MINISTRY OF EDUCATION REPUBLIC OF MOLDOVA

TECHNICAL UNIVERSITY OF MOLDOVA

REPORT

Laboratory work no. 1

On IPP

“Creational patterns”

Performed by: **st. gr. FAF-141**

Cristea Victor

Verified by: **Anastasia Șerșun**

Chisinau 2017

Creational Patterns

- Singleton
- Prototype
- Builder
- Factory Method
- Abstract Factory

Singleton

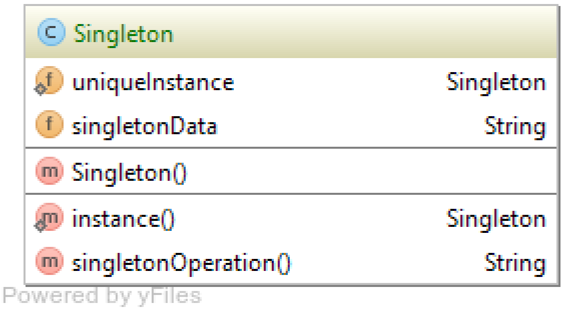
- Story

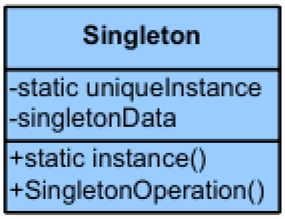
Singleton ensures that only one(single) object can be created from the class.

Men's 100 meters world record holder is a singleton. There can be at most one active "Men's 100 meters world record holder" at any given time. Regardless of who that person is the title, "Men's 100 meters world record holder" is a global point of access that identifies the fastes person in the world.

- Implementation

UML:





Source Code:

Clone repo: `$  git clone https://github.com/cristeav49/IPP.git .`

Move to singleton folder:

```
$  cd /src/main/java/com/github/cristea/basepatterns/creational/singleton
```

