**PROJECT DESIGN**

**Factory Method Pattern**

**Fitchburg State University.**

**Prof |Nguyen Thai.**

**CSC 7400.**

**Object Oriented Analysis and Design.**

**By**

**Prashanthi Sudha Kosgi.**

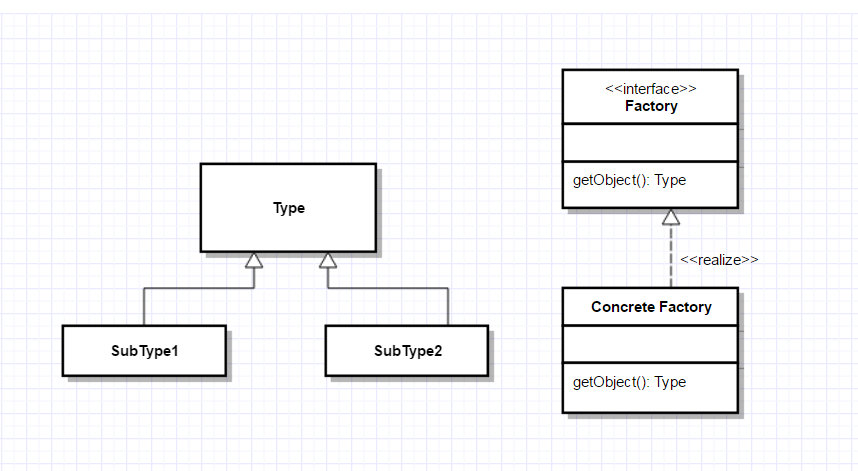
**Factory Method Pattern**

Factory method pattern is one of the most used design patterns in Java. Factory pattern is most suitable when there is some complex object creation steps are involved. To ensure that these steps are centralized and not exposed to composing classes, factory pattern should be used. Factory Method Pattern allows the sub-classes to choose the type of objects to create. It promotes the loose-coupling by eliminating the need to bind application-specific classes into the code.

#### The Factory Method design pattern usage:

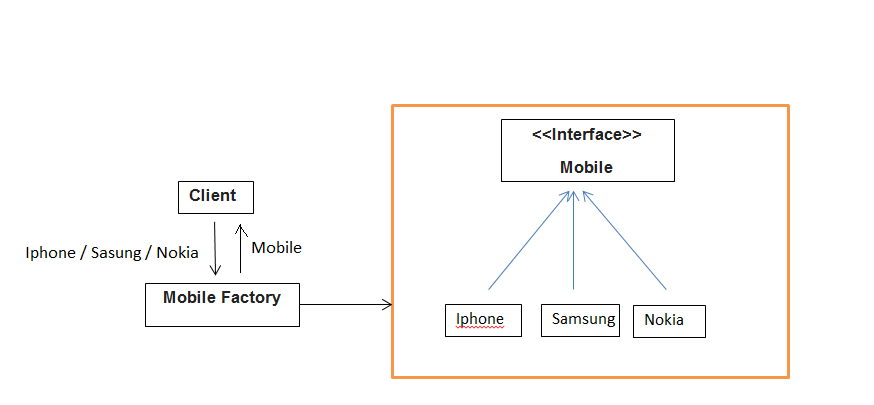
* When a class doesn't know what sub-classes will be required to create
* When a class wants that its sub-classes specify the objects to be created.
* When the parent classes choose the creation of objects to its sub-classes.

Simplified UML diagram for Factory Pattern.

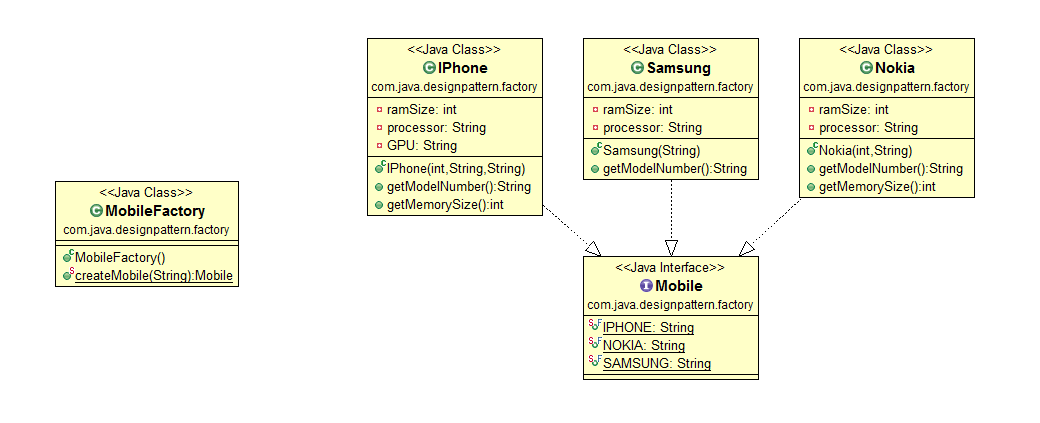


This project is about factory Method design pattern. In this project factory method pattern implementation is shown using a sample code. The following is an overall idea of my project.

