什么是 Java 虚拟机？为什么 Java 被称作是“平台无关的编程语言”？

参考答案

Java 虚拟机是一个可以执行 Java 字节码的虚拟机进程。Java 源文件被编译成能被 Java 虚拟机执行的字节码文件。
Java 被设计成允许应用程序可以运行在任意的平台，而不需要程序员为每一个平台单独重写或者是重新编译。Java 虚拟机让这个变为可能，因为它知道底层硬件平台的指令长度和 其他特性。

JDK 和 JRE 的区别是什么？

参考答案

Java 运行时环境(JRE)是将要执行 Java 程序的 Java 虚拟机。它同时也包含了执行 applet 需要的浏览器插件。Java 开发工具包(JDK)是完整的 Java 软件开发包，包含了 JRE，编译器和其他的工具(比如：JavaDoc，Java 调试器)，可以让开发者开发、编译、执行 Java 应 用程序。

”static”关键字是什么意思？Java 中是否可以覆盖(override)一个 private 或者是 static 的方法？

参考答案

“static”关键字表明一个成员变量或者是成员方法可以在没有所属的类的实例变量的情况下被访问。
Java 中 static 方法不能被覆盖，因为方法覆盖是基于运行时动态绑定的，而 static 方法是编译时静态绑定的。static 方法跟类的任何实例都不相关，所以概念上不适用。

是否可以在 static 环境中访问非 static 变量？

参考答案

static 变量在 Java 中是属于类的，它在所有的实例中的值是一样的。当类被 Java 虚拟机载入的时候,会对 static 变量进行初始化.如果你的代码尝试不用实例来访问非 static 的变量，编译器会报错，因为这些变量还没有被创建出来，还没有跟任何实例关联上。

Java 支持的数据类型有哪些？什么是自动拆装箱？

参考答案

Java 语言支持的 8 种基本数据类型是：
byte
short
int
long
float
double

boolean
char
自动装箱是 Java 编译器在基本数据类型和对应的对象包装类型之间做的一个转化。比如：把 int 转化成 Integer，double 转化成 Double，等等。反之就是自动拆箱。

Java 中的方法覆盖(Overriding)和方法重载(Overloading)是什么意思？

参考答案

Java 中的方法重载发生在同一个类里面两个或者是多个方法的方法名相同但是参数不同的情况。与此相对，方法覆盖是说子类重新定义了父类的方法。方法覆盖必须有相同的方法名，参数列表和返回类型。覆盖者可能不会限制它所覆盖的方法的访问。

Java 中，什么是构造函数？什么是构造函数重载？什么是复制构造函数？

参考答案

当新对象被创建的时候，构造函数会被调用。每一个类都有构造函数。在程序员没有给类提 供构造函数的情况下，Java 编译器会为这个类创建一个默认的构造函数。
Java 中构造函数重载和方法重载很相似。可以为一个类创建多个构造函数。每一个构造函 数必须有它自己唯一的参数列表。
Java 不支持像 C++中那样的复制构造函数，这个不同点是因为如果你不自己写构造函数的情况下，Java 不会创建默认的复制构造函数。

Java 支持多继承么？

参考答案

Java 中类不支持多继承，只支持单继承（即一个类只有一个父类）。 但是 java 中的接口支持多继承，，即一个子接口可以有多个父接口。（接口的作用是用来扩展对象的功能，一个子接口继承多个父接口，说明子接口扩展了多个功能，当类实现接口时，类就扩展了相应 的功能）。

接口和抽象类的区别是什么？

参考答案

Java 提供和支持创建抽象类和接口。它们的实现有共同点，不同点在于：
接口中所有的方法隐含的都是抽象的。而抽象类则可以同时包含抽象和非抽象的方法。
类可以实现很多个接口，但是只能继承一个抽象类
类可以不实现抽象类和接口声明的所有方法，当然，在这种情况下，类也必须得声明成是抽 象的。
抽象类可以在不提供接口方法实现的情况下实现接口。
Java 接口中声明的变量默认都是 final 的。抽象类可以包含非 final 的变量。
Java 接口中的成员函数默认是 public 的。抽象类的成员函数可以是 private，protected 或者是 public。
接口是绝对抽象的，不可以被实例化。抽象类也不可以被实例化，但是，如果它包含 main 方法的话是可以被调用的。
也可以参考 JDK8 中抽象类和接口的区别

什么是值传递和引用传递？

参考答案

值传递是对基本型变量而言的,传递的是该变量的一个副本,改变副本不影响原变量.

