ПРИКЛАДНЫЕ ПАКЕТЫ КОМПЪЮТЕРНОЙ ГРАФИКИ

# Применение графики в системе Windows

**Лекция №3**

# УЧЕБНЫЕ ВОПРОСЫ:

1. **Основные объекты используемые при применении компьютерной графики в современной ПЭВМ**
2. **WINDOWS и ОКНА**
3. **СООБЩЕНИЯ**
4. **Особенности графического вывода в многозадачной среде**

# Составные части Современного Компьютера?

- Процессор
- Материнская плата
- Корпус с блоком питания
- Видеокарта
- Монитор
- Память
- Жесткий диск
- Клавиатура
- Мышь
- CD/DVD привод

# Системные требования предъявляемые к графике

**Операционная система**

Последние редакторы компьютерной графики —работают только с полностью 32-битными операционными системами:
Windows 2000 (с установленным четвертым пакетом обновлений);
Windows XP (с установленным первым пакетом обновлений)
Windows 7 и старше.
В связи с этим многим придется сменить операционную систему. Это, как правило, связано и с модернизацией самого компьютера — ведь Windows XP требуется намного больше памяти, чем ранних версиях Windows NT/98.

**Процессор**

Минимальный процессор (определенный разработчиками, который сможет работать с этим графическими пакетами — Pentium с тактовой частотой 200 МГц. Однако поспешим вас огорчить — на практике это не так. Реально самый слабый процессор для работы с графикой — это Pentium II с частотой выше 600 МГц. Полностью комфортная работа будет обеспечена только на новейших процессорах, частота которых более 2000 МГц.

**Оперативная память**
Программе CorelDRAW было бы вполне достаточно 128 Мбайт оперативной памяти. Однако не забывайте о том, что программа работает только в современных операционных системах Windows 2000 и Windows XP, которые являются требовательными к объему памяти. Поэтому лучше ориентироваться на 256 Мбайт.Если же вы планируете запускать одновременно несколько графических приложений, например, CorelDRAW и Photoshop, лучше сразу приобрести 512 Мбайт оперативной памяти.

**Жесткий диск**
Для установки программы CorelDRAW потребуется около 250 Мбайт свободного места на жестком диске. Если же вы планируете использовать весь пакет CorelDRAW Graphics Suite, то выделяйте сразу около 500 Мбайт. Также не забывайте о том, что при работе CorelDRAW активно использует жесткий диск для хранения временных файлов, поэтому лучше, чтобы на вашем системном диске было хотя бы несколько гигабайт свободного места.

**Монитор**
Поспешим огорчить обладателей раритетов, работающих в разрешении 800x600 (если такие есть среди любителей компьютерной графики): минимальное разрешение для работы с CorelDRAW — это 1024x768. Для действительно комфортной работы даже этого разрешения будет недостаточно.

**Манипулятор**

Обязательно наличие мыши или любого устройства, способного ее заменять (например, графического планшета). Также CorelDRAW отлично работает с планшетными компьютерами, существует поддержка Windows XP Tablet Edition.

Если ваш компьютер соответствует перечисленным системным требованиям,можно смело переходить к установке программы, если не соответствует — лучше предварительно произвести модернизацию или использовать более ранние версии CorelDRAW.

# Программное обеспечение (ПО)

- **Системное ПО** управляет работой компьютера и периферийными устройствами, обеспечивая среду выполнения **Прикладного ПО**
- **Прикладное ПО** решает реальные задачи пользователей

# Операционная система

- основной вид системного ПО, комплекс программ, обеспечивающий
  - управление аппаратными средствами компьютера,
  - работу с файлами,
  - ввод и вывод данных,
  - выполнение прикладных программ и утилит.

- Одна из основных задач ОС – предоставить набор системных вызовов (system calls) для удобного управления ресурсами компьютера
  - Как правило, реализация системных вызовов работает в привилегированном режиме процессора
  - Пользовательские программы работают в непривилегированном режиме

# Особенности реализации ОС Windows NT/2000/XP

- ОС состоит из двух частей:
  - Привилегированная часть режима ядра (kernel mode part)
    - 0 уровень привилегий
    - Полный набор инструкций процессора
    - Доступ к 4Гб адресного пространства
    - Доступ к портам ввода-вывода
  - Непривилегированная часть режима пользователя (user mode part):
    - 3 уровень привилегий
    - Ограниченный набор инструкций процессора
    - Доступ к нижним 2 Гб адресного пространств

# Компоненты режима ядра ОС Windows NT/2000/XP

- Слой Аппаратных Абстракций (Hardware Abstraction Layer)
- Микроядро
- Драйверы устройств
- Система управления окнами и графикой
- Исполнительная часть

# Слой Аппаратных Абстракций

- Слой кода, изолирующий части ядра ОС от различий, специфичных для конкретных платформ:
  - Архитектура Ввода/вывода
  - Управление прерываниями
  - Операции с DMA (Direct Memory Access)
  - Системные часы и таймеры

# Микроядро

- Планирование потоков
- Переключение задач
- Обработка прерываний и исключительных ситуаций
- Мультипроцессорная синхронизация

# Драйверы устройств

- Драйверы аппаратных устройств
- Драйверы файловых систем
- Поддержка сети

# Система управления окнами и графическая система

- Функции, обеспечивающие работу графического интерфейса пользователя
  - Окна
  - Элементы управления
  - Рисование
  - Печать

# Исполнительная система

- Базовые службы ОС:
  - Управление памятью
  - Управление процессами и потоками
  - Управление безопасностью
  - Ввод/вывод
  - Межпроцессное взаимодействие

