<<ключевой переход
:00420EE5 51      push ecx
:00420EE6 8BCC      mov ecx, esp
:00420EE8 8965E0      mov dword ptr [ebp-20], esp
:00420EEB 68742E4600 push 00462E74
:00420EEC E80D310200 call 00443FDA
:00420EEF 8945C0      mov dword ptr [ebp-40], eax
:00420EF0 6A25      push 00000025
:00420EF2 8B45DC      mov eax, dword ptr [ebp-24]
:00420EF5 8B4820      mov ecx, dword ptr [eax+20]
:00420EF8 51      push ecx
:00420EF9 E8AB350000 call 00424489 ;<<пароль не
:00420EFD 83C40C      add esp, 0000000C
:00420F01 EB08      jmp 00420F0B
:00420F0B 7426      je 00420F0B ;<<ключевой переход
:00420F0D 51      push ecx
:00420F0E 8BCC      mov ecx, esp
:00420F10 8965E0      mov dword ptr [ebp-20], esp
:00420F13 68742E4600 push 00462E74
:00420F16 E80D310200 call 00443FDA
:00420F19 8945C0      mov dword ptr [ebp-40], eax
:00420F1C 6A25      push 00000025
:00420F1E 8B45DC      mov eax, dword ptr [ebp-24]
:00420F21 8B4820      mov ecx, dword ptr [eax+20]
:00420F24 51      push ecx
:00420F25 E8AB350000 call 00424489 ;<<пароль не
:00420F29 83C40C      add esp, 0000000C
:00420F2D EB08      jmp 00420F37
:00420F37 7426      je 00420F37 ;<<ключевой переход
:00420F39 51      push ecx
:00420F3A 8BCC      mov ecx, esp
:00420F3C 8965E0      mov dword ptr [ebp-20], esp
:00420F3F 68742E4600 push 00462E74
:00420F42 E80D310200 call 00443FDA
:00420F45 8945C0      mov dword ptr [ebp-40], eax
:00420F48 6A25      push 00000025
:00420F4A 8B45DC      mov eax, dword ptr [ebp-24]
:00420F4D 8B4820      mov ecx, dword ptr [eax+20]
:00420F50 51      push ecx
:00420F51 E8AB350000 call 00424489 ;<<пароль не
:00420F55 83C40C      add esp, 0000000C
:00420F59 EB08      jmp 00420F6B
:00420F6B 7426      je 00420F6B ;<<ключевой переход
:00420F6D 51      push ecx
:00420F6E 8BCC      mov ecx, esp
:00420F70 8965E0      mov dword ptr [ebp-20], esp
:00420F73 68742E4600 push 00462E74
:00420F76 E80D310200 call 00443FDA
:00420F79 8945C0      mov dword ptr [ebp-40], eax
:00420F7C 6A25      push 00000025
:00420F7E 8B45DC      mov eax, dword ptr [ebp-24]
:00420F81 8B4820      mov ecx, dword ptr [eax+20]
:00420F84 51      push ecx
:00420F85 E8AB350000 call 00424489 ;<<пароль не
:00420F89 83C40C      add esp, 0000000C
:00420F8D EB08      jmp 00420F97
:00420F97 7426      je 00420F97 ;<<ключевой переход
:00420F99 51      push ecx
:00420F9A 8BCC      mov ecx, esp
:00420F9C 8965E0      mov dword ptr [ebp-20], esp
:00420F9F 68742E4600 push 00462E74
:00420FA2 E80D310200 call 00443FDA
:00420FA5 8945C0      mov dword ptr [ebp-40], eax
:00420FA8 6A25      push 00000025
:00420FAA 8B45DC      mov eax, dword ptr [ebp-24]
:00420FAD 8B4820      mov ecx, dword ptr [eax+20]
:00420FAE 51      push ecx
:00420FAF E8AB350000 call 00424489 ;<<пароль не
:00420FB3 83C40C      add esp, 0000000C
:00420FB7 EB08      jmp 00420F89
:00420F89 7426      je 00420F89 ;<<ключевой переход
:00420F8B 51      push ecx
:00420F8C 8BCC      mov ecx, esp
:00420F8E 8965E0      mov dword ptr [ebp-20], esp
:00420F91 68742E4600 push 00462E74
:00420F94 E80D310200 call 00443FDA
:00420F97 8945C0      mov dword ptr [ebp-40], eax
:00420F9A 6A25      push 00000025
:00420F9C 8B45DC      mov eax, dword ptr [ebp-24]
:00420F9F 8B4820      mov ecx, dword ptr [eax+20]
:00420FA0 51      push ecx
:00420FA1 E8AB350000 call 00424489 ;<<пароль не
:00420FA5 83C40C      add esp, 0000000C
:00420FA9 EB08      jmp 00420F8B
:00420F8B 7426      je 00420F8B ;<<ключевой переход
:00420F8D 51      push ecx
:00420F8E 8BCC      mov ecx, esp
:00420F90 8965E0      mov dword ptr [ebp-20], esp
:00420F93 68742E4600 push 00462E74
:00420F96 E80D310200 call 00443FDA
:00420F99 8945C0      mov dword ptr [ebp-40], eax
:00420F9C 6A25      push 00000025
:00420F9E 8B45DC      mov eax, dword ptr [ebp-24]
:00420FA1 8B4820      mov ecx, dword ptr [eax+20]
:00420FA2 51      push ecx
:00420FA3 E8AB350000 call 00424489 ;<<пароль не
:00420FA7 83C40C      add esp, 0000000C
:00420FAB EB08      jmp 00420F8D
:00420F8D 7426      je 00420F8D ;<<ключевой переход
:00420F8F 51      push ecx
:00420F90 8BCC      mov ecx, esp
:00420F92 8965E0      mov dword ptr [ebp-20], esp
:00420F95 68742E4600 push 00462E74
:00420F98 E80D310200 call 00443FDA
:00420F9B 8945C0      mov dword ptr [ebp-40], eax
:00420F9E 6A25      push 00000025
:00420FA0 8B45DC      mov eax, dword ptr [ebp-24]
:00420FA3 8B4820      mov ecx, dword ptr [eax+20]
:00420FA4 51      push ecx
:00420FA5 E8AB350000 call 00424489 ;<<пароль не
:00420FA9 83C40C      add esp, 0000000C
:00420FAD EB08      jmp 00420F8F
:00420F8F 7426      je 00420F8F ;<<ключевой переход
:00420F91 51      push ecx
:00420F92 8BCC      mov ecx, esp
:00420F94 8965E0      mov dword ptr [ebp-20], esp
:00420F97 68742E4600 push 00462E74
:00420F9A E80D310200 call 00443FDA
:00420F9D 8945C0      mov dword ptr [ebp-40], eax
:00420FA0 6A25      push 00000025
:00420FA2 8B45DC      mov eax, dword ptr [ebp-24]
:00420FA5 8B4820      mov ecx, dword ptr [eax+20]
:00420FA6 51      push ecx
:00420FA7 E8AB350000 call 00424489 ;<<пароль не
:00420FAB 83C40C      add esp, 0000000C
:00420FAD EB08      jmp 00420F91
:00420F91 7426      je 00420F91 ;<<ключевой переход
:00420F93 51      push ecx
:00420F94 8BCC      mov ecx, esp
:00420F96 8965E0      mov dword ptr [ebp-20], esp
:00420F99 68742E4600 push 00462E74
:00420F9C E80D310200 call 00443FDA
:00420F9F 8945C0      mov dword ptr [ebp-40], eax
:00420FA2 6A25      push 00000025
:00420FA4 8B45DC      mov eax, dword ptr [ebp-24]
:00420FA7 8B4820      mov ecx, dword ptr [eax+20]
:00420FA8 51      push ecx
:00420FA9 E8AB350000 call 00424489 ;<<пароль не
:00420FAD 83C40C      add esp, 0000000C
:00420FAF EB08      jmp 00420F93
:00420F93 7426      je 00420F93 ;<<ключевой переход
:00420F95 51      push ecx
:00420F96 8BCC      mov ecx, esp
:00420F98 8965E0      mov dword ptr [ebp-20], esp
:00420F9B 68742E4600 push 00462E74
:00420F9E E80D310200 call 00443FDA
:00420FA1 8945C0      mov dword ptr [ebp-40], eax
:00420FA4 6A25      push 00000025
:00420FA6 8B45DC      mov eax, dword ptr [ebp-24]
:00420FA9 8B4820      mov ecx, dword ptr [eax+20]
:00420FAA 51      push ecx
:00420FAB E8AB350000 call 00424489 ;<<пароль не
:00420FAF 83C40C      add esp, 0000000C
:00420FB1 EB08      jmp 00420F95
:00420F95 7426      je 00420F95 ;<<ключевой переход
:00420F97 51      push ecx
:00420F98 8BCC      mov ecx, esp
:00420F9A 8965E0      mov dword ptr [ebp-20], esp
:00420F9D 68742E4600 push 00462E74
:00420F9E E80D310200 call 00443FDA
:00420FA1 8945C0      mov dword ptr [ebp-40], eax
:00420FA4 6A25      push 00000025
:00420FA6 8B45DC      mov eax, dword ptr [ebp-24]
:00420FA9 8B4820      mov ecx, dword ptr [eax+20]
:00420FAA 51      push ecx
:00420FAB E8AB350000 call 00424489 ;<<пароль не
:00420FAD 83C40C      add esp, 0000000C
:00420FAD EB08      jmp 00420F97
:00420F97 7426      je 00420F97 ;<<ключевой переход
:00420F99 51      push ecx
:00420F9A 8BCC      mov ecx, esp
:00420F9C 8965E0      mov dword ptr [ebp-20], esp
:00420F9F 68742E4600 push 00462E74
:00420FA0 E80D310200 call 00443FDA
:00420FA3 8945C0      mov dword ptr [ebp-40], eax
:00420FA6 6A25      push 00000025
:00420FA8 8B45DC      mov eax, dword ptr [ebp-24]
:00420FAB 8B4820      mov ecx, dword ptr [eax+20]
:00420FAC 51      push ecx
:00420FAD E8AB350000 call 00424489 ;<<пароль не
:00420FB1 83C40C      add esp, 0000000C
:00420FB3 EB08      jmp 00420FA3
:00420FA3 7426      je 00420FA3 ;<<ключевой переход
:00420FA5 51      push ecx
:00420FA6 8BCC      mov ecx, esp
:00420FA8 8965E0      mov dword ptr [ebp-20], esp
:00420FAB 68742E4600 push 00462E74
:00420FAC E80D310200 call 00443FDA
:00420FAD 8945C0      mov dword ptr [ebp-40], eax
:00420FAE 6A25      push 00000025
:00420FAC 8B45DC      mov eax, dword ptr [ebp-24]
:00420FAD 8B4820      mov ecx, dword ptr [eax+20]
:00420FAE 51      push ecx
:00420FAF E8AB350000 call 00424489 ;<<пароль не
:00420FB3 83C40C      add esp, 0000000C
:00420FB5 EB08      jmp 00420FA5
:00420FA5 7426      je 00420FA5 ;<<ключевой переход
:00420FA7 51      push ecx
:00420FA8 8BCC      mov ecx, esp
:00420FAA 8965E0      mov dword ptr [ebp-20], esp
:00420FAD 68742E4600 push 00462E74
:00420FAE E80D310200 call 00443FDA
:00420FAD 8945C0      mov dword ptr [ebp-40], eax
:00420FAE 6A25      push 00000025
:00420FAF 8B45DC      mov eax, dword ptr [ebp-24]
:00420FB0 8B4820      mov ecx, dword ptr [eax+20]
:00420FB1 51      push ecx
:00420FB2 E8AB350000 call 00424489 ;<<пароль не
:00420FB6 83C40C      add esp, 0000000C
:00420FB8 EB08      jmp 00420FA7
:00420FA7 7426      je 00420FA7 ;<<ключевой переход
:00420FA9 51      push ecx
:00420FAA 8BCC      mov ecx, esp
:00420FAC 8965E0      mov dword ptr [ebp-20], esp
:00420FAD 68742E4600 push 00462E74
:00420FAE E80D310200 call 00443FDA
:00420FAD 8945C0      mov dword ptr [ebp-40], eax
:00420FAE 6A25      push 00000025
:00420FAF 8B45DC      mov eax, dword ptr [ebp-24]
:00420FB0 8B4820      mov ecx, dword ptr [eax+20]
:00420FB1 51      push ecx
:00420FB2 E8AB350000 call 00424489 ;<<пароль не
:00420FB6 83C40C      add esp, 0000000C
:00420FB8 EB08      jmp 00420FA9
:00420FA9 7426      je 00420FA9 ;<<ключевой переход
:00420FAB 51      push ecx
:00420FAC 8BCC      mov ecx, esp
:00420FAE 8965E0      mov dword ptr [ebp-20], esp
:00420FB1 68742E4600 push 00462E74
:00420FB2 E80D310200 call 00443FDA
:00420FB1 8945C0      mov dword ptr [ebp-40], eax
:00420FB2 6A25      push 00000025
:00420FB3 8B45DC      mov eax, dword ptr [ebp-24]
:00420FB4 8B4820      mov ecx, dword ptr [eax+20]
:00420FB5 51      push ecx
:00420FB6 E8AB350000 call 00424489 ;<<пароль не
:00420FBA 83C40C      add esp, 0000000C
:00420FBC EB08      jmp 00420FAB
:00420FAB 7426      je 00420FAB ;<<ключевой переход
:00420FAD 51      push ecx
:00420FAE 8BCC      mov ecx, esp
:00420FB0 8965E0      mov dword ptr [ebp-20], esp
:00420FB1 68742E4600 push 00462E74
:00420FB2 E80D310200 call 00443FDA
:00420FB1 8945C0      mov dword ptr [ebp-40], eax
:00420FB2 6A25      push 00000025
:00420FB3 8B45DC      mov eax, dword ptr [ebp-24]
:00420FB4 8B4820      mov ecx, dword ptr [eax+20]
:00420FB5 51      push ecx
:00420FB6 E8AB350000 call 00424489 ;<<пароль не
:00420FBA 83C40C      add esp, 0000000C
:00420FBC EB08      jmp 00420FAD
:00420FAD 7426      je 00420FAD ;<<ключевой переход
:00420FAD 51      push ecx
:00420FAE 8BCC      mov ecx, esp
:00420FB0 8965E0      mov dword ptr [ebp-20], esp
:00420FB1 68742E4600 push 00462E74
:00420FB2 E80D310200 call 00443FDA
:00420FB1 8945C0      mov dword ptr [ebp-40], eax
:00420FB2 6A25      push 00000025
:00420FB3 8B45DC      mov eax, dword ptr [ebp-24]
:00420FB4 8B4820      mov ecx, dword ptr [eax+20]
:00420FB5 51      push ecx
:00420FB6 E8AB350000 call 00424489 ;<<пароль не
:00420FBA 83C40C      add esp, 0000000C
:00420FBC EB08      jmp 00420FAD
:00420FAD 7426      je 00420FAD ;<<ключевой переход
:00420FAD 51      push ecx
:00420FAE 8BCC      mov ecx, esp
:00420FB0 8965E0      mov dword ptr [ebp-20], esp
:00420FB1 68742E4600 push 00462E74
:00420FB2 E80D310200 call 00443FDA
:00420FB1 8945C0      mov dword ptr [ebp-40], eax
:00420FB2 6A25      push 00000025
:00420FB3 8B45DC      mov eax, dword ptr [ebp-24]
:00420FB4 8B4820      mov ecx, dword ptr [eax+20]
:00420FB5 51      push ecx
:00420FB6 E8AB350000 call 00424489 ;<<пароль не
:00420FBA 83C40C      add esp, 0000000C
:00420FBC EB08      jmp 00420FAD
:00420FAD 7426      je 00420FAD ;<<ключевой переход
:00420FAD 51      push ecx
:00420FAE 8BCC      mov ecx, esp
:00420FB0 8965E0      mov dword ptr [ebp-20], esp
:00420FB1 68742E4600 push 00462E74
:00420FB2 E80D310200 call 00443FDA
:00420FB1 8945C0      mov dword ptr [ebp-40], eax
:00420FB2 6A25      push 00000025
:00420FB3 8B45DC      mov eax, dword ptr [ebp-24]
:00420FB4 8B4820      mov ecx, dword ptr [eax+20]
:00420FB5 51      push ecx
:00420FB6 E8AB350000 call 00424489 ;<<пароль не
:00420FBA 83C40C      add esp, 0000000C
:00420FBC EB08      jmp 00420FAD
:00420FAD 7426      je 00420FAD ;<<ключевой переход
:00420FAD 51      push ecx
:00420FAE 8BCC      mov ecx, esp
:00420FB0 8965E0      mov dword ptr [ebp-20], esp
:00420FB1 68742E4600 push 00462E74
:00420FB2 E80D310200 call 00443FDA
:00420FB1 8945C0      mov dword ptr [ebp-40], eax
:00420FB2 6A25      push 00000025
:00420FB3 8B45DC      mov eax, dword ptr [ebp-24]
:00420FB4 8B4820      mov ecx, dword ptr [eax+20]
:00420FB5 51      push ecx
:00420FB6 E8AB350000 call 00424489 ;<<пароль не
:00420FBA 83C40C      add esp, 0000000C
:00420FBC EB08      jmp 00420FAD
:00420FAD 7426      je 00420FAD ;<<ключевой переход
:00420FAD 51      push ecx
:00420FAE 8BCC      mov ecx, esp
:00420FB0 8965E0      mov dword ptr [ebp-20], esp
:00420FB1 68742E4600 push 00462E74
:00420FB2 E80D310200 call 00443FDA
:00420FB1 8945C0      mov dword ptr [ebp-40], eax
:00420FB2 6A25      push 00000025
:00420FB3 8B45DC      mov eax, dword ptr [ebp-24]
:00420FB4 8B4820      mov ecx, dword ptr [eax+20]
:00420FB5 51      push ecx
:00420FB6 E8AB350000 call 00424489 ;<<пароль не
:00420FBA 83C40C      add esp, 0000000C
:00420FBC EB08      jmp 00420FAD
:00420FAD 7426      je 00420FAD ;<<ключевой переход
:00420FAD 51      push ecx
:00420FAE 8BCC      mov ecx, esp
:00420FB0 8965E0      mov dword ptr [ebp-20], esp
:00420FB1 68742E4600 push 00462E74
:00420FB2 E80D310200 call 00443FDA
:00420FB1 8945C0      mov dword ptr [ebp-40], eax
:00420FB2 6A25      push 00000025
:00420FB3 8B45DC      mov eax, dword ptr [ebp-24]
:00420FB4 8B4820      mov ecx, dword ptr [eax+20]
:00420FB5 51      push ecx
:00420FB6 E8AB350000 call 00424489 ;<<пароль не
:00420FBA 83C40C      add esp, 0000000C
:00420FBC EB08      jmp 00420FAD
:00420FAD 7426      je 00420FAD ;<<ключевой переход
:00420FAD 51      push ecx
:00420FAE 8BCC      mov ecx, esp
:00420FB0 8965E0      mov dword ptr [ebp-20], esp
:00420FB1 68742E4600 push 00462E74
:00420FB2 E80D310200 call 00443FDA
:00420FB1 8945C0      mov dword ptr [ebp-40], eax
:00420FB2 6A25      push 00000025
:00420FB3 8B45DC      mov eax, dword ptr [ebp-24]
:00420FB4 8B4820      mov ecx, dword ptr [eax+20]
:00420FB5 51      push ecx
:00420FB6 E8AB350000 call 00424489 ;<<пароль не
:00420FBA 83C40C      add esp, 0000000C
:00420FBC EB08      jmp 00420FAD
:00420FAD 7426      je 00420FAD ;<<ключевой переход
:00420FAD 51      push ecx
:00420FAE 8BCC      mov ecx, esp
:00420FB0 8965E0      mov dword ptr [ebp-20], esp
:00420FB1 68742E4600 push 00462E74
:00420FB2 E80D310200 call 00443FDA
:00420FB1 8945C0      mov dword ptr [ebp-40], eax
:00420FB2 6A25      push 00000025
:00420FB3 8B45DC      mov eax, dword ptr [ebp-24]
:00420FB4 8B4820      mov ecx, dword ptr [eax+20]
:00420FB5 51      push ecx
:00420FB6 E8AB350000 call 00424489 ;<<пароль не
:00420FBA 83C40C      add esp, 0000000C
:00420FBC EB08      jmp 00420FAD
:00420FAD 7426      je 00420FAD ;<<ключевой переход
:00420FAD 51      push ecx
:00420FAE 8BCC      mov ecx, esp
:00420FB0 8965E0      mov dword ptr [ebp-20], esp
:00420FB1 68742E4600 push 00462E74
:00420FB2 E80D310200 call 00443FDA
:00420FB1 8945C0      mov dword ptr [ebp-40], eax
:00420FB2 6A25      push 00000025
:00420FB3 8B45DC      mov eax, dword ptr [ebp-24]
:00420FB4 8B4820      mov ecx, dword ptr [eax+20]
:00420FB5 51      push ecx
:00420FB6 E8AB350000 call 00424489 ;<<пароль не
:00420FBA 83C40C      add esp, 0000000C
:00420FBC EB08      jmp 00420FAD
:00420FAD 7426      je 00420FAD ;<<ключевой переход
:00420FAD 51      push ecx
:00420FAE 8BCC      mov ecx, esp
:00420FB0 8965E0      mov dword ptr [ebp-20], esp
:00
```

И.ШИРКО,
г.Минск, БГУИР, ФИТУ.

Microsoft Agent Часть II

В [1] мы научились применять технологию Microsoft Agent в web-страничках. В этот раз мы будем работать с Delphi. Убедитесь, что установлены все необходимые компоненты (см. первую часть), и можно отправляться в увлекательное путешествие по миру MsAgent.

Запустите Delphi, и в меню "Component" выберите пункт "Import ActiveX Control...". В появившемся диалоговом окне нужно выделить строку "Microsoft Agent Control..." и нажать кнопку "Install". Далее произойдет стандартный процесс установки нового компонента. После окончания инсталляции создайте новое приложение и поместите на форму полученный компонент — Agent1:TAgent (он должен находиться на закладке ActiveX).



Вот этот почти "ноль-ноль-семь" будет помогать нам создавать приложения, которые станут работать с технологией MsAgent. Начнем с небольшого примера — маленького "костяка", который можно будет использовать при разработке более сложных программ.

Установите свойству Connected компонента Agent1 значение True, затем объявите две глобальные переменные:

```
Var
//для хранения персонажа
Character:IAgentCtlCharacterEx;
//для получения состояния персонажа
Request:IAgentCtlRequest;

Для события OnCreate формы запишите такую процедуру:
procedure TForm1.FormCreate(Sender: TObject);
begin
//загружаем персонаж "Джин"
Request:=Agent1.Characters.Load('genie','genie.acs');
//получаем объект персонажа Джин
Character:=Agent1.Characters.Character('genie') as
IAgentCtlCharacterEx;
//Джин появляется
Request:=Character.Show(False);
//Джин здоровается и вкратце рассказывает о себе
Request:=Character.Speak('Здравствуйте!'+Character.Description, EmptyParam);
//Джин играет анимацию "Greet"
Request:=Character.Play('Greet');
end;
```

При закрытии формы нужно выгрузить персонаж:

```
procedure TForm1.FormClose(Sender: TObject; var Action:
TCloseAction);
begin
Agent1.Characters.Unload('genie');
end;
```

Если запустить эту программу, то вместе с формой появится Джин, который поздоровается и на несколько секунд застынет в приветственном поклоне. Теперь будем дополнять данный пример.

Как уже упоминалось в первой части, персонажи умеют проигрывать различные анимации, при этом для каждого персонажа могут определяться свои анимации. Поэтому иногда требуется получить все допустимые анимации для персонажа, что и делает следующая процедура:

```
Procedure GetAgentAnim(st:TStrings);
var
AnEnum: IEnumVariant;
flag: Cardinal;
V: OleVariant;
begin
```

```
//получаем интерфейс анимаций агента
AnEnum:=(Character.AnimationNames.Enum) as IEnumVariant;
//Сбрасываем указатель списка на начало
AnEnum.Reset;
//перебираем все анимации и добавляем их в наш список
repeat
AnEnum.Next(1, V, flag);
if VarToStr(V) <> '' then
st.Add(V);
until flag=0;
end;
```

Давайте проверим работоспособность данной процедуры. Добавьте на форму кнопку Button1 и список ListBox1. Дайте кнопке заголовок "Список анимаций" и запишите процедуру обработки ее нажатия:

```
procedure TForm1.Button1Click(Sender: TObject);
begin
GetAgentAnim(ListBox1.Items);
end;
```

Данная процедура записывает список анимаций загруженного персонажа в компонент ListBox1. А теперь сделаем так, чтобы при двойном щелчке мышкой по ListBox'y Джин проигрывал выбранную анимацию:

```
procedure TForm1.ListBox1DblClick(Sender: TObject);
begin
Request := Character.Play(ListBox1.Items[ListBox1.ItemIndex]);
end;
```

Теперь посмотрим, зачем нам переменная Request (ее мы объявили как глобальную). Именно она служит для синхронизации действий персонажа (либо персонажей, если их несколько). Интерфейс IAgentCtlRequest имеет одно очень полезное свойство — Status. В зависимости от состояния запроса, который в данный момент выполняет персонаж, это свойство принимает значения, приведенные в табл.1.

Табл. 1

Status	Пояснения
0	Запрос успешно выполнен
1	Запрос "провалился"
2	Запрос не выполняется, т.к. ждет завершения обработки других запросов
3	Запрос прерван
4	Запрос обрабатывается

На основе этой таблички легко написать процедуру, которая будет дожидаться выполнения всех запросов к персонажу, а затем вернет программе управление:

```
procedure WaitFor(Request:IAgentCtlRequest);
begin
repeat
Application.ProcessMessages;
Status := Request.Status;
until (Request.Status <> 2) and (Request.Status <> 4);
end;
```

Для примера добавьте на форму кнопку "Пример WaitFor":

```
procedure TForm1.Button3Click(Sender: TObject);
begin
//Джин произносит фразу
Request := Character.Speak('Доброе утро', EmptyParam);
(waitfor(Request));
//Показываем сообщение
ShowMessage('Все');
end;
```

Если нажать на кнопку, то Джин начнет говорить, и сразу же появится сообщение. А если снять комментарии со строки waitfor(Request);, то сообщение появится только после того как Джин закончит говорить.

А сейчас "научим" Джина вслух произносить текущее время. Для этого ему нельзя подсунуть строку типа "21 час 44 минуты 32 секунды", эту строку он произнесет так: "двадцать один час, сорок четвертого минуты, тридцать второго секунды". Джину нужно полностью "разжевать",



Табл. 2

Язык	ID	Язык	ID
Саудовской Аравии	\$401	Итальянский	\$410
Баскский	\$42D	Японский	\$411
Китайский (упрощенный)	\$804	Корейский	\$412
Китайский (традиционный)	\$404	Норвежский	\$414
Хорватский	\$41A	Польский	\$415
Чешский	\$405	Португальский (Португалия)	\$816
Датский	\$406	Португальский (Бразилия)	\$416
Английский (британский)	\$809	Румынский	\$418
Английский (американский)	\$409	Русский	\$419
Финский	\$40B	Словенский	\$424
Французский	\$40C	Испанский	\$C0A
Немецкий	\$407	Шведский	\$41D
Греческий	\$408	Таиландский	\$41E
Венгерский	\$40E	Турецкий	\$41F

Character.Speak('Pit=18400\Spd=100\Wap="" +str+"="" +tinetostr(DT)+"\ ", EmptyParam); end;

И напоследок рассмотрим некоторые параметры голосового движка. На один компьютер можно установить поддержку нескольких языков. Чтобы переключаться между ними, нужно изменять параметр персонажа LanguageID. В табл.2 приведены идентификаторы к некоторым языкам. Символ "\$" означает, что число представлено в шестнадцатеричной системе счисления.

Пример:

Character.LanguageID:=\$409; — переключаемся на английский язык.

Также можно изменять и сам голос. Для этого имеется свойство TTSModeID. В табл.3 приведен список голосов для русского и английского языков.

Табл. 3

Голос	Идентификатор
Женский #1, английский (американский)	{CA141FD0-AC7F-11D1-97A3-006008273008}
Женский #2, английский (американский)	{CA141FD0-AC7F-11D1-97A3-006008273009}
Мужской #1, английский (американский)	{CA141FD0-AC7F-11D1-97A3-006008273000}
Мужской #2, английский (американский)	{CA141FD0-AC7F-11D1-97A3-006008273001}
Мужской #3, английский (американский)	{CA141FD0-AC7F-11D1-97A3-006008273002}
Мужской #4, английский (американский)	{CA141FD0-AC7F-11D1-97A3-006008273003}
Мужской #5, английский (американский)	{CA141FD0-AC7F-11D1-97A3-006008273004}
Мужской #6, английский (американский)	{CA141FD0-AC7F-11D1-97A3-006008273005}
Мужской #7, английский (американский)	{CA141FD0-AC7F-11D1-97A3-006008273006}
Мужской #8, английский (американский)	{CA141FD0-AC7F-11D1-97A3-006008273007}
Кэрл, английский (британский)	{227A0E40-A92A-11d1-B17B-0020AFED142E}
Питер, английский (британский)	{227A0E41-A92A-11d1-B17B-0020AFED142E}
Светлана, русский	{06377F80-D48E-11d1-B17B-0020AFED142E}
Борис, русский	{06377F81-D48E-11d1-B17B-0020AFED142E}

Пример:

Character.TTSModeID:='{06377F80-D48E-11d1-B17B-0020AFED142E}'; — устанавливается женский голос для русского языка. Так что можете поиздеваться над Джином, заставляя его разговаривать женским голосом ;)).

На сегодня все. В следующей части мы научимся управлять Джином при помощи голосовых команд.

Литература

1. И.Ширко. MicrosoftAgent. Часть I. — Радиомир. Ваш компьютер, 2004, №3, С.35.

что именно он должен произносить. Так что будем давать ему такую строку: "21 час, 40 четыре минуты, 30 две секунды". Т.е. число, стоящее в женском роде (44 минуты, 32 секунды), разделяем на составляющие (40 четыре минуты, 30 две секунды).

Добавьте на форму очередную кнопку "Сколько времени?" и запишите следующую процедуру:

