



Creational Design Pattern << Prototype >>



Creational Design Patterns :: About



Concerned with Object-Oriented concept

Offer how to create instances form objects in:

... Flexible

... Efficient

... Maintainable

Provide INTERFACE for creating Instances in SUPPER CLASS

Allows SUBCLASSES to alter TYPE of instance to create

Great Decoupling with other code. HIGH Flexibility

Centralize creational logic

Creational Design Patterns

1 Singleton

... One INSTANCE.

... Global access

2 Factory

... Interface to create instances in supper class.

... Subclass decides type of Instance`

3 Abstract Factory

... Group similar Factory based on Instances types

4 Builder

... Create complex instance step by step

5 Prototype

... Clone Existing instance

Prototype Design Pattern

1 In Brief....

- ... make caller code to copy existing instance with ease.
- ... helps to get rid of repeated initialization
- ... **CLONING**

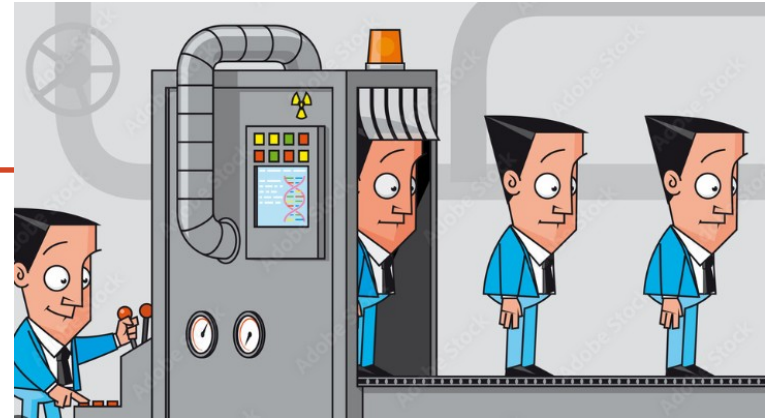
2 HOW ??

- Define Abstract/Interface class with “**clone()**” and other required methods/properties
 - ... *this will be the uniform for concrete classes*

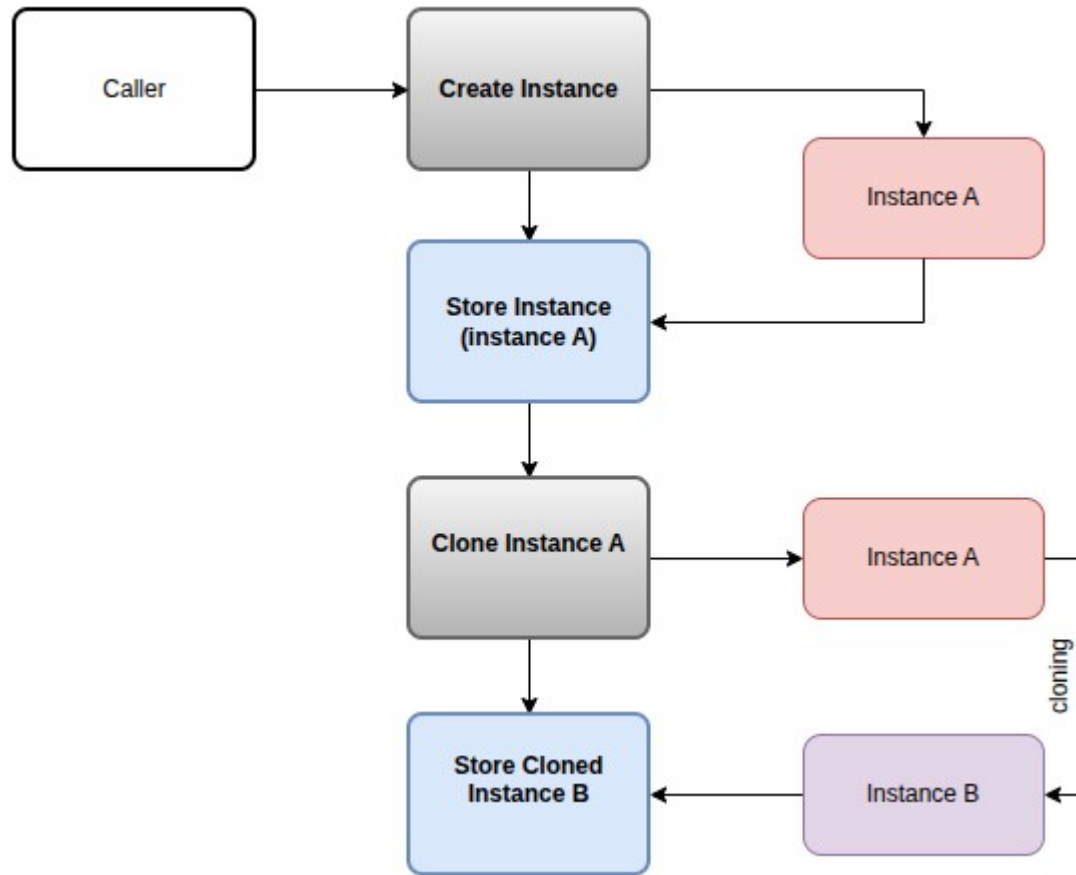
Implement concrete class by implementing the Defined Abstract/Interface