# Компоненты пользовательского режима ОС Windows NT/2000/XP

- Системные процессы
- Службы
  - Журнал событий
  - Планировщик заданий
- Подсистема окружения
  - Обеспечивает пользовательские программы интерфейсом к функциям ОС:
    - Win32 API
    - POSIX
    - OS/2 1.2
    - DOS
    - Win16

# Компоненты Win32 API

- Базовые службы
- Пользовательский интерфейс
- Графика и Мультимедиа
- COM/OLE/ActiveX
- Базы данных и обмен сообщениями
- Сетевые коммуникации и распределенные службы
- Службы Internet/Intranet/Extranet
- Службы установки и управления настройками системы

# Простейшее приложение Windows

```
int APIENTRY WinMain(HINSTANCE /*hInstance*/,
                     HINSTANCE /*hPrevInstance*/,
                     LPSTR     /*lpCmdLine*/,
                     int       /*nCmdShow*/)
{
   // возвращаемся в операционную систему
   return 0;
}
```

# Функция WinMain()

- Подобно тому как функция main() является точкой входа консольного приложения, функция WinMain является **точкой входа** в оконное приложение системы Windows
- Задачи функции WinMain:
  - Инициализация приложения
  - Отображение главного окна приложения
  - Цикл выборки и обработки сообщений
  - Возврат целочисленного значения в операционную систему

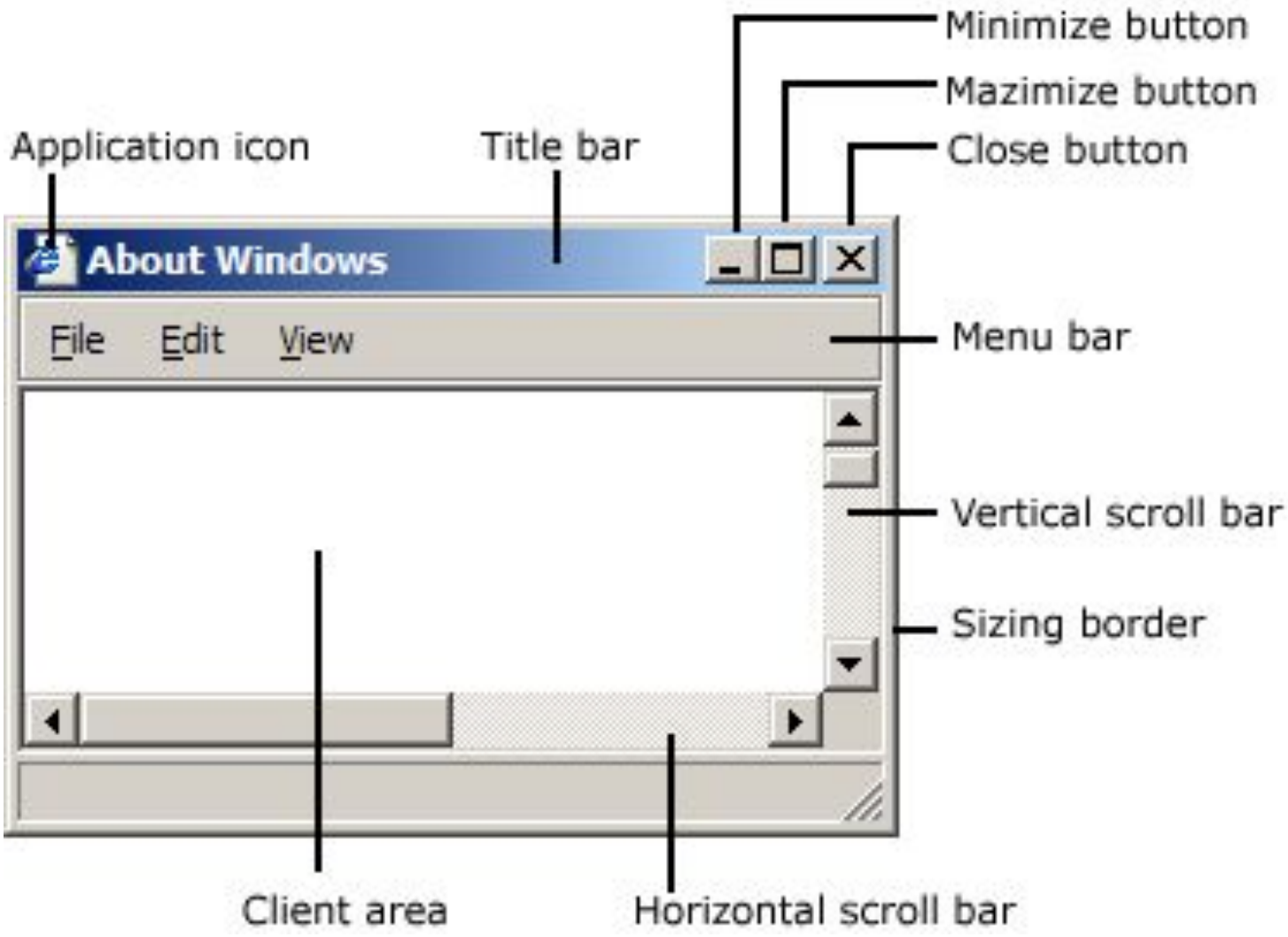
# Windows и Окна

- В графическом приложении Windows **окно** – это прямоугольная область на экране, служащая для отображения выводимой информации и получения пользовательского ввода
- Область экрана – общая для всех окон в системе
  - Только одно окно может в заданный момент времени получать вводимую информацию от пользователя

