23. 카메라 스크롤

Camera.h

```h
#pragma once

#include "../Game.h"

class CCamera
{
private:
	POSITION		m_tPos;
	RESOLUTION		m_tClientRS;
	RESOLUTION		m_tWorldRS;
	POSITION		m_tPivot;
	class CObject* m_pTarget;

public:
	POSITION GetPos() const
	{
		return m_tPos;
	}

public:
	void SetTarget( class CObject* pObj );

	void SetPivot( const POSITION& tPivot )
	{
		m_tPivot = tPivot;
	}
	void SetPivot( float x, float y )
	{
		m_tPivot = POSITION( x, y );
	}

	void SetPos( const POSITION& tPos )
	{
		m_tPos = tPos;
	}
	void SetPos( float x, float y )
	{
		m_tPos = POSITION( x, y );
	}
	void SetClientResolution( const RESOLUTION& tRs )
	{
		m_tClientRS = tRs;
	}
	void SetClientResolution( int x, int y )
	{
		m_tClientRS = RESOLUTION( x, y );
	}
	void SetWorldResolution( const RESOLUTION& tRs )
	{
		m_tWorldRS = tRs;
	}
	void SetWorldResolution( int x, int y )
	{
		m_tWorldRS = RESOLUTION( x, y );
	}

public:
	bool Init( const POSITION& tPos, const RESOLUTION& tRS, const RESOLUTION& tWorldRS );

	void Input( float DeltaTime );

	void Update( float DeltaTime );

	DECLARE_SINGLE( CCamera )
};
```

Camera.cpp

```cpp
#include "Camera.h"
#include "../Object/Object.h"

DEFINITION_SINGLE( CCamera )


CCamera::CCamera()
	:
	m_pTarget( nullptr )
{
}

CCamera::~CCamera()
{
	SAFE_RELEASE( m_pTarget );
}

void CCamera::SetTarget( CObject* pObj )
{
	SAFE_RELEASE( m_pTarget );
	m_pTarget = pObj;

	if( m_pTarget )
	{
		m_pTarget->AddRef();
	}
}

bool CCamera::Init( const POSITION& tPos,
	const RESOLUTION& tRS, const RESOLUTION& tWorldRS )
{
	m_tPos = tPos;
	m_tClientRS = tRS;
	m_tWorldRS = tWorldRS;
	m_tPivot = POSITION( 0.5f, 0.5f );

	return true;
}

void CCamera::Input( float DeltaTime )
{
	if( !m_pTarget )
	{

	}
}
```

타겟이 설정되지 않았을 때
프리 카메라로 입력을 받을 수 있도록 함.

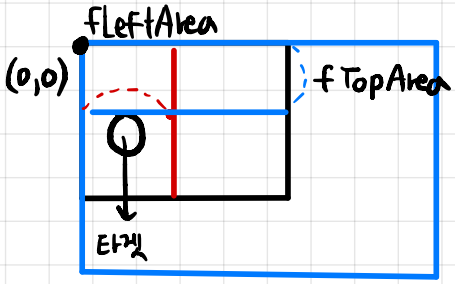
Core.cpp

```cpp
bool CCore::Init( HINSTANCE hInst )
{
    m_hInst = hInst;

    // 윈도우 클래스 등록
    MyRegisterClass();

    // 해상도 설정
    m_tRs.iW = 1280;
    m_tRs.iH = 720;

    // 윈도우 창 생성
    Create();

    // 화면 DC 만들어주기
    m_hDC = GetDC( m_hWnd );

    // 타이머 초기화
    if( !GET_SINGLE( CTimer )->Init() )
    {
        return false;
    }

    // 경로 관리자 초기화
    if( !GET_SINGLE( CPathManager )->Init() )
    {
        return false;
    }

    // 리소스 관리자 초기화
    if( !GET_SINGLE( CResourcesManager )->Init( m_hInst, m_hDC ) )
    {
        return false;
    }

    // 카메라 관리자 초기화
    if( !GET_SINGLE( CCamera )->Init( POSITION( 0.f, 0.f ),
        m_tRs, RESOLUTION( 1500, 1200 ) ) )
    {
        return false;
    }
```

