The Factory Pattern

Gerald Britton

IT SPECIALIST

@GeraldBritton www.linkedin.com/in/geraldbritton

Overview

Classification: Creational

Define interface for creating an object

Let subclasses decide which object

Defer instantiation to subclasses

Also known as Virtual Constructor

Demo

Motivating Example:

Create an object for a model of car

Support several car models

Don’t know which one we want until runtime

Introduction to Sublime Text

Notes for Demo 1

Simple Factory Pattern Structure

Notes for Demo 2

Problems solved so far:

Eliminated the open/closed violation

Eliminated the dependencies

Separated concerns

Limited to one autofactory

What if we want more factories?

The classic Factory Pattern is what we need.

Factory Pattern Structure

Factory Pattern Structure

Notes for demo 5

Added an abstract factory base class

Many factories can be implemented

The implementations can vary

A complex factory might use other patterns

- Builder pattern

Encapsulated the factory loader

Summary

Encapsulates object instantiation

Supports dependency inversion

Clients don’t depend on implementation

Depend on abstraction instead

Simple Factory Pattern – one factory

- Often just what you need

Classic Factory Pattern – one or more

- Must get an instance of the factory
- Then make the object
- Most flexible