引用传递一般是对于对象型变量而言的,传递的是该对象地址的一个副本, 并不是原对象本身 。 所以对引用对象进行操作会同时改变原对象.
一般认为,java 内的传递都是值传递.

进程和线程的区别是什么？

参考答案

进程是执行着的应用程序，而线程是进程内部的一个执行序列。一个进程可以有多个线程。 线程又叫做轻量级进程。

创建线程有几种不同的方式？你喜欢哪一种？为什么？

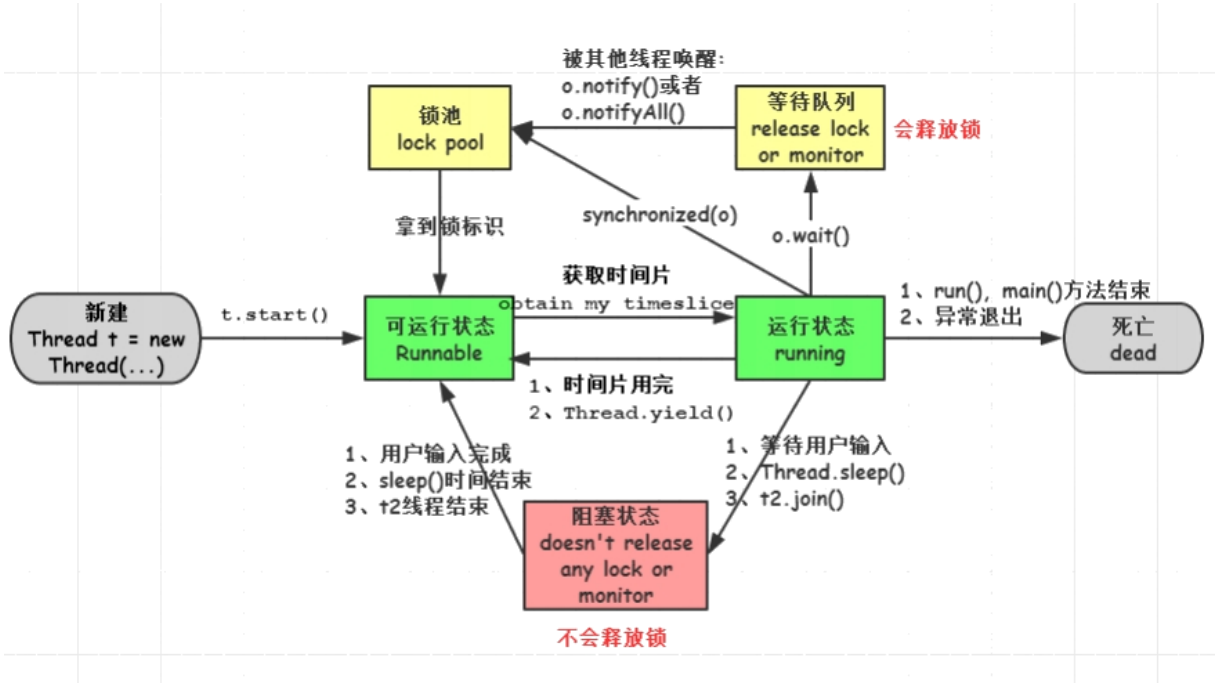
参考答案

有三种方式可以用来创建线程：
继承 Thread 类
实现 Runnable 接口
应用程序可以使用 Executor 框架来创建线程池
实现 Runnable 接口这种方式更受欢迎，因为这不需要继承 Thread 类。在应用设计中已经继承了别的对象的情况下，这需要多继承（而 Java 不支持多继承），只能实现接口。同时， 线程池也是非常高效的，很容易实现和使用。

