1. 熟练掌握单例模式的常见写法。

见代码

1. 总结每种单例写法的优、缺点。
2. 饿汉式

优点：原子操作，无锁化线程安全；效率高

缺点：

1. 不管类是否使用，都先加载。如果不用，则一定程度上浪费资源。
2. 反射攻击，序列化，反序列化破坏
3. 懒汉式

优点：使用的时候再初始化，节省资源

缺点：线程不安全；如果再方法上加锁，但是如果线程较多，则cpu分配压力上升，会出现大量线程阻塞，导致运行性能下降。

双重检查单例

优点：线程安全，不阻塞整个类，而是方法内阻塞。

缺点：反射攻击，序列化，反序列化破坏。由于synchronnized是排它锁，性能还是会有缺陷

1. 内部类

优点：线程安全（利用内部类的加载机制）

缺点：反射攻击，序列化，反序列化破坏

1. 注册式
2. 枚举

优点：线程安全（jdk层面防止枚举被重复初始化）

缺点：序列化，反序列化破坏

1. 容器式

优点：创建大量对象的时候，有缓存机制，方便管理

缺点： 线程不安全，因为在put的时候需要先进行判断，类似于懒汉。所以需要加锁

1. 伪单例（利用threadLocal特性，实现线程内部的线程安全）
2. 思考破坏单例模式的方式有哪些？并且归纳总结。
3. 反射：可以通过在构造函数的时候，增加判断，如果有对象了，则抛出异常阻止调用
4. 序列化/深克隆：可以通过重写readResove()方法，使用第一次初始化的对象，覆盖新生产的对象

4、梳理内部类的执行逻辑，并画出时序图