# Главное окно приложения

- Служит для обработки сообщений и получения данных, введенных пользователем
- Является родительским для окон, создаваемых в приложении
- Возможно существование нескольких главных окон в рамках одного приложения
  - Приложение Microsoft Word

# Элементы окна



# Клиентская область окна

- Часть окна, служащая для вывода текста и графики, а также для получения пользовательского ввода

# Неклиентская область окна

- Включает в себя область **заголовка**, **меню**, **рамку**, области прокрутки, кнопки минимизации/максимизации и закрытия окна
- Как правило, управление неклиентской областью берет на себя ОС, в то время как задача приложения – управление содержимым клиентской области

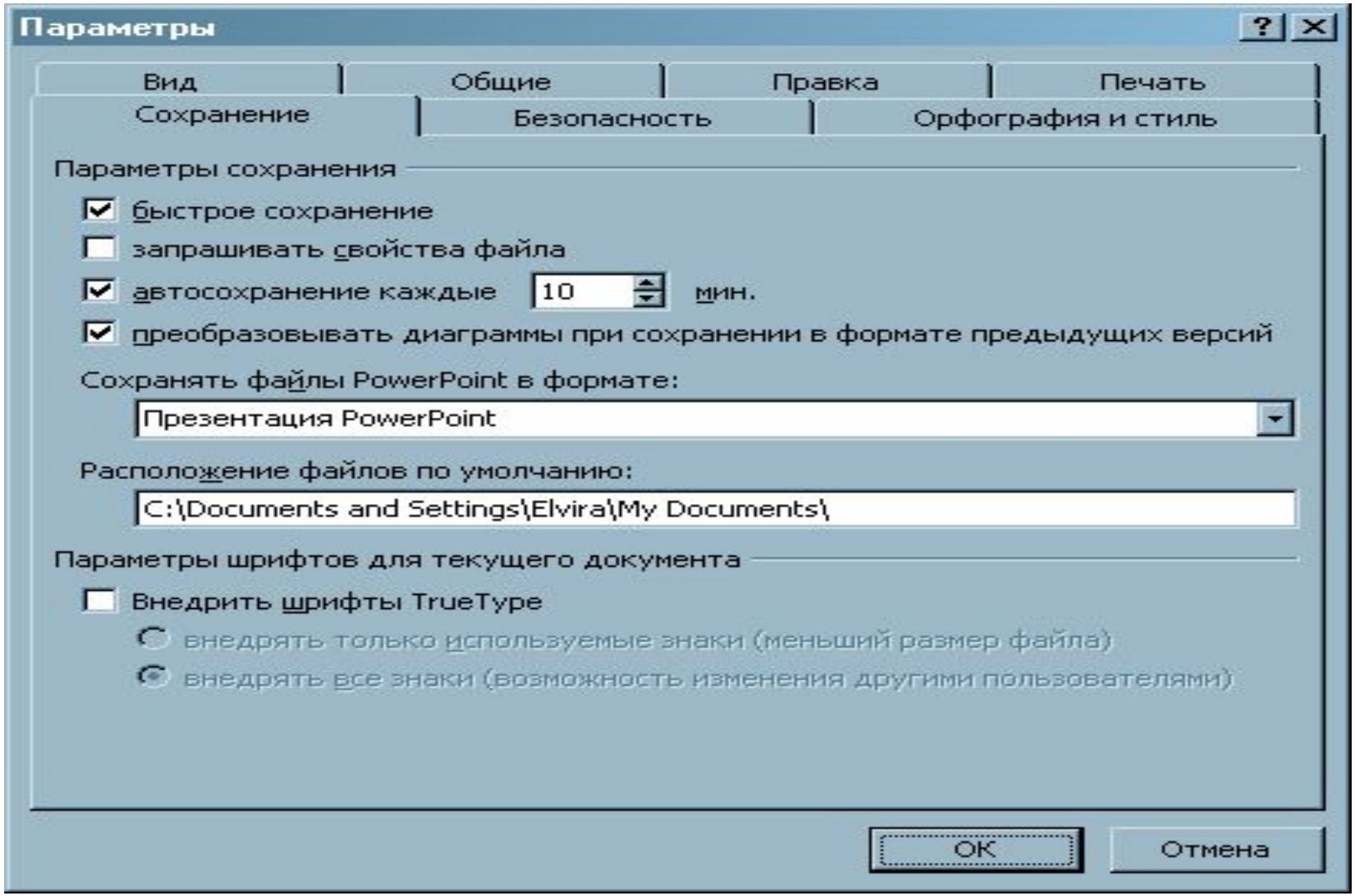
# Элементы управления (controls)

- Вспомогательные окна, служащие для ввода и вывода определенной информации
  - Поля редактирования (Edit box)
  - Статический текст (Static)
  - Кнопки (Button, Radio button, Check Box)
  - Списки (List box, Combo box, List)
  - Деревья (Tree control)
  - Создаваемые пользователем компоненты

# Диалоговые окна

- **Диалоговое окно** – это окно, содержащее один или более элементов управления
  - Диалоговые окна часто не содержат меню или полос прокрутки, обычно не имеют кнопок минимизации/максимизации
  - Используются для ввода пользователем различных данных
- Стандартные диалоговые окна
  - Окно выбора файла
  - Окно выбора шрифта
  - Окно выбора цвета
  - Окно выбора принтера
  - Окно выбора папки
  - Окно настройки свойств страницы

