**Factory Pattern**

In Factory pattern, we create object without exposing the creation logic to the client and refer to newly created object using a common interface.

**Implementation**

To use factory pattern, we code to the interface and not to the implementation. Following are the steps to implement a factory pattern:

**Step 1**

Create a generic interface that will be implemented by all concrete classes that we want to instantiate using the factory pattern.

**Step 2**

Create concrete classes implementing the same interface.

**Step 3**

Create a Factory to generate object of concrete class based on given information. Example: Create a method that takes a string to decide which type of object is to be instantiated and returned.

**Step 4**

Use the Factory to get object of concrete class by passing information such as type.

**Advantage:**

By coding to the interface, rather than implementation, we have de-coupled the application. Now, we can add new concrete implementation of the interface, without breaking the existing functionality.

Also, by providing factory to create an object, we have hidden the logic that is involved in creation of such an object. For example, there might be many steps involved in creating a usable copy of that object, and we mention all those steps in factory method, so that the end user does not have to worry about writing all that code over and over again, and hence code reusability is established.