Prototype

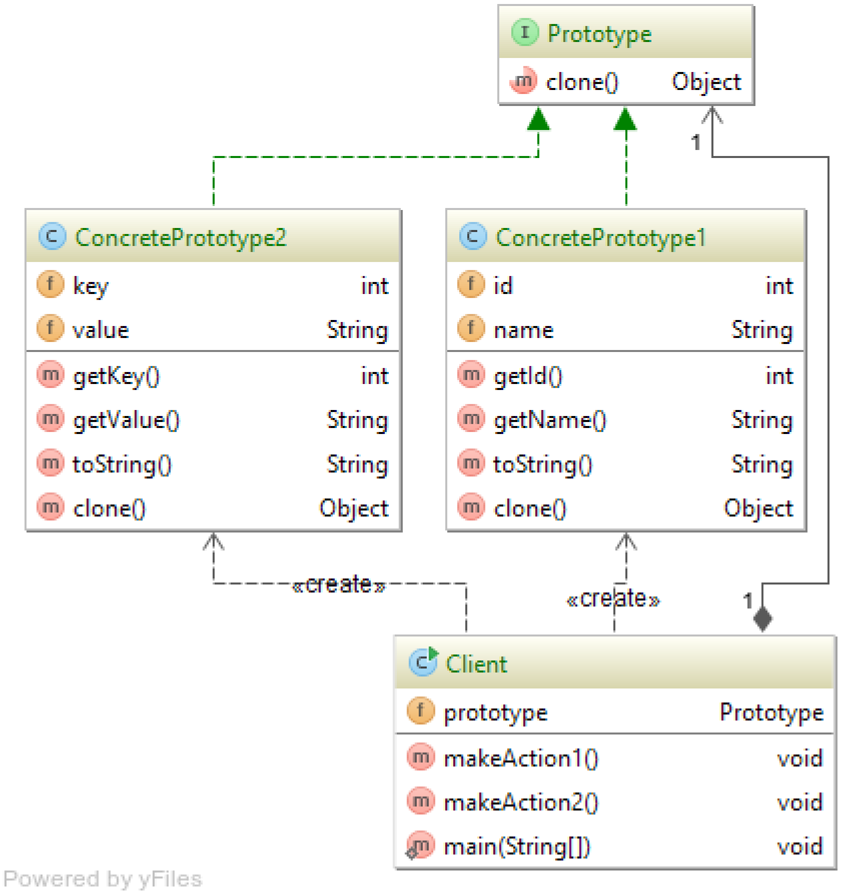
- Story

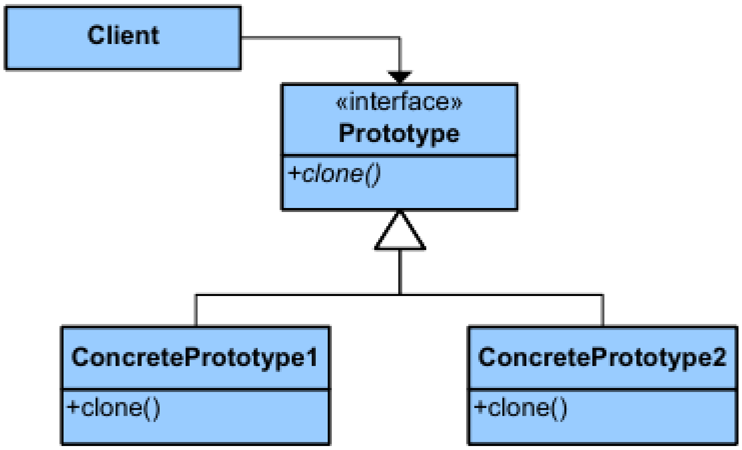
Clone itself.

Sheep Dolly is the first mammal to be cloned, so Dolly is a duplicate.

- Implementation

UML:





Source Code:

Clone repo:

```
$  git clone https://github.com/cristeav49/IPP.git .
```

Move to prototype folder:

```
$  cd /src/main/java/com/github/cristea/basepatterns/creational/prototype
```

Builder

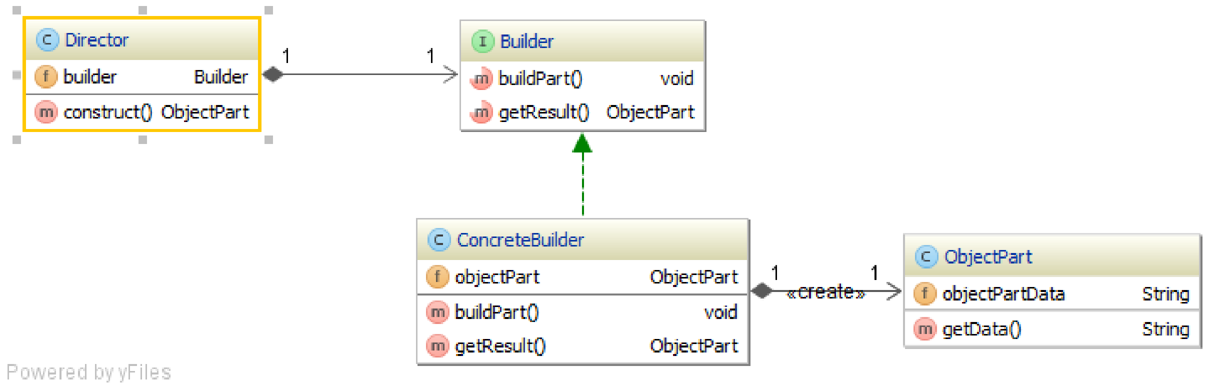
- Story

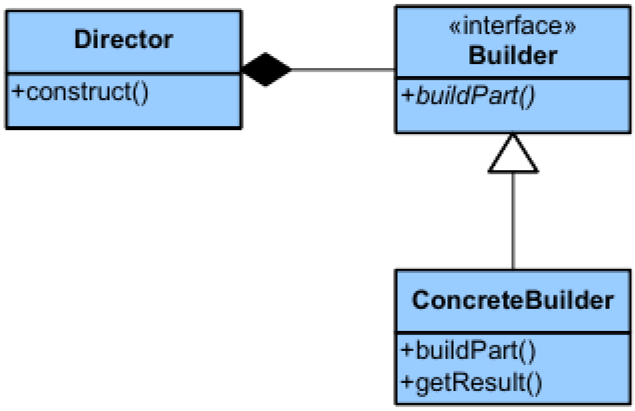
Separates the construction of a complex object from its representation so that the same construction process can create different representations.