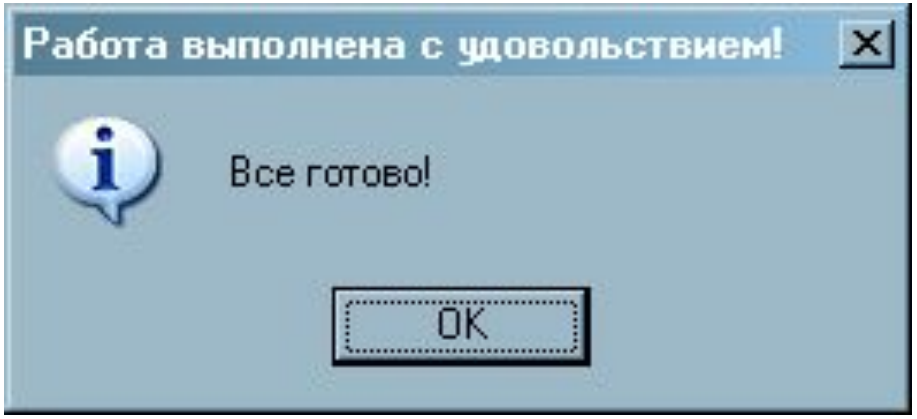
# Пример диалогового окна



# Окна сообщений (Message boxes)

- Специальные диалоговые окна, служащие для отображения коротких сообщений пользователю:
  - Предупреждение об ошибке
  - Уведомление о завершении результатах операции

# Пример окна сообщений



# Приложение HelloWorld

```
int APIENTRY WinMain(HINSTANCE /*hInstance*/,
                     HINSTANCE /*hPrevInstance*/,
                     LPSTR     /*lpCmdLine*/,
                     int       /*nCmdShow*/)
{
   MessageBox(
      NULL,
      "Hello World",
      "Hi", MB_OK | MB_ICONINFORMATION
      );
   return 0;
}
```

# Окно рабочего стола (Desktop Window)

- Данное окно автоматически создается при загрузке ОС
- Окно рабочего стола служит для отображения заднего фона окна и служит основой для размещения других окон

# Создание окна

- При создании окна приложение должно указать следующие атрибуты окна:
  - Имя класса окна
  - Имя окна
  - Стиль окна и дополнительный стиль окна
  - Положение и размер окна
  - Дескриптор родительского окна или окна-владельца
  - Дескриптор меню или дочернего окна
  - Дескриптор экземпляра приложения
  - Данные, характерные для приложения

# Имя класса окна (Window Class Name)

- **Класс окна** – это определенный набор атрибутов, используемый в качестве шаблона при создании окон
  - Каждое окно относится к определенному классу
  - На основе класса окна приложение может создать произвольное количество окон
  - Окна, принадлежащие к одному классу ведут себя сходным образом

# Предопределенные классы окон

- В системе зарегистрированы предопределенные классы окон
  - BUTTON,
  - LISTBOX,
  - COMBOBOX,
  - STATIC,
  - EDIT,
  - MDICLIENT,
  - RICHEDIT_CLASS,
  - SCROLLBAR

# Регистрация класса окна

- Главное окно приложения не имеет предопределенного класса, поэтому приложению необходимо зарегистрировать класс своего главного окна самостоятельно
- Регистрация класса окна осуществляется при помощи функции RegisterClassEx()

# Имя окна

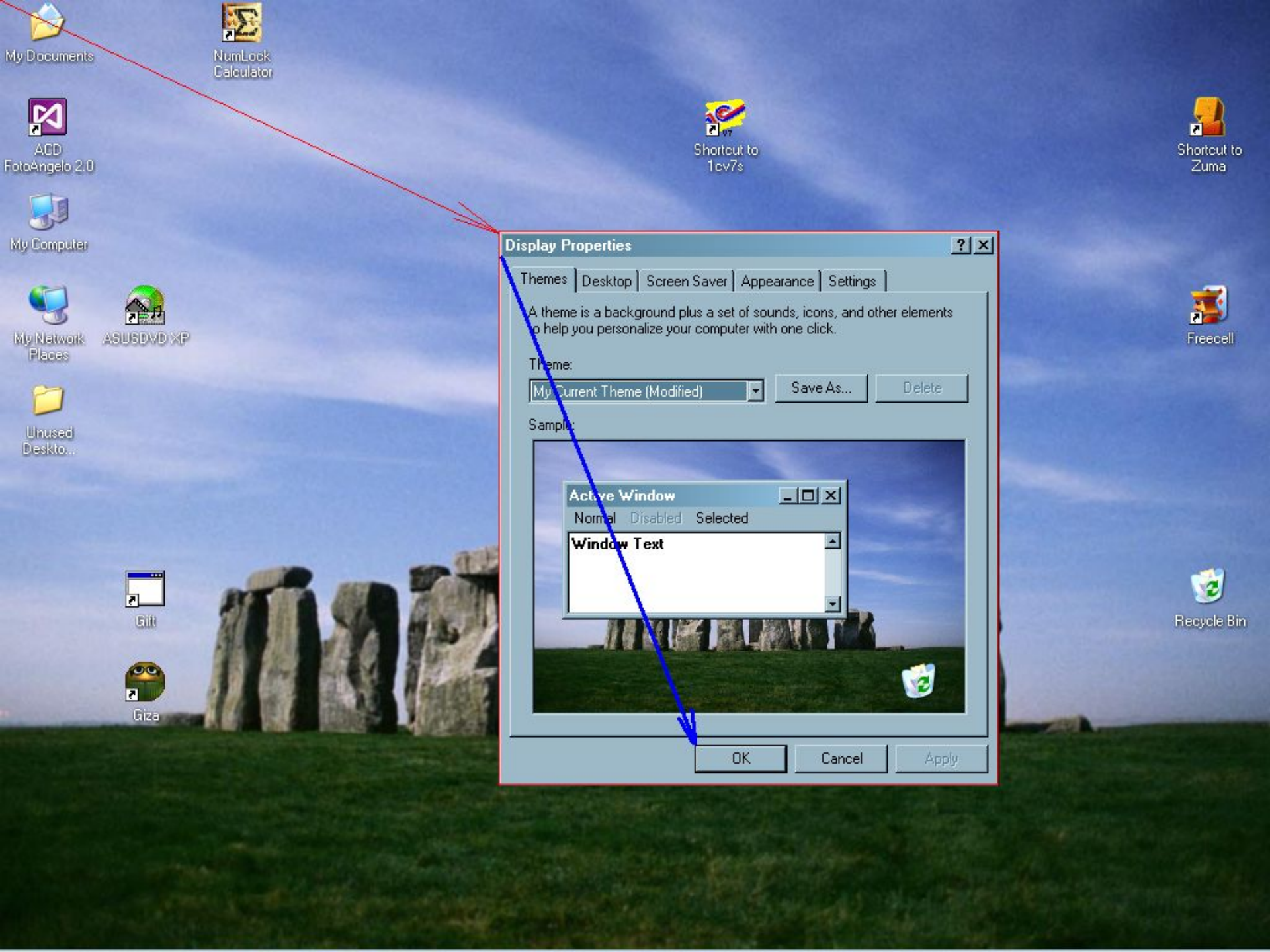
- Строка, идентифицирующая данное окно для пользователя
  - Главное окно приложения и диалоговые окна отображают имя окна в строке заголовка
  - Некоторые элементы управления – внутри клиентской области
    - Edit box, Static text, Button
  - Некоторые элементы управления – не отображают вообще
    - List box, Combo box

