Лекция 4.  
Технологии программирования.

Введение в программирование под Windows

Операционная система Windows является много задачной, что позволяет одновременно запускать несколько системных и пользовательских процессов, при этом процессами могут быть как различные приложения, так и копии одной программы.

Особенностями windows следует считать:

* Файловая система, которая использует таблицу размещения файлов защищенного режима. При работе с файлами программист может сделать его виртуальным до размера 1 Гб. После этого с отображаемым файлом можно работать как с участком ОЗУ. Отображаемый файл можно сделать одновременно доступным нескольким приложениям, что позволяет совершать обмен файлами между приложениями
* Вытесняющая многозадачность.
* Не сегментированная модель памяти.
* Библиотеки динамической загрузки. Так как существует необходимость вызова одних и тех же функции, то возникает необходимость выполнения динамической загрузки и компоновки библиотек. Они обеспечивают однократную загрузку в систему с возможностью организации к нему общего доступа, при этом выгрузка модуля осуществляется в том случае, если ни одно из запущенных приложений не выполняет обращение к нему.
* Использование очереди сообщений.
* Регистрационная база данных. Содержит информацию о всех приложениях в системе.

Многозадачность – свойство ОС или среды программирования обеспечивать возможность параллельной или псевдо параллельной обработки нескольких процессов. Реальная многозадачность возможна только в распределенных системах. В настоящее время выделяют 3 типа многозадачности:

* Не вытесняющая. Тип при которой ОС способная загрузить в память 2 или более приложения но процессорное время выделяется только одному приложению. Для выполнения другого приложения оно должно быть активировано либо специальной командой ОС, либо после завершения активного приложения.
* Совместная (кооперативная). Тип многозадачности при которой следующая задача выполняется только после того как текущая задача **явно объявит себя** готовой отдать процессорное время другим задачам. При этом задачи в фоновом режиме полностью не замораживаются. Приложения захватывает процессорное время на тот интервал, которое считает нужное, но периодически происходит переключение контекста, т.е. передача управления другим задачам. Основным приемуществом является отсуствие необходимости защищать разделяемые ресурсы, что упрощает управление такими задачами и упрощает перенос с много задачных. Основным недостатком является то, что при потере работоспособности одного приложения может уронить всю систему.
* Вытесняющая (приоритетная) многозадачность. Вид многозадачности в которой **ОС сама передает управление** от одной программы к другой в случае завершении тех или иных операций или истечение времени таймера.

**Процесс и поток**

Процесс создается всякий раз когда программа загружается в ОЗУ для выполнения процессу выделяется… Сразу после запуска процесса создается задача или поток. Поток – фрагмент кода приложения, который может выполнятся автономно независимо от других потоков в системе в рамках одного процесса. При необходимости поток может порождать новые потоки для реализации собственного алгоримтма работы. Все потоки имеют доступ к памяти выделенной процессу и именно потоку выделяется процессорное время.

Структура сообщения

|  |
| --- |
| *Typedef struct MSG {*  *HWND hwnd; // Идетефикатор окна, к которому полетит сообщение на обработку*  *UINT message; // Ид сообщения на основания которого в оконной процедуре определяется обработчик*  *WPARAM wParam;*  *LPARAM lParam; // Содержат необходимую информацию для обработки сообщения*  *DWORD time; // Время добавления сообщения в очередь*  *Point pt; // Определяет положение курсора мыши во время события.*  *}* |

Общий алгоритм рботы любого приложения выглядит следущющим образом. ОС использует WinApi запускает приложение порождая при этом процесс, а затем поток WinMain(). Внутри функции инициализируется необходимые переменные, определяются характеристики используемых окон и непосредственно создается требуемое окно. Запускается цикл получения сообщений. При получений сообщения вызывается процедура wndProc(). Представляет собой case структуру, позволяющую выбрать обработчик на основании ид сообщения. Цикл обработки сообщеения так же поддеживается на основе вызова winAPI.

Функция WinMain

Int APIENTRY WinMain(

HINSTACE hIinstance // ид приложения, является уникальным, используется системой для //органицазии переключения контектса и защиты памяти

HPINSTANCE hPrevInstance // является ид запущенной копии предыдущего приложения.

LPSTR pCmd // указывает на командную строку

Int nCMDShow // Определяет сбособ отбражения окна.

) {}

Для каждого окна состается структура типа wndClassex в которой определяется набор параметров:

* cbSize = 123; // определяет размер окна в байтах
* hIconSm = 0; // Определяет ссылку на значек приложения
* style; // Определяет стиль окна
* lpfnWndProc; // Определяется ссылка на функцию wndProc (обработка сообщений)
* cbClsExtra; // Число байт запращиваемых у ОС для собственных данных окна.
* hInstance; // Ид приложения к которому принадлежит окно
* hIcon; // Иконка окна
* hCursor; // Определяет курсор над окном
* hbrBackground; // Фон окна
* lpszMenuName; // Указатель на главное меню
* lpszClassName; // Определяет имя создаваемого класса, которое может быть доступно другим приложениям. Через вызов системных функций.

Для регистраций окна вызывается функция RegisterClassex().

Внутри главной функции приложений цикл обработки сообщения запускается путем вызова функции GetMessage(). Если в очереди сообщений нет сообщений, то передается управление другому приложению. Если сообщение есть, то проверяется на сообщение-завершения. Если это не приложение-завершение, то выполнение передается в цикл обработки. В цикле обработки выполняется 2 основных функций.

DispatchMessage(&msg) - распределяет сообщения по окнам приложения.

TranslateMessage(&msg) - преобразует коды клавиш в структуры символьного сообщения.

getMessage (

* Указатель на сообщение
* индетефикатор окна которому в момент сообщения принадлежал фокус ввода. Если Null, то вернутся все сообщения приложения.
* Фильтры сообщений. При 0-лях нет фильтра.

)

WM\_QUERYENDSESSION – сообщение при завершении работы windows.

HWND FindWindow(LPCTSTR lpClassName, LPCTSTR lpWindowName) – поиск окна.