Factory Pattern

https://www.codeproject.com/Tips/737117/Factory-Pattern-Example-in-Csharp

Factory Pattern Example in C#

[**Vipin\_Arora**](https://www.codeproject.com/script/Membership/View.aspx?mid=8728087), 2 Mar 2014 [CPOL](http://www.codeproject.com/info/cpol10.aspx)

|  |  |
| --- | --- |
|  |  |
|  | |  |  |  | | --- | --- | --- | | Rate: | [vote 1vote 2vote 3vote 4vote 5](https://www.codeproject.com/Tips/737117/Factory-Pattern-Example-in-Csharp) |  | |

This tip is about how to implement factory design pattern

https://www.codeproject.com/images/warning48.png**Is your email address OK?** You are signed up for our newsletters but your email address is either unconfirmed, or has not been reconfirmed in a long time. Please **click**[**here**](https://www.codeproject.com/script/Membership/SendConfirmRequest.aspx?rp=%2fTips%2f737117%2fFactory-Pattern-Example-in-Csharp)**to have a confirmation email sent** so we can confirm your email address and start sending you newsletters again. Alternatively, you can [**update your subscriptions**](https://www.codeproject.com/script/Membership/Subscribe.aspx?rp=%2fTips%2f737117%2fFactory-Pattern-Example-in-Csharp).

* [**Download source code (no EXE) - 18.3 KB**](https://www.codeproject.com/KB/cs/737117/ProFactoryPattern-noexe.zip)
* [**Download source code - 39.7 KB**](https://www.codeproject.com/KB/cs/737117/ProFactoryPattern.zip)

Introduction

This tip is about how to implement factory design pattern.

Background

Design patterns are general reusable solutions to common problems that occurred in software designing. There are broadly 3 categories of design patterns, i.e., Creational, Behavioral and Structural.

Now, Factory Design Pattern falls under the category of creational design pattern.  
It deals with the problem of creating objects (products) without specifying the exact class of object that will be created. The essence of this pattern is to "Define an interface for creating an object, but let the classes that implement the interface decide which class to instantiate. The Factory method lets a class defer instantiation to sub classes."  
  
(Source: [Factory method pattern](http://en.wikipedia.org/wiki/Factory_method_pattern)[[^](http://en.wikipedia.org/wiki/Factory_method_pattern)])

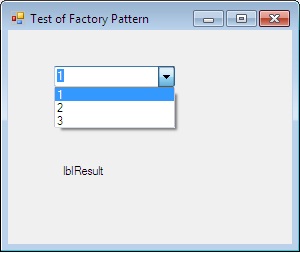
Where to Use?

It would be tedious when the client needs to specify the class name while creating the objects. So, to resolve this problem, we can use Factory pattern. It provides the client a simple way to create the object. The example below will elaborate the factory pattern in detail.

Using the Code

Here is the step by step procedure to create an application with Factory Pattern:

1. Create a new Windows project (say: ProFactoryPattern).
2. Add a ComboBox (name it cmbSelect) and one Label (for displaying the Result (name it lblResult)) as shown below:



1. Add the below interface and two classes implement this interface. Please note that both classes have the same methods.

**interface IGet**

**{**

**string ConC(string s1, string s2);**

**}**

**class clsFirst : IGet**

**{**

**public string ConC(string s1, string s2)**

**{**

**string Final = "From First: " + s1 + " and " + s2;**

**return Final;**

**}**

**}**

**class clsSecond : IGet**

**{**

**public string ConC(string s1, string s2)**

**{**

**string Final = "From Second: " + s1 + " and " + s2;**

**return Final;**

**}**

**}**

1. Next, add the factory class, in which conditions are there to create the objects. Method in this class basically decides object of class to be created, as shown below:

**class clsFactory**

**{**

**static public IGet CreateandReturnObj(int cChoice)**

**{**

**IGet ObjSelector = null;**

**switch (cChoice)**

**{**

**case 1:**

**ObjSelector = new clsFirst();**

**break;**

**case 2:**

**ObjSelector = new clsSecond();**

**break;**

**default:**

**ObjSelector = new clsFirst();**

**break;**

**}**

**return ObjSelector;**

**}**

**}**

1. Finally, the client code looks like shown below. In this, the client does not bother about the classes and class name and does not worry if any new class will be added:

private void cmbSelect\_SelectedIndexChanged(object sender, EventArgs e)

{

IGet ObjIntrface = null;

ObjIntrface = clsFactory.CreateandReturnObj(cmbSelect.SelectedIndex + 1);

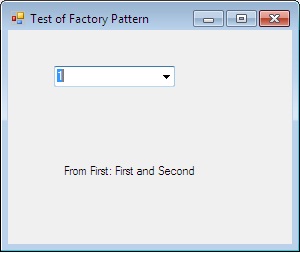
string res = ObjIntrface.ConC("First", "Second");

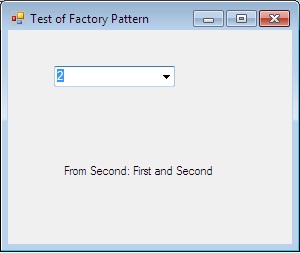
lblResult.Text = res;

}

1. Now the output will be:

If 1 will be selected, then object of clsFirst will be created and in case of 2, clsSecond will be created:





Points of Interest

Now consider, if we have to add one more class (say: clsThird) and we want to add one more case in switch condition (under class: CreateandReturnObj), then client code will not change. Also, you can get the input from config.