Design Defects and Restructuring

LECTURE 07

SAT, OCT 12, 2019

Creational Patterns

Abstract Factory

Builder

Factory Method

Prototype

Singleton

Factory Method

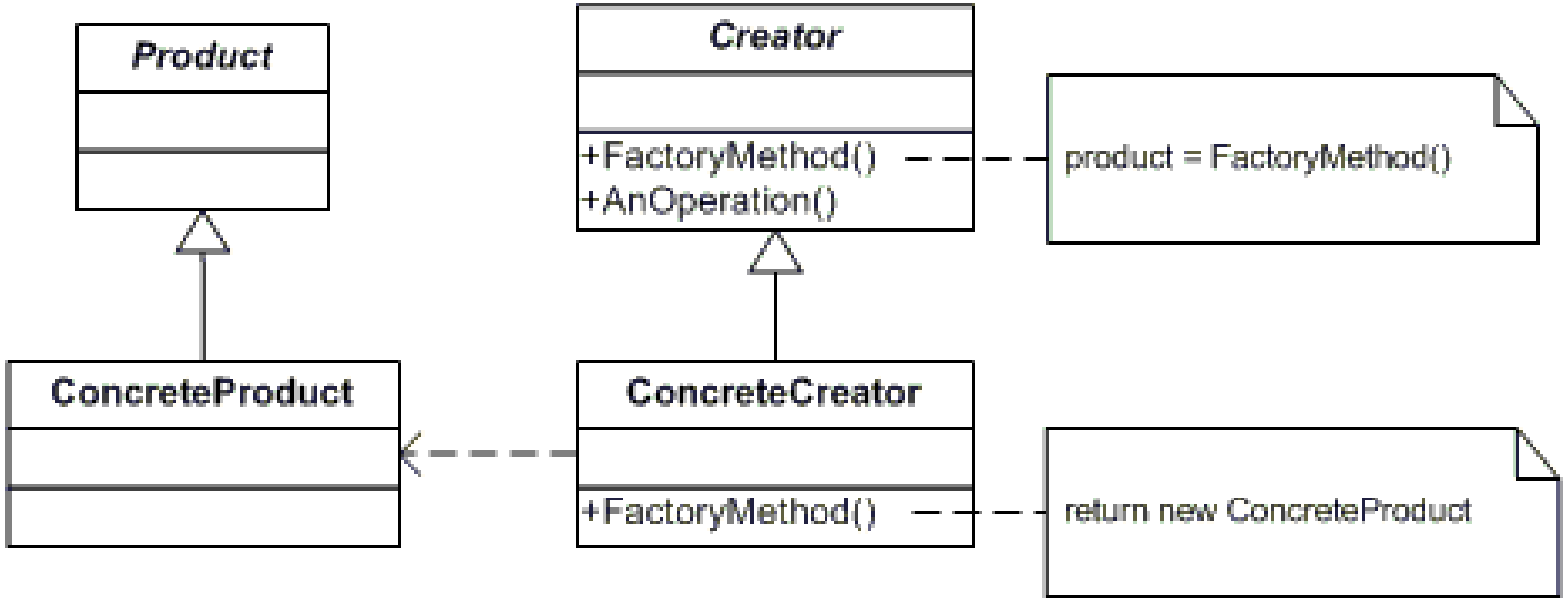
Intent

- Define an interface for creating an object, but let subclasses decide which class to instantiate
- Factory Method lets a class defer instantiation to subclasses

Applicability

- A class can’t anticipate the class of objects it must create
- A class wants its subclasses to specify the objects it creates
- Classes delegate responsibility to one of several helper subclasses, and you want to localize the knowledge of which helper subclass is the delegate

