ЛАБОРАТОРНА РОБОТА 4
ПРОГРАМУВАННЯ ВЛАСТИВОСТЕЙ ОКРЕМИХ ВИДІВ РЕСУРСІВ WINDOWS

4.1 Мета заняття

Познайомитися з видами деяких ресурсів Windows та основними прийомами роботи з системними функціями Win API на прикладі складання та налагодження програми маніпулювання цими ресурсами.

4.2 Методичні вказівки з організації самостійної роботи студентів

4.1.1 Опис ресурсів та визначення

Значки, курсори, вікна меню і діалогу пов'язані між собою. Все це види ресурсів (resources) Windows. Ресурси є даними, і вони зберігаються в .EXE файлі програми, але розташовані вони не в області даних, де зазвичай зберігаються дані виконуваних програм. Таким чином, до ресурсів немає безпосереднього доступу через змінні, визначені у початковому тексті програми. Вони повинні бути явно завантажені з файлу з розширенням .EXE до пам'яті.

Коли Windows завантажує до пам'яті код і дані програми для її виконання, вона зазвичай залишає ресурси на диску. Тільки тоді, коли Windows потрібен конкретний ресурс, вона завантажує його до пам'яті. Ви могли звернути увагу на таке динамічне завантаження ресурсів під час роботі з Windows-програмами. Коли ви перший раз викликаєте вікно діалогу програми, Windows зазвичай звертається до диска для копіювання ресурсу вікна діалогу з файлу з розширенням .EXE програми до оперативної пам'яті.

Існують наступні ресурси:

- Значки (icons);
- Курсори (cursors);
- Бітові образи (bitmaps);
- Символьні рядки (characterstrings);
- Ресурси, які визначаються користувачем (userdefinedresources);
- Меню (menus);
- Швидкі комбінації клавіш (keyboardaccelerators);
- Вікна діалогу (dialogboxes) тощо.

На даному практичному занятті будуть розглянуті перші два ресурси.

Піктограма – це графічне зображення невеликого розміру, яке складається з окремих пікселів. Піктограми зазвичай використовуються для позначення згорнутого вікна програми.

Ярлик (Shortcut) — файл зв'язку, який служить покажчиком на об'єкт (наприклад, файл, який потрібно певним чином обробити), програму або команду і містить додаткову інформацію. Найчастіше ярлики створюються

на робочому столі для швидкого (зручного) запуску програм, що знаходяться у «незручних» місцях. Ярлик має містити посилання на файл або каталог, що викликає. Ярлик може містити також посилання на зображення, піктограму (з-за чого між ярликами є відмінність).

У Windows застосовують різні види значків. У програмі можна отримати горизонтальний (X) і вертикальний (Y) розміри значків та курсорів, використовуючи Win API функцію GetSystemMetrics з параметрами SM_CXICON і SM_CYICON (для стандартного значка), SM_CXSMICON і SM_CYSMICON (для маленького значка) і SM_CXCURSOR і SM_CYCURSOR для курсорів миші.

Функція витягує настройки метрики або системи конфігурації зазначеної системи. Зверніть увагу, що всі розміри, які повертаються GetSystemMetrics, у пікселях.

Синтаксис C++:

```
int WINAPI GetSystemMetrics(
_In_ int nIndex
);
```

Параметри:

nIndex [in]

Type: **int**

Система метрик або настройки конфігурації повинні бути вилучені. Цей параметр може приймати одне з наступних значень. Зверніть увагу, що всі значення SM_CX* - ширина, а значення SM_CY* - висота. Також зверніть увагу, що всі налаштування, призначені для повернення булевих даних представляють собою TRUE, як і будь-яке ненульове значення, і FALSE як нульове значення.

Перелік та опис параметрів можна отримати на:

https://docs.microsoft.com/ru-ru/windows/win32/api/winuser/

Для завантаженя такого ресурсу як власний значок, потрібно його створити та зберегти у файлі з розширенням .ico, а потім викликати за допомогою функції LoadResource. Ця функція витягує дескриптор, який може використовуватися, щоб отримати покажчик на перший байт зазначеного ресурсу у пам'яті. Проте, для негайного використання потрібного ресурсу доведеться скористатися допоміжними функціями:

Function	Action	Toremoveresource
FormatMessage	Loadsandformats a message-table entry	Noactionneeded
LoadAccelerators	Loadsanacceleratortable	**DestroyAcceleratorTable**
LoadBitmap	Loads a bitmapresource	**DeleteObject**
LoadCursor	Loads a cursorresource	**DestroyCursor**
LoadIcon	Loadsaniconresource	**DestroyIcon**

LoadMenu	Loads a menuresource	**DestroyMenu**
LoadString	Loads a stringresource	Noactionneeded

Перелік та опис параметрів можна отримати на: https://docs.microsoft.com/ru-ru/windows/win32/devnotes/developer-notes та https://docs.microsoft.com/ru-ru/windows/win32/devnotes/-loadcursor

Хоча для позначення програм Windows використовує значки декількома способами, у безлічі програм, значок задається тільки при визначенні класу вікна. Ви можете в обох інструкціях послатися на один і той самий значок стандартного розміру, і Windows, при виведенні маленького значка на екран просто збільшить його до необхідного розміру.

