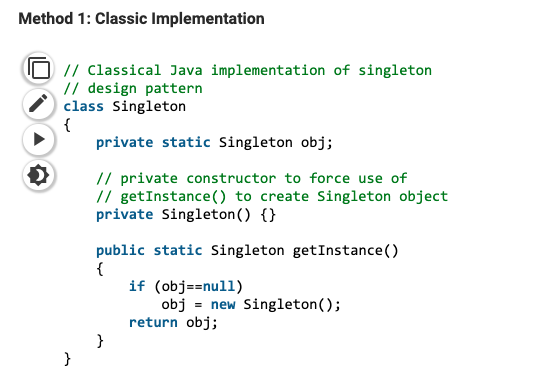
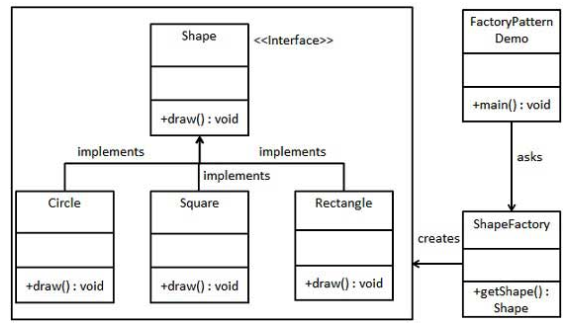


**Common Design Patterns:**

* Singleton: only one instance for the class; can be accessed globally
  + **pro**: only one instance 🡪 save memory; no need to create&delete instance repeatly
  + **con:** no interface, cannot inherit; not thread safe

****

* Factory Method: create object without exposing logic to the client using the common interface
  + when? we want to make different instance basing on the condition
  + how: **children class** make the factory interface



* Abstract Factory: factory of factories; best way to create an object
  + purpose: provide an interface to create multiple dependent objects; no need to explicitly specifying their classes
  + solve: interface choosing problem
* Observer:
  + purpose&when: when there is one-to-many relationship between objects such as if one object is modified, its dependent objects are to be notified automatically
  + solve: one object change; others are notified 🡪 high cohesion & low coupling