ZGUI 드래그 시스템 아키텍처

개요

ZGUI는 다이얼로그를 마우스로 드래그하여 이동할 수 있는 기능을 제공합니다. 이 문서는 드래그 기능의 전체 아키텍처와 작동 흐름을 설명합니다.

---

1. 시스템 구성 요소

1.1 주요 클래스

- **BasicRenderState**: 게임 상태 클래스, 마우스 입력을 받아 ZGUIManager로 전달
- **ZGUIManager**: GUI 시스템 관리자, 모든 다이얼로그를 관리하고 메시지를 분배
- **ZGUIDialog**: 개별 다이얼로그 클래스, 드래그 로직을 실제로 구현

1.2 관련 멤버 변수

ZGUIDialog.h

```
class ZGUIDialog
{
    // 다이얼로그 위치 및 크기
    int m_iX, m_iY;
    int m_iWidth, m_iHeight;

    // 드래그 관련
    BOOL m_bDrag;           // 현재 드래그 중인가?
    POINT m_ptMouseLast;    // 마지막 마우스 위치 (드래그용)
    BOOL m_bDragable;       // 이 다이얼로그를 드래그할 수 있는가?

    // 포커스 관리
    static ZGUIDialog* s_pDialogFocus;  // 현재 포커스를 가진 다이얼로그
};
```

---

2. 드래그 작동 흐름

2.1 전체 메시지 흐름

```
Windows 메시지 (WM_LBUTTONDOWN, WM_MOUSEMOVE, WM_LBUTTONUP)
    ↓
WinMain.cpp (WndProc)
    ↓
BasicRenderState::OnMouseDown/OnMouseMove/OnMouseUp
    ↓
ZGUIManager::MsgProc(HWND, UINT, WPARAM, LPARAM)
    ↓
ZGUIDialog::MsgProc(HWND, UINT, WPARAM, LPARAM)
    ↓
드래그 처리 (WM_LBUTTONDOWN, WM_MOUSEMOVE, WM_LBUTTONUP)
```

---

3. 드래그 시작 (WM_LBUTTONDOWN)

3.1 BasicRenderState에서 메시지 변환

BasicRenderState.cpp

```
void BasicRenderState::OnMouseDown(int x, int y, int button)
{
    if (m_pGUIManager && m_pGUIManager->IsInit())
    {
        // Windows 메시지로 변환
        UINT uMsg = (button == 0) ? WM_LBUTTONDOWN : WM_RBUTTONDOWN;
        LPARAM lParam = MAKELPARAM(x, y);
        WPARAM wParam = (button == 0) ? MK_LBUTTON : MK_RBUTTON;

        HWND hWnd = _pGraphicsRef->GetHWND();
        m_pGUIManager->MsgProc(hWnd, uMsg, wParam, lParam);
    }
}
```

3.2 ZGUIManager에서 다이얼로그로 분배

ZGUIManager.cpp

```
BOOL ZGUIManager::MsgProc(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam)
{
    if (!m_bInit) return FALSE;

    int iCount = m_pResource->GetDialogCount();

    // 최상단 다이얼로그부터 역순으로 처리 (화면상 위에 있는 것 먼저)
    for (int i = (iCount-1); i >= 0; i--)
    {
        ZGUIDialog* pDialog = m_pResource->GetDialog(i);
        if (pDialog && pDialog->GetVisible())
            pDialog->MsgProc(hWnd, uMsg, wParam, lParam);
    }

    return TRUE;
}
```

3.3 ZGUIDialog에서 드래그 시작 처리

ZGUIDialog.cpp (MsgProc 내부)

```
case WM_LBUTTONDOWN:
    if (!s_pDialogFocus)
        break;

    if (m_bDragable && !m_bDrag && s_pDialogFocus == this)
    {
        POINT mousePoint = { short(LOWORD(lParam)), short(HIWORD(lParam)) };

        // 다이얼로그 범위 체크
        if (mousePoint.x >= m_iX && mousePoint.x < m_iX + m_iWidth &&
            mousePoint.y >= m_iY && mousePoint.y < m_iY + m_iHeight)
        {
            m_bDrag = TRUE;                 // 드래그 상태 활성화
            SetCapture(GetHWND());          // 마우스 캡처 (화면 밖에서도 추적)
            m_ptMouseLast.x = mousePoint.x; // 마지막 마우스 위치 저장
            m_ptMouseLast.y = mousePoint.y;

            return TRUE;
        }
    }
    break;
```

조건:

1. `m_bDragable == TRUE`: 이 다이얼로그가 드래그 가능해야 함
2. `m_bDrag == FALSE`: 아직 드래그 중이 아니어야 함
3. `s_pDialogFocus == this`: 이 다이얼로그가 포커스를 가져야 함

4. 마우스가 다이얼로그 영역 내에 있어야 함

---

4. 드래그 중 (WM_MOUSEMOVE)

