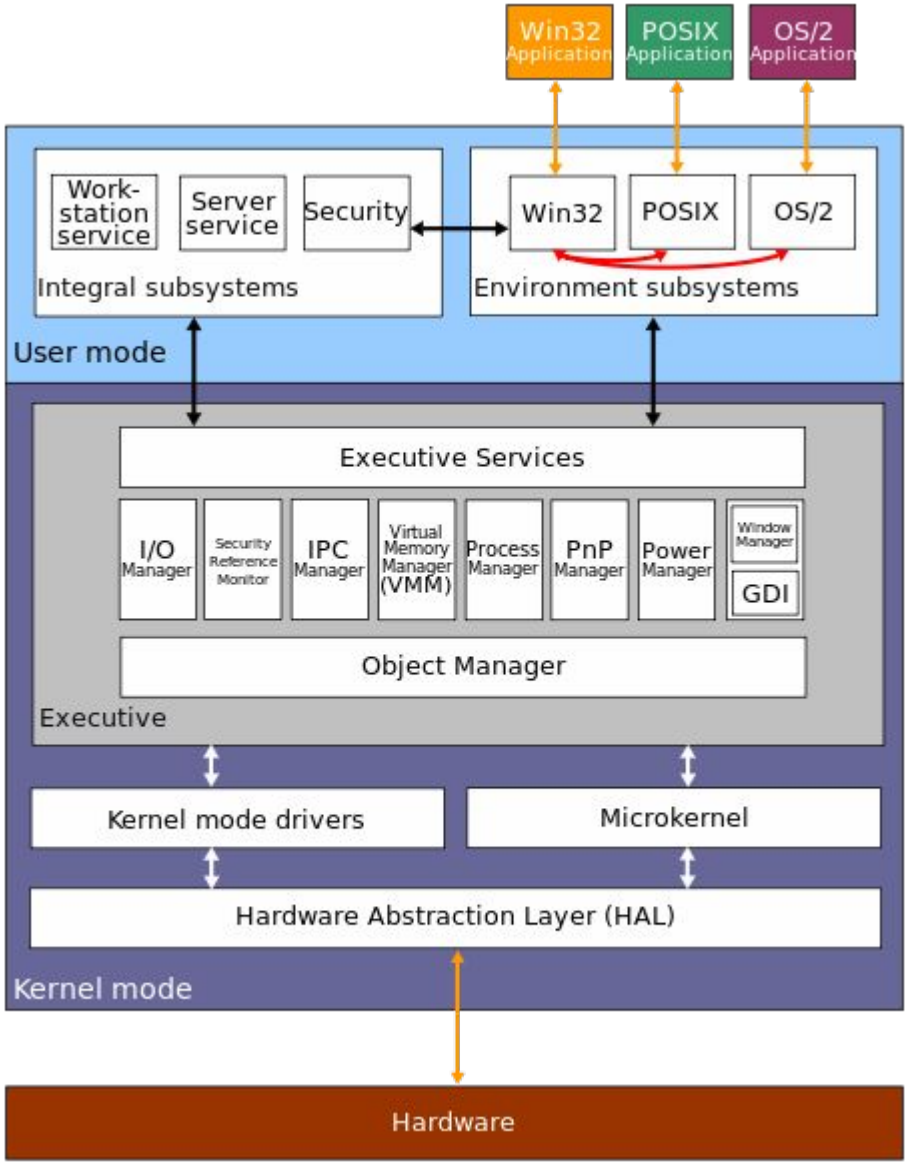
Системное программирование. Лекция 2. Введение в Windows API

Введение в Windows API

- *Архитектура ОС Windows 2000/XP:*
  - *Реализация Win32:*
    - kernel32.dll
    - gdi32.dll
    - user32.dll