```
procedure TForm1.Button1Click(Sender: TObject);
var
```

```
buf, str: string;
Hour, Min, Sec, Msec: Word;
Hours, Mins, Secs: string;
hs, ms, ss: string;
DT: TDateTime;
```

```
begin
```

```
hs:='';
ms:='';
ss:='';
//узнаем текущее время
DT:=Now;
```

```
//разбиваем его на составляющие
DecodeTime(DT, Hour, Min, Sec, MSec);
```

```
//смотрим, какое окончание должно быть у слова "час"
```

```
case hour of
  2..4, 22, 23: hs:='a';
  0, 5..20: hs:='ov';
```

```
end;
```

```
//окончание слова "минута"
```

```
case min of
  1, 21, 31, 41, 51: ms:='a';
  2..4, 22..24, 32..34, 42..44, 52..54: ms:='y';
```

```
end;
```

```
//окончание слова "секунда"
```

```
case sec of
  1, 21, 31, 41, 51: ss:='a';
  2..4, 22..24, 32..34, 42..44, 52..54: ss:='y';
```

```
end;
```

```
//часы не будем преобразовывать, т.к. слово "час" мужского рода
```

```
hours:=inttostr(hour);
```

```
//при необходимости разбиваем минуты на слова
mins:=inttostr(min);
```

```
if (mins[length(mins)]='1') and (min<>11) then
```

```
begin
```

```
if min>10 then buf:=mins[1]+'0 '
else buf:='';
mins:=buf+'одна';
```

```
end
```

```
else
```

```
begin
```

```
if (mins[length(mins)]='2') and (min<>12) then
begin
if min>10 then buf:=mins[1]+'0 '
else buf:='';
mins:=buf+'две';
```

```
end
```

```
end;
```

```
//при необходимости разбиваем секунды на слова
```

```
secs:=inttostr(sec);
```

```
if (secs[length(secs)]='1') and (sec<>11) then
```

```
begin
```

```
if sec>10 then buf:=secs[1]+'0 '
else buf:='';
secs:=buf+'одна';
```

```
end
```

```
else
```

```
begin
```

```
if (secs[length(secs)]='2') and (sec<>12) then
begin
if sec>10 then buf:=secs[1]+'0 '
else buf:='';
secs:=buf+'две';
```

```
end;
```

```
end;
```

```
//составляем строку, которую должен произнести Джин
str:=hours+' час'+hs+', '+mins+' минут'+ms+', '+secs+' секунд'+ss;
```

```
//Джин произносит строку str, а над ним отображается
```

```
//текущее время (рис.1)
```

```
Request :=
```

23:29:50

Рис. 1





В.ПОНОМАРЕВ,
г.Витебск,
ВГУ им.П.М.Машерова.

Верификация и спецификация в программировании

Вряд ли кто-то мог предположить в начале XX века, когда зарождалась математическая логика, и даже в 30...40-е годы, в пору ее расцвета, что эта чисто теоретическая дисциплина, относящаяся скорее к области "философии" математики, найдет практическое применение. И тем не менее, это произошло уже в 50-е годы, в первое десятилетие компьютерной эры, а в 70-е годы появился даже термин "вычислительная логика" [1].

Основным объектом изучения в математической логике являются различные исчисления. В понятие "исчисления" входят такие основные компоненты:

- язык (формальный) исчисления;
- аксиомы исчисления;
- правила вывода.

Понятие "исчисление" позволяет дать строгое математическое определение понятию "доказательство" и получить точные утверждения о невозможности доказательства тех или иных предложений теории. Еще одним замечательным достижением математической логики является нахождение математического определения понятия "алгоритм", т.е. эффективной процедуры для решения задач из того или иного (бесконечного) класса задач. Интуитивно понятие "алгоритм" использовалось очень давно. Выдающийся мыслитель XVII...XVIII вв. Г.Лейбниц даже мечтал о нахождении универсального алгоритма для решения всех математических проблем. Точное определение понятия "алгоритм" позволило довольно быстро разрушить эту красивую утопию. А.Черг в 1936 г. показал, что невозможен алгоритм, который по произвольному утверждению, записанному на формальном языке элементарной арифметики, отвечал бы на вопрос: будет ли это утверждение истинно на натуральных числах? Далее оказалось, что даже в системе, описывающей "чистую логику" (исчисление предикатов), проблема доказуемости алгоритмически неразрешима. В последующие годы было обнаружено большое многообразие алгоритмически неразрешимых проблем во многих разделах математи-

ки. Большой вклад в разработку теории алгоритмов и решение алгоритмических проблем внесли Э.Пост, А.Тьюринг, С.Клини и советские математики А.И.Мальцев, П.С.Новиков и А.А.Марков.

Изучение исчислений составляет синтаксическую часть математической логики. Наиболее глубокое изучение понятия (синтаксического) доказательства в тех или иных исчислениях составляет самостоятельный раздел математической логики, который носит название теории доказательств. Наряду с синтаксическим изучением исчислений, проводится также семантическое изучение формальных языков математической логики. Основным понятием семантики является понятие истинности для выражений (формул, секвенций и т.п.) формального языка. Семантические понятия также получили точные математические определения, что дало возможность систематического и строгого изучения понятий истинности. Классическая семантика языка исчисления предикатов составила весьма богатый раздел математической логики — теорию моделей, которая активно развивается, а ее методы и результаты успешно применяются и в других областях математики (алгебре, анализе). Основателями теории моделей являются А.Тарский и А.И.Мальцев [2].

Исчисления позволяют формализовать многие разделы математики и других наук. Исчисление высказываний и исчисление предикатов являются формализациями логики — древнейшей науки о законах правильного мышления. Создание и изучение этих формализаций явилось важным этапом в развитии логики как науки. Первые попытки формализации логики связаны с именем Аристотеля и Дж.Буля, но действительная (и действенная) формализация логики была осуществлена только с созданием математической логики. Итальянский математик Пеано много сделал для разработки и популяризации формальных языков логики.

Написать большую программу не-

легко. Еще труднее написать ее правильно. Как убедиться в том, что программа работает именно так, как нам хочется? Как правило, в нашем распоряжении имеется лишь одно средство — отладка тестированием, т.е. исполнение программы на некоторых контрольных входных данных. Однако, как заметил Д.Грие, само слово "отладка" подразумевает наличие ошибок, избавиться от которых полностью очень трудно — сколько бы мух мы ни перебили, они снова появятся.

Чтобы быть абсолютно уверенным в том, что программа работает правильно, лучше всего было бы иметь доказательство ее правильности. Здесь существуют две возможности: доказывать правильность готовых программ (этот подход называется верификацией программ) и строить программы а priori правильно, параллельно с доказательством их правильности (такой подход называют синтезом программ). Синтез программ тесно связан с формальными спецификациями.

Идея построения доказательства правильности программы вместо ее отладки путем тестирования на отдельных входах принадлежит Дж.Маккарти. Соответствующая формальная система доказательства была построена в статье Т.Хоара, в которой использовались идеи более ранних работ П.Наура и Р.Флойда. Язык этой формальной системы, названный впоследствии логикой Хоара, является расширением языка логики предикатов утверждениями вида $P\{a\}Q$, где P и Q — формулы логики предикатов, а a — программа на некотором языке программирования. Это утверждение означает, что если до выполнения программы a была верна формула P , то после завершения a (если она завершается) будет верна формула Q (частичная правильность программы a относительно предположения P и постулирования Q).

Правила вывода логики Хоара содержат все аксиомы и правила вывода классического исчисления предикатов первого порядка, а также правила вывода формул вида



$P\{a\}Q$, зависящие от того языка, на котором пишутся программы а. С примером логики Хоара для простого языка программирования можно ознакомиться в [1].

В настоящее время главными представителями формального направления в спецификации про-

грамм является метод VDM (Vienna Development Method) и алгебраический метод.

Об этих, а также других понятиях, относящихся к теоретической информатике, я надеюсь прочитать в статьях постоянных авторов журнала.

Литература

1. Математическая логика в программировании. Сборник статей. Под ред. М.В.Захарьянцева и Ю.И.Янова. — М.: Мир, 1991 г.
2. Ю.Л.Ершов, Е.А.Палютин. Математическая логика. — М.: Наука, 1987 г.

Настройка COM-порта

О.ВАЛЬПА,

E-mail: sandh@yandex.ru

Однажды мне потребовалось состыковать компьютер с периферийным устройством по COM-порту. Но поскольку у этого устройства была нестандартная скорость обмена, невозможно было решить эту задачу стандартным способом...

Устройство позволяло работать с последовательным портом на скорости 50 бод. Но как известно, стандартные функции DOS (прерывание INT14h) и системные установки Windows позволяют настроить последовательный порт компьютера для работы только со скоростями 110, 300, 1200, 2400, 4800, 9600, 19200, 38400, 57600 и 115200 бод. Пришлось прибегнуть к прямой установке внутренних регистров микросхемы контроллера COM-порта. В частности, нужно было записать в регистры делителей контроллера новые значения для получения скорости 50 бод.

Адреса регистров контроллера известны из многих литературных источников, например, из [1]. Но для вычисления значения делителя необходимо знать тактовую частоту контроллера порта:

$$D = F[\Gamma\omega] : S[\text{бод}] : 16,$$

где:

- D — значение делителя;
- F — тактовая частота контроллера порта;
- S — скорость порта
- 16 — константа внутреннего делителя контроллера.

Тактовая частота контроллера порта не имеет стандартной величины и зависит от производителя материнской платы компьютера или адаптера последовательного COM-порта. Вот тут-то на выручку и пришла смекалка. А что если пойти от обратного — сначала установить стандартную скорость порта доступными средствами, и считать значение делителей? Тогда легко будет вычислить тактовую частоту контроллера порта по формуле:

$$F[\Gamma\omega] = D * S[\text{бод}] * 16.$$

Наконец, вычислив тактовую частоту, находим новые коэффициенты деления для необходимой нестандартной скорости и записываем их в регистры контроллера. В результате порт будет работать с необходимой нам скоростью, в том числе и нестандартной.

Описанное выше решение задачи было оформлено мною в небольшую программу, текст которой приведен ниже. Эта программа, запущенная на компьютере, автоматически производит настройку порта компьютера на скорость 1200 бод, считывает значения делителей из регистров контроллера и производит описанные

выше операции по вычислению тактовой частоты контроллера. После этого она выдает результат работы на экран монитора.

Текст программы

```

////////////////////////////////////
// Программа вычисления тактовой частоты контроллера COM-порта
//
// Файл: rate_com.cpp
// Дата: 25.10.2003
// Автор: Вальпа Олег
////////////////////////////////////
#include <stdio.h>
#include <conio.h>
#include <bios.h>
#include <stdlib.h>
#include <dos.h>

