БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Факультет информационных технологий

и робототехники

Кафедра «Программное обеспечение вычислительной техники и автоматизированных систем»

О Т Ч Ё Т

о лабораторной работе № 7

Дисциплина

«Системное программирование»

Тема

«Межпроцессорное взаимодействие. Буфер обмена»

Выполнил: студент гр. 10702217 Храмков Д. С.

Проверил: Разорёнов Н. А.

Минск 2019

***Лабораторная работа № 7***

**МЕЖПРОЦЕССОРНОЕ ВЗАИМОДЕЙСТВИЕ. БУФЕР ОБМЕНА.**

**Цель работы:** изучить основы передачи между процессами информации на базе сообщения WM\_COPYDATA. Изучить основы работы с объектом – буфер обмена.

**Изучаемые вопросы**

1. Структура COPYDATASTRUCT.
2. Передача текстовой информации.
3. Приём текстовой информации.
4. Передача структурированной информации.
5. Приём структурированной информации.
6. Форматы данных.
7. Запись информации в буфер обмена.
8. Чтение информации из буфера обмена.
9. Передача информации пользовательского типа \*.

***Постановка задачи***

1. Создать приложение, состоящее из двух процессов:

* Первый процесс посылает текстовую и структурированную информацию;
* Второй процесс принимает информацию и отображает её в окне процесса.

1. Создать приложение, состоящее из двух процессов:

* Первый процесс записывает текстовый файл и растровый рисунок в буфер обмена;
* Второй процесс считывает информацию из буфера обмена и отображает в окне процесса.

Текстовый файл и файл с растровым рисунком взять из предыдущих лабораторных работ.

1. **Структура COPYDATASTRUCT.**

Структура COPYDATASTRUCT используется при передаче данных другому процессу через сообщение WM\_COPYDATA. COPYDATASTRUCT содержит размер передаваемых данных, размер данных, указанных в lpData и указатель на передаваемые данные.

COPYDATASTRUCT cds1;

cds1.dwData = STRUCT\_TEXT\_ID;

cds1.cbData = sizeof(text1);

cds1.lpData = &text1;

1. **Передача текстовой информации.**

Передача текстовой информации меджу процессами осуществляется с использованием функции SendMessage с сообщением WM\_COPYDATA. При этом, передаваемая текстовая информация пакуется в структуру COPYDATASTRUCT, указатель на которую передаётся четвёртым параметром в SendMessage.

SendMessage(wnd, WM\_COPYDATA, (WPARAM)(HWND)hWnd, (LPARAM)(LPVOID)&cds1);

1. **Приём текстовой информации.**

Программа, ожидающая сообщения, находится в режиме мониторинга, проверяя, не было ли передано сообщение WM\_COPYDATA. Если использование этого сообщения было замечено, то из принятой структуры COPYDATASTRUCT достаются данные.

SetDlgItemTextA(hWnd, EDIT\_TEXT, (char\*)pcds->lpData);

1. **Передача структурированной информации.**

Процесс передачи структурированной информации отличается от процесса передачи текстовой информации только тем, что в качестве третьего параметра структуры COPYDATASTRUCT задаётся указатель на другую структуру.

struct userStruct {

int code1;

int code2;

} info;

info.code1 = 3223;

info.code2 = 782;

COPYDATASTRUCT cds2;

cds2.dwData = STRUCT\_STRUCT\_ID;

cds2.cbData = sizeof(info);

cds2.lpData = &info;

SendMessage(wnd, WM\_COPYDATA, (WPARAM)(HWND)hWnd, (LPARAM)(LPVOID)&cds2);

1. **Приём структурированной информации.**

Как и в случае с передачей структурированной информации, отличие в приёме информации также незначительно. После активации сообщения WM\_COPYDATA из указателя на структуру COPYDATASTRUCT забирается параметр pwData, соответствующий MYSTRUCT. Далее из передаваемой структуры забирается структурированная информация.

userStruct \*info = (userStruct\*)pcds->lpData;

int code1 = info->code1;

int code2 = info->code2;

1. **Форматы данных.**

Форматы данных определяют способ хранения и интерпретации этих данных. Одними из наиболее используемых являются CF\_BITMAP, CF\_DPSTEXT, CF\_TEXT, CF\_WAVE и CF\_UNICODETEXT.

1. **Запись информации в буфер обмена.**

Чтобы записать информацию в буфер обмена нужно сначала открыть его функцией OpenClipboard, затем очистить его функцией EmptyClipboard, и наконец выделить память, необходимую для хранения выбранной информации, функцией GlobalAlloc(). После этого, скопируем нужную нам информацию, запишем её в буфер функцией SetClipboardData и закроем буфер функцией CloseClipboard().

char buffer[] = "buffer";

HGLOBAL hGl = GlobalAlloc(GMEM\_DDESHARE, sizeof(buffer));

memcpy(hGl, buffer, sizeof(buffer));

SetClipboardData(CF\_TEXT, hGl);

CloseClipboard();

1. **Чтение информации из буфера обмена.**

Последовательность действий при чтении информации из буфера обмена: открываем буфер (OpenClipboard), проверяем тип данных, находящийся в нём (IsClipboardFormatAvailable) и извлекаем данные (GetClipboardData). По завершению работы с буфером закрываем его (CloseClipboard).

if (OpenClipboard(hWnd)) {

HANDLE hData = GetClipboardData(CF\_TEXT);

SetDlgItemTextA(hWnd, EDIT\_BUFFER\_TEXT, (char\*)hData);

CloseClipboard();

}

1. **Передача информации пользовательского типа \*.**

Передача пользовательского типа данных осуществляется также, как и передача обычных данных. Создание своего типа данных происходит через RegisterClipboardFormat, в который передаётся строковое название типа данных. Теперь, для принятии данных пользовательского типа другим приложением, нужно в нём задать этот тип данных тем же способом.

if (IsClipboardFormatAvailable(jora)) {

HANDLE hData = GetClipboardData(jora);

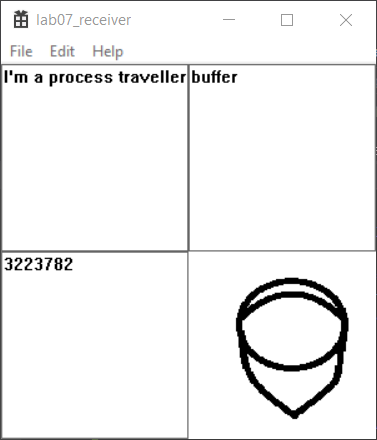
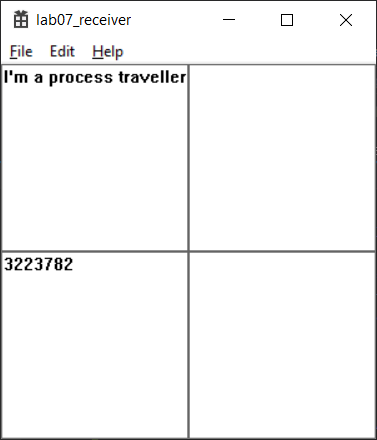
string msg = to\_string((DWORD)hData);

SetDlgItemTextA(hWnd, EDIT\_BUFFER\_TEXT, msg.c\_str());

CloseClipboard();

}

**Результаты выполнения:**



**Вывод:**

* 1. Изучены передача и приём текстовой информации между процессами.
  2. Изучены передача и приём структурной информации между процессами.
  3. Изучены передача и приём текстовой информации через буфер обмена.
  4. Изучены передача и приём графической информации через буфер обмена.
  5. Изучена возможность создания пользовательского типа данных в буфере обмена.