Abstract Factory ID

Student:

Izabela Kuźniar

Teacher:

Andrea Corradini

Course:

Software Design Patterns

# Name and category

Abstract Factory is a creational pattern.

# Intent:

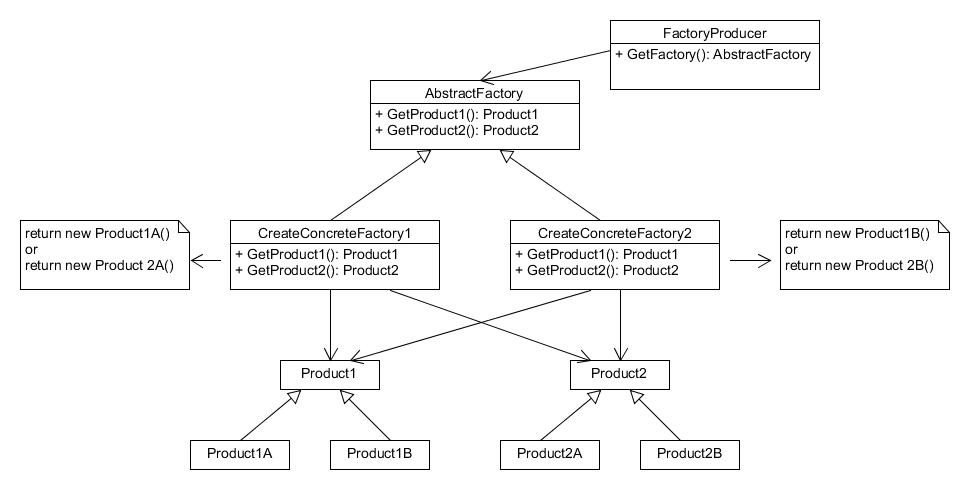
It provides an interface for creating families of related or dependent object without specifying their concrete.

# Motivation:

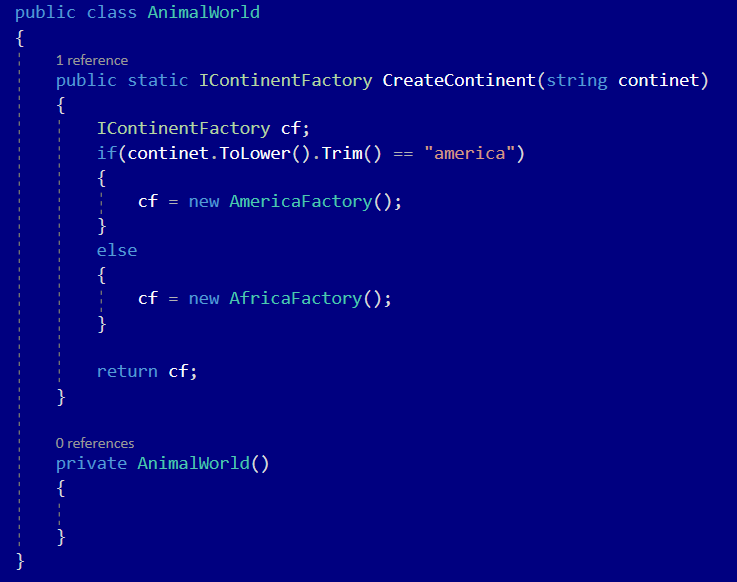
Abstract Factory Pattern is simply Factory of Factory. With this pattern one interface is used for creating different products on different scenarios

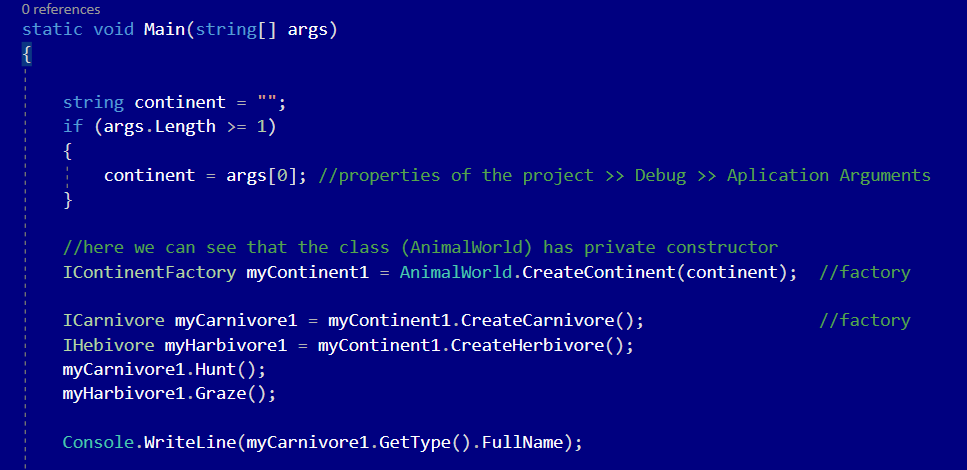
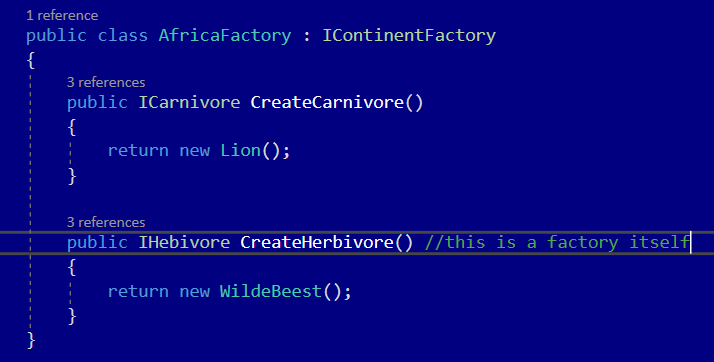
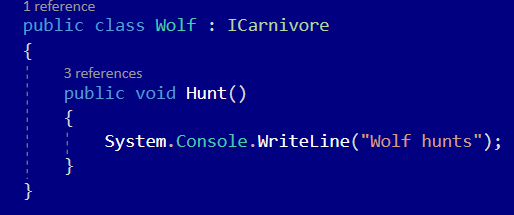
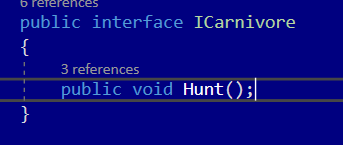
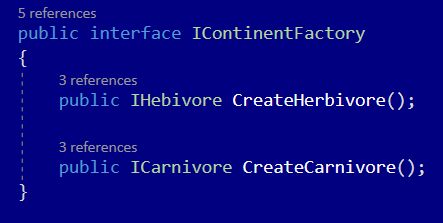
* Clients get and use an interface of the abstract factory object to create products
* Clients are decoupled from the actual concrete products returned by each factory

# Structure as a UML class diagram



# Implementation:





# Consequences:

Benefits:

* Products getting form a factory are compatible with each other
* Loose coupling between concrete product and client code
* Separating responsibilities
* Open/Close Principle. You can introduce new variants of products without breaking existing client code

Drawbacks:

* Code being more complicated than it should be (many interfaces and classes introduced)

# Known uses

This pattern can be used to build GUI elements. GUI for Windows and Apple will have the same items as for example a Button, but it will implement it differently.

# Related patterns

1. Abstract Factory is evolved version of Factory Method Pattern.
2. Builder focuses on constructing complex objects step by step. Abstract Factory specializes in creating families of related objects.
3. Prototype Pattern can be used to compose the methods of Abstract Factory classes.
4. Can be an alternative of Façade.
5. You can use Abstract Factory along with Bridge.
6. Abstract Factory Pattern can be implemented as Singleton.