Edward Plesa

Abstract Factory Design Pattern

To demonstrate the Abstract Factory Design Pattern in a simple and effective way, I created a mockup order form application allowing the user to "order" a weapon of mass destruction.

The way this works is that the order form first tells the user to pick a "middleman"(AbstractFactory) for the creation of their specified vehicle factory. You are allowed to pick from jet, boat, or tank factory options.

Then once this middleman has been established, the user is then asked for which model of their specified vehicle should be ordered from the "middleman"(AbstractFactory). In this instance, every vehicle has two different choices. The factory then returns an object of the class of the vehicle they chose.

Finally, this vehicle object can output the string dictating that their specific vehicle choice was created and that it is ready to be shipped.

Creating the code in the abstract factory method has its drawbacks and its benefits. Creating a program using this design pattern makes it more scalable and easier to separate and add different classes and how they interact with the factory easier. However, doing it this way complicates the code in a sense that you must add many interfaces and classes that allow for the design pattern to be used compared to a more traditional implementation.