Prototype Pattern

The **Prototype Pattern** is a creational pattern that is used to create duplicate objects while keeping performance in mind. It allows a client to copy an existing object without having to become dependent on the class of the object. The pattern is useful when creating the object directly is costly. For things like database objects, it is better to keep a copy of the object in cache and return the copy as required, thus reducing database calls.

In Java, cloning an object is possible via the *Cloneable* interface.

public abstract class Shape implements *Cloneable* {  
 private final String id;  
 public Shape(String id) { this.id = id; }  
 public String getId() { return id; }  
 public Object clone() {  
 Object clone = null;  
 try {  
 clone = super.clone();  
 } catch (CloneNotSupportedException e) {  
 e.printStackTrace();  
 }  
 return clone;  
 }  
}  
  
public class Rectangle extends Shape {  
 // example class  
}

JAVA

Caching objects can be achieved using a hash table.

public class ShapeCache {  
 private static Hashtable<String, Shape> *shapeMap* = new Hashtable<>();  
 public static Shape getShape(String id) {  
 return (Shape) *shapeMap*.get(id).clone();  
 }  
 public void cacheShape(Shape shape) {  
 *shapeMap*.put(shape.getId(), shape);  
 }  
}

JAVA