This pattern is used by PC shops to contruct PC's. PC is combination of various parts like CPU, motherboard, memory, storage, power supply, video card, etc. To build a PC same construction process is used even for each part we have different variation. Whether a customer picks a classical hard disk or SSD for storage, the construction process is the same.

- Implementation

UML:





Source Code:

Clone repo:

```
$  git clone https://github.com/cristeav49/IPP.git .
```

Move to builder folder:

```
$  cd /src/main/java/com/github/cristea/basepatterns/creational/builder
```

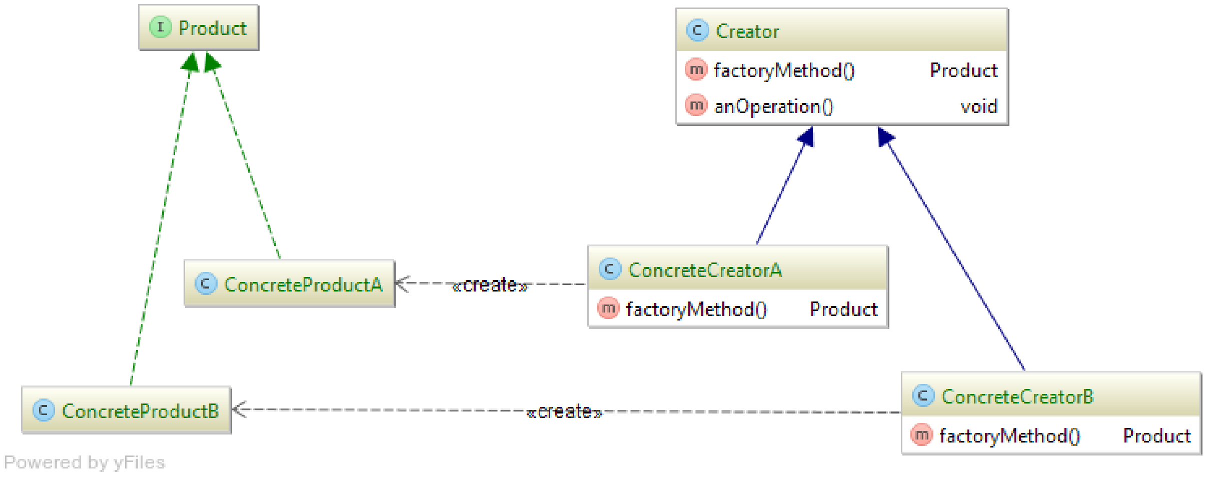
Factory Method

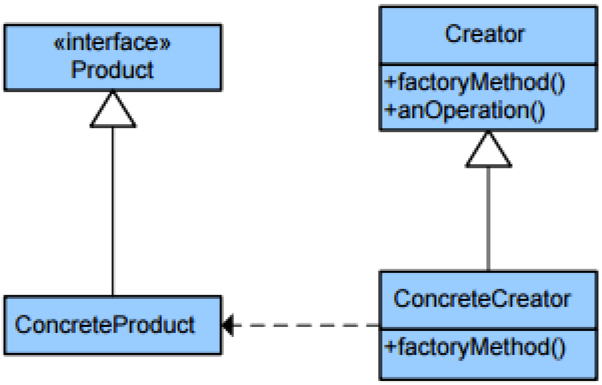
- Story

Defines an interface for creating objects, but lets subclasses decides which class to instantiate. Plasticine is used for children's play. Plasticine is injected into predefined molds. The class of end product(ball, toy, sculpture, cake) is determined by the mold.