概括的解释下线程的几种可用状态。

参考答案

1. 新建(new)：新创建了一个线程对象。
2. 可运行(runnable):线程对象创建后,其他线程(比如 main 线程)调用了该对象 的 start ()方法。该状态的线程位于可运行线程池中，等待被线程调度选中，获 取 cpu 的使用权 。 3. 运行(running)：可运行状态(runnable)的线程获得了 cpu 时间片（ timeslice ） ，执行 程序代码。
4. 阻塞(block)：阻塞状态是指线程因为某种原因放弃了 cpu 使用权，也即让出了 cpu timeslice ，暂时停止运行。直到线程进入可运行(runnable)状态，才有 机会再次获得 cpu timeslice 转到运行(running)状态。阻塞的情况分三种：
(一). 等待阻塞：运行(running)的线程执行 o . wait ()方法， JVM 会把该线程放 入等待队列(waitting queue)中。
(二). 同步阻塞：运行(running)的线程在获取对象的同步锁时，若该同步锁 被别的线程占用，则 JVM 会把该线程放入锁池(lock pool)中。
(三). 其他阻塞: 运行(running)的线程执行 Thread . sleep (long ms)或 t . join ()方法，或者发出了 I / O 请求时， JVM 会把该线程置为阻塞状态。 当 sleep ()状态超时、join ()等待线程终止或者超时、或者 I / O 处理完毕时，线程重新转入可运行(runnable)状态。 5. 死亡(dead)：线程 run ()、 main () 方法执行结束，或者因异常退出了 run ()方法，则该 线程结束生命周期。死亡的线程不可再次复生。



同步方法和同步代码块的区别是什么？

参考答案

区别：

同步方法默认用 this 或者当前类 class 对象作为锁；

同步代码块可以选择以什么来加锁，比同步方法要更细颗粒度，我们可以选择只同步会发生同步问题的部分代码而不是整个方法；

在监视器(Monitor)内部，是如何做线程同步的？程序应该做哪种级别的同步？

参考答案

监视器和锁在 Java 虚拟机中是一块使用的。监视器监视一块同步代码块，确保一次只有一个线程执行同步代码块。每一个监视器都和一个对象引用相关联。线程在获取锁之前不允许执行同步代码。

什么是死锁(deadlock)？

参考答案

两个线程或两个以上线程都在等待对方执行完毕才能继续往下执行的时候就发生了死锁。结果就是这些线程都陷入了无限的等待中。

如何确保 N 个线程可以访问 N 个资源同时又不导致死锁？

参考答案

使用多线程的时候，一种非常简单的避免死锁的方式就是：指定获取锁的顺序，并强制线程

按照指定的顺序获取锁。因此，如果所有的线程都是以同样的顺序加锁和释放锁，就不会出现死锁了。

Java 集合类框架的基本接口有哪些？

参考答案

集合类接口指定了一组叫做元素的对象。集合类接口的每一种具体的实现类都可以选择以它自己的方式对元素进行保存和排序。有的集合类允许重复的键，有些不允许。
Java 集合类提供了一套设计良好的支持对一组对象进行操作的接口和类。Java 集合类里面最基本的接口有：
Collection：代表一组对象，每一个对象都是它的子元素。
Set：不包含重复元素的 Collection。
List：有顺序的 collection，并且可以包含重复元素。
Map：可以把键(key)映射到值(value)的对象，键不能重复。

为什么集合类没有实现 Cloneable 和 Serializable 接口？

参考答案

克隆(cloning)或者是序列化(serialization)的语义和含义是跟具体的实现相关的。因此，应该由集合类的具体实现来决定如何被克隆或者是序列化。

什么是迭代器(Iterator)？

参考答案

Iterator 接口提供了很多对集合元素进行迭代的方法。每一个集合类都包含了可以返回迭代器实例的
迭代方法。迭代器可以在迭代的过程中删除底层集合的元素,但是不可以直接调用集合的 remove(Object Obj)删除，可以通过迭代器的 remove()方法删除。

Iterator 和 ListIterator 的区别是什么？

参考答案

下面列出了他们的区别：
Iterator 可用来遍历 Set 和 List 集合，但是 ListIterator 只能用来遍历 List。
Iterator 对集合只能是前向遍历，ListIterator 既可以前向也可以后向。
ListIterator 实现了 Iterator 接口，并包含其他的功能，比如：增加元素，替换元素，获取前一个和后一个元素的索引，等等。

