**Creating an MMC Snap-in Using ATL in C++**

This guide provides a step-by-step approach to creating a **Microsoft Management Console (MMC) Snap-in** using **ATL (Active Template Library)** in **Visual Studio**.

**Step 1: Create a New ATL Project**

1. **Open Visual Studio** and click **File → New → Project**.
2. Select **Visual C++ → ATL → ATL Project**.
3. Name the project **DemoSnap** and click **Create**.
4. In the ATL Project Wizard, choose:
   * **Dynamic-Link Library (DLL)**
   * Enable **Support COM+ 1.0**
   * Click **Finish**.

**Step 2: Add an ATL Object for MMC Snap-in**

1. In **Solution Explorer**, right-click the **DemoSnap** project.
2. Select **Add → Class**.
3. Choose **ATL → ATL Simple Object**.
4. Name it **SnapinComponent** and click **Finish**.

This creates SnapinComponent.h and SnapinComponent.cpp.

**Step 3: Implement MMC Interfaces**

Modify SnapinComponent.h to include the required interfaces:

cpp

#pragma once

#include "resource.h"

#include <mmc.h>

class ATL\_NO\_VTABLE CSnapinComponent :

public CComObjectRootEx<CComSingleThreadModel>,

public CComCoClass<CSnapinComponent, &CLSID\_SnapinComponent>,

public IComponent,

public IComponentData

{

public:

CSnapinComponent() {}

DECLARE\_REGISTRY\_RESOURCEID(IDR\_SNAPINCOMPONENT)

BEGIN\_COM\_MAP(CSnapinComponent)

COM\_INTERFACE\_ENTRY(IComponent)

COM\_INTERFACE\_ENTRY(IComponentData)

END\_COM\_MAP()

// IComponentData Methods

STDMETHOD(Initialize)(LPUNKNOWN pUnknown);

STDMETHOD(CreateComponent)(LPCOMPONENT\* ppComponent);

STDMETHOD(DestroyComponent)();

// IComponent Methods

STDMETHOD(Notify)(LPDATAOBJECT lpDataObject, MMC\_NOTIFY\_TYPE event, LPARAM arg, LPARAM param);

STDMETHOD(GetDisplayInfo)(RESULTDATAITEM\* pResultDataItem);

};

// Object creation

OBJECT\_ENTRY\_AUTO(\_\_uuidof(SnapinComponent), CSnapinComponent)

**Step 4: Implement Snap-in Logic**

Modify SnapinComponent.cpp:

cpp

#include "pch.h"

#include "SnapinComponent.h"

// Initialize the Snap-in

STDMETHODIMP CSnapinComponent::Initialize(LPUNKNOWN pUnknown) {

return S\_OK;

}

// Create the MMC Component

STDMETHODIMP CSnapinComponent::CreateComponent(LPCOMPONENT\* ppComponent) {

\*ppComponent = this;

AddRef();

return S\_OK;

}

// Destroy the Component

STDMETHODIMP CSnapinComponent::DestroyComponent() {

Release();

return S\_OK;

}

// Handle MMC Events

STDMETHODIMP CSnapinComponent::Notify(LPDATAOBJECT lpDataObject, MMC\_NOTIFY\_TYPE event, LPARAM arg, LPARAM param) {

return S\_OK;

}

// Provide Display Info

STDMETHODIMP CSnapinComponent::GetDisplayInfo(RESULTDATAITEM\* pResultDataItem) {

return S\_OK;

}

**Step 5: Register the Snap-in**

Modify DemoSnap.idl to add the Snap-in to MMC:

idl

[

uuid(12345678-1234-5678-1234-567812345678),

version(1.0)

]

library DemoSnapLib

{

importlib("stdole32.tlb");

[

uuid(87654321-4321-8765-4321-876543218765)

]

coclass SnapinComponent

{

[default] interface IComponentData;

interface IComponent;

};

};

Compile and register using:

sh

regsvr32 DemoSnap.dll

**Step 6: Load the Snap-in in MMC**

1. Open **Run (Win + R)** → Type mmc.exe → Press Enter.
2. Click **File → Add/Remove Snap-in**.
3. Select **DemoSnap** and click **Add → OK**.
4. Verify that the Snap-in loads without errors.

**Project File Structure**

bash

DemoSnap/

│── DemoSnap.sln # Solution file

│── DemoSnap.vcxproj # Project file

│── SnapinComponent.h # Snap-in header

│── SnapinComponent.cpp # Snap-in logic

│── dllmain.cpp # DLL entry point

│── resource.h # Resource definitions

│── DemoSnap.idl # Interface definition

│── DemoSnap.rc # Resource script

│── Debug/ # Compiled output

│ │── DemoSnap.dll # DLL file

│ │── DemoSnap.exp # Export file

│ │── DemoSnap.lib # Library file

│ │── DemoSnap.pdb # Debug symbols