Java Checklist

wacky6 / Jiewei Qian (C) CC-BY-NC-SA-3.0

Permission hereby granted to you:
The freedom to:

- Share, redistribute this document in any media
- Modify the content of this document

Under the following conditions:

- You **MUST** give credit to original author, in a appropriate way. (eg: give a link to original document)
- You **MUST NOT** use this document for commercial purpose.
- If you modify this document, you **MUST** share under the same license.

In Addition:
You MUST NOT upload this document to any of following:

- Baidu Cloud（百度云、网盘、文库）
- 360 Cloud Disk（360 云盘）
- Sina Microblog Share（新浪微博共享）
- Thunder Network Services（迅雷快传）
- Any of document sharing services（docin 等）

This document is originally written to prepare myself for NJU-CS's Java exam.

:D

- **error vs. exception**

exception 是可控或不可控的，程序级别的错误，由程序处理
error 是不可控的，表示系统错误或底层资源错误，（比较严重的问题）程序不应该处理

- **overload vs. override**

overload 重载： 是一个方法接受多种类型的参数, eg: setSize(Dimension) / setSize(width, height)
override 覆盖： 子类的方法覆盖父类的方法，eg: Vector.get() overrides List.get()

- **abstract class vs. interface**

实现多个接口，只能继承一个类
抽象类能提供方法的实现（跟普通类一样），接口只能提供方法的声明、和常量定义(final)
抽象类是 OOP 抽象的概念（具体类的抽象），接口是类提供功能的概念（类的职责）
extends vs. implements
类可以三种访问修饰符，接口只能 public

- **heap vs. stack**

stack 比 heap 快，stack 比 heap 小，
stack 属于某个线程，heap 是进程公用的
对象是在 heap 中分配的，对象的引用是在 stack 中分配的
基础类型在 stack，类在 heap
String s = “abc” → stack　　String s = new String(“abc”) → heap
this → stack (对自身的**引用**)
static XXX → stack　　局部变量 → stack

- **final vs. finally vs. finalize**

final : 类成员修饰符，表示这个方法、成员不能被子类扩展(不能被 override)
finally : 异常处理表示最后执行的动作（无论是否有异常发生）
finalize : GC 回收之前调用，类似析构函数，用于释放非 JVM 资源（如 WIN 的 Handle）

- **HashMap vs. Hashtable**

都实现 Map 接口
父类不同： HashMap → AbstractMap, Hashtable → Dictionary
Hashtable 是同步的（线程安全的），HashMap 默认不是同步的
Hashtable 中不允许出现 null，HashMap 可以
都有 Iterator，Hashtable 还支持 Enumeration

- **Collection vs. Collections**

Collection:java.util 下的接口　　Collections:java.util 下的类(Utility Class)
Collections 提供了对[实现了 Collection 接口的类]进行操作的静态方法（比如 binarySearch）
Collection 定义了集合类（List，Vector 等）常见的接口

- **sleep(), wait(), suspend()**

sleep() : 不能被唤醒，Thread 的方法，不会释放同步锁，
wait() : 可以 notify()唤醒，Object 的方法，会释放同步锁，必须在 synchronized 块中
suspend() : 挂起线程，需要 resume()恢复，不能设置时间，现在不推荐使用

- **默认类型（字面值）**

浮点：double， 整数：int
定义 long 需要： long a = 123L; 定义 float 需要 float b=1.23F;
运算时，会升格为 int/double，short b = (short) 1*2; （强制类型转换）

- **final**

final 方法： 不能被子类覆盖(Override)，方法可以被 inline
final 类： 不能被继承

- **对象创建过程：**

父类构造函数，本体构造函数

- **内部类，匿名类：**

```
new ClassName() {
    public : void method() {}
}
```

- **动态绑定实现： 方法表**
- **String 是不可变的（immutable）**
- **对象输入输出流**

- **Applet vs. Application**

Applet：包含在 Web 页面内部，没有 main 方法，有安全限制
class MainApplet extends java.applet.Applet
init(), start(), stop(), destroy()
数据是 HTML<apple><param>标签传入的

- **StringBuffer**

每次使用都会改变自身，内存效率比 String 好，类似的：StringBuilder
StringBuffer 线程安全， StringBuilder 非线程安全， 功能一样

- **字符串比较要用.equals(), ==比较的是内存位置**
- **Graphics**
- **AWT、Swing**

awt 是使用平台底层的。Swing 是不依赖于平台的。
Swing 使用了 awt 一些核心组件
部分 swing 组件是直接依赖于 awt 的
重量级组件：依赖系统底层的
轻量级组件：不依赖于底层的

- **Serializable**
- **线程同步：**

synchronized 关键字
Object.wait / notify() 方法， 必须在 synchronized 块中调用
Lock 接口， Condition 接口
suspend() resume()