Static singleton

```
public class StaticSingleton {
  public int theAnswer;

  // Private constructor
  private StaticSingleton() { theAnswer = 42; }
  // The one instance
  public static StaticSingleton instance = new StaticSingleton();

  public static StaticSingleton getInstance() { return instance; }
}
```

Lazy/Dynamic singleton

```
public class LazySingleton {
  public int theAnswer;

  // Private constructor
  private LazySingleton() { theAnswer = 42; }
  // The one instance
  private static instance = null;

  public static LazySingleton getInstance() {
    if (instance == null)
      instance = new LazySingleton();
    return instance;
  }
}
```

Slow Thread-Safe singleton

```
public class SlowThreadSafeSingleton {
  public int theAnswer;

  // Private constructor
  private SlowThreadSafeSingleton() { theAnswer = 42; }
  // The one instance
  private static instance = null;

  synchronized public static SlowThreadSafeSingleton getInstance() {
    if (instance == null)
      instance = new SlowThreadSafeSingleton();
    return instance;
  }
}
```

Faster Thread-Safe singleton

```
public class FasterThreadSafeSingleton {
  public int volatile theAnswer;

  // Private constructor
  private FasterThreadSafeSingleton() { theAnswer = 42; }
  // The one instance
  private static instance = null;

  synchronized public static FasterThreadSafeSingleton getInstance() {
    if (instance == null) {
      synchronized (Lock.class) {
```

```
        if (instance == null) {
          instance = new FasterThreadSafeSingleton();
        }
      }
    }
    return instance;
  }
}
```