#define BAIT_INIT 0x8b // Байт инициализации COM-порта:
                        // 1200 бод, нечет., 8 бит, 1 стоп
#define INIT       0x00 // Адрес регистра инициализации COM-порта
#define PERED      0x01 // Адрес регистра приемника COM-порта
#define PRIEM      0x02 // Адрес регистра передатчика COM-порта
#define STAT       0x03 // Адрес регистра статуса COM-порта
#define COM        1    // Назначение номера COM-порта
#define SPEED      1200 // Скорость COM-порта

int i;
unsigned long f;
unsigned char s;
unsigned char b[8];

main()
{
    bioscom ( INIT, BAIT_INIT, COM); // Инициализация COM-порта
    // Читаем статус порта, и если буфер не пуст, очистим его
    if ((bioscom(STAT,0,COM)) & 0x100) != 0) bioscom(PRIEM,0,COM);

    s=(inp(0x2f8+3) | 0x80); outportb(0x2f8+3,s);
    for(i=0;i<=7;i++) b[i]=inp(0x2f8+i);
    s=(inp(0x2f8+3) & 0x7f); outportb(0x2f8+3,s);

    f=b[1]<<8 | b[0]; f=f * SPEED * 16;
    printf("\n\nТактовая частота[f]=%ldГц",f);
    printf("\n\nДля скорости порта=%dbод делитель ст.=%2x делитель"
           "мл.=%2x\n",SPEED,b[1],b[0]);
    printf("\n\nДля скорости порта=N бод делитель = f/16\n\n");

    printf("\n\nНажмите клавишу Enter для завершения...");
    getch();
}

```

Надеюсь, что эта программа поможет и вам при решении подобных задач, которые нет-нет, да и встречаются в нашей жизни.

Литература:

1. Гук М. Аппаратные средства IBM PC. Энциклопедия. — СПб.: ПитерКом, 1998. — 816 с.



DOWNGRADE

Р.КАРПАЧ,

E-mail: Commander-Norton@tut.by
www.fdd5-25.net

Записки пользователя Win 3.11

Downgrade — это когда люди, которым нечего делать, пытаются оживить мертвые компьютеры.

Из моей почты

Когда у меня появился первый notebook, то я подумал, что Windows 3.11 — это операционная система для него. Из разговора

1. Записка с рассуждениями

Скука смертная обуяла меня. Не могу уже смотреть на разноцветные Windows XP, читать обзоры непонятных ОС и пачками скачивать новые программы. Скучно... Эксперименты с DOS уже давно закончились. А все равно хочется чего-то свежего и нестандартного, хочется вырваться из этого засилья силикона и попасть в мир эдакого компьютерного модерна начала 90-х годов! Конечно, на чужих компьютерах, да и на работе сильно не разгонишься. Там все чинно и серьезно, а вот дома можно и позабавиться :-).

«Так про что же он пытается начать свой рассказ? — наверное, задумался читатель. Да, про то, как сделать так, чтобы Windows 3.11 работал на компьютерах с файловой системой FAT 32 и процессором с частотой выше 166 МГц. Кому же это нужно? Какая от этого практическая польза? Да как всегда в downgrade — никакой. Downgrade — это лишь способ развлечения и набора нового компьютерного опыта вперемешку со свежими впечатлениями. Почему же выбор пал именно на Windows 3.11? Все дело в том, что мне нужно было попытаться сделать что-то невозможное и невероятное. Просто чтобы удивить знакомых и читателей чем-нибудь нестандартным. Ну, например, разрешение рабочего стола 1024x768 пикселей и 75 Гц. Это реально? В Windows 95 с этим уже нет никаких проблем, но вот в системах Win 3.x уже вызовет большое затруднение. А работа в FAT 32, так это вообще фантастика!

Для полноты ощущений этот материал набирался исключительно под Windows 3.11 и MS WORKS 2.0, а затем конвертировался в *.doc файл Word 2000. Чтобы, как говорится, не отрываться от предмета рассказа (рис.1).

Сейчас стало модно расписывать те или иные возможности новой операционной системы, с понтом и смаком рассказывать о каждой детали, удивляя всех непонятным видом и нестандартными функциями. Я же решил доказать всем, и в первую очередь себе, что красивую картинку и супервозможности можно получить и с помощью старых и, на первый взгляд, ни для чего не пригодных программ. Было бы немного знаний и умений, а остальное придет! Для этого не нужно быть Гуром.

2. Записка технологическая — установка Windows 3.11

Дальше все будем делать вместе. Если вы захотели установить систему, то не рвитесь вперед, читайте весь текст внимательно, чтобы не возникло никаких неожиданных проблем. Вначале ознакомьтесь с конфигурацией моего компьютера, на котором проводился эксперимент.

Тестовая система:

AMD 450, 128 Mb RAM, 20 Gb HDD, 12 Mb 3dfx, 2 Mb video, sound MF 1000, monitor 17" Samtron 76E, CD-RW, network NE2000, modem Hays Accura 33600, OS WIN98-FAT32, DOS 7.10.

Чтобы проинсталлировать Windows 3.11 на компьютер, вам не нужно покупать диски, просто зайдите в Internet и воспользуйтесь этой ссылкой — www.fdd5-25.net/os/win311.zip. Затем распакуйте содержимое архива на отдельный раздел диска. Допустим, его имя будет D:\. Запишем Windows 3.11 в каталог D:\WIN311. Распаковали архив? Теперь нужно будет подготовить системные файлы для инсталляции. Еще раз лезем в Internet и пользуемся еще и этой ссылкой — www.fdd5-25.net/drivers/osr2kit.exe. Это сервис-пак для запуска Win 3.x на FAT32. Мы его тоже распаковываем на диск и запускаем из каталога PATCHES\WIN3.BUG\WIN3XSTART.EXE. Перезагружаем компьютер и производим изменения в загрузочных файлах. К сожалению, Windows 3.11 — это оболочка DOS, хотя и красивая, и многофункциональная одновременно. Поэтому перед установкой на наш компьютер мы должны будем подготовиться к некоторым нюансам, связанным с работой этой среды. В стандартный **autoexec.bat** добавим следующие строки.

```
@ECHO OFF
```

C:\WINDOWS\SMARTDRV.EXE — загружает драйвер устройства smartdrv.exe для выполнения двойной буферизации. Двойная буферизация обеспечивает совместимость с отдельными контроллерами жесткого диска, которые не могут работать с памятью, поддерживаемой EMM386 или Windows 3.x в улучшенном режиме 386. SMARTDRV может также выполнять кэширование диска, ускоряющее работу компьютера.

C:\WINDOWS\command\mscdex /d:mscd001 — обеспечивает доступ к дискам CD-ROM.

```
mode con codepage prepare=((866) C:\Windows\COMMAND\ega3.cpi)
```

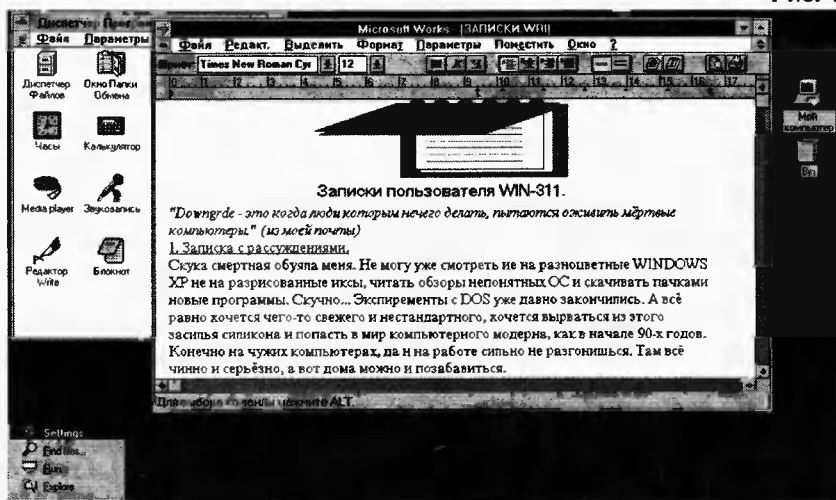
```
mode con codepage select=866
```

keyb ru,,C:\WINDOWS\COMMAND\keybrd3.sys — стандартные строки выбора кодовой страницы языка Windows 98.

Файл **config.sys** тоже не должен оставаться без внимания. DEVICE=C:\WINDOWS\HIMEM.SYS — это администратор дополнительной памяти, программа, координирующая использование дополнительной памяти вашего компьютера, включая верхнюю память (HMA), благодаря чему никакие прикладные программы и драйверы устройств не будут одновременно использовать одну и ту же область памяти.