Якщо потім ви захочете змінити значок програми, це можна зробити за допомогою функції SetClassLong.

Маленький значок можна замінити за допомогою GCL_HICONSM. Якщо ви зберегли описатель значка, повернутий функцією LoadIcon, то ви також можете і намалювати значок у робочій області вашого вікна:

DrawIcon(hdc, x, y, hIcon);

Сама ОС Windows використовує функцію DrawIcon під час виведення значка вашої програми у відповідне місце. Windows отримує описувач значка зі структури класу вікна. Ви можете отримати описувач у той самий спосіб:

DrawIcon(hdc, x, y, GetClassLong(hwnd, GCL_HICON));

Аналогічні дії необхідно провести для створення альтернативного курсору: створити курсор та зберегти у файлі з розширенням .cur.

Якщо ви використовуєте дочірні вікна, можна зробити так, щоб курсор виглядав по-різному у різних вікнах. Якщо у вашій програмі визначаютьсяhttps://docs.microsoft.com/ru-ru/windows/win32/api/_menurc/ класи цих дочірніх вікон, то для кожного класу ви можете використовувати свій курсор шляхом відповідної установки у кожному класі вікна. А якщо ви використовуєте зумовлені дочірні елементи управління, то змінити необхідні поля класу вікна можна за допомогою функції SetClassLong.

Опис функції та значень структури класу можна отримати на: https://docs.microsoft.com/ru-ru/search/?terms=SetClassLong&category=Reference&scope=Desktop

Якщо ви розподілили робочу область вікна вашої програми на маленькі логічні області без використання дочірніх вікон, тоді для зміни курсору миші можна використовувати функцію SetCursor, яка встановлює форму курсору.

Функцію SetCursor слід викликати під час обробки повідомлення WM_MOUSEMOVE. В іншому випадку для перемальовування курсору під час його рухів Windows використовує курсор, раніше заданий у класі вікна.

Перелік та опис параметрів можна отримати на:
https://docs.microsoft.com/ru-ru/windows/win32/api/_menurc/
https://docs.microsoft.com/ru-ru/windows/win32/api/

4.3 Контрольні завдання.

За варіантом створити програму, яка:

Варіант 1

Розташовує свій ярлик на робочому столі (ярлик створити самостійно).

Варіант 2

Змінює колір стандартного курсору програми.

Варіант 3

Відображає стандартний курсор програми з рамкою навколо.

Варіант 4

Змінює стандартний курсор вікна програми на власний (авторський).

Варіант 5

Змінює стандартні розміри ярлика будь-якої програми на робочому столі.

Варіант 6

Видаляє на робочому столі ярлик будь-якої програми.

Варіант 7

Змінює стандартний ярлик будь-якої програми на власний (авторський).

Варіант 8

Під час запуску виводить на екран власний курсор та власний ярлик (авторські).

Варіант 9

У різних вікнах різних додатків відображає різні курсори.

Варіант 10

Примушує курсор перетворюватися на курсор створений самостійно, тоді коли він опиняється за межами робочій поверхні екранної форми програми.

Варіант 11

Примушує курсор перетворюватися на курсор створений самостійно, тоді коли він опиняється на робочій поверхні екранної форми програми.

Варіант 12

Під час згортання програми виводить на екран курсор створений самостійно.

Варіант 13

Змінює розмір ярлика будь-якої програми.

Варіант 14

Змінює розмір курсору на робочому столі.

Варіант 15

Змінює розмір курсору у визначеному місці на робочому столі.

Варіант 16

Виводить ярлик програми у визначене місце на робочому столі.

Варіант 17

Робить курсор невидимим, коли він опиняється на екранній формі.

Варіант 18

Робить курсор невидимим, коли він опиняється за межами екранної форми.

Варіант 19

Робить курсор прозорим, коли він опиняється за межами екранної форми.

Варіант 20

Робить курсор прозорим, коли він опиняється на екранній формі.

Варіант 21

Під час згортання програми виводить на екран ярлик створений самостійно.

Варіант 22

Робить невидимим на робочому столі ярлик будь-якої програми.

Варіант 23

Робить прозорим на робочому столі ярлик будь-якої програми.

Варіант 24

Переміщує на робочому столі ярлик будь-якої програми у визначене місце.