



C++. Базовый уровень

Создание оконных приложений



**Минцифры
России**



**ЦИФРОВАЯ
ЭКОНОМИКА**

20.35
УНИВЕРСИТЕТ

Создание приложений

Прежде чем начать, убедитесь, что у вас установлена среда разработки Visual Studio с поддержкой MFC. Если у вас еще нет Visual Studio, вы можете скачать ее с официального сайта Microsoft.

1. Откройте Visual Studio и выберите "Создать проект".
2. В окне "Создание проекта" выберите "MFC Application" и нажмите "Далее".
3. Введите имя проекта и выберите место сохранения. Нажмите "Далее".
4. В окне "Выбор типа приложения" выберите "На основе диалоговых окон" и нажмите "Далее".
5. В окне "Выбор опций MFC" убедитесь, что выбрана опция "Использовать MFC в виде разделяемой DLL" и нажмите "Далее".
6. В окне "Функции пользовательского интерфейса" укажите заголовок диалогового окна и нажмите "Далее".
7. Нажмите "Готово" для

Заголовок диалогового окна

MFC APP EXAMPLE



Создание приложений

Создайте диалоговое окно с элементом управления CStatic, который показывает текущее время в формате часы:минуты:секунды.

При нажатии на кнопку **Update** обновите метку с текущим временем.

При нажатии на кнопку **Stop** остановите обновление метки и выведите сообщение “Обновление времени остановлено”.

При нажатии на кнопку **Start** возобновите обновление метки и выведите сообщение “Обновление времени возобновлено”.

Создание приложений

```
// Добавьте в заголовочный файл диалогового окна следующие объявления:  
// Идентификаторы элементов управления  
#define IDC_STATIC 1001  
#define IDC_UPDATE_BUTTON 1002  
#define IDC_STOP_BUTTON 1003  
#define IDC_START_BUTTON 1004  
// Переменная для хранения идентификатора таймера  
UINT_PTR m_Timer;  
// Обработчики нажатия на кнопки Update, Stop и Start, а также таймера  
afx_msg void OnUpdateButton();  
afx_msg void OnStopButton();  
afx_msg void OnStartButton();  
afx_msg void OnTimer(UINT_PTR nIDEvent);  
  
// Добавьте в карту сообщений диалогового окна следующие записи:  
ON_BN_CLICKED(IDC_UPDATE_BUTTON, &CMyDialog::OnUpdateButton)  
ON_BN_CLICKED(IDC_STOP_BUTTON, &CMyDialog::OnStopButton)  
ON_BN_CLICKED(IDC_START_BUTTON, &CMyDialog::OnStartButton)  
ON_WM_TIMER()
```

Создание приложений

// Добавьте в исходный файл диалогового окна следующие реализации:

// Обработчик инициализации диалогового окна

```
BOOL CMyDialog::OnInitDialog(){
    CDialog::OnInitDialog();

    // Инициализируем идентификатор таймера нулем
    m_Timer = 0;
    // Обновляем метку с текущим временем
    OnUpdateButton();

    return TRUE;
}
```

// Обработчик нажатия на кнопку Update

```
void CMyDialog::OnUpdateButton(){
    // Получаем текущее время
    CTime time = CTime::GetCurrentTime();
    // Форматируем время в виде часы:минуты:секунды
    CString text = time.Format(_T("%H:%M:%S"));
    // Устанавливаем текст метки
    SetDlgItemText(IDC_STATIC, text);
}
```

Создание приложений

```
// Обработчик нажатия на кнопку Stop
void CMyDialog::OnStopButton(){
    // Если таймер запущен, то останавливаем его
    if (m_Timer != 0){
        // Удаляем таймер
        KillTimer(m_Timer);
        // Обнуляем идентификатор таймера
        m_Timer = 0;
        // Выводим сообщение о том, что обновление времени остановлено
        AfxMessageBox(_T("Обновление времени остановлено"));
    }
}

// Обработчик нажатия на кнопку Start
void CMyDialog::OnStartButton(){
    // Если таймер не запущен, то запускаем его
    if (m_Timer == 0){
        // Создаем таймер с интервалом в 1000 миллисекунд (1 секунда)
        m_Timer = SetTimer(1, 1000, NULL);
        // Выводим сообщение о том, что обновление времени возобновлено
        AfxMessageBox(_T("Обновление времени возобновлено"));
    }
}
```

Создание приложений

```
// Обработчик таймера
void CMyDialog::OnTimer(UINT_PTR nIDEvent){
    // Проверяем, что таймер соответствует нашему идентификатору
    if (nIDEvent == m_Timer){
        // Обновляем метку с текущим временем
        OnUpdateButton();
    }
    // Вызываем базовый обработчик
    CDialog::OnTimer(nIDEvent);
}
```

Создание приложений

Создайте диалоговое окно с элементом управления CScrollBar, который позволяет прокручивать изображение, загруженное из файла.