快速失败(fail-fast)和安全失败(fail-safe)的区别是什么？

参考答案

Iterator 的安全失败是基于对底层集合做拷贝，因此，它不受源集合上修改的影响。java.util 包下面的所有的集合类都是快速失败的，而 java.util.concurrent 包下面的所有的类都是安全失败的。快速失败的迭代器会抛出 ConcurrentModificationException 异常，而安全失败的 迭代器永远不会抛出这样的异常。

Java 中的 HashMap 的工作原理是什么？

参考答案

Java 中的 HashMap 是以键值对(key-value)的形式存储元素的。HashMap 需要一个 hash 函数，它使用 hashCode()和 equals()方法来向集合/从集合添加和检索元素。当调用 put() 方法的时候，HashMap 会计算 key 的 hash 值，然后把键值对存储在集合中合适的索引上。 如果 key 已经存在了，value 会被更新成新值。HashMap 的一些重要的特性是它的容量(capacity)，负载因子(load factor)和扩容极限(threshold resizing)。

hashCode()和 equals()方法的重要性体现在什么地方？

参考答案

Java 中的 HashMap 使用 hashCode()和 equals()方法来确定键值对的索引，当根据键获取值的时候也会用到这两个方法。如果没有正确的实现这两个方法，两个不同的键可能会有相同的 hash 值，因此，可能会被集合认为是相等的。而且，这两个方法也用来发现重复元素。 所以这两个方法的实现对 HashMap 的精确性和正确性是至关重要的。

HashMap 和 Hashtable 有什么区别？

参考答案

HashMap 和 Hashtable 都实现了 Map 接口，因此很多特性非常相似。但是，他们有以下不 同点：

HashMap 允许键和值是 null，而 Hashtable 不允许键或者值是 null。

Hashtable 是同步的，而 HashMap 不是。因此，HashMap 更适合于单线程环境，而 Hashtable 适合于多线程环境。

HashMap 提供了可供应用迭代的键的集合，因此，HashMap 是快速失败的。另一方面，Hashtable 提供了对键的列举(Enumeration)。

一般认为 Hashtable 是一个遗留的类。

数组(Array)和列表(ArrayList)有什么区别？什么时候应该使用 Array 而不是 ArrayList？

参考答案

下面列出了 Array 和 ArrayList 的不同点：

Array 可以包含基本类型和对象类型，ArrayList 只能包含对象类型。

Array 大小是固定的，ArrayList 的大小是动态变化的。

ArrayList 提供了更多的方法和特性，比如：addAll()，removeAll()，iterator()等等。

对于基本类型数据，集合使用自动装箱来减少编码工作量。但是，当处理固定大小的基本数 据类型的时候，这种方式相对比较慢。

ArrayList 和 LinkedList 有什么区别？

参考答案

ArrayList 和 LinkedList 都实现了 List 接口，他们有以下的不同点：

ArrayList 是基于索引的数据接口，它的底层是数组。它可以以 O(1)时间复杂度对元素进行

随机访问。与此对应，LinkedList 是以元素列表的形式存储它的数据，每一个元素都和它的 前一个和后一个元素链接在一起，在这种情况下，查找某个元素的时间复杂度是 O(n)。
相对于 ArrayList，LinkedList 的插入，添加，删除操作速度更快，因为当元素被添加到集合 任意位置的时候，不需要像数组那样重新计算大小或者是更新索引。

LinkedList 比 ArrayList 更占内存，因为 LinkedList 为每一个节点存储了两个引用，一个指 向前一个元素，一个指向下一个元素。
也可以参考 ArrayList vs. LinkedList。

Comparable 和 Comparator 接口是干什么的？列出它们的区别。

参考答案

Java 提供了只包含一个 compareTo()方法的 Comparable 接口。这个方法可以个给两个对象排序。具体来说，它返回负数，0，正数来表明输入对象小于，等于，大于已经存在的对 象。Java 提供了包含 compare()和 equals()两个方法的 Comparator 接口。compare()方法用来给两个输入参数排序，返回负数，0，正数表明第一个参数是小于，等于，大于第二个参数。equals()方法需要一个对象作为参数，它用来决定输入参数是否和 comparator 相等。只有当输入参数也是一个 comparator 并且输入参数和当前 comparator 的排序结果是相同的时候，这个方法才返回 true。