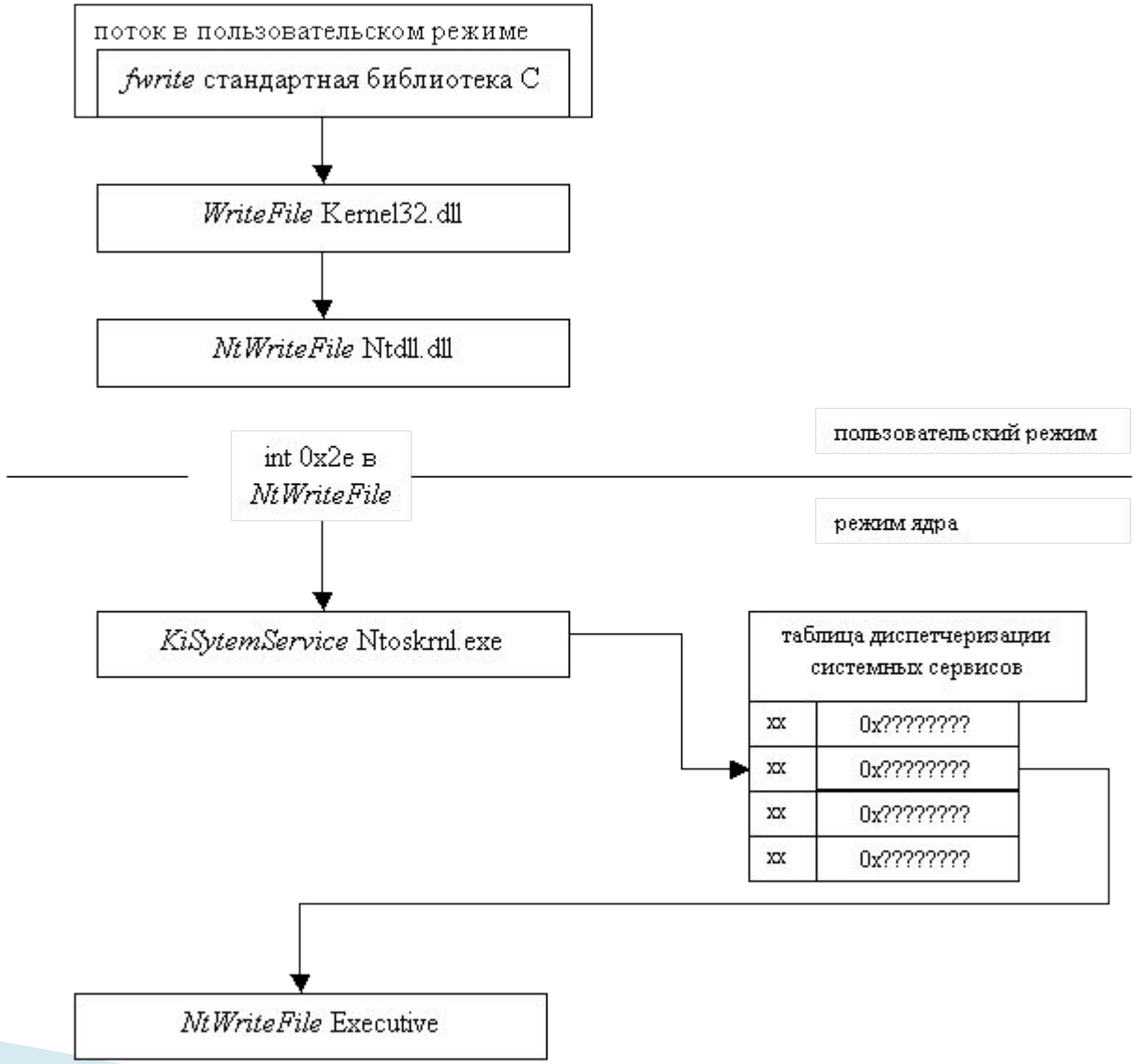


Введение в Windows API

- *Схема обработки функций ввода/вывода:*

Введение в Windows API

- *Схема обработки системного сервиса:*



Введение в Windows API

- *Принципы, лежащие в основе Windows API:*
  - Системные ресурсы представляются в виде объектов:
    - Объект Windows – структура данных, представляющая системный ресурс
  - Основные классы объектов Windows:
    - Объекты ядра (файлы, процессы, потоки, сокеты…)
    - Объекты интерфейса пользователя (окна, курсоры, меню,…)
    - Объекты графического интерфейса (перья, кисти,…)
  - Манипуляции с объектами Windows – только через Windows API
  - Для идентификации объектов используются дескрипторы – специальные структуры, указывающие на объекты ОС и хранящие информацию о них
    - Дескриптор – тип `HANDLE`
    - Создание объектов: `Create<имя>` (например

Введение в Windows API

- *Принципы, лежащие в основе Windows API:*
  - Имеется собственный набор типов данных:
    - Типы пишутся заглавными буквами (для Си)
    - Примеры типов: `HANDLE`, `BOOL`, `DWORD`, `LPTSTR`
  - Типы Windows API – «переобозначенные» базовые типы с учетом параметров компиляции:
    - `typedef unsigned long DWORD`
  - В именах типов Windows API «*» не используется:
    - `LPTSTR` – это `TCHAR *`
    - `LPCTSTR` – это `const TCHAR *`
    - `LPDWORD` – это `DWORD *`
  - Типы данных, представляющие собой указатели могут записываться в двух вариантах:
    - `LPDWORD` = `PDWORD`
    - `LPVOID` = `PVOID`

Введение в Windows API

- *Принципы, лежащие в основе Windows API:*
  - Для возможности использовать Windows API нужно подключать библиотеки (модули)
    - Библиотека <windows.h> – для C/C++
    - Модуль Windows – для Delphi
  - В библиотеках (модулях) содержатся внешние определения функций («мостик» для обращения к соответствующим DLL-библиотекам)