# Стили окна

- **Стиль окна** – именованная целочисленная константа, определяющая определенные аспекты поведения и внешнего вида окна
- Окно может иметь несколько стилей окна
- Некоторые стили могут быть применены ко всем окнам, в то время как другие – к окнам определенного класса

# Положение и размеры окна

- Положение и размеры окна задаются в пикселях
- Положение окна – координаты верхнего левого угла окна относительно верхнего левого угла экрана, либо относительно родительского окна (для дочерних окон)


My Documents
NumLock Calculator
ACD FotoAngelo 2.0
Shortcut to 1cv7s
Shortcut to Zuma
My Computer
My Network Places
ASUSDVD XP
Freecell
Unused Deskto...
Gift
Giza
Recycle Bin
Display Properties
Themes
Desktop
Screen Saver
Appearance
Settings
A theme is a background plus a set of sounds, icons, and other elements to help you personalize your computer with one click.
Theme:
My Current Theme (Modified)
Save As...
Delete
Sample:
Active Window
Normal
Disabled
Selected
Window Text
OK
Cancel
Apply

# Дескриптор родительского окна

- Некоторые окна могут иметь родителей
  - такие окна называются дочерними
  - Родительские окна определяют систему координат своих дочерних окон
- Окна без родителей – окна верхнего уровня (Top Level Window)

# Идентификатор меню или идентификатор дочернего окна

- Дочерние окна могут иметь **идентификатор дочернего окна** – уникальное значение, ассоциированное с данным дочерним окном
  - Система позволяет получить дескриптор, соответствующий окну с указанным идентификатором при помощи функции
  - Окна, не являющиеся дочерними, могут иметь меню

# Дескриптор экземпляра приложения (HINSTANCE)

- Является идентификатором экземпляра запущенного приложения
- Передается в качестве параметра функции WinMain
- Используется при создании окон, работы с ресурсами, потоками выполнения и т.п.

# Дескриптор Окна (Window Handle)

- Каждое окно в системе после своего создания получает **дескриптор** – уникальный идентификатор, однозначно определяющий данное окно в системе
  - Данный идентификатор имеет тип **HWND**
  - Получив дескриптор окна приложение может передавать его в различные функции работы с окнами

# Создание окна на основе зарегистрированного класса

- Создание окна осуществляется при помощи функции CreateWindowEx()
- После своего создания окно является невидимым
  - Чтобы показать его, необходимо вызвать функцию ShowWindow()

# Сообщения

- Обработка сообщений лежит в основе работы приложений Windows
  - Сообщения создаются системой и приложением в ответ на каждое событие, происходящее в Windows:
    - Движения мыши, нажатие клавиш, события таймера и т.п.
- Многие приложения проводят большую часть времени в ожидании сообщений
  - Это позволяет оптимально использовать ресурсы процессора в условиях параллельной работы нескольких приложений

# Что такое сообщение?

- Это структура данных, определенная следующим образом:

```
typedef struct  tagMSG
{
    HWND hwnd;   // дескриптор окна
    UINT message;   // идентификатор сообщения
    WPARAM wParam;  // параметр wParam
    LPARAM lParam;  // параметр lParam
    DWORD time;  // время отправки сообщения
    POINT pt;    // положение курсора мыши
} MSG;
```

# Очередь сообщений

- Система передает сообщения в очередь сообщений нити
  - Нить (thread) – это независимый путь выполнения программы внутри процесса
  - Внутри процесса существует как минимум одна нить выполнения