什么是 Java 优先级队列(Priority Queue)？

参考答案

PriorityQueue 是一个基于优先级堆的无界队列，它的元素是按照自然顺序(natural order)排序的。在创建的时候，我们可以给它提供一个负责给元素排序的比较器。PriorityQueue 不 允许 null 值，因为他们没有自然顺序，或者说他们没有任何的相关联的比较器。最后，PriorityQueue 不是线程安全的，入队和出队的时间复杂度是 O(log(n))。

你了解大 O 符号(big-Ǒ notation)么？你能给出不同数据结构的例子么？

参考答案

大 O 符号描述了当数据结构里面的元素增加的时候，算法的规模或者是性能在最坏的场景下有多么好。
大 O 符号也可用来描述其他的行为，比如：内存消耗。因为集合类实际上是数据结构，我 们一般使用大 O 符号基于时间，内存和性能来选择最好的实现。大 O 符号可以对大量数据 的性能给出一个很好的说明。

如何权衡是使用无序的数组还是有序的数组？

参考答案

有序数组最大的好处在于查找的时间复杂度是 O(log n)，而无序数组是 O(n)。有序数组的缺点是插入操作的时间复杂度是 O(n)，因为值大的元素需要往后移动来给新元素腾位置。相反，无序数组的插入时间复杂度是常量 O(1)。

Java 集合类框架的最佳实践有哪些？

参考答案

根据应用的需要正确选择要使用的集合的类型对性能非常重要，比如：假如元素的大小是固

定的，而且能事先知道，我们就应该用 Array 而不是 ArrayList。
有些集合类允许指定初始容量。因此，如果我们能估计出存储的元素的数目，我们可以设置 初始容量来避免重新计算 hash 值或者是扩容。

为了类型安全，可读性和健壮性的原因总是要使用泛型。同时，使用泛型还可以避免运行时的 ClassCastException。
使用 JDK 提供的不变类(immutable class)作为 Map 的键可以避免为我们自己的类实现 hashCode()和 equals()方法。
编程的时候接口优于实现。
底层的集合实际上是空的情况下，返回长度是 0 的集合或者是数组，不要返回 null。

Enumeration 接口和 Iterator 接口的区别有哪些？

参考答案

Enumeration 速度是 Iterator 的 2 倍，同时占用更少的内存。但是，Iterator 远远比Enumeration 安全，因为其他线程不能够修改正在被 iterator 遍历的集合里面的对象。同时， Iterator 允许调用者删除底层集合里面的元素，这对 Enumeration 来说是不可能的。

HashSet 和 TreeSet 有什么区别？

参考答案

HashSet 是由一个 hash 表来实现的，因此，它的元素是无序的。add()，remove()，contains()方法的时间复杂度是 O(1)。
另一方面，TreeSet 是由一个树形的结构来实现的，它里面的元素是有序的。因此，add()，remove()，contains()方法的时间复杂度是 O(logn)。

Java 中垃圾回收有什么目的？什么时候进行垃圾回收？

参考答案

垃圾回收的目的是识别并且丢弃应用不再使用的对象来释放和重用资源。

System.gc()和 Runtime.gc()会做什么事情？

参考答案

这两个方法用来提示 JVM 要进行垃圾回收。但是，立即开始还是延迟进行垃圾回收是取决于 JVM 的。

finalize()方法什么时候被调用？析构函数(finalization)的目的是什么？

参考答案

垃圾回收器(garbage colector)决定回收某对象时，就会运行该对象的 finalize()方法 但是在 Java 中很不幸，如果内存总是充足的，那么垃圾回收可能永远不会进行，也就是说 filalize() 可能永远不被执行，显然指望它做收尾工作是靠不住的。 那么 finalize()究竟是做什么的呢？ 它最主要的用途是回收特殊渠道申请的内存。Java 程序有垃圾回收器，所以一般情况下内存问题不用程序员操心。但有一种 JNI(Java Native Interface)调用 non-Java 程序（C 或 C++），finalize()的工作就是回收这部分的内存。

如果对象的引用被置为 null，垃圾收集器是否会立即释放对象占用的内存？

参考答案

不会，在下一个垃圾回收周期中，这个对象将是可被回收的。

Java 堆的结构是什么样子的？什么是堆中的永久代(Perm Gen space)?