Введение в Windows API

- *Основные типы данных в Windows API:*
  - Типы данных объявлены в:
    - <WinDef.h>, <WinNT.h>, <BaseTsd.h> и некоторых других
  - Константы:
    - `#define CONST const`
  - Пустой (любой) тип:
    - `#define VOID void`
  - Целочисленные типы:
    - `typedef unsigned char BYTE;`
    - `typedef unsigned short WORD;`
    - `typedef unsigned long DWORD;`
    - `typedef short SHORT;`
      `typedef unsigned short USHORT;`
    - `typedef int INT; typedef unsigned int UINT;`
    - `typedef long LONG; typedef unsigned long ULONG;`
  - Вещественные типы:
    - `typedef float FLOAT;`

Введение в Windows API

- *Основные типы данных в Windows API:*
  - Логические типы:
    - `typedef int` **`BOOL;`**
    - `typedef BYTE` **`BOOLEAN;`**
  - Символьные типы:
    - `typedef char` **`CHAR;`** `typedef unsigned char` **`UCHAR;`**
    - `typedef wchar_t` **`WCHAR;`**
    - `#ifdef UNICODE typedef WCHAR` **`TCHAR;`** `#else typedef char` **`TCHAR;`**
  - Указатели:
    - `typedef BOOL *`**`PBOOL,*LPBOOL;`**
    - `typedef BYTE *`**`PBYTE,*LPBYTE;`**
    - `typedef int *`**`PINT,*LPINT;`**
    - `typedef WORD *`**`PWORD,*LPWORD;`**
    - `typedef DWORD *`**`PDWORD,*LPDWORD;`**
    - `typedef long *`**`PLONG,*LPLONG;`**
    - `typedef FLOAT *`**`PFLOAT;`**
    - `typedef UINT *`**`PUINT,*LPUINT;`** и др.

Введение в Windows API

- *Основные типы данных в Windows API:*
  - Указатели:
    - `typedef void *PVOID,*LPVOID;`
    - `typedef CONST void *PCVOID,*LPCVOID;`
    - `typedef CHAR *PCHAR;`
    - `typedef CHAR *PSTR,*LPSTR;`
    - `typedef WCHAR *PWSTR,*LPWSTR;`
    - `typedef CONST CHAR *PCSTR,*LPCSTR;`
    - `typedef CONST WCHAR *PCWSTR,*LPCWSTR;`
    - `#ifdef UNICODE typedef LPWSTR PTSTR,LPTSTR;`
      `#else typedef LPSTR PTSTR,LPTSTR;`
    - `#ifdef UNICODE typedef LPCWSTR PCTSTR,LPCTSTR;`
      `#else typedef LPCSTR PCTSTR,LPCTSTR;`
  - Дескриптор объектов:
    - `typedef PVOID HANDLE;`
  - Win32/Win64 (пара примеров):
    - `typedef unsigned int DWORD32;`
    - `typedef unsigned __int64 DWORD64;`

Введение в Windows API

- *Символы ASCII и Unicode (UTF-16):*
  - 8-битовые символы (ASCII): `char` = `CHAR`
  - 16-битовые символы (UTF-16): `wchar_t` = `WCHAR`
- Для написания обобщенных приложений нужно:
  1. Определить все символы и строки с использованием обобщенных типов: `TCHAR`, `LPTSTR`, `LPCTSTR`
  2. Включить в самом начале во все модули (для UTF-16):
     - `#define UNICODE` – для управления компиляцией библиотек Windows
     - `#define _UNICODE` – для управления компиляцией стандартных библиотек C
     - **Примечание**: Лучше управлять выбором через тип проекта
  3. Размеры буферов в операциях ввода/вывода и

Введение в Windows API

- Для написания обобщенных приложений нужно:
  4. Включить библиотеку `<tchar.h>` перед `<Windows.h>`
  5. Для ввода/вывода и преобразования строк использовать функции библиотеки `<tchar.h>`:
     - `_tprintf` вместо `printf`
     - `_tscanf` вместо `scanf`
     - `_totupper` вместо `toupper`
     - `_totlower` вместо `tolower`
     - `_ttoi` вместо `atoi`
     - и т.д.
     - **Примечание**: в библиотеке `<tchar.h>` определен тип `_TCHAR` – это аналог `TCHAR` Windows API
  6. Использовать макрос `_T(<строка>)` для строковых констант:
     - Пример: `_T("Hello world")`
     - **Примечание**: 16-битовую строковую константу можно

Введение в Windows API

- Для написания обобщенных приложений нужно:
  7. Использовать обобщенную главную функцию:
     - `_tmain` вместо `main` и `wmain` – для консольных
     - `_tWinMain` вместо `WinMain` и `wWinMain` – для Win32
- Windows API предоставляет свои функции для работы с обобщенными строками и символами:
  - `CharUpper`, `CharLower`, `IsCharAlphaNumeric` и др.
  - Учитываются региональные особенности
- Функции Windows API автоматически являются обобщенными:
  - Например, для функции `CreateFile`:
    - `CreateFileA` – вариант с использованием ASCII-строк
    - `CreateFileW` – вариант с использованием UNICODE-строк