**Factory Method Pattern**

****

**UML Class Diagrams:**



**Description of class diagrams:**

Samsung phone ramsize

Samsung processor

Parameterised constructor.

Returns model number.

Returns memory size.

Samsung

-ramsize : int

-processor : String

+Samsung(int, string) : void

+getModelNumber() : String

+grtMemorySize() : int

Iphone ramsize

Iphone processor

Iphone GPU

Parameterised constructor.

Returns model number.

Iphone

-ramsize : int

-processor : String

-GPU : String

+Iphone(int, String, String) : void

+getModelNumber() : String

Nokia phone ramsize

Nokia processor

Parameterised constructor.

Returns model number.

Returns memory size.

Nokia

-ramsize : int

-processor : String

+Nokia(int, string) : void

+getModelNumber() : String

+grtMemorySize() : int

IPHONE Constant

SAMSUNG Constant

NOKIA Constant.

Returns color.

Returns Dimensions.

Mobile

-IPHONE : String

-SAMSUNG : String

-NOKIA : String

+getColor() : String

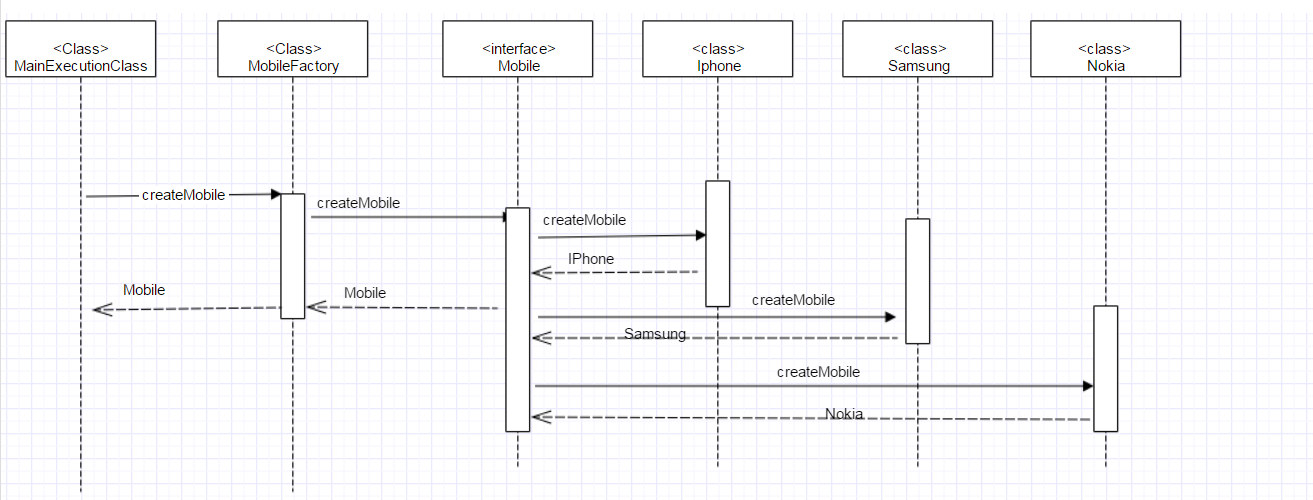
+getDimensions() : int int int

Returns a new Mobile object.

MobileFactory

+createMobile(string) : Mobile

**UML Sequence Diagrams:**



To demonstrate a Factory method pattern, a MainExecutionClass is taken to call a **MobileFactory** class which implements Factory method pattern. The MobileFactory class returns the object of depending upon the input request parameter .

If input parameter is sent as “samsung”, MobileFactory will return the Samsung class instance, if input parameter is sent as “iphone”, then MobileFactory will return Iphone class instance, if input parameter is sent as “nokia” then MobileFactory will return Nokia class instance.

So depending upon the input parameter to Factory class, its returns the different class instance.