参考答案

JVM 的堆是运行时数据区，所有类的实例和数组都是在堆上分配内存。它在 JVM 启动的时候被创建。对象所占的堆内存是由自动内存管理系统也就是垃圾收集器回收。
堆内存是由存活和死亡的对象组成的。存活的对象是应用可以访问的，不会被垃圾回收。死亡的对象是应用不可访问尚且还没有被垃圾收集器回收掉的对象。一直到垃圾收集器把这些对象回收掉之前，他们会一直占据堆内存空间。

串行(serial)收集器和吞吐量(throughput)收集器的区别是什么？

参考答案

吞吐量收集器使用并行版本的新生代垃圾收集器，它用于中等规模和大规模数据的应用程序。而串行收集器对大多数的小应用(在现代处理器上需要大概 100M 左右的内存)就足够了。

在 Java 中，对象什么时候可以被垃圾回收？

参考答案

当对象对当前使用这个对象的应用程序变得不可触及的时候，这个对象就可以被回收了。

JVM 的永久代中会发生垃圾回收么？

参考答案

垃圾回收不会发生在永久代，如果永久代满了或者是超过了临界值，会触发完全垃圾回收(Full GC)。如果你仔细查看垃圾收集器的输出信息，就会发现永久代也是被回收的。这就是为什么正确的永久代大小对避免 Full GC 是非常重要的原因。请参考下 Java8：从永久代 到元数据区
(注：Java8 中已经移除了永久代，新加了一个叫做元数据区的 native 内存区)

Java 中的两种异常类型是什么？他们有什么区别？

参考答案

Java 中有两种异常：受检查的(checked)异常和不受检查的(unchecked)异常。不受检查的异常不需要在方法或者是构造函数上声明，就算方法或者是构造函数的执行可能会抛出这样的异常，并且不受检查的异常可以传播到方法或者是构造函数的外面。相反，受检查的异常必须要用 throws 语句在方法或者是构造函数上声明。这里有 Java 异常处理的一些小建议。

Java 中 Exception 和 Error 有什么区别？

参考答案

Exception 和 Error 都是 Throwable 的子类。Exception 用于用户程序可以捕获的异常情况。Error 定义了不期望被用户程序捕获的异常。

throw 和 throws 有什么区别？

参考答案

throw 关键字用来在程序中明确的抛出异常，相反，throws 语句用来表明方法不能处理的异常。每一个方法都必须要指定哪些异常不能处理，所以方法的调用者才能够确保处理可能发生的异常，多个异常是用逗号分隔的。

异常处理完成以后，Exception 对象会发生什么变化？

参考答案

Exception 对象会在下一个垃圾回收过程中被回收掉。

finally 代码块和 finalize()方法有什么区别？

参考答案

无论是否抛出异常，finally 代码块都会执行，它主要是用来释放应用占用的资源。finalize()方法是 Object 类的一个 protected 方法，它是在对象被垃圾回收之前由 Java 虚拟机来调用的。

什么是 JDBC？

参考答案

JDBC 是允许用户在不同数据库之间做选择的一个抽象层。JDBC 允许开发者用 JAVA 写数据库应用程序，而不需要关心底层特定数据库的细节。

解释下驱动(Driver)在 JDBC 中的角色。

参考答案

JDBC 驱动提供了特定厂商对 JDBC API 接口类的实现，驱动必须要提供 java.sql 包下面这些类的实现：Connection, Statement, PreparedStatement,CallableStatement, ResultSet 和 Driver。

Class.forName()方法有什么作用？

参考答案

初始化参数指定的类，并且返回此类对应的 Class 对象

PreparedStatement 比 Statement 有什么优势？

参考答案

PreparedStatements 是预编译的，因此，性能会更好。同时，不同的查询参数值，PreparedStatement 可以重用。

什么时候使用 CallableStatement？用来准备 CallableStatement 的方法是什么？

参考答案

CallableStatement 用来执行存储过程。存储过程是由数据库存储和提供的。存储过程可以接受输入参数，也可以有返回结果。非常鼓励使用存储过程，因为它提供了安全性和模块化。 准备一个 CallableStatement 的方法是：
CallableStament.prepareCall();;