4.1 마우스 이동 처리

BasicRenderState.cpp

```
void BasicRenderState::OnMouseMove(int x, int y)
{
    _curX = x;
    _curY = y;

    if (m_pGUIManager && m_pGUIManager->IsInit())
    {
        LPARAM lParam = MAKELPARAM(x, y);
        WPARAM wParam = 0;

        HWND hWnd = _pGraphicsRef->GetHWND();
        m_pGUIManager->MsgProc(hWnd, WM_MOUSEMOVE, wParam, lParam);
    }
}
```

4.2 ZGUIDialog에서 위치 업데이트

ZGUIDialog.cpp (MsgProc 내부)

```
case WM_MOUSEMOVE:
    if (!s_pDialogFocus)
        break;

    if (m_bDragable && m_bDrag && s_pDialogFocus == this)
    {
        RECT rcClient;
        POINT mousePoint = { short(LOWORD(lParam)), short(HIWORD(lParam)) };
        POINT ptOffSet;

        GetClientRect(m_pResourceRef->GetHWND(), &rcClient);

        // 이동 거리 계산
        ptOffSet.x = mousePoint.x - m_ptMouseLast.x;
        ptOffSet.y = mousePoint.y - m_ptMouseLast.y;

        // 다이얼로그 위치 업데이트
        m_iX += ptOffSet.x;
        m_iY += ptOffSet.y;

        // 화면 경계 제한
        if (m_iX < 0)
            m_iX = 0;
        else if (m_iX >= ((rcClient.right - rcClient.left) - m_iWidth))
            m_iX = (rcClient.right - rcClient.left) - m_iWidth;

        if (m_iY < 0)
            m_iY = 0;
        else if (m_iY >= (rcClient.bottom - rcClient.top) - m_iHeight)
            m_iY = (rcClient.bottom - rcClient.top) - m_iHeight;

        // 마지막 마우스 위치 업데이트
        m_ptMouseLast.x = mousePoint.x;
        m_ptMouseLast.y = mousePoint.y;

        return TRUE;
    }
    break;
```

동작:

1. 현재 마우스 위치와 마지막 위치의 차이(offset) 계산
2. 다이얼로그 위치(`m_iX`, `m_iY`)를 offset만큼 이동
3. 화면 경계를 벗어나지 않도록 제한
4. 마지막 마우스 위치를 현재 위치로 업데이트

---

5. 드래그 종료 (WM_LBUTTONUP)

5.1 BasicRenderState에서 메시지 전달

BasicRenderState.cpp

```
void BasicRenderState::OnMouseUp(int x, int y, int button)
{
    if (m_pGUIManager && m_pGUIManager->IsInit())
    {
        UINT uMsg = (button == 0) ? WM_LBUTTONUP : WM_RBUTTONUP;
        LPARAM lParam = MAKELPARAM(x, y);
        WPARAM wParam = 0;  // UP 메시지에서는 버튼 플래그 제거

        HWND hWnd = _pGraphicsRef->GetHWND();
        m_pGUIManager->MsgProc(hWnd, uMsg, wParam, lParam);
    }
}
```

5.2 ZGUIDialog에서 드래그 종료

ZGUIDialog.cpp (MsgProc 내부)

```
case WM_LBUTTONUP:
    if (m_bDragable && m_bDrag)
    {
        ReleaseCapture();   // 마우스 캡처 해제
        m_bDrag = FALSE;    // 드래그 상태 비활성화

        return TRUE;
    }
    break;
```

5.3 캡처 손실 처리

ZGUIDialog.cpp (MsgProc 내부)

```
case WM_CAPTURECHANGED:
    // 애플리케이션이 마우스 캡처를 잃었을 때
    // WM_MOUSEUP을 받지 못했을 수 있으므로 드래그 상태를 강제로 해제
    if ((HWND)lParam != hWnd)
        m_bDrag = FALSE;
    break;
```

---

6. 초기화 및 설정

6.1 ZGUIDialog 생성자

ZGUIDialog.cpp

```
ZGUIDialog::ZGUIDialog(ZGraphics& gfx)
{
```

```
    m_iX = 0;
    m_iY = 0;
    m_iWidth = 0;
    m_iHeight = 0;

    m_bVisible = TRUE;
    m_bDrag = FALSE;        // 드래그 중이 아님
    m_bDragable = TRUE;     // 드래그 가능 (기본값)
    m_ptMouseLast.x = 0;
    m_ptMouseLast.y = 0;

    // ... 기타 초기화
}
```

6.2 리소스 파일에서 드래그 설정 로드

DialogRes.ift

```
[0]
DialogName: MainDialog
X: 100
Y: 100
Width: 400
Height: 300
Keyboard: 1
Mouse: 1
Dragable: 1      # 1 = 드래그 가능, 0 = 드래그 불가능
Visible: 1
```

