1、redis的特点：

基于内存、单线程、IO多路复用

2、IO多路复用：

经典的Reactor设计模式，也称为异步阻塞IO，



用户首先将需要进行IO操作的socket添加到select函数中，然后阻塞等待select系统调用返回。当数据到达时，socket被激活，select函数返回。用户线程正式发起read请求，读取数据并继续执行。用户可以注册多个socket，然后不断地调用select读取被激活的socket，即可达到在同一个线程内同时处理多个IO请求的目的。

通过Reactor的方式，用户线程注册事件处理器之后可以继续执行做其他的工作（异步），而Reactor线程负责调用内核的select函数检查socket状态。当有socket被激活时，则通知相应的用户线程，执行handle\_event进行数据读取、处理的工作。由于select函数是阻塞的，因此多路IO复用模型也被称为异步阻塞IO模型。注意，这里的所说的阻塞是指select函数执行时线程被阻塞，而不是指socket。

**3 、redis单线程问题**

    单线程指的是网络请求模块使用了一个线程（所以不需考虑并发安全性），即一个线程处理所有网络请求，其他模块仍用了多个线程。采用单线程,避免了不必要的[上下文切换](https://www.baidu.com/s?wd=%E4%B8%8A%E4%B8%8B%E6%96%87%E5%88%87%E6%8D%A2&tn=24004469_oem_dg&rsv_dl=gh_pl_sl_csd)

4、redis数据结构

String、List、Set、hash、zSet

5、spring的注入方式

常用的注入方式主要有三种：构造方法注入，setter注入，基于注解的注入。

<https://blog.csdn.net/yinni11/article/details/80212614>

6、java虚拟机参数指令

-Xms20m 堆最小值20MB

-Xmx20m堆最大值20MB

-Xss128k 栈容量128KB

-XX:MaxPermSize=10M 永久代(方法区)最大值为10MB

7、类加载过程（类的生命周期）

类加载过程包括 加载、验证、准备、解析、初始化五个阶段。

加载：（三步）

a、通过类全限定名获取类的二进制字节流。

b、将该字节流所代表的静态存储结构转化为方法区的运行时数据结构。

c、在内存中生成一个代表该类的java.lang.Class对象，作为方法区这个类的访问入口。

验证：验证阶段的第一步，为了确保Class文件的字节流符合虚拟机的安全。

准备：为类变量（static修饰）分配内存并设置初值（0值）。

解析：将常量池内的符号引用替换为直接引用的过程。

符号引用: 用一组符号来描述所引用的目标，符号引用与虚拟机实现的内存布局无关，引用的目标不一定已经加载到内存中。

直接引用：可以是直接指向目标的指针、相对偏移量或句柄。直接引用和虚拟机实现的内存布局相关，如果有直接引用，那应用的目标已存在在内存中。

初始化：类加载过程的最后一步，执行类构造器<clinit>()方法的过程。

8、类加载器

启动类加载器（Bootstrap ClassLoader）:

加载<JAVA\_HOME>\lib目录中的

扩展类加载器（ExtensionClassLoader）：

加载<JAVA\_HOME>\lib\ext目录中的

应用程序类加载器（ApplicationClassLoader）

加载ClassPath所指定的类库，也称系统类加载器