数据库连接池是什么意思？

参考答案

像打开关闭数据库连接这种和数据库的交互可能是很费时的，尤其是当客户端数量增加的时候，会消耗大量的资源，成本是非常高的。可以在应用服务器启动的时候建立很多个数据库连接并维护在一个池中。连接请求由池中的连接提供。在连接使用完毕以后，把连接归还到 池中，以用于满足将来更多的请求。

什么是 RMI？

参考答案

Java 远程方法调用(Java RMI)是 Java API 对远程过程调用(RPC)提供的面向对象的等价形式，支持直接传输序列化的 Java 对象和分布式垃圾回收。远程方法调用可以看做是激活远程正在运行的对象上的方法的步骤。RMI 对调用者是位置透明的，因为调用者感觉方法是 执行在本地运行的对象上的。看下 RMI 的一些注意事项。

什么是分布式垃圾回收(DGC)？它是如何工作的？

参考答案

DGC 叫做分布式垃圾回收。RMI 使用 DGC 来做自动垃圾回收。因为 RMI 包含了跨虚拟机的远程对象的引用，垃圾回收是很困难的。DGC 使用引用计数算法来给远程对象提供自动 内存管理。

什么是分布式垃圾回收(DGC)？它是如何工作的？

参考答案

DGC 叫做分布式垃圾回收。RMI 使用 DGC 来做自动垃圾回收。因为 RMI 包含了跨虚拟机的远程对象的引用，垃圾回收是很困难的。DGC 使用引用计数算法来给远程对象提供自动 内存管理。

RMI 中使用 RMI 安全管理器(RMISecurityManager)的目的是什么？

参考答案

RMISecurityManager 使用下载好的代码提供可被 RMI 应用程序使用的安全管理器。如果没有设置安全管理器，RMI 的类加载器就不会从远程下载任何的类。

解释下 Marshalling 和 demarshalling。

参考答案

当应用程序希望把内存对象跨网络传递到另一台主机或者是持久化到存储的时候，就必须要把对象在内存里面的表示转化成合适的格式。这个过程就叫做 Marshalling，反之就是 demarshalling。

解释下 Serialization 和 Deserialization。

参考答案

Java 提供了一种叫做对象序列化的机制，他把对象表示成一连串的字节，里面包含了对象的数据，对象的类型信息，对象内部的数据的类型信息等等。因此，序列化可以看成是为了把对象存储在磁盘上或者是从磁盘上读出来并重建对象而把对象扁平化的一种方式。反序列化是把对象从扁平状态转化成活动对象的相反的步骤。

Servlet

什么是 Servlet?

参考答案

Servlet 是用来处理客户端请求并产生动态网页内容的 Java 类。Servlet 主要是用来处理或者是存储 HTML 表单提交的数据，产生动态内容，在无状态的 HTTP 协议下管理状态信息。

说一下 Servlet 的体系结构。

参考答案

所有的 Servlet 都必须要实现的核心的接口是 javax.servlet.Servlet。每一个 Servlet 都必须要直接或者是间接实现这个接口，或者是继承 javax.servlet.GenericServlet 或者 javax.servlet.http.HTTPServlet。最后，Servlet 使用多线程可以并行的为多个请求服务。

GenericServlet 和 HttpServlet 有什么区别？

参考答案

GenericServlet 是一个通用的协议无关的 Servlet，它实现了 Servlet 和 ServletConfig 接口。继承自 GenericServlet 的 Servlet 应该要覆盖 service()方法。最后，为了开发一个能用在网页上服务于使用 HTTP 协议请求的 Servlet，你的 Servlet 必须要继承自 HttpServlet。这里有 Servlet 的例子。

解释下 Servlet 的生命周期。

参考答案

对每一个客户端的请求，Servlet 引擎载入 Servlet，调用它的 init()方法，完成 Servlet 的初始化。然后，Servlet 对象通过为每一个请求单独调用 service()方法来处理所有随后来自客 户端的请求，最后，调用 Servlet(译者注：这里应该是 Servlet 而不是 server)的 destroy() 方法把 Servlet 删除掉。

doGet()方法和 doPost()方法有什么区别？

参考答案

doGet：GET 方法会把名值对追加在请求的 URL 后面。因为 URL 对字符数目有限制，进而限制了用在客户端请求的参数值的数目。并且请求中的参数值是可见的，因此，敏感信息不 能用这种方式传递。
doPOST：POST 方法通过把请求参数值放在请求体中来克服 GET 方法的限制，因此，可以发送的参数的数目是没有限制的。最后，通过 POST 请求传递的敏感信息对外部客户端 是不可见的。

