Flyweight Design Pattern

A Flyweight Pattern says that just **"to reuse already existing similar kind of objects by storing them and create new object when no matching object is found"**.

Advantage of Flyweight Pattern

* It reduces the number of objects.
* It reduces the amount of memory and storage devices required if the objects are persisted

Usage of Flyweight Pattern

* When an application uses number of objects
* When the storage cost is high because of the quantity of objects.
* When the application does not depend on object identity.

Implementation

We are going to create a *Shape* interface and concrete class *Circle*implementing the *Shape* interface. A factory class *ShapeFactory* is defined as a next step.

*ShapeFactory* has a *HashMap* of *Circle* having key as color of the *Circle* object. Whenever a request comes to create a circle of particular color to *ShapeFactory*, it checks the circle object in its *HashMap*, if object of *Circle*found, that object is returned otherwise a new object is created, stored in hashmap for future use, and returned to client.

*FlyWeightPatternDemo*, our demo class, will use *ShapeFactory* to get a *Shape*object. It will pass information (*red / green / blue/ black / white*) to *ShapeFactory* to get the circle of desired color it needs.