Factory Method



Prototype

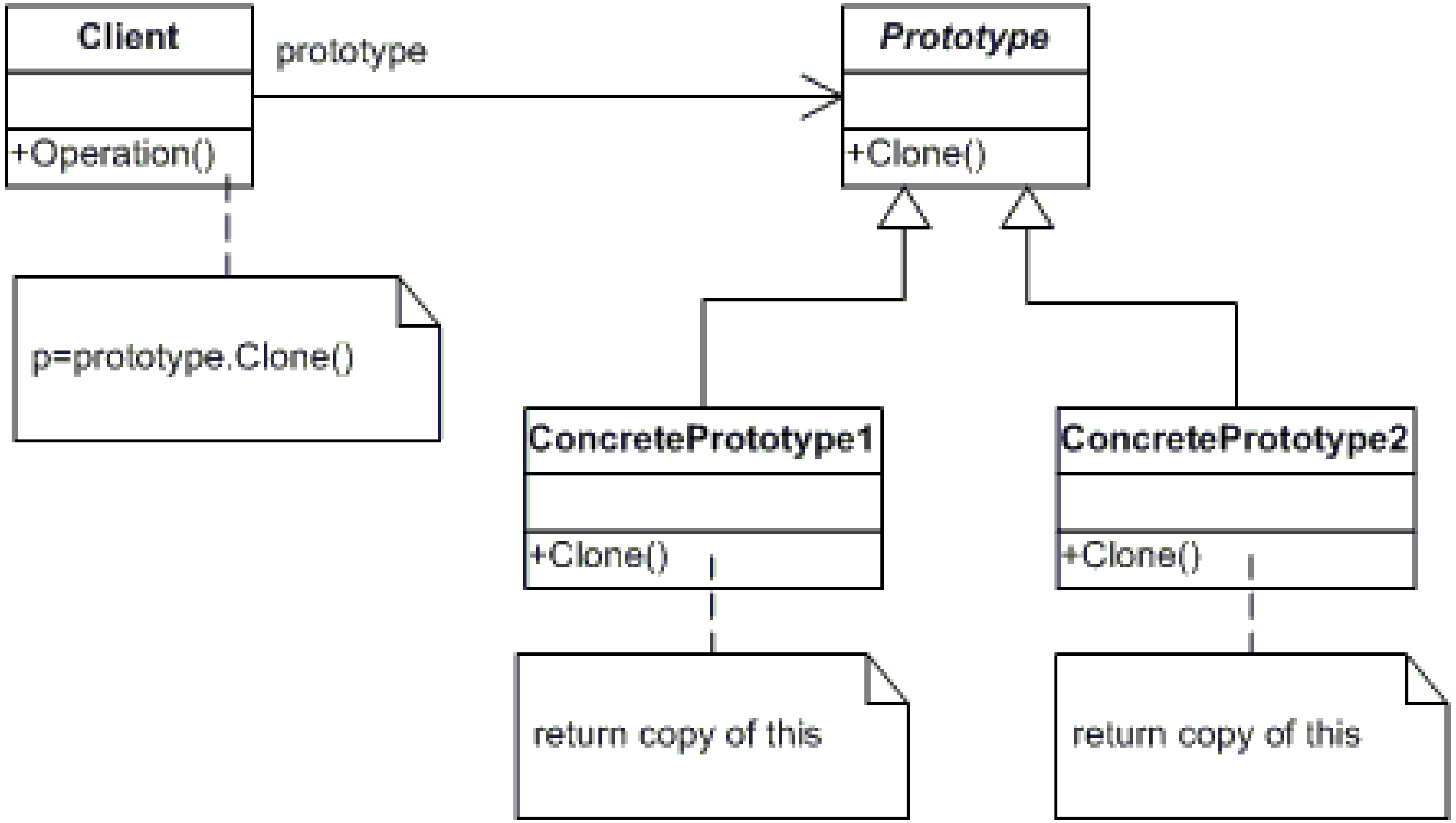
Intent

- Specify the kinds of objects to create using a prototypical instance, and create new objects by copying this prototype

Applicability

- When the classes to instantiate are specified at run-time, for example, by dynamic loading; or
- To avoid building a class hierarchy of factories that parallels the class hierarchy of products; or
- When instances of a class can have one of only a few different combinations of state
  - It may be more convenient to install a corresponding number of prototypes and clone them rather than instantiating the class manually, each time with the appropriate state

Prototype


Client
+Operation()
prototype
Prototype
+Clone()
p=prototype.Clone()
ConcretePrototype1
+Clone()
ConcretePrototype2
+Clone()
return copy of this
return copy of this

Singleton

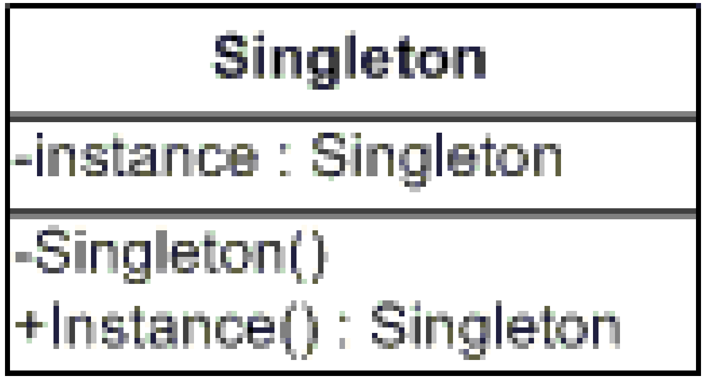
Intent

- Ensure a class only has one instance, and provide a global point of access to it

Applicability

- There must be exactly one instance of a class, and it must be accessible to clients from a well-known access point
- When the sole instance should be extensible by sub-classing, and clients should be able to use an extended instance without modifying their code

Singleton



Singleton

-instance : Singleton

-Singleton()

+Instance() : Singleton