# Обработка сообщений

- Нить или процесс выбирает сообщения из очереди с использованием цикла обработки сообщений
- Обычно цикл сообщений имеет следующий вид:

```
MSG msg;
while (GetMessage(&msg, NULL, 0,
0))
{
    TranslateMessage(&msg);
    DispatchMessage(&msg);
}
```

# Цикл обработки сообщений

- Функция **GetMessage()** осуществляет выбор сообщения из очереди сообщений
- Функция **TranslateMessage()** осуществляет перевод нажатий виртуальных клавиш в символьные сообщения, которые будут считаны при следующем вызове GetMessage() или PeekMessage()
- Функция **DispatchMessage()** перенаправляет сообщение из цикла обрабоки в соответствующую **оконную процедуру**

# Сообщение WM_CREATE

- Уведомление о том, что окно создается.
- Вызывается в ходе работы CreateWindowEx()

# Сообщение WM_MOVE

- Посылается окну **после** его перемещения
- Сообщение **WM_MOVING** посылается окну во время его перемещения
  - Приложение может изменить положение окна в обработчике данного сообщения

# Сообщение WM_SIZE

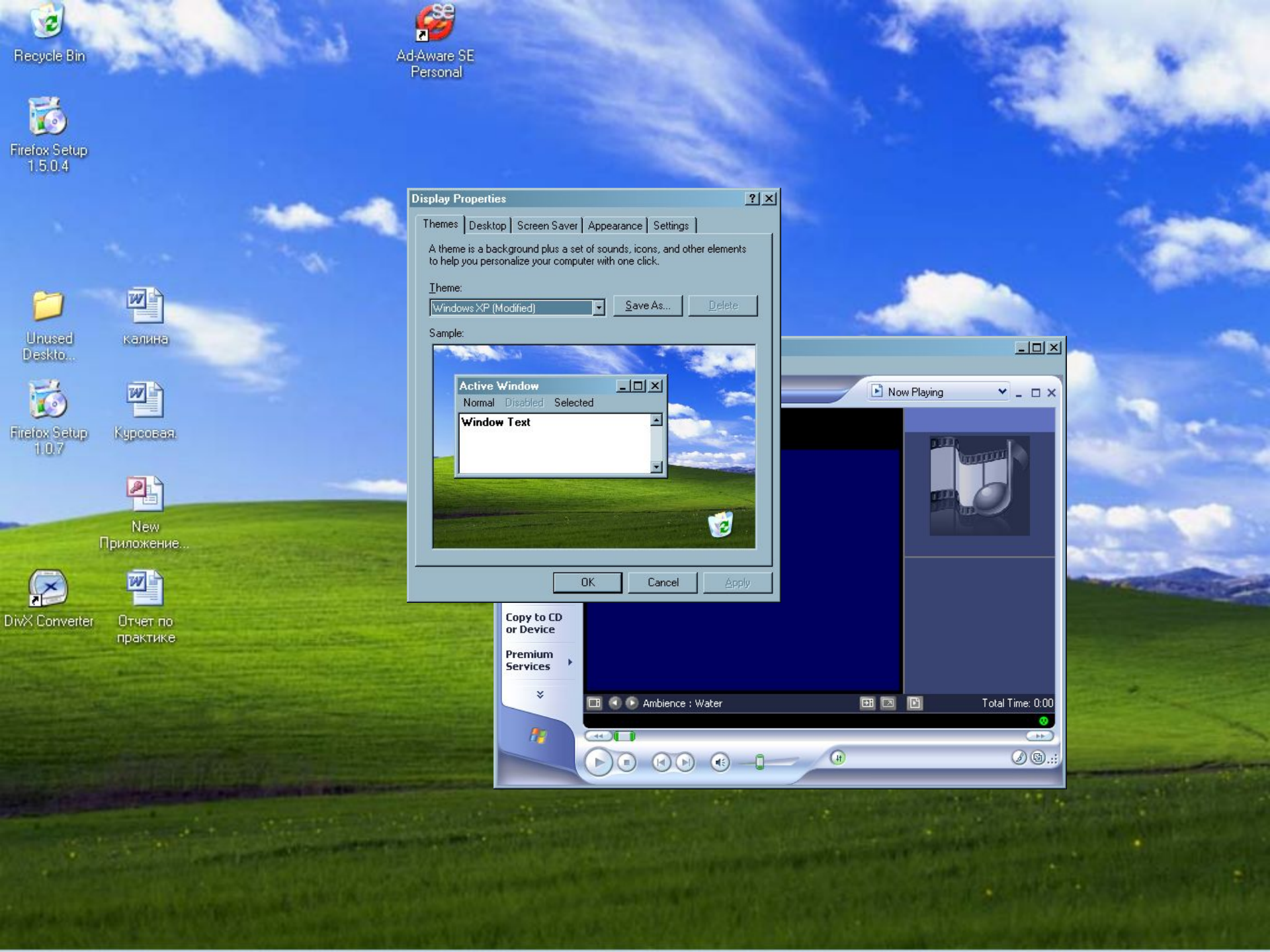
- Посылается окну **после** изменения его размеров
- Сообщение **WM_SIZING** посылается **во время** изменения размеров окна
  - Приложение может изменить размеры и положение окна в обработчике данного сообщения

# Особенности графического вывода в многозадачной среде

- В многозадачных ОС одновременно могут работать несколько приложений
- В ОС Windows приложения могут создавать окна для визуального отображения информации и получения ввода пользователя
- Данные окна используют общую площадь экрана монитора, порой перекрывая друг друга