При нажатии на кнопку **Load** откройте диалог выбора файла и загрузите выбранное изображение в память.

При нажатии на кнопку **Show** отобразите изображение в диалоговом окне, подстраивая размеры скролл-баров в зависимости от размеров изображения.

При перемещении ползунков скролл-баров прокручивайте изображение в соответствующем направлении.

Создание приложений

```
// Добавьте в заголовочный файл диалогового окна следующие объявления:  
// Идентификаторы элементов управления  
#define IDC_SCROLL 1001  
#define IDC_LOAD_BUTTON 1002  
#define IDC_SHOW_BUTTON 1003  
// Переменная для хранения указателя на скролл-бар  
CScrollBar* m_pScroll;  
// Переменная для хранения загруженного изображения  
CImage m_Image;  
// Обработчики нажатия на кнопки Load и Show, а также перемещения ползунка  
скролл-бара  
afx_msg void OnLoadButton();  
afx_msg void OnShowButton();  
afx_msg void OnVScroll(UINT nSBCode, UINT nPos, CScrollBar* pScrollBar);  
  
// Добавьте в карту сообщений диалогового окна следующие записи:  
ON_BN_CLICKED(IDC_LOAD_BUTTON, &CMyDialog::OnLoadButton)  
ON_BN_CLICKED(IDC_SHOW_BUTTON, &CMyDialog::OnShowButton)  
ON_WM_VSCROLL()
```

Создание приложений

```
// Добавьте в исходный файл диалогового окна следующие реализации:  
// Обработчик инициализации диалогового окна  
BOOL CMyDialog::OnInitDialog(){  
    CDialog::OnInitDialog();  
  
    // Получаем указатель на скролл-бар  
    m_pScroll = (CScrollBar*)GetDlgItem(IDC_SCROLL);  
    // Скрываем скролл-бар до тех пор, пока не будет загружено изображение  
    m_pScroll->ShowWindow(SW_HIDE);  
  
    return TRUE;  
}
```

Создание приложений

```
// Обработчик нажатия на кнопку Load
void CMyDialog::OnLoadButton(){
    // Создаем диалог выбора файла
    CFileDialog dlg(TRUE, NULL, NULL, OFN_FILEMUSTEXIST, _T("Image Files
(*.bmp;*.jpg;*.png)|*.bmp;*.jpg;*.png|All Files (*.*)|*.*||"));
    // Если пользователь выбрал файл, то загружаем его в память
    if (dlg.DoModal() == IDOK){
        // Получаем путь к выбранному файлу
        CString path = dlg.GetPathName();
        // Освобождаем память от предыдущего изображения, если оно было загружено
        m_Image.Destroy();
        // Загружаем новое изображение из файла
        HRESULT hr = m_Image.Load(path);
        // Если загрузка прошла успешно, то выводим сообщение об этом
        if (SUCCEEDED(hr)){
            AfxMessageBox(_T("Изображение загружено"));
        }
        // Иначе выводим сообщение об ошибке
        else{
            AfxMessageBox(_T("Не удалось загрузить изображение"));
        }
    }
}
```

Создание приложений

```
// Обработчик нажатия на кнопку Show
void CMyDialog::OnShowButton(){
    // Проверяем, что изображение было загружено
    if (!m_Image.IsNull()){
        // Получаем размеры изображения
        int width = m_Image.GetWidth();
        int height = m_Image.GetHeight();
        CRect rect; // Получаем размеры клиентской области диалогового окна
        GetClientRect(&rect);
        // Если высота изображения больше высоты диалогового окна, то показываем скролл-бар
        if (height > rect.Height()){
            // Устанавливаем диапазон значений скролл-бара от 0
            // до разницы между высотами изображения и диалогового окна
            m_pScroll->SetScrollRange(0, height - rect.Height());
            // Устанавливаем начальное положение ползунка скролл-бара в 0
            m_pScroll->SetScrollPos(0);
            m_pScroll->ShowWindow(SW_SHOW); // Показываем скролл-бар
        }
        else{// Иначе скрываем скролл-бар
            m_pScroll->ShowWindow(SW_HIDE);
        }
        // Отображаем изображение в диалоговом окне
        Invalidate();
    }
}
```

