21. 더블 버퍼와 피벗

ResourceManager.h

```h
#pragma once

#include "../Game.h"

class CResourcesManager
{
private:
	unordered_map<string, class CTexture*> m_mapTexture;
	HINSTANCE	m_hInst;
	HDC			m_hDC;
	CTexture*	m_pBackBuffer;

public:
	CTexture* GetBackBuffer() const;

public:
	bool Init( HINSTANCE hInst, HDC hDC );
	class CTexture* LoadTexture( const string& strKey,
		const wchar_t* pFileName, const string& strPathKey = TEXTURE_PATH );
	CTexture* FindTexture( const string& strKey );

	DECLARE_SINGLE( CResourcesManager )
};
```

ResourceManager.cpp

```cpp
#include "ResourcesManager.h"
#include "Texture.h"

DEFINITION_SINGLE( CResourcesManager )


CResourcesManager::CResourcesManager()
{
}

CResourcesManager::~CResourcesManager()
{
	SAFE_RELEASE( m_pBackBuffer );
	Safe_Release_Map( m_mapTexture );
}

CTexture* CResourcesManager::GetBackBuffer() const
{
	m_pBackBuffer->AddRef();
	return m_pBackBuffer;
}

bool CResourcesManager::Init( HINSTANCE hInst, HDC hDC )
{
	m_hInst = hInst;
	m_hDC = hDC;

	// Load Back Buffer
	m_pBackBuffer = LoadTexture( "BackBuffer", L"BackBuffer.bmp" );

	return true;
}
```

백 버퍼를 하나 만들어서 바로 메인 화면 버퍼에 그리는 것이 아닌 이 백 버퍼에 그린 후에, 메인 화면을 백 버퍼로 교체한다.

이를 매 프레임마다 반복해 줌.

Core.cpp

```cpp
void CCore::Render( float fDeltaTime )
{
	CTexture* pBackBuffer = GET_SINGLE( CResourcesManager )->GetBackBuffer();

	Rectangle( pBackBuffer->GetDC(), 0, 0, m_tRs.iW, m_tRs.iH );
	// Draw to back buffer
	GET_SINGLE( CSceneManager )->Render( pBackBuffer->GetDC(), fDeltaTime );

	// back buffer to main dc
	BitBlt( m_hDC, 0, 0, m_tRs.iW, m_tRs.iH, pBackBuffer->GetDC(), 0, 0, SRCCOPY );


	SAFE_RELEASE( pBackBuffer );
}
```

→ 백 버퍼 받아오고

→ 한번 화면 크기의 사각형으로 덮어주고) 백 버퍼 DC에 그린다.

→ 버퍼에 그리고

→ 메인 DC에 백 버퍼 DC 복사

Player.cpp

```cpp
bool CPlayer::Init()
{
	SetPos( 0.f, 0.f );
	SetSize( 203.f, 203.f );
	SetSpeed( 400.f );
	SetPivot( 0.5f, 0.5f );

	SetTexture( "Player", L"Kirby.bmp" );

	return true;
}
```

Object.cpp

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

if( m_pTexture )
{
	BitBlt( hDC, m_tPos.x - m_tSize.x * 0.5f, m_tPos.y - m_tSize.x * 0.5f, m_tSize.x, m_tSize.y, m_p
}
```

이미지 위치의 기준점을 이동시키는 것임.