# Эта ситуация динамически изменяется

- Подобно тому, как на обычном столе человек может перекладывать бумажные документы с места на место, пользователь графической ОС может переключаться между окнами, перемещать их по экрану
  - При этом ранее закрытые области экрана и окон должны своевременно показывать пользователю ранее скрытую часть информации
  - Такой подход позволяет обеспечить интуитивно понятное поведение графического интерфейса пользователя

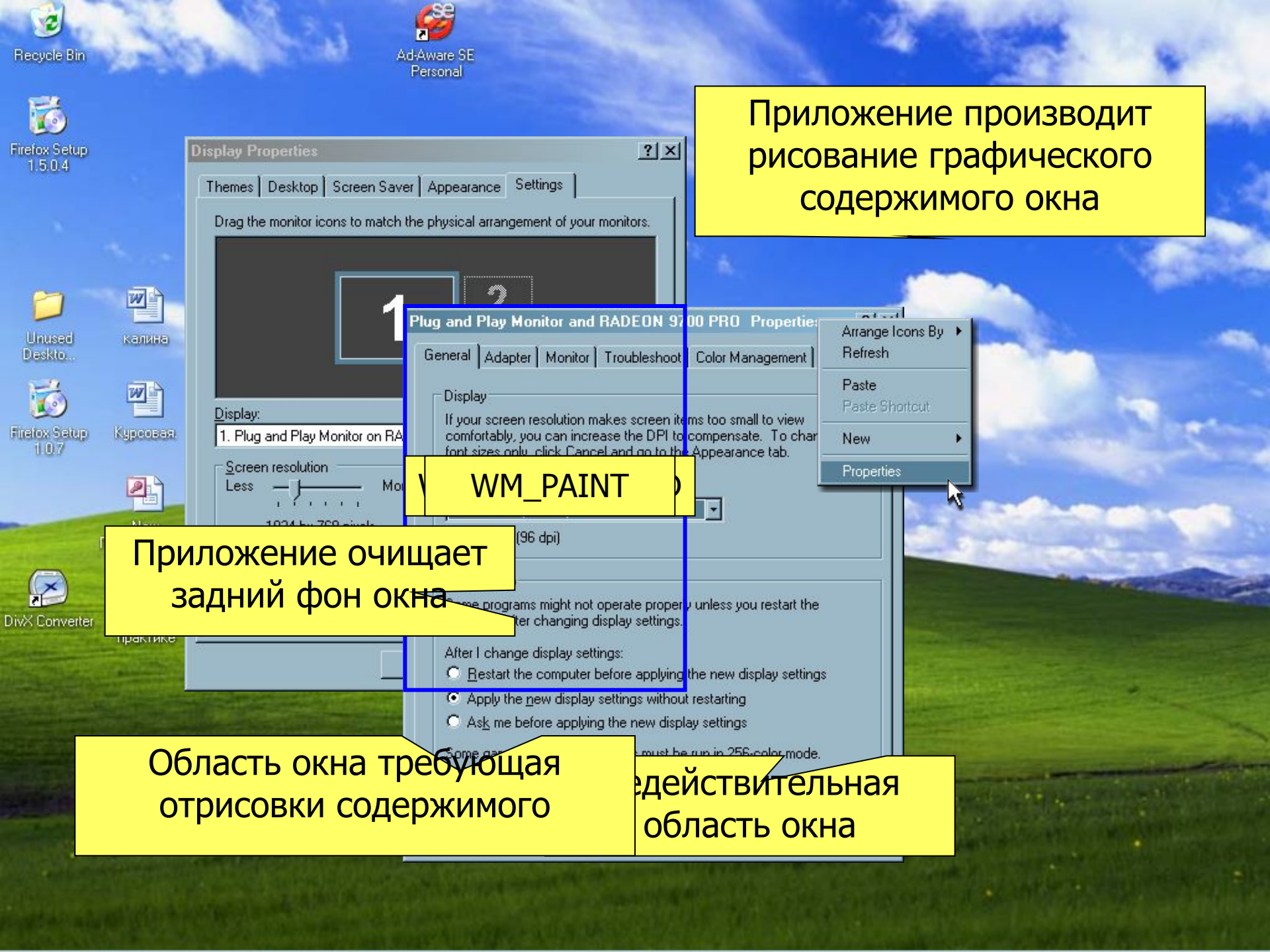

Recycle Bin
Ad-Aware SE Personal
Firefox Setup 1.5.0.4
Unused Deskto...
калина
Firefox Setup 1.0.7
Курсовая
New Приложение...
DivX Converter
Отчет по практике
Display Properties
Themes
Desktop
Screen Saver
Appearance
Settings
A theme is a background plus a set of sounds, icons, and other elements to help you personalize your computer with one click.
Theme:
Windows XP (Modified)
Save As...
Delete
Sample:
Active Window
Normal
Disabled
Selected
Window Text
OK
Cancel
Apply
Now Playing
Copy to CD or Device
Premium Services
Ambience : Water
Total Time: 0:00

# Как этого достичь?

- Каждое окно должно быть ответственным за корректное и своевременное отображение содержащейся в нем информации

# Роль операционной системы Windows

- ОС Windows хранит информацию обо всех окнах в системе
  - Пользовательские приложения не имеют прямого доступа к этой информации
  - Все операции над окнами возможны только через функции ОС, принимающие **дескриптор окна**
  - Это позволяет системе отслеживать изменения во взаимном положении окон, определять области требующие перерисовки и уведомлять приложения посредством отсылки сообщений