Implement **clone()** method

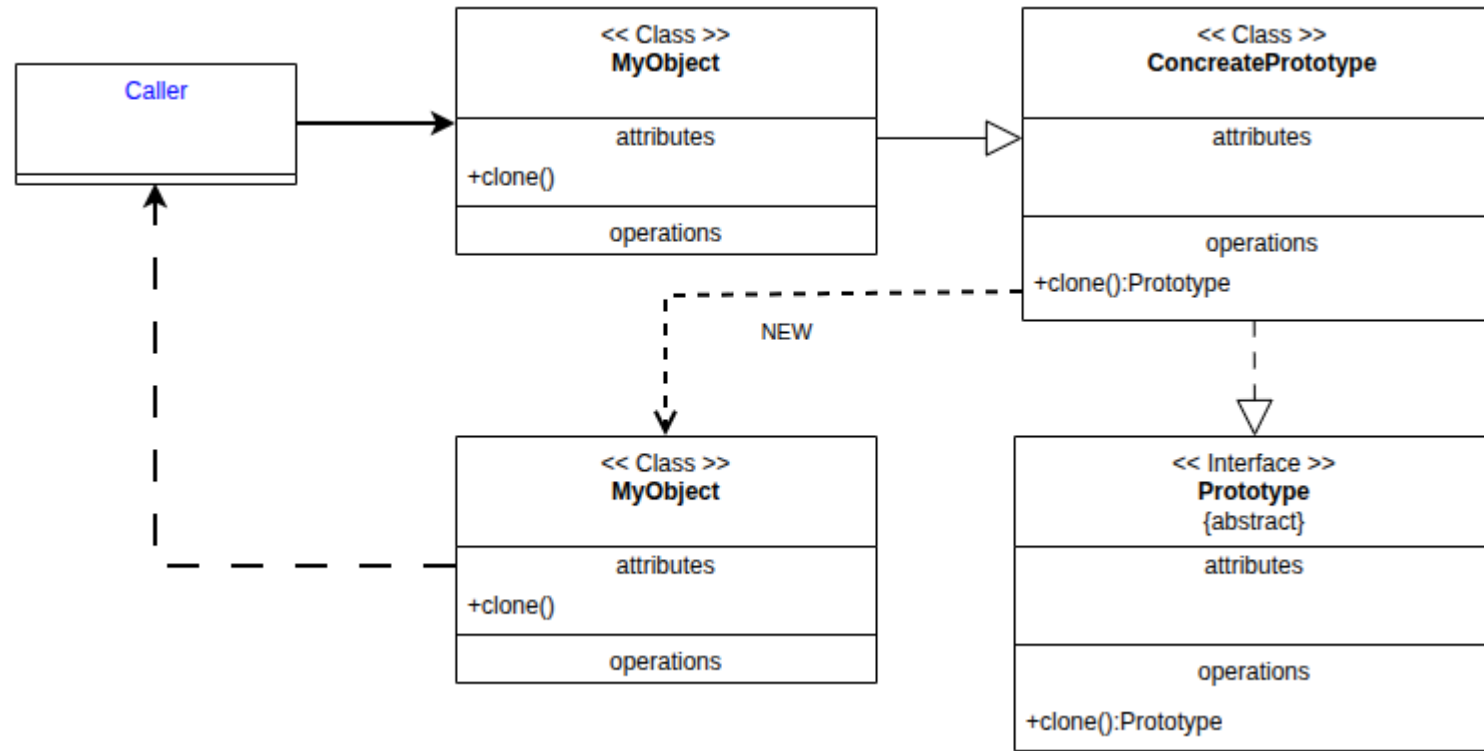
- ... *copy all the properties and fields of current instance to NEW instance*
- ... *return the new **cloned** instance*



Prototype Design Pattern :: Flowchart



Factory Design Pattern :: UML



Prototype Design Pattern :: How to Implement

- 1 Define the interface with **clone()** and other required methods & properties.
- 2 Implement the interface in concrete class
- 3 Implement the clone() method based on the programming language
... must copy all fields, properties (private & public) of current instance to NEW instance
- 4 Client/caller will use the clone() method to retrieve exact copy of required required instance

Prototype Design Pattern :: Practical

- 1 Check/study the first given sample
- 2 Check/study the second sample
- 3 Run/Debug and check how it works
- 3 Extend concrete implementation