ZGUIManager.cpp (Init 메서드)

```
pDialog->SetDragable(std::stoi(m_pDialogRes->GetValue(iDialogIndex, "Dragable")));
```

---

7. 주요 API

7.1 공개 메서드

ZGUIDialog.h

```
// 드래그 가능 여부 설정
BOOL GetDragable() { return m_bDragable; }
void SetDragable(BOOL bDragable) { m_bDragable = bDragable; }

// 위치 설정 (프로그래밍 방식)
void GetLocation(POINT &Pt) const { Pt.x = m_iX; Pt.y = m_iY; }
void SetLocation(int x, int y) { m_iX = x; m_iY = y; }

// 크기 설정
void SetSize(int width, int height) { m_iWidth = width; m_iHeight = height; }
int GetWidth() { return m_iWidth; }
int GetHeight() { return m_iHeight; }
```

7.2 사용 예제

```
// 드래그 가능 다이얼로그 생성
ZGUIDialog* pDialog = new ZGUIDialog(gfx);
pDialog->Init(0, "MyDialog", pResource);
pDialog->SetLocation(100, 100);
pDialog->SetSize(400, 300);
pDialog->SetDragable(TRUE);  // 드래그 활성화

// 런타임에 드래그 비활성화
pDialog->SetDragable(FALSE);
```

```
// 현재 위치 얻기
POINT pos;
pDialog->GetLocation(pos);
std::cout << "Dialog at (" << pos.x << ", " << pos.y << ")" << std::endl;
```

---

8. 디버깅

8.1 콘솔 출력

드래그 중 콘솔에 다음과 같은 디버그 메시지가 출력됩니다:

```
[Dialog] LastXY(150,200)
[Dialog] [MOVE] Able(1) Drag(1)
[Dialog] Focus: 0x12345678 This: 0x12345678
[Dialog] OffsetXY(5,3)
[Dialog] NewLastXY(155,203)
[Dialog] [UP] Able(1) Drag(0)
```

8.2 문제 해결

드래그가 작동하지 않을 때:

1. **m_bDragable 확인**

   ```
   pDialog->SetDragable(TRUE);
   ```

2. **포커스 확인**
   - 다이얼로그를 클릭하여 `s_pDialogFocus`가 설정되었는지 확인

   ```
   std::cout << "Focus: " << s_pDialogFocus << " This: " << this << std::endl;
   ```

3. **마우스 입력 활성화 확인**

   ```
   pDialog->EnableMouseInput(TRUE);
   ```

4. **메시지 전달 확인**
   - BasicRenderState::OnMouseDown/Move/Up이 호출되는지 확인
   - ZGUIManager::MsgProc이 호출되는지 확인

---

9. 성능 및 최적화

9.1 마우스 캡처

- `SetCapture(hWnd)`: 마우스가 윈도우 밖으로 나가도 메시지를 받을 수 있음
- 드래그 중에만 캡처를 활성화하여 성능 영향 최소화

9.2 경계 체크

- 매 프레임마다 화면 경계를 체크하여 다이얼로그가 화면 밖으로 나가지 않도록 함
- 클라이언트 영역(`GetClientRect`)을 기준으로 제한

9.3 메시지 처리 순서

- 최상단 다이얼로그부터 역순으로 처리하여 올바른 클릭 대상 선택

---

10. 확장 가능성

10.1 추가 가능한 기능

1. **스냅 기능**: 다이얼로그를 화면 가장자리에 붙이기
2. **제한 영역**: 특정 영역 내에서만 드래그 허용
3. **드래그 핸들**: 타이틀바 영역에서만 드래그 가능
4. **애니메이션**: 드래그 시작/종료 시 부드러운 애니메이션
5. **도킹**: 다른 다이얼로그에 붙이기

10.2 확장 예제

```
// 드래그 핸들 기능 추가
class ZGUIDialog
{
    RECT m_rcDragHandle;  // 타이틀바 영역

    BOOL IsInDragHandle(POINT pt)
    {
        return PtInRect(&m_rcDragHandle, pt);
    }
};

// WM_LBUTTONDOWN에서
if (m_bDragable && !m_bDrag && IsInDragHandle(mousePoint))
{
    // 타이틀바를 클릭했을 때만 드래그 시작
    m_bDrag = TRUE;
    // ...
}
```

---

11. 요약

ZGUI 드래그 시스템은 다음과 같은 특징을 가집니다:

✅ **계층적 메시지 처리**: BasicRenderState → ZGUIManager → ZGUIDialog
✅ **마우스 캡처**: 드래그 중 화면 밖에서도 추적 가능
✅ **경계 제한**: 다이얼로그가 화면 밖으로 나가지 않음
✅ **포커스 관리**: 클릭한 다이얼로그가 자동으로 포커스를 받음
✅ **설정 가능**: 리소스 파일이나 코드로 드래그 가능 여부 설정

작성일: 2025-11-01
버전: 1.0
작성자: Cascade AI