Приложение производит рисование графического содержимого окна
WM_PAINT
Приложение очищает задний фон окна
Область окна требующая отрисовки содержимого
едействительная область окна
Display Properties
Themes
Desktop
Screen Saver
Appearance
Settings
Drag the monitor icons to match the physical arrangement of your monitors.
Plug and Play Monitor and RADEON 9700 PRO Properties
General
Adapter
Monitor
Troubleshoot
Color Management
Display
Screen resolution
Less
After I change display settings:
Restart the computer before applying the new display settings
Apply the new display settings without restarting
Ask me before applying the new display settings
Arrange Icons By
Refresh
Paste
Paste Shortcut
New
Properties
Recycle Bin
Ad-Aware SE Personal
Firefox Setup 1.5.0.4
Unused Deskto...
калина
Firefox Setup 1.0.7
Курсовая
DivX Converter

# Сообщение WM_ERASEBKGND

- Посылается окну в тот момент, когда его фон должен быть очищен
- Это подготовка недействительной области окна к отрисовке
- Приложение может использовать это сообщение для заполнения заднего фона окна каким-нибудь узором
  - Оконная процедура по умолчанию просто заполняет задний фон кистью, указанной при регистрации класса окна

# Сообщение WM_PAINT

- Посылается окну в том случае, когда какая-то его область требует перерисовки
  - Приложение вызывало функцию RedrawWindow() или UpdateWindow()
  - Область окна стала недействительной при сворачивании окна на верхнем уровне

# Обработка сообщения WM_PAINT

```
LRESULT CALLBACK MainWndProc(HWND hWnd, UINT uMsg,
WPARAM wParam, LPARAM lParam)
{
    switch (uMsg)
    {
    case WM_PAINT:
        {
            PAINTSTRUCT ps;
            HDC dc = BeginPaint(hWnd, &ps);
            // выполняем рисование
            // ...
            EndPaint(hWnd, &ps);
        }
        break;
    // обработка остальных сообщений
    }
}
```

# Сообщение WM_DESTROY

- Посылается окну после того как оно убрано с экрана перед его уничтожением
  - Дочерние окна получают данное сообщение вслед за родительским
- При уничтожении главного окна приложения обычно происходит вызов функции **PostQuitMessage()**

# Сообщение WM_QUIT

- Данное сообщение посылается **приложению** (а не окну) в качестве запроса на завершение его работы
- Функция GetMessage() возвращает значение 0 при извлечении сообщения WM_QUIT

# Оконная процедура

- Каждое окно имеет связанную с ним **оконную процедуру**
  - Все окна, относящиеся к данному классу используют одну и ту же оконную процедуру по умолчанию
  - Поведение и внешний вид окна определяются реакцией оконной процедуры на получаемые сообщения

# Регистрация оконной процедуры

- Регистрация оконной процедуры происходит одновременно с регистрацией класса окна
  - Приложение может создать произвольное количество окон, относящихся к данному классу и использующих одну и ту же оконную процедуру
    - Аккуратная работа с глобальными переменными
  - Оконная процедура должна быть **реентерабельной**, т.е. должна позволять ее повторный вызов явным или неявным образом из самой себя.
    - Минимум локальных переменных внутри оконной процедуры

# Структура оконной процедуры

- Оконная процедура имеет следующий прототип:
  - LRESULT CALLBACK WindowProc(
    HWND hwnd,
    UINT uMsg,
    WPARAM wParam,
    LPARAM lParam);

# Оконная процедура по умолчанию

- Для многих оконных сообщений в системе предусмотрено поведение по умолчанию
  - Перемещение окон
  - Минимизация/максимизация
  - Реакция на движения мыши
- Все сообщения, не обрабатываемые явно в оконной процедуре приложения, должны направляться в оконную процедуру по умолчанию при помощи вызова функции
    - DefWindowProc()

# Пример простейшей оконной процедуры

```
LRESULT CALLBACK MainWndProc(
    HWND hWnd,
    UINT uMsg,
    WPARAM wParam,
    LPARAM lParam)
{
    switch(uMsg)
    {
    case WM_DESTROY:
        PostQuitMessage(0);
        return 0;
    }
    return DefWindowProc(hWnd, uMsg, wParam, lParam);
}
```

# Перегрузка (Subclassing) оконной процедуры

- При создании окна система выделяет память для хранения информации об окне, включая адрес оконной процедуры
- При посылке сообщения окну система извлекает адрес оконной процедуры из информации, специфичной для окна с указанным дескриптором, и вызывает оконную процедуру
- **Subclassing** – технология, позволяющая перехватывать и обрабатывать в обход оригинальной оконной процедуры

# Принцип SubClassing’a

- Приложение может заменить адрес оконной процедуры отдельно взятого окна или оконного класса адресом собственной оконной процедуры подкласса (subclass procedure)
  - Subclassing возможет лишь в рамках одного процесса
- После этого новая оконная процедура сможет обрабатывать сообщения окна:
  - Передача их в оригинальную оконную процедуру без изменения
  - Модификация сообщения с последующей передачей его в оригинальную оконную процедуру
  - Обработка сообщения

# Виды Subclassing’а

- Instance subclassing
  - У созданного окна заменяется адрес оконной процедуры
- Global Subclassing
  - Подменяется адрес оконной процедуры целого класса окон

# Superclassing

- Это технология, позволяющая создать новый класс окна с базовой функциональностью существующего класса + добавить собственную функциональность
  - Часто применятся для расширения функционала существующих стандартных элементов управления

# Шаги для создания оконного приложения для ОС Windows

- Регистрация класса окна
- Создание главного окна приложения
- Выборка сообщений из цикла обработки сообщений
- Обработка сообщений в оконной процедуре окна