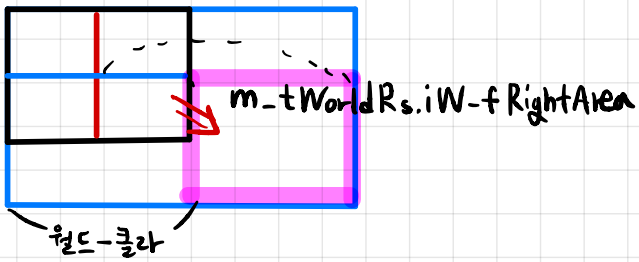
클라 영역 사이즈

월드 사이즈

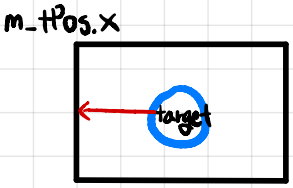
```cpp
void CCamera::Update( float DeltaTime )
{
	if( m_pTarget )
	{
		POSITION tPos = m_pTarget->GetPos();
		POSITION tPivot = m_pTarget->GetPivot();
		_SIZE	tSize = m_pTarget->GetSize();

		float fL = tPos.x - tPivot.x * tSize.x;
		float fT = tPos.y - tPivot.y * tSize.y;
		float fR = fL + tSize.x;
		float fB = fT + tSize.y;

		float fLeftArea = m_tClientRS.iW * m_tPivot.x;
		float fRightArea = m_tClientRS.iW - fLeftArea;
		float fTopArea = m_tClientRS.iH * m_tPivot.y;
		float fBottomArea = m_tClientRS.iH - fTopArea;

		// X
		if( tPos.x <= fLeftArea )
		{
			m_tPos.x = 0.f;
		}
		else if( tPos.x >= m_tWorldRS.iW - fRightArea )
		{
			m_tPos.x = m_tWorldRS.iW - m_tClientRS.iW;
		}
		else
		{
			m_tPos.x = tPos.x - m_tClientRS.iW * m_tPivot.x;
		}
		// Y
		if( tPos.y <= fTopArea )
		{
			m_tPos.y = 0.f;
		}
		else if( tPos.y >= m_tWorldRS.iH - fBottomArea )
		{
			m_tPos.y = m_tWorldRS.iH - m_tClientRS.iH;
		}
		else
		{
			m_tPos.y = tPos.y - m_tClientRS.iH * m_tPivot.y;
		}
	}
}
```



타겟의 위치가 fLeftArea 혹은 fTopArea에 들어오면 x 혹은 y를 0으로 설정



타겟의 위치가 Right or Bottom 영역인 경우



그 외에 경우 타겟 위치에서 카메라에 피봇을 적용한 위치를 빼서 타겟 중심으로 카메라가 이동되게 함.

Stage.cpp

```cpp
void CStage::Render( HDC hDC, float fDeltaTime )
{
	//CStaticObject::Render( hDC, fDeltaTime );

	if( m_pTexture )
	{
		POSITION tCamPos = GET_SINGLE( CCamera )->GetPos();

		BitBlt(
			hDC, m_tPos.x - m_tSize.x * m_tPivot.x,
			m_tPos.y - m_tSize.x * m_tPivot.y,
			GETRESOLUTION.iW, GETRESOLUTION.iH, m_pTexture->GetDC(),
			tCamPos.x, tCamPos.y, SRCCOPY );
	}
}
```

가져올 이미지의 시작점으로 카메라 위치를 넘김.

Object.cpp

```cpp
ject::Render( HDC hDC, float fDeltaTime )

if( m_pTexture )
{
	POSITION camPos = GET_SINGLE( CCamera )->GetPos();

	BitBlt( hDC, m_tPos.x - m_tSize.x * m_tPivot.x - camPos.x,
		m_tPos.y - m_tSize.y * m_tPivot.y - camPos.y, m_tSize.x, m_tSize.y, m_pTexture->GetDC(), 0
}
```