- Implementation

UML:





Source Code:

Clone repo:

```
$  git clone https://github.com/cristeav49/IPP.git .
```

Move to factorymethod folder:

```
$  cd /src/main/java/com/github/cristea/basepatterns/creational/factorymethod
```

Abstract Factory

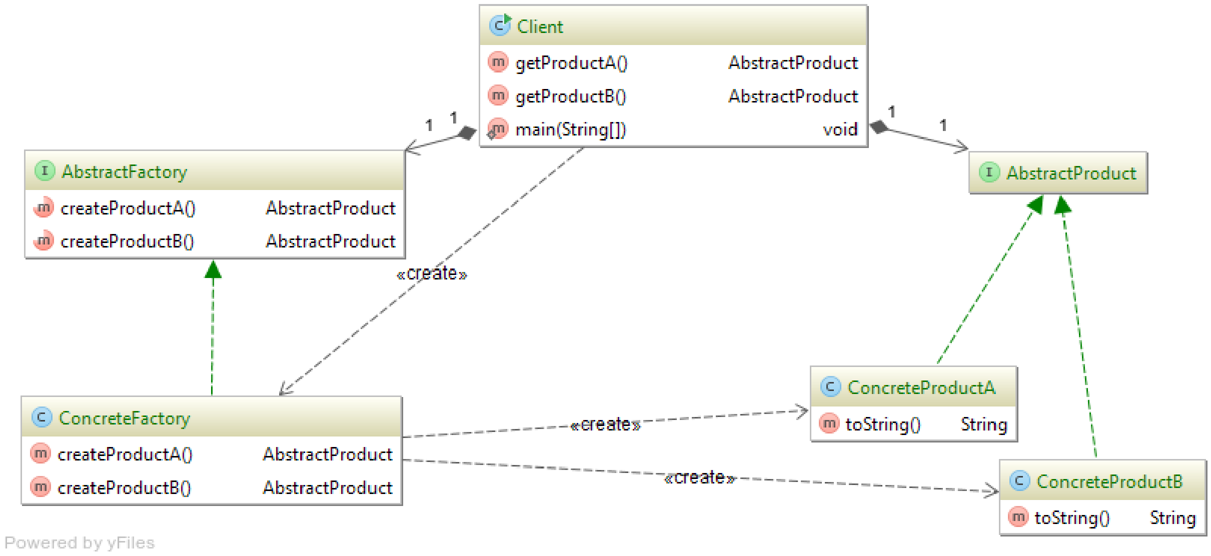
- Story

Provides an interface for creating families of related objects, without specifying concrete classes.

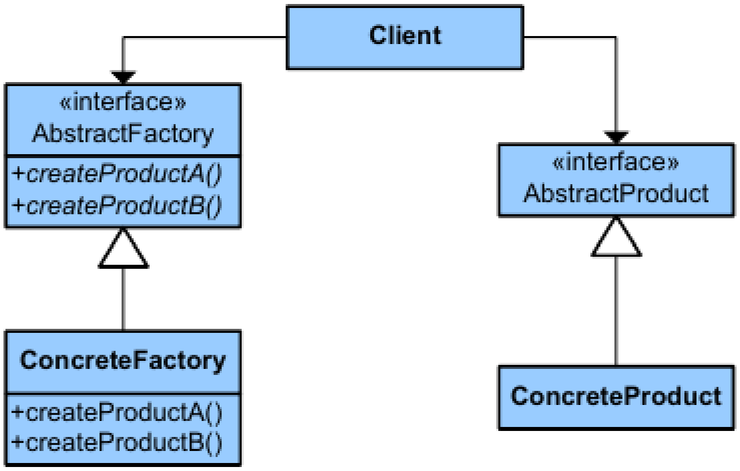
This pattern is found in the cards stamping equipment used in the manufacture in order to produce playing cards. Cards stamping machine is an Abstract Factory which produces a cards. The same machine is used to stamp French, Italian or German cards.

- Implementation

UML:



Client
getProductA() AbstractProduct
getProductB() AbstractProduct
main(String[]) void
AbstractFactory
createProductA() AbstractProduct
createProductB() AbstractProduct
AbstractProduct
ConcreteFactory
createProductA() AbstractProduct
createProductB() AbstractProduct
ConcreteProductA
toString() String
ConcreteProductB
toString() String
«create»
«create»
«create»
1
1
1
1
Powered by yFiles



Source Code:

Clone repo:

```
$  git clone https://github.com/cristeav49/IPP.git .
```

Move to abstractfactory folder:

```
$  cd
/src/main/java/com/github/cristea/basepatterns/creational/abstractfactory
```