```
DEVICEHIGH=C:\WINDOWS\COMMAND\DISPLAY.SYS CON=(EGA,,1)
COUNTRY=007,866,C:\WINDOWS\COMMAND\COUNTRY.SYS
```

Рис. 1



DEVICEHIGH=: \WINDOWS\ SETVER.EXE — загружает в память таблицу версий MS-DOS, позволяя избежать надписи Incorrect DOS version.

LASTDRIVE=Z — задает максимальное число дисководов, к которым вы можете обращаться. Задаваемое значение представляет последний дисковод, распознаваемый MS-DOS.

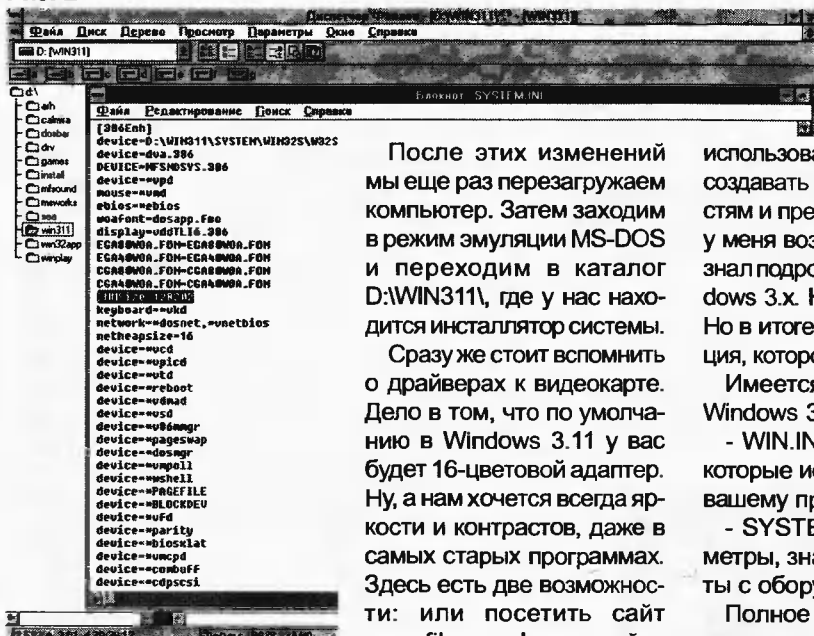
DEVICEHIGH=C:\GSCDROM.SYS /D:MSCD001 — это драйвер CD-ROM, чтобы под Windows 3.11 можно было работать с компакт-дисками. GSCDROM.SYS — файл драйвера. К каждому приводу CD-ROM есть свой драйвер DOS, который поставляется на диске вместе с новым устройством.

STACKS=9,256 — поддержка динамического использования стеков данных для работы с аппаратными прерываниями.

BUFFERS=60 — при запуске системы выделяет память для указанного числа дисковых буферов.

DOS=HIGH — определяет, что MS-DOS должна поддерживать связь с расширенной памятью, частично загружаться в старшую память, или и то, и другое.

Рис. 2



После этих изменений мы еще раз перезагружаем компьютер. Затем заходим в режим эмуляции MS-DOS и переходим в каталог D:\WIN31\, где у нас находится инсталлятор системы.

Сразу же стоит вспомнить о драйверах к видеокarte. Дело в том, что по умолчанию в Windows 3.11 у вас будет 16-цветовой адаптер. Ну, а нам хочется всегда яркости и контрастов, даже в самых старых программах. Здесь есть две возможности: или посетите сайт www.filesearch.ru и найти

версию драйверов к вашей видеокарте для Windows 3.x, или скачать по ссылке www.fdd5-25.net/drivers/svga.exe универсальный драйвер для SVGA-адаптера с поддержкой 256 цветов и разрешения 1024x768. Я выбрал вариант под номером один и нашел родные драйвера для своей видеокарты TSENG LABS 6000, в которых есть возможность регулировки частоты обновления развертки монитора. Это позволяет работать в режиме 1024x768/75 Гц/16 бит. При таких параметрах работа с Windows 3.11 превращается в сплошное развлечение.

Итак, продолжим. Мы остановились на установке самой системы. В этом ничего сложного нет. Запускаем файл `setup.exe` и далее следуем указаниям программы. У нас появилось текстовое меню, в котором есть возможность выбрать тип видеоадаптера. По умолчанию это VGA. Мы же поступаем следующим образом. Если у нас универсальные SVGA-драйвера, то указываем путь к ним для Windows 3.11, а если они "родные", то не производим никаких изменений. Достаточно будет потом распаковать их на диск и запустить `install.exe`, и они в большинстве случаев все выполнят сами.

И сразу два предупреждения! Первое — не устанавливайте Windows 3.x на диск C:\, если у вас там установлены WIN95...ME, и не давайте вносить изменения в загрузочные

файлы во избежание ошибок. Искать установленные программы будет бесполезно, т.к. в большинстве своем они окажутся WIN32, и не будут работать под Windows 3.x.

Второе — если вы устанавливаете Windows 3.11 со ссылки, указанной мною, то столкнетесь с тем, что оболочка поддерживает только английский язык. В этом ничего страшного нет. По ссылке www.fdd5-25.net/os/parawin.zip можно скачать программу PARAWIN, которая позволит установить поддержку восточнославянских языков, в том числе русского, белорусского и украинского. Она очень проста в установке. Вам нужно будет лишь запустить инсталляцию и указать, при помощи какой комбинации клавиш вы хотите переключать тот или иной язык. Дальше PARAWIN все сделает за вас.

После всех злоключений с инсталляцией и русификацией у вас должны быть видны рабочий стол и диспетчер программ. Разрешение рабочего стола и "герцовка" монитора полностью зависят от вас, т.к. у всех разные видеоадаптеры. Со звуковой картой все намного проще. Можно воспользоваться стандартными драйверами. Но в начале работы требуется сказать пару слов о главных системных файлах Windows 3.11 (рис.2).

Инициализационные файлы содержат информацию, которая определяет вашу рабочую среду. Windows 3.11 и приложения на ее основе могут использовать информацию, записанную в этих файлах, чтобы создавать конфигурацию, удовлетворяющую вашим потребностям и предпочтениям. Чаще всего неразрешимые проблемы у меня возникали лишь потому, что ни я, ни кто-то другой не знал подробных значений параметров системных файлов Windows 3.x. Конечно, пришлось многое искать и редактировать. Но в итоге набралась действительно эксклюзивная информация, которой, на мой взгляд, обязательно следует поделиться.

Имеется два стандартных инициализационных файла Windows 3.x:

- WIN.INI, который первоначально содержит параметры, которые используются для того, чтобы управлять средой по вашему предпочтению;

- **SYSTEM.INI**, который первоначально содержит параметры, значения которых настраивают систему для работы с оборудованием вашего компьютера.

Полное описание параметров, позволяющее провести тонкую настройку системы, займет не одну страницу, поэтому с ним можно ознакомиться в приложении к этой статье на сайте журнала <http://radio-mir.com>. Тонкая настройка системы предполагает знание самых потаенных ее уголков. Может, кому-то покажется бесполезным описание параметров файлов WIN.INI и SYSTEM.INI. Но на самом деле это не так. Простой жизненный пример. Windows 3.11 справедливо полагает, что у вас на компьютере установлено вместо 128 Мб оперативной памяти всего 64 Мб, и никакой драйвер под DOS эту проблему решить не сможет! А параметр EMMSize=128202 секции [386Enh] файла SYSTEM.INI наглядно докажет нашей системной оболочке, что она имеет дело с намного более мощной системой. Хотя если честно, производительность и без этого просто потрясающая.

При установке Windows 3.11 со всеми драйверами у вас появляется возможность работать с локальной сетью из чистого MS-DOS. В старших версиях системы, как и в предыдущих, есть замечательная команда NET, при помощи которой можно подключать сетевые диски, отправлять нужные файлы, пользуясь при этом всеми законами локальных сетей. Самые нужные команды:

- NET START — запуск служб для работы в локальной сети;
- NET STOP — остановка работы служб;

Рис. 3



- NET VIEW — вывод списка компьютеров, входящих в рабочую группу, или списка общих ресурсов на заданном компьютере. Для вывода полного списка имеющихся общие ресурсы компьютеров из рабочей группы, в которой зарегистрирован данный компьютер, используется команда NET VIEW без параметров;

- NET USE — без параметров выводит список всех подключенных ресурсов локальной сети. Для того чтобы подключить удаленный диск компьютера, мы должны руководствоваться следующей логикой построения команды: NET USE ИМЯ_ДИСКА \ИМЯ_КОМПЬЮТЕРА\ИМЯ_РАСШАРЕННОГО_КАТАЛОГА.

Это, конечно, самая малость, но уже благодаря ей можно будет без проблем работать в чистой DOS-сети. Главное, загрузить перед этим драйвера сетевой карты.

Теперь перейдем к вопросу о настройке вашей звуковой карты. Как и для видеоадаптера, существуют два варианта — или вы ищете "родные" драйвера для вашего "звука", или пользуетесь

MCI sound—Добавить—Creative Sound Blaster Labs 1.5. При этом нужно указать прерывания, которые использует ваша звуковая карта. Для большинства ISA-карт это A220 I7 D1. Чаще всего как раз драйвер от Creative подходит под большинство звуковых карт (рис.3).

После всех описанных манипуляций остается лишь перезапустить ваш Windows 3.11 и наслаждаться поддержкой мультимедиа.

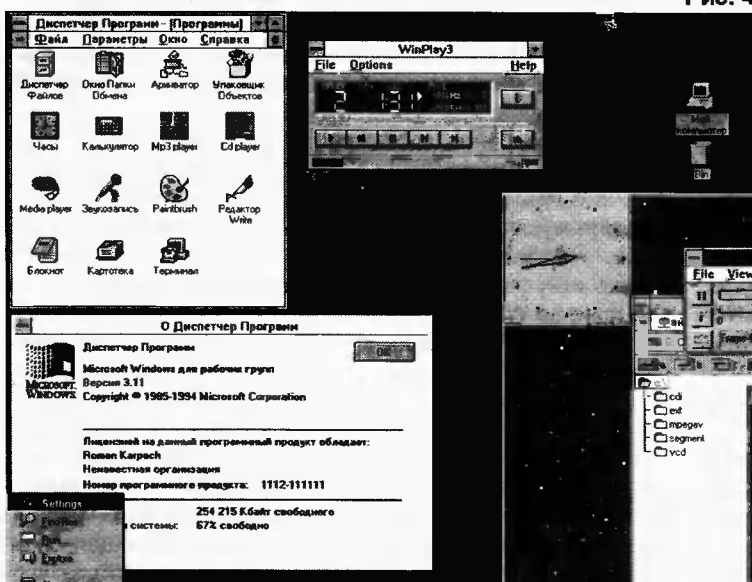
3. Записка развлечений и повседневной работы

Сейчас я буду вас убеждать в том, что Windows 3.11 может эксплуатироваться не хуже, чем Windows XP. Допустим, у вас "упала" система, а переставлять ее лень, да и времени на это нет. И здесь, к нашему великому удовольствию, на диске D:\ находится Windows

3.11. Она жаждет работы, она просто горит желанием, чтобы вы с ней что-нибудь сделали! Возможности Windows 3.11 просто потрясут. Например, вы хотите слушать MP3-музыку или смотреть MP3-фильмы? Нет проблем. Две маленькие программки — и вы прекрасно проводите время.

WINPLAY (www.iis.fhg.de/audio) — очень удобный и простой проигрыватель MP3-музыки под Windows 3.x. Главным его преимуществом перед другими подобными программами является неприхотливость. Ведь некоторые, прочитав эту статью, сидят отнюдь не за такой, как у меня, компьютер, а, возможно, за что-то более старое и намного менее производительное. Так вот, WINPLAY работает даже на 486-х компьютерах, главное, правильно настроить сам проигрыватель. Для этого заходим в Options->Output и выбираем такой режим, в котором музыка будет качественно проигрываться и при этом не перегружать вашу систему. Причем можно прямо указать тип вашего процессора. На первый взгляд кажется, что у WINPLAY нет листа

рис. 4 проигрывания файлов. Так оно и есть, но группу файлов все равно можно прослушивать, правда, это будет больше похоже на проигрыватель CD-дисков. Для того чтобы проиграть группу файлов, следует выбрать File->Open, перейти в каталог с музыкой и, нажав клавишу SHIFT, выделять мышкой MP3-файлы, которые вы хотите ус-



тесь стандартным набором Windows 3.11. Прошу учесть, что "родные" драйвера намного предпочтительнее для PCI-устройств, т.к. стандартные могут вызвать определенные проблемы. Если вы хотите установить поддержку звука к вашей ISA-плате, для которой у вас нет драйверов, следует зайти в Панель управления и выбрать Драйверы—



Рис. 5

Рис. 6



лышать. На рис.4 показана работа WINPLAY на компьютере с процессором AMD 450. Как видите, замечательно смотрится и слушается!

XING MPEG PLAYER — некогда популярнейший проигрыватель фильмов. Поддерживает несколько стандартов видеофайлов: dat, mp...mp3, vcd. Также есть возможность тонкой настройки под ваш компьютер, для этого в XING встроен специальный тест производительности видеоадаптера. Имеется и возможность создания скриншотов из фильмов, которые, между прочим, всегда получаются (рис.5).

Если вы все еще терзаетесь смутными сомнениями относительно установки Windows 3.11, предлагаю вам посетить сайт <http://www.win31.de>. Конечно, он на немецком языке, но зато там есть множество полезных утилит, которые облегчат вашу повседневную работу. А я пишу лишь о самом нужном и, на мой скромный взгляд, незаменимом.

Текстовые редакторы под Windows 3.11 — это отдельный разговор. Можно пользоваться Word 6.0 из комплекта MS Office, можно — пакетом MS Works, в который все включено, а можно — простенькими редакторами, которые заменят Блокнот и еще укажут вам нужную кодировку текста.

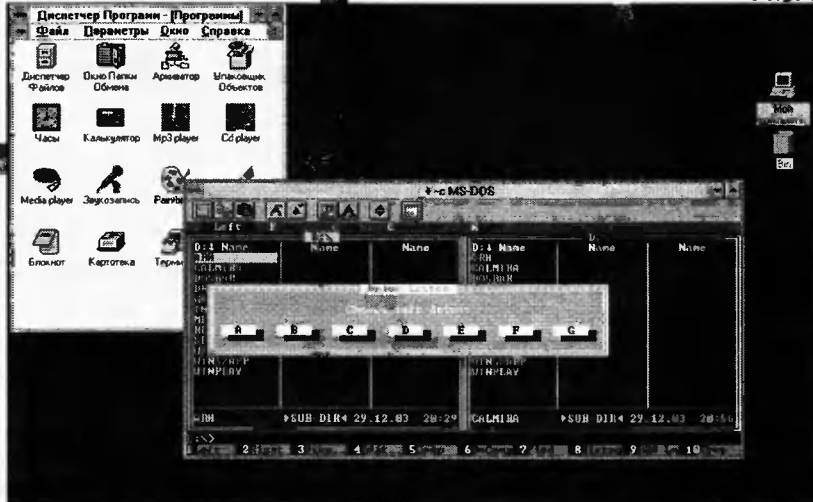
MS Works 2.0 (www.fdd5-25.net/text/works.zip) — еще раз отмечу, что эта статья изначально была набрана в этом программном продукте. В пакете Works имеется пять инструментов. С их помощью вы сможете выполнить большинство экономических, школьных и домашних задач. Немаловажным достоинством WORKS является поддержка *.rtf файлов, чего в других текстовых редакторах не наблюдалось.

EDITEUR (www.fdd5-25.net/text/ed25.zip) — отличная замена стандартных WRITE и Блокнота вместе взятых. Текстовый редактор EDITEUR обладает весьма полезными функциями, главная из которых — поддержка нескольких кодировок (Windows, DOS, UNIX). Также есть поддержка макросов. Программа имеет огромное количество нужных и не очень настроек. Среди них возможность работы с несколькими окнами сразу (рис.6).

После небольшого экскурса в возможности прикладных программ Windows 3.11 я расскажу вам еще о некоторых интересных вещах. Так, не стоит ни в коем случае забывать об архиваторах. Можно пользоваться как DOS-, так и Windows-версиями. Но лучше всего работу с файлами и архивами объединить в одной программе. И в этом случае к нам на помощь придет незабвенный Windows Commander. Да, тот самый, который вы каждый день запускаете у себя на

компьютере! Оказывается, есть версия и для Windows 3.11 (www.fdd5-25.net/shell/wincom3x.zip). Как видите, комфортная работа возможна и в Windows 3.x. И все же пользователей всегда что-то смущает. В данном случае — внешний вид рисунков в статье. Все, наверное, заметили, что на них присутствует значок Корзины и странный ярлык Мой компьютер. Ничего странного в этом нет, я не пытаюсь

Рис. 7



никого обмануть, на самом деле это все та же Windows 3.11, только немного модернизированная программой CALMIRA (www.fdd5-25.net/shell/calmir.zip). Она позволяет изменить интерфейс пользователя, сделать его как в Windows 95 — появляется taskbar и стартовое меню, сведенные вместе во встроенной среде, что преобразует старушку 3.11. Для этого требуется 386/4 Mb RAM/VGA/ Win3.1/3.11. Хотя мне удалось запустить CALMIRA даже на IBM 286/4 Mb Ram (рис.7).

Маленький совет. Если вы удалили группу запуска, но хотите, чтобы при загрузке Windows 3.11 загружалась CALMIRA и еще что-нибудь, то в файле WIN.INI напишите аналогичное `load=D:\DRV\ET6000\vi_grm.exe D:\Calmira\calmira.exe D:\DOSBAR\DOSBAR.EXE`

Имена программ для запуска должны быть перечислены через пробел в том порядке, в котором вы хотите видеть их очередность загрузки.

Вообще говоря, за время экспериментов с Windows 3.11 у меня сложилось такое впечатление, будто вовсе она и не устаревшая ДОС'овская оболочка. Ведь чем дальше я пытался находить, казалось бы, невозможные для нее вещи, тем чаще находились соответствующие программы. Будь то музыка и фильмы, графика, текст, Internet. Для любой повседневной задачи можно найти программу, работающую под Windows 3.11. Это говорит о том, что Microsoft в далеком 1993 г. создала оболочку, обогнавшую время на много лет вперед. Ведь многие ее функции свойственны любой новой операционной системе. Хотя годы уже взяли свое — техническая поддержка снята, а по законам рынка это значит, что теперь Windows 3.11 можно пользоваться свободно, без всяких ограничений и поправок в сторону пиратского ПО.

4. Internet

Теперь я расскажу, как с помощью Windows 3.11 без особых проблем работать с Internet. Развеиваем легенды. Под Windows 3.11 есть браузеры: Internet Explorer 1.0...5.0, Opera 1.0...3.61,

Netscape 1.0...4.0 etc; и почтовые программы: Outlook, Eurodora и etc. Между прочим, ICQ тоже имеется. Но дело не в этом. Сейчас я подробно расскажу, как же выйти в Internet из-под Windows 3.11.

Сначала следует отыскать любую версию IE. Я предлагаю 3.0 (www.fdd5-25.net/inet/iew31.exe), т.к. нам нет никакой разницы, с чего мы будем начинать эксперимент. Браузер Internet Explorer нужен будет лишь для того, чтобы установить протокол TCP/IP. Для серфинга по сайтам он уже староват. Хотя существуют версии 4.0 и 5.0, но, каюсь, на свой сайт я их не закачал, а ссылку потерял. Поэтому предлагается вариант, которым изначально воспользовался автор. Итак, после установки IE 3.0 нам будет нужен более молодой браузер, поддерживающий массу кодировок. На такую роль годится Netscape 4.0. Его версии для разных операционных систем можно найти по адресу ftp.netscape.com.

После установки IE 3.0 у вас появится Wizard подключения к сети, но мы лишь настроим модем, а остальное пропустим. Затем перезагрузим наш Windows 3.11 и запустим программу *iedial* из каталога IE. Там ничего сложного нет, кроме двух вещей. Во-первых, вам следует убрать галочку с опции использовать код города/страны. Во-вторых, вам нужно будет указать DNS сервера вашего провайдера. Белорусам я смогу подсказать:

DNS BELTELECOM она же "беспаролька" — tel: 8w600100, 2092662

DNS1: 193.232.248.2

DNS2: 193.232.248.45

После этого мы сохраняем наше соединение в той рабочей группе, которая вам будет по душе. Затем устанавливаем Netscape 4.0. Не устанавливайте его раньше IE — будут проблемы с запуском и работой! После этого выбираем Netscape как браузер по умолчанию. Затем нам следует провести с ним кое-какие манипуляции. Во-первых, зайти в Preference (Настройки), выбрать меню Fonts и указать, что нужно использовать шрифты пользователя. После этого выбрать три основных шрифта для программы с пометкой Суг (кириллица). Если вы этого не сделаете, то даже при указании кодировки "Windows-1251" в русской и русифицированной версии Windows 3.11 у вас на сайтах будут сплошные каракули. Во-вторых, вам потребуется Firewall или, проще говоря, защита от несанкционированного доступа к вашему компьютеру в сети. И вот вам еще одна ссылка — www.fdd5-25.net/inet/trumpet.rar. Trumpet — это довольно мощный firewall и звонилка. Возможности: установка IP, скорости передачи данных, разнообразные режи-

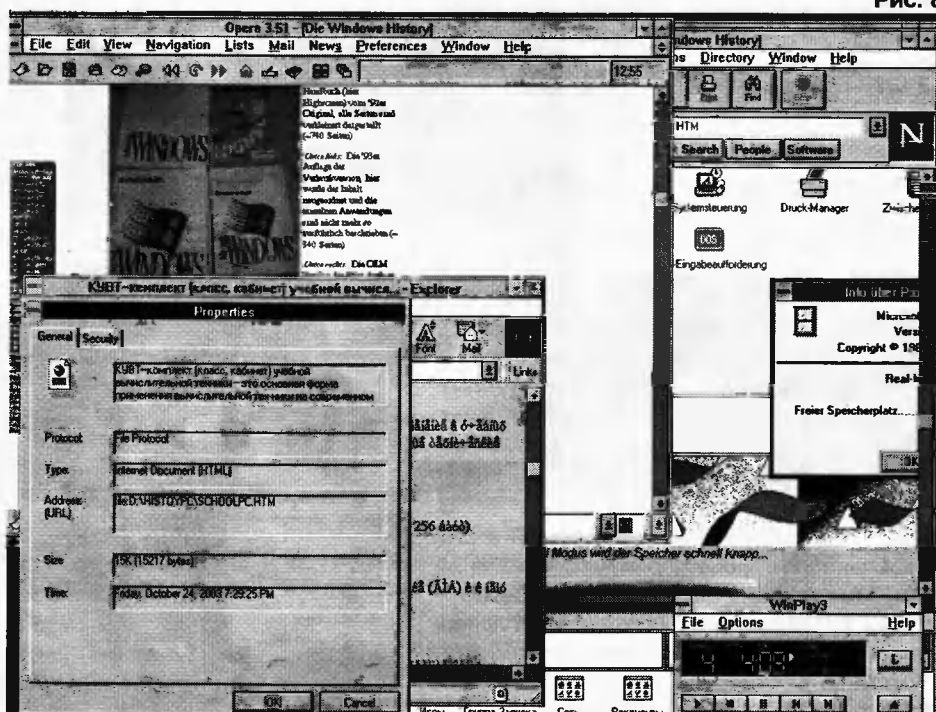


Рис. 8

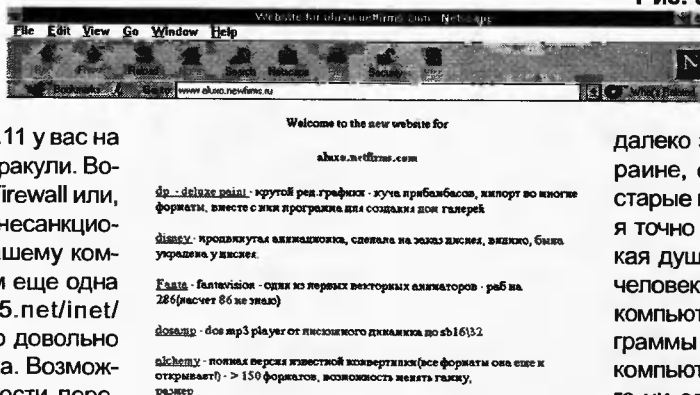
мы настройки работы с сетью, выбор диапазона охраняемых IP. Но больше всего мне понравилось то, что программа поддерживает скорость, с которой работают нормальные современные модемы. Так что с этой звонилкой-firewall вы будете себя чувствовать в сети не хуже, чем пользователь с какими-нибудь навороченными Windows XP или Linux'ами (рис.8 и 9).

Электронная почта. Что же написать про нее? И стоит ли? Ведь в Netscape есть почтовый клиент, а по адресу www.fdd5-25.net/inet/eudor152.exe есть Eurodora для Windows 3.11. Наверное, про это и говорить не следует. Ведь если вы разберетесь, как сделать доступным из старых "Окон" Internet, то настроить почтовую программу вам не составит никакого труда. Мне бы еще хотелось рассказать о многих вещах, но, боюсь, ни журнал, ни мое время больше этого не позволяют.

5. Эпилог

У этого явления до некоторого времени не было названия, как не было и сайта в сети. Но этим странным и, на первый взгляд, бредовым развлечением — downgrade — занимались сотни и даже тысячи компьютерщиков. Но когда по воскресеньям у меня на сайте в чате начал собираться народ, не для

Рис. 9



того чтобы в очередной раз покататься туповатыми сообщениями, а просто чтобы пообщаться и обсудить свои пристрастия, то оказалось, что и далеко за Уралом, и совсем рядом, в Украине, есть пользователи, собирающие старые программы и компьютеры. И тогда я точно узнал, как широка наша славянская душа, как умен и изобретателен наш человек! Ведь старые программы на новых компьютерах — это одно, а вот новые программы и новые возможности на старых компьютерах — это совсем другое! До этого ни один хваленый Запад не дорастет! Но об этом как-нибудь в следующий раз.



Построение таблицы громкости в плеере Pro Tracker 3

И.РОЩИН,

г.Москва,

Fido: 2:5020/689.53

ZXNet: 500:95/462.53

E-mail: bestview@mtu-net.ru

WWW: http://www.ivr.da.ru

Музыкальные модули, написанные в редакторе Pro Tracker 3 (далее PT3), проигрываются с помощью специального набора процедур, называемого плеером. В плеере (по крайней мере, для версий 3.3...3.6) имеется так называемая таблица громкости, расположенная по смещению #110 от начала плеера и занимающая 240 (#F0) байтов. Эта таблица представляет собой 15 строк по 16 значений в строке, каждое значение занимает один байт и является числом от 0 до 15.

В PT3 каждый отсчет сэмпла имеет громкость от 0 до 15, а сам сэмпл может быть проигран с громкостью от 1 до 15 (нулевая громкость не используется). Вот для определения громкости проигрывания очередного отсчета по значениям громкости сэмпла и громкости этого отсчета в сэмпле и нужна таблица. На пересечении строки с номером, равным громкости сэмпла, и столбца с номером, равным громкости отсчета в сэмпле, и находится искомое значение громкости.

Если обнулить в плеере место, занимаемое таблицей громкости, при этом добавив в программу фрагмент для построения этой таблицы, то при упаковке программы получится выигрыш в размере. Действительно, последовательность нулей на месте таблицы сожмется гораздо лучше самой таблицы, а длина добавленного фрагмента программы, строящего таблицу, очень невелика (как мы увидим далее), к тому же, он тоже может быть сжат.

Pro Tracker 3 — наиболее широко используемый в настоящее время музыкальный редактор для "ZX-Spectrum", написанные в нем модули используются во множестве программ (играх, электронных газетах и журналах, intro, demo). Соответственно, область применения такого способа уменьшения длины программы обещает быть весьма широкой.

Ниже приведен участок дампа стандартного плеера Pro Tracker 3.3...3.4, содержащий таблицу громкости:

```
#0110: 0000 0000 0000 0000 0101 0101 0101 0101
#0120: 0000 0000 0000 0101 0101 0102 0202 0202
#0130: 0000 0000 0101 0101 0202 0202 0303 0303
#0140: 0000 0000 0101 0102 0202 0303 0304 0404
#0150: 0000 0001 0101 0202 0303 0304 0404 0505
#0160: 0000 0001 0102 0203 0303 0404 0505 0606
#0170: 0000 0101 0202 0303 0404 0505 0606 0707
#0180: 0000 0101 0202 0303 0405 0506 0607 0708
#0190: 0000 0101 0203 0304 0505 0606 0708 0809
#01A0: 0000 0102 0203 0404 0506 0607 0808 090A
#01B0: 0000 0102 0303 0405 0606 0708 0909 0A0B
#01C0: 0000 0102 0304 0405 0607 0808 090A 0B0C
#01D0: 0000 0102 0304 0506 0707 0809 0A0B 0C0D
#01E0: 0000 0102 0304 0506 0708 090A 0B0C 0D0E
#01F0: 0001 0203 0405 0607 0809 0A0B 0C0D 0E0F
```

Я не знаю, какую формулу использовал для расчета этой таблицы автор Pro Tracker'a, но такая вот простая

формула подошла абсолютно точно:

$$A(i, j) = \left[\frac{(i+1) \cdot j}{16} \right]$$

В этой формуле квадратные скобки обозначают выделение целой части, i — номер строки таблицы (строки нумеруются с 1), j — номер столбца (столбцы нумеруются с 0).

Так как плеер PT3 всегда начинается с адреса #XX00, а таблица расположена по смещению #110 от начала плеера и, таким образом, всегда начинается с адреса #XX10, получается, что координаты i и j элемента таблицы можно получить из младшего байта адреса этого элемента: значение старшего полубайта — это i , а младшего — j .

Учитывая все вышесказанное, удалось написать достаточно короткий фрагмент программы, строящий таблицу:

```
LD HL, VOL_TAB ;Адрес таблицы — число вида #XX10.
LD DE, #000F ;D — начальное значение i. Проще хранить
;и увеличивать i отдельно, а не
;получать из L. Сейчас i=0, но перед
;вычислением первого элемента таблицы
;i увеличится до 1. E — константа #0F.

;Главный цикл.

M1 LD A, L ;Младший байт адреса текущего элемента.
AND E ;=AND #0F — получили j.
LD C, A ;Запомнили j.
JR NZ, M2 ;Каждый раз, когда j=0 (т.е. когда
;начинаем обрабатывать новую строку),
;увеличиваем i на 1.
INC D ;Счетчик=i.
;Сейчас A=j.
M2 ADD A, C ;A=A+j.
DJNZ M3

;Вычислили A=(i+1)*j. Теперь делим на 16.

RRCA
RRCA
RRCA
RRCA
AND E ;=AND #0F.
LD (HL), A ;Запись значения элемента.
INC L ;Переход к адресу следующего элемента.
;Если вся таблица заполнена, то L=0.
JR NZ, M1 ;L<>0 — переходим к началу главного
;цикла (продолжение заполнения таблицы).
```

Длина этого фрагмента — всего 25 байтов, а длина исходной таблицы, сжатой с помощью HRUST 1.3 — 94 байта (без учета служебных 6 байтов получаемого файла, в которых хранятся идентификатор "HR", длина исходного блока и длина упакованного блока). Тогда выигрыш от программного построения таблицы составит примерно $94-25=69$ байтов (примерно — по нескольким причинам: во-первых, при сжатии всей программы, а не только таблицы, таблица может сжаться не в точности до 94 байтов; во-вторых, сжатая последовательность нулей на месте таблицы тоже займет сколько-то места, пусть и очень мало; в-третьих, рассматриваемый фрагмент программы также может быть сжат).



Кстати, если известны значения каких-либо регистров к моменту выполнения этого фрагмента, то в некоторых случаях возможна дополнительная оптимизация. Например, если D=0, то можно заменить LD DE,#000F на LD E,#0F, а если E=0, то можно переписать фрагмент так, чтобы поменять местами функции регистров D и E, тогда вместо LD DE,#000F будет LD DE,#0F00, а эту команду уже можно заменить на LD D,#0F. Также при необходимости можно поменять местами функции регистров пар HL и DE (например, если к моменту выполнения фрагмента в DE уже содержится адрес VOL_TAB).

В плеере Pro Tracker 3.5...3.6 используется несколько другая таблица громкости:

```
#0110: 0000 0000 0000 0000 0101 0101 0101 0101
#0120: 0000 0000 0101 0101 0101 0101 0202 0202
#0130: 0000 0001 0101 0101 0202 0202 0203 0303
#0140: 0000 0101 0101 0202 0202 0303 0303 0404
#0150: 0000 0101 0102 0202 0303 0304 0404 0505
#0160: 0000 0101 0202 0203 0304 0404 0505 0606
#0170: 0000 0101 0202 0303 0404 0505 0606 0707
#0180: 0001 0102 0203 0304 0405 0506 0607 0708
#0190: 0001 0102 0203 0404 0505 0607 0708 0809
#01A0: 0001 0102 0303 0405 0506 0707 0809 090A
#01B0: 0001 0102 0304 0405 0607 0708 090A 0A0B
#01C0: 0001 0202 0304 0506 0607 0809 0A0A 0B0C
#01D0: 0001 0203 0304 0506 0708 090A 0A0B 0C0D
#01E0: 0001 0203 0405 0607 0708 090A 0B0C 0D0E
#01F0: 0001 0203 0405 0607 0809 0A0B 0C0D 0E0F
```

В этом случае не так просто подобрать формулу для вычисления элементов таблицы. Но, к счастью, мне удалось найти подпрограмму построения такой таблицы в [1]. Там был приведен исходный текст плеера Pro Tracker 2.101, в котором таблица громкости строится специальной подпрограммой, а при сравнении обнаружилось, что эта таблица — такая же, как в плеере Pro Tracker 3.5...3.6. Кстати, странно — исходный текст плеера в [1] был прокомментирован как "стандартный проигрыватель", а между тем, в оригинальном плеере PT2 таблица содержится непосредственно, а не строится подпрограммой.

Приведенную в [1] подпрограмму длиной 43 байта мне удалось существенно оптимизировать, несколько изменив логику ее работы и учитывая, что адрес расположения таблицы — число вида #XX10. В результате получилась вот такая подпрограмма длиной 32 байта:

```
LD BC,VOL_TAB ;Адрес таблицы — число вида #XX10.
XOR A
LD D,A

INITV2 ADD A,#11 ;Если в результате этого сложения
;получится #100 (т.е. A=0 и флаг C=1),
;то 0 превратится в #FF после SBC A,D.
SBC A,D
LD E,A
LD H,D
LD L,D ;HL=0 (D всегда равно 0).