Создание приложений

```
// Обработчик перемещения ползунка скролл-бара
void CMyDialog::OnVScroll(UINT nSBCode, UINT nPos, CScrollBar* pScrollBar){
    // Проверяем, что элемент управления - это скролл-бар
    if (pScrollBar->GetDlgCtrlID() == IDC_SCROLL){
        int pos = m_pScroll->GetScrollPos();// Получаем текущее положение ползунка скролл-бара
        // В зависимости от кода перемещения изменяем положение ползунка
        switch (nSBCode){
            case SB_LINEUP: // Перемещение на одну линию вверх
                pos -= 10;break;
            case SB_LINEDOWN: // Перемещение на одну линию вниз
                pos += 10;break;
            case SB_PAGEUP: // Перемещение на одну страницу вверх
                pos -= 50;break;
            case SB_PAGEDOWN: // Перемещение на одну страницу вниз
                pos += 50;break;
            case SB_THUMBTRACK: // Перемещение ползунка
                pos = nPos;break;
            default: break;
        }
        m_pScroll->SetScrollPos(pos);// Устанавливаем новое положение ползунка скролл-бара
        Invalidate(); // Перерисовываем изображение с учетом прокрутки
    }
    CDialog::OnVScroll(nSBCode, nPos, pScrollBar);// Вызываем базовый обработчик
}
```

Создание приложений

```
// Обработчик рисования диалогового окна
void CMyDialog::OnPaint(){
    // Вызываем базовый обработчик
    CDialog::OnPaint();
    // Проверяем, что изображение было загружено
    if (!m_Image.IsNull()){
        // Получаем контекст устройства для рисования
        CClientDC dc(this);
        // Получаем текущее положение ползунка скролл-бара
        int pos = m_pScroll->GetScrollPos();
        // Рисуем изображение в диалоговом окне с учетом прокрутки
        m_Image.Draw(dc, 0, -pos);
    }
}
```

Создание приложений

Создайте диалоговое окно с элементом управления CListBox, который показывает список городов.

При нажатии на кнопку **Add** добавьте в список город, введенный в поле ввода.

При нажатии на кнопку **Delete** удалите из списка город, выбранный в списке.

При нажатии на кнопку **Clear** очистите список от всех городов.

Создание приложений

```
// Добавьте в заголовочный файл диалогового окна следующие объявления:  
// Идентификаторы элементов управления  
#define IDC_LIST 1001  
#define IDC_EDIT 1002  
#define IDC_ADD_BUTTON 1003  
#define IDC_DELETE_BUTTON 1004  
#define IDC_CLEAR_BUTTON 1005  
// Переменная для хранения указателя на список  
CListBox* m_pList;  
// Обработчики нажатия на кнопки Add, Delete и Clear  
afx_msg void OnAddButton();  
afx_msg void OnDeleteButton();  
afx_msg void OnClearButton();  
  
// Добавьте в карту сообщений диалогового окна следующие записи:  
ON_BN_CLICKED(IDC_ADD_BUTTON, &CMyDialog::OnAddButton)  
ON_BN_CLICKED(IDC_DELETE_BUTTON, &CMyDialog::OnDeleteButton)  
ON_BN_CLICKED(IDC_CLEAR_BUTTON, &CMyDialog::OnClearButton)
```


Создание приложений

```
// Добавьте в исходный файл диалогового окна следующие реализации:
// Обработчик инициализации диалогового окна
BOOL CMyDialog::OnInitDialog(){
    CDialog::OnInitDialog();

    // Получаем указатель на список
    m_pList = (CListBox*)GetDlgItem(IDC_LIST);

    return TRUE;
}

// Обработчик нажатия на кнопку Add
void CMyDialog::OnAddButton(){
    // Получаем текст из поля ввода
    CString text;
    GetDlgItemText(IDC_EDIT, text);
    // Добавляем текст в список
    m_pList->AddString(text);
}
```

Создание приложений

```
// Обработчик нажатия на кнопку Delete
void CMyDialog::OnDeleteButton(){
    // Получаем индекс выбранного элемента в списке
    int index = m_pList->GetCurSel();
    // Если индекс не равен -1, то удаляем элемент из списка
    if (index != -1){
        m_pList->DeleteString(index);
    }
}

// Обработчик нажатия на кнопку Clear
void CMyDialog::OnClearButton(){
    // Очищаем список от всех элементов
    m_pList->ResetContent();
}
```