INITV1 LD A,L
RLA
LD A,H
ADC A,D ;=ADC A,0.
LD (BC),A
ADD HL,DE
```

```
INC C
RET Z
LD A,C
AND 15
JR NZ,INITV1

LD A,E
CP #77
JR NZ,INITV2
INC A
JR INITV2
```

Чтобы использовать ее в виде непосредственно встраиваемого в программу фрагмента (чтобы не трагиться на команду CALL), достаточно заменить команду RET Z на JR Z,M1 (где M1 — адрес следующей за фрагментом команды). Получится фрагмент программы на один байт длиннее — 33 байта.

Исходная таблица сжимается с помощью HRUST 1.3 до 97 байтов (также без учета служебных 6 байтов). Таким образом, примерный выигрыш от программного построения таблицы PT 3.5...3.6 составит 97-33=64 байта.

При рассмотрении этой подпрограммы я понял, что можно написать похожую подпрограмму для построения уже рассмотренной ранее таблицы PT 3.3...3.4:

```
LD BC,VOL_TAB ;Адрес таблицы — число вида #XX10.
LD DE,#10

INITV2 LD HL,#10
ADD HL,DE ;После сложения флаг C сброшен.
EX DE,HL ;DE=DE+HL.
SBC HL,HL ;HL=0.

INITV1 LD A,H
LD (BC),A
ADD HL,DE
INC C
RET Z
LD A,C
AND 15
JR NZ,INITV1

JR INITV2
```

Длина этой подпрограммы — 25 байтов, а при оформлении ее в виде фрагмента программы — 26 байтов. В сравнении с 25-байтовым фрагментом, приведенным в начале статьи, выигрыша в размере не получилось. Но все-таки работа была проделана не зря: возникла идея объединить две похожие подпрограммы — эту и предыдущую — в одну универсальную подпрограмму построения таблицы громкости.

Если приходится проигрывать одним и тем же плеером модули, написанные в различных версиях PT3, то желательно перед проигрыванием каждого модуля проверять, в какой версии PT3 он написан, и в соответствии с этим строить нужную таблицу громкости. Тогда каждый модуль будет звучать так, как задумывал его автор. Именно так сделано в моей программе BestView.

Первые байты проигрываемого PT3-модуля представляют собой строку "ProTracker 3.x", где x — номер подверсии. Таким образом, можно, проверив байт, в котором хранится символ x (он находится по смеще-



нию 13 от начала модуля), узнать, какую из двух таблиц необходимо сформировать.

Ниже приведен текст универсальной подпрограммы, которая именно это и делает — определяет, какую из двух таблиц надо построить, и строит нужную таблицу. Метка MODULE — это адрес проигрываемого модуля.

```
LD A,(MODULE+13);Проверка версии.
CP "5"
LD HL,#11;Подготовка данных для построения
LD D,H;таблицы PT 3.5...3.6.
LD E,H;DE=0.
LD A,#17;Код команды RLA.
JR NC,M1;Переход, если версия - 3.5 или 3.6.
DEC L;HL=#10.
LD E,L;DE=#10.
XOR A;A=0 (код команды NOP).
M1 LD (M2),A;По адресу M2 будет или NOP, или RLA.

LD BC,VOL_TAB;Адрес таблицы - число вида #XX10.

INITV2 PUSH HL;Значение слагаемого (#10 или #11)
;храним в стеке.
ADD HL,DE;После сложения флаг C сброшен.
EX DE,HL;DE=DE+#10 или DE=DE+#11, смотря какую
;таблицу строим.
SBC HL,HL;HL=0.

INITV1 LD A,L
M2 DB #7D;Здесь будет RLA или NOP.
LD A,H
```

;Если по адресу M2 находится команда NOP, то флаг C сейчас точно сброшен (либо после команды SBC HL,HL, либо после команды AND 15 в конце цикла), и следующая команда не изменит A.

```
ADC A,0
LD (BC),A
ADD HL,DE
INC C
LD A,C
AND 15
JR NZ,INITV1

POP HL;Восстановили значение слагаемого.
LD A,E;Если строим таблицу PT 3.3...3.4,
CP #77;то E никогда не будет равняться #77.
JR NZ,M3
INC E

M3 LD A,C;Если вся таблица заполнена, то C=0.
AND A
JR NZ,INITV2

RET
```

Длина этой подпрограммы — 53 байта, а при оформлении ее в виде фрагмента программы — 52 байта. Как видим, это меньше, чем суммарная длина двух отдельных подпрограмм и команд для определения, какую из них надо вызывать.

Небольшой комментарий к этой подпрограмме. Перед построением таблицы происходит модификация кода: по адресу M2 заносится либо код команды NOP, либо код команды RLA, в зависимости от того, какую таблицу надо строить. Как можно видеть, изначально по адресу M2 находится байт #7D. Почему именно он? Потому что он равен значению предыдущего байта фрагмента — коду команды LD A,L. Таким образом, в программе будут два следующих друг за

другом одинаковых байта, что положительно скажется на степени ее сжатия. Подробнее о таком способе оптимизации рассказывается в [2].

Если таблицу надо строить только один раз, либо если подпрограмма построения таблицы хранится в упакованном виде (возможно, вместе с плеером) и распаковывается заново перед каждым проигрыванием нового модуля, то можно дополнительно ее оптимизировать. По адресу M2 тогда можно сразу поместить команду RLA, а из начала подпрограммы убрать команду LD A,#17 и переставить метку M1 на следующую команду (LD BC,VOL_TAB).

Осталось напомнить, что вопрос уменьшения размера программ, в которых используются музыкальные модули (не обязательно написанные в Pro Tracker 3), я уже затрагивал ранее. Так, в [3] рассмотрено улучшение сжатия музыкальных модулей за счет автоматического определения тех ячеек модуля и плеера, первоначальные значения которых не используются при проигрывании, и заполнения этих ячеек значением из предыдущей используемой ячейки. А в [4, 5], среди прочего, приводится процедура построения частотной таблицы — таким образом, можно не хранить эту таблицу в плеере и за счет этого улучшить сжатие программы.

Литература

1. VtNG/NEW. Описание формата модулей Pro Tracker 2.101. — Электронная газета "Echo" #2.
2. И.Рощин. Улучшение сжатия программ на ассемблере Z80. — Радиомир. Ваш компьютер, 2003, №4, С.44.
3. И.Рощин. Повышение степени сжатия музыкальных модулей. — Радиомир. Ваш компьютер, 2002, №7, С.39.
4. И.Рощин. Частотная таблица с нулевой погрешностью. — Радиолюбитель. Ваш компьютер, 2001, №6.
5. И.Рощин. Частотная таблица с нулевой погрешностью. — Радиомир. Ваш компьютер, 2001, №7, 8.

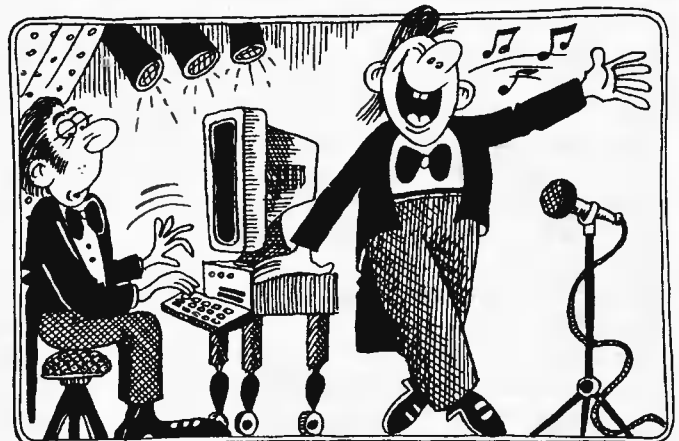


Рисунок О.Попова

А.ВЕНДИЛОВСКИЙ,
г.Минск.

Call of Duty

В небе над Нормандией

Ночь была темной и весьма не приветливой. Еще в воздухе, качаясь на парашютных стропах, Дедушка Геймер понял, что как всегда их выбросили далеко от намеченного района, и сейчас он вместе с Малдером и остальными бойцами отряда собирается приземлиться прямехонько на немецкие огневые точки, где их вряд ли встретят хлебом-солью. Едва он поблагодарил Бога за ночной мрак, как небо разразилось сотнями огней от взрывов зенитных снарядов, вспышек пламени и яростным смертоносным грохотом. Краем глаза он успел заметить, что самолет, из которого они десантировались, получил снаряд в крыло и с нарастающим воем устремился к земле. Молясь, чтобы его не постигла судьба живой мишени, болтающейся на веревочках посреди этого безобразия, и мысленно желая того же своему напарнику, он всеми возможными способами пытался направить свой парашют на более или менее тихое место, где, по его разумению, можно было укрыться и занять хорошую огневую позицию.

Чем особенно запоминаются ночные прыжки с парашютом, так это тем, что как бы ты ни готовился к приземлению, оно всегда неожиданно, что подтверждает сильный удар по ногам и не менее сильный щелчок зубами. Крики немецких солдат, звуки автоматных очередей и свист пролетающих пуль встретили Дедушку Геймера на брэнной земле. Практически невозможно было поднять голову без риска получить в своем теле лишнее отверстие, так что оставалось ползти, пытаясь сориентироваться, откуда и кем ведется огонь. В этот момент он внезапно вспомнил дом и напутствие Автора...

— Значит так, уважаемые, — Автор уткнулся в стопку свежееотпечатанных листов, — ваша парочка уже давно требует, чтобы я отправил вас в какой-нибудь крутой

экшен от первого лица. Будем считать, что ваша мечта сбылась. Вас ждет большое приключение во времена Второй мировой войны. Судя по описанию, вам придется стать самыми обычными солдатами и выполнять самую обычную роль — стараться добыть побольше славы для военачальников и сохранить свою шкуру в целостности и сохранности. Игра позиционируется как командный экшен, а это значит, хотите вы или нет, но вам придется полагаться на ваших коллег-ботов, иначе пройти миссию будет невозможно. О героических подвигах "a-la wolfenstein" лучше сразу забыть. Играть, возможно, будете за три стороны — американцев, русских и немцев. По традиции пройдете подготовительный курс и начнете за американцев с их уже набившей оскомину высадкой в Нормандии. Осмотритесь и доложите все очень подробно. И еще... — тут Автор несколько замялся — советую предварительно изучить исторические хроники, так, на всякий случай.

— Я тебе покажу исторические хроники! — подумал Дед, переваливаясь в яму, где его тут же подхватили руки Малдера. "Привет Дедуль! Спорим, такого приема мы не встречали со времен десанта на Строггосе?!" — Малдер старательно прицеливался в смутную фигурку пулеметчика — профессиональный кемпер остается профи в любой ситуации. Грохнул выстрел, еще дымящаяся гильза упала во влажную глину, и один из пулеметов замолчал.

— А ну вставайте, сукины дети! — голос сержанта перекрыл канонаду, как и положено голосу любого уважающего себя сержанта, — Быстро вперед...! Или что, решили жить вечно? Бегом, ползком, но вперед! Заткните этот пулемет как хотите! Бросьте туда пару гранат или засуньте им эти гранаты в горло, но чтобы через двадцать минут вы все были в том окопе!

— Тебе не кажется, Дед — философски заметил Малдер, — что во все времена и во всех армиях сержанты страшно похожи друг на друга?

Пара гранат заставила пулемет на мгновение замолчать, и бойцы по-пластунски, от укрытия к укрытию, стараясь не попасть под шквальный огонь, поползли к своей цели. Вскоре в перекрестье прицела Малдера показался немецкий пулеметчик... Но пауза была недолгой, с севера подкатила небольшая колонна фашистов, сходу вступившая в бой, да и дома оказались здорово укреплены. А тут еще приказ поступил, любой ценой заткнуть вражеские зенитки.

— Один вывод уже напрашивается, Малдер — тихо произнес Дед — AI, как противника, так и наших компьютерных напарников прописан на весьма достойном уровне. Под пули не бросаются, хорошо пользуются оружием, и вообще, не изображают ходячую мишень, только мешающую играть.

— Я тоже себя спокойней чувствую, когда за моей спиной находится парочка наших бойцов, тем более, что игра изначально рассчитана так, что герой-одиночка ее в жизни не пройдет. Коллективная поддержка и... Дед, ложись!

Откуда-то с чердака ударил пулемет, и пока бойцы, прижавшись к стене, старались отвлечь внимание противника, Малдер бросил на чердак гранату. Позже он не раз повторял, что не нужно бояться гранат, они ведь ручные!

Через несколько дней

Нам этот мост в жизни не удержать, если подмога не придет! Малдер еще раз проверил оптику трофейной винтовки. И говорю тебе, Дед, бросай ты эту американскую винтовку и возьми "Шмайсер"! Благо, этого добра навалом. и с патронами нет проблем. А по качеству, уж простите меня Смит и Вессон, получше будет. Кстати, о птичках,



мне кажется или пальба уже началась? Давай-ка вернемся к работе, а то Автор нас опять в какой-нибудь квест сошлет. Движок игры достаточно прост, без графически навороченных изысков: дырки от пуль не остаются, предметы не ломаются, взрывом особенно ничего не покалечишь. Но он хорошо оптимизирован, и шустренько летает даже на средней конфигурации, что позволяет играть с достаточным комфортом. Зато звук здесь на все десять баллов: объемный, очень реальный, и свист каждой пули буквально ощущаешь печенкой. Оружие — реальные прототипы образцов вооружения трех воюющих держав. И что приятно — в бою можешь легко поменять одно на другое. А, к примеру, немецкие станковые пулеметы...

Договорить им не дал немецкий десант, который с трех сторон атаковал защитников. Укрываясь за наспех сделанной обороной, и все время оглядываясь, так как враг очень старался прокрасться к ним в тыл, обороняющиеся крутились как ужи на сковородке. От взрывов гранат они практически оглохли, пороховой дым заставлял глаза слезиться. "Отступаем за мост! Живее!" — кричал сержант, когда они под градом пуль бросились по этому злосчастному мосту, понимая, что в мишени в тире попасть намного сложнее, чем сейчас в них. Малдер буквально кубарем влетел в укрепленную башню и бросился вверх по лестнице. Оттуда ему открылась картина, от которой ему стало весьма нехорошо. Больше половины бойцов не смогли проскочить через мост и оказались под перекрестным пулеметным огнем немцев, они даже не могли поднять головы. Откуда-то издали раздавалось грохотание танкового двигателя. Укрепление, в котором были фауст-патроны и станковые пулеметы, было на другой стороне улицы. Геймер и несколько бойцов попытались проскочить, но плотность огня заставила их вернуться обратно, и только время от времени они огрызались короткими автоматными очередями, бессильные сделать что-либо еще.

"Если сейчас сюда выползет танк,

и Дед не успеет его отправить в рай для машин, все будет кончено буквально за несколько минут. Его пулемет просто растерзает тех, кто остался на том берегу, а потом его пушка поставит точку на нас здесь. Если только..."

Малдер схватил свою снайперскую винтовку и, не обращая внимания на пролетающие рядом пули, методично стал отстреливать пулеметчиков. Выстрел, другой, третий..., на мгновение пулеметный шквал ослабел, и он увидел, как Дед вместе с другими бойцами бросились к укрытию. Лязг гусениц становился все сильнее и сильнее, но обращать на это внимание просто не было времени — он быстро находил в прицеле очередного пулеметчика и мягко нажимал на спусковой крючок. Тяжелейший удар выбил из рук оружие и бросил его на пол, весь мир закачался в тумане, и он еще успел подумать, бывают ли в этой игре респауны...

Дед видел, как в башню, где сидел Малдер, попал снаряд. Тишина, наступившая после этого, была просто пугающей. Он поймал в ромб наведения гранатомета танк, и через несколько секунд тот запылал, превращаясь в огромную грудку металлолома. Но тут из-за поворота появились еще несколько "Тигров" с пехотной поддержкой, и Дед с чувством, близким к отчаянию, наблюдал, как они с ходу сносят их первую и практически последнюю линию обороны. Внезапный взрыв стал неожиданностью как для него, так и для немцев. Из плотных облаков вырвались американские самолеты-штурмовики, несущие на крыльях смерть и разрушение. Подмога на этот раз пришла вовремя, и Дед бросился к укрытию, в котором был Малдер.

— Так вот, о геймплее — Малдер со всех сторон был обложен аптечками и медленно приходил в себя. — Появляется ощущение, что ты просто посажен на капельницу с адреналином — ни минуты расслабления, только динамика и напряжение, аж свист в ушах стоит. Не

каждая игра может таким похвастаться. Кстати, Дед, тут мне ребята сказали, что все это просто детский сад и разминка по сравнению с тем, что нам придется пережить, играя за русских!

— Хороший хит получился! — резюмировал Дед, ожидая, когда загрузится очередной уровень...

Minimum system requirements

Windows 98, 98 Second Edition, ME, 2000 Professional, XP Professional or XP Home operating system

Intel Pentium III (or compatible) CPU running at 700Mhz or higher

32 MB DirectX 9 compatible video card (Geforce2 or higher) using a supported chipset (see below)

128 MB of RAM (256 MB using Windows 2000 or XP)

DirectX compatible sound card
2 GB of HD space
8X CD-ROM

Minimum system requirements for online play

Windows 98, 98 Second Edition, ME, 2000 Professional, XP Professional, or XP Home operating system

Intel Pentium III (or compatible) CPU running at 700Mhz or higher

32 MB DirectX 9 compatible video card (Geforce2 or higher) using a supported chipset (see below)

256 MB of RAM

DirectX compatible sound card
2 GB of HD space
8X CD-ROM

Network card for broadband connectivity

Игра предоставлена интернет-магазином "MediaCraft"
www.mediacraft.shop.by



Рисунок О.Попова

**КУПЛЮ, ПРОДАМ, ОБМЕНЯЮ**

Для публикации бесплатных объявлений **некоммерческого характера** о покупке и продаже компьютерных комплектующих и программ, их текст можно присылать в письме по адресам:



119454, г.Москва, а/я 37 или
220095, г.Минск-95, а/я 199,
передавать по тел. в Минске (017) 249-41-47,
либо через E-mail:

rl@radiopage.by
rm@radio-mir.com
WWW: <http://radio-mir.com>

Продаю:

- микросхемы новые (серий от 134 до 1810);
- диоды, стабилитроны, транзисторы, (большой выбор);
- автомобильный телевизор "Мара-2", новый (270х140х70 мм, размер по диагонали — 7 см);
- дисконд MF580 (Венгрия) с дисками 5,25;
- ПЗВМ "Сантака-001";
- блок питания импульсный +5В/17,5Вт и +12В/48Вт;
- блок питания +5 В/10 Вт.

Тел. (017) 243-02-97, Саша.

Куплю процессор ATHLON только с разъемом SLOT-A.

E-mail: santinal@yandex.ru

Школьник примет в дар дистрибутив Delphi версии 5 или более поздней. Дмитрий.
E-mail: svidinskii@iren.ru

"Электронный магазин компакт-дисков для радиолюбителей". Доставка почтой.
<http://elcomcd.hostmos.ru>

Продам CD-диски. Список дисков:
www.m01.nm.ru/shop_cd.htm

Ищу техническую документацию на ноутбук ф. Siemens mod. Simatic PG 720P v.2148 или аналогичный.
211033, г.Могилев, ул.Гришина, 110 — 54,
Юношев В.Г.
Тел. (8-10375-0222) 23-80-00,
с 8 до 10 или с 20 до 24 часов.

Куплю контроллер жесткого диска и программное обеспечение для ПК "Сантака-002".
Приму в дар системный блок компьютера.
425568, Респ.Марий Эл,
Куженерский р-н, д.Шудумарь, Нагаев М.А.

Куплю в Москве HDD MK1403MAV TOSHIBA на 1443 Мб для notebook, можно "битый" (нужен только контроллер).
E-mail: vitpravd@hotmail.com

Куплю два картриджа (чистый оптический диск) типа OC301-1 или OC301-2, желательно фирмы "Maxell".
456785, г.Озерск, Челябинская обл., а/я 643.
E-mail: dkflvbh@telecom.ozersk.ru

Продаю материнскую плату Epox EP-8KEM2 (встроенная видеоподсистема) с гарантией. Возможен торг.
Тел. в Минске 224-48-55, Михаил.

Продаю компьютер 486 по частям: материнская плата; ОЗУ — 12 Мб SDRAM; процессор — Intel 80486 DX2S, 120 МГц; видеокарта — Cirrus Logic (SVGA/4 Мб); винчестер — 472 Мб; мультикарта — HDD/FDD (5,25", 3,5")/2xLPTxCOM; дисководы 3,5" и 5,25"; мышь — Genius, 3 кнопки, для COM-порта (DB9); клавиатура — Microsoft Natural (разъем 5 DIN).
Для ответа приложить чистый конверт.
692665, Приморский край, Михайловский р-н, с.Ивановка, ул.Восточная, 19 — 2, Гевик А.И.

Подписные индексы:**Радиомир**

- для жителей России и стран СНГ (кроме Беларуси): 48996 — подписка по каталогу Агентства "Роспечать" (72370 — годовая); 25785 — подписка по Объединенному каталогу "Пресса России" с адресной рассылкой (электронный адрес подписки в INTERNET - www.presssafe.ru);
- для жителей Беларуси: 48996, 489962 (для организаций) по каталогу РО "Белпочта" "Газеты и журналы Республики Беларусь" (раздел "Издания РФ, распространяемые по прямому договору").

Радиомир. КВ и УКВ

- для жителей России и стран СНГ (кроме Беларуси): 48924 — подписка по каталогу Агентства "Роспечать", (71545 — годовая);
- для жителей Беларуси: 48924, 489242 (для организаций) по каталогу РО "Белпочта" "Газеты и журналы Республики Беларусь" (раздел "Издания РФ, распространяемые по прямому договору").

Радиомир. Ваш компьютер

- для жителей России и стран СНГ (кроме Беларуси): 48925 — подписка по каталогу Агентства "Роспечать" (45995 — годовая);
- для жителей Беларуси: 48925, 489252 (для организаций) по каталогу РО "Белпочта" "Газеты и журналы Республики Беларусь" (раздел "Издания РФ, распространяемые по прямому договору").

Кроме того, предприятия регионов России, а также ближнего и дальнего зарубежья могут оформить подписку на журналы "Радиомир", "Радиомир. КВ и УКВ", "Радиомир. Ваш компьютер" через ООО "Корпоративная Почта" по телефонам: (095) 953-92-62, 953-92-02, 953-93-20.

Получить подписку или заказать имеющиеся в наличии отдельные номера журналов можно из редакции. Текущие цены приведены в таблице.

Год	Можно заказать следующие номера журналов			Расценки на 1 экз. (с учетом пересылки)		
	Радиолюбитель	Радиолюбитель. Ваш компьютер	Радиолюбитель. КВ и УКВ	По России и СНГ (рос. руб.)	По Беларуси (бел. руб.)	По Украине (гривны)
1999	11	1 — 6, 10, 11, 12	6	20	700	5
2000	3 — 6, 8 — 12	2 — 8, 10 — 12	6, 7, 9, 11, 12	20	800	5
2001	2, 4 — 6	2, 5, 6	2, 3, 5, 6	25	900	5
	Радиомир	Радиомир. Ваш компьютер	Радиомир. КВ и УКВ	По России и СНГ (рос. руб.)	По Беларуси (бел. руб.)	По Украине (гривны)
2001	7 — 10, 12	7, 9 — 12	7, 8, 10, 11	30	1000	6
2002	1 — 12	1 — 12	1 — 3, 5 — 12	35	1500	7
2003	1 — 12	1 — 12	1, 5 — 8, 10 — 12	40	1800	8
2004	1 — 12	1 — 12	1 — 12	45	2100	9

Наши платежные реквизиты:

- для жителей России и стран СНГ (кроме Беларуси) —
получатель: ООО "Радиомир Пресс", ИНН 7729404741, КПП 772901001,
р/с 40702810900010000084 в ООО КБ "Агропромкредит", ф-л Центральный, г.Москва,
корр. счет 30101810500000000109, БИК 044525109
- Адрес банка: 125315, г.Москва, Ленинградский пр-т, дом 76, корп. 4;
- для жителей Беларуси —
получатель: УП "РЛД", УНН 190218688
р/с 3012000004882 в ф-ле №524 АСБ "Беларусбанк" в г.Минске код 121.

Адрес банка: 220028, г.Минск, ул.Физкультурная, 31.
При оплате через Сбербанк в графе "назначение платежа" необходимо написать свой почтовый индекс, полный адрес, фамилию, имя и отчество полностью и точно перечислить, какие конкретно номера какого из журналов Вы заказываете.

При оплате почтовым переводом все сведения о заказе необходимо указать в графе "Для письма".
Для ускорения процесса получения журналов заказ можно продублировать по E-mail: rm-sales@radio-mir.com. Вся информация — там же или по тел. в г.Минске (017) 221-01-10, (017) 505-13-65.

Приобретение отдельных номеров журналов**В РОССИИ:**

В ЗАО "Интерпресс — Экспресс": тел. в Москве (095) 208-85-50, 208-67-55,
e-mail: inter@aif.ru, <http://www.interpress-express.com>.

В магазинах радиодеталей "ЧИП и ДИП" (единая справочная — тел. (095) 780-95-09):
- г.Москва, ул.Гиларовского, д.39 (ст. метро "Проспект Мира" — радиальная);
- г.Москва, ул.Ивана Франко, д.40, к.1, стр. 2 (платф.Рабочий поселок, 15 мин. от Белорусского вокзала);
- г.Москва, ул.Беговая, д.2;
- г.Ярославль, ул.Нахимсона, 12, тел. (0852) 27-57-15.

На радиорынках в Москве: Митинском (места R4, S8, K52, E50, G56), Царицынском (место 121).

В ООО "ТРЭНТЭК": 111024, г.Москва, 4-я Кабельная, 2А (ст. метро "Авиамоторная"), тел. (095) 258-91-94, 258-91-95. E-mail: abook@mail.ru, abook@inbox.ru
На радиорынках: "Царицыно" (место 13/А), "Митино" (ряд 1, контейнер 17 и место Т-8); в ТК "Савеловский" (ст. метро "Савеловская", место А4, А5); на книжной ярмарке на "Тульской" (ст. метро "Тульская", место 515-19).

На УКРАИНЕ:

В Киеве в фирме "Торм", тел. (044) 227-72-73.

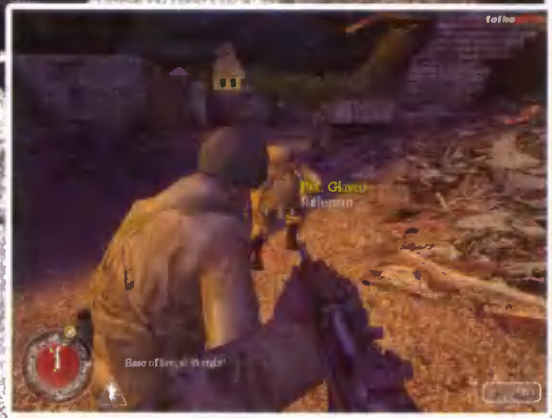
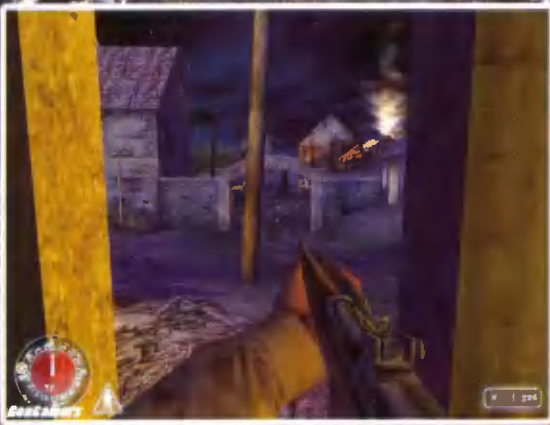
В КАЗАХСТАНЕ:

В фирме ТОО СП "Аргументы и факты. Казахстан": тел. в Алма-Ате (3272) 50-22-60, 50-21-64.

В БЕЛАРУСИ:

В Минске в магазинах "Книга XXI век", пр.Ф.Скорины, д.92, тел. (017) 264-27-97 (ст.метро "Московская") и "Глобус", ул.Володарского, д.16, тел. (017) 227-30-67 (ст.метро "Площадь Независимости").

CALL OF DUTY



CALL OF DUTY



- Ремонт и восстановление неисправных пейджеров
- Перепрограммирование пейджеров
- Большой выбор новых пейджеров (от **35000** р.)

7 способов отправки сообщений!

Стационарные охранные системы для дома и офиса

Получение на пейджер
полной информации о
состоянии охранной системы,
установленной в квартире,
офисе или коттедже



am® Атлант Моторс



Спутниковые автомобильные охранные системы

Осуществление мониторинга за
автомобилем при угоне:
определение его географических
координат по всему миру