什么是 Web 应用程序？

参考答案

Web 应用程序是对 Web 或者是应用服务器的动态扩展。有两种类型的 Web 应用：面向表现的和面向服务的。面向表现的 Web 应用程序会产生包含了很多种标记语言和动态内容的交互的 web 页面作为对请求的响应。而面向服务的 Web 应用实现了 Web 服务的端点(endpoint)。一般来说，一个 Web 应用可以看成是一组安装在服务器 URL 名称空间的特定 子集下面的 Servlet 的集合。

什么是服务端包含(Server Side Include)？

参考答案

服务端包含(SSI)是一种简单的解释型服务端脚本语言，大多数时候仅用在 Web 上，用 servlet 标签嵌入进来。SSI 最常用的场景把一个或多个文件包含到 Web 服务器的一个 Web

页面中。当浏览器访问 Web 页面的时候，Web 服务器会用对应的 servlet 产生的文本来替换 Web 页面中的 servlet 标签。

什么是 Servlet 链(Servlet Chaining)？

参考答案

Servlet 链是把一个 Servlet 的输出发送给另一个 Servlet 的方法。第二个 Servlet 的输出可以发送给第三个 Servlet，依次类推。链条上最后一个 Servlet 负责把响应发送给客户端。

如何知道是哪一个客户端的机器正在请求你的 Servlet？

参考答案

ServletRequest 类可以找出客户端机器的 IP 地址或者是主机名。getRemoteAddr()方法获取客户端主机的 IP 地址，getRemoteHost()可以获取主机名。看下这里的例子。

HTTP 响应的结构是怎么样的？

参考答案

HTTP 响应由三个部分组成：
状态码(Status Code)：描述了响应的状态。可以用来检查是否成功的完成了请求。请求失败的情况下，状态码可用来找出失败的原因。如果 Servlet 没有返回状态码，默认会返回成功的状态码 HttpServletResponse.SC_OK。
HTTP 头部(HTTP Header)：它们包含了更多关于响应的信息。比如：头部可以指定认为响应过期的过期日期，或者是指定用来给用户安全的传输实体内容的编码格式。如何在 Serlet 中检索 HTTP 的头部看这里。
主体(Body)：它包含了响应的内容。它可以包含 HTML 代码，图片，等等。主体是由传输在 HTTP 消息中紧跟在头部后面的数据字节组成的。

什么是 cookie？session 和 cookie 有什么区别？

参考答案

cookie 是 Web 服务器发送给浏览器的一块信息。浏览器会在本地文件中给每一个 Web 服务器存储 cookie。以后浏览器在给特定的 Web 服务器发请求的时候，同时会发送所有为该服务器存储的 cookie。下面列出了 session 和 cookie 的区别：
无论客户端浏览器做怎么样的设置，session 都应该能正常工作。客户端可以选择禁用 cookie，但是，session 仍然是能够工作的，因为客户端无法禁用服务端的 session。
在存储的数据量方面 session 和 cookies 也是不一样的。session 能够存储任意的 Java 对象，cookie 只能存储 String 类型的对象。

浏览器和 Servlet 通信使用的是什么协议？

参考答案

浏览器和 Servlet 通信使用的是 HTTP 协议。

什么是 HTTP 隧道？

参考答案

HTTP 隧道是一种利用 HTTP 或者是 HTTPS 把多种网络协议封装起来进行通信的技术。因此，HTTP 协议扮演了一个打通用于通信的网络协议的管道的包装器的角色。把其他协议的请求掩盖成 HTTP 的请求就是 HTTP 隧道。

sendRedirect()和 forward()方法有什么区别？

参考答案

sendRedirect()方法会创建一个新的请求，而 forward()方法只是把请求转发到一个新的目标上。重定向(redirect)以后，之前请求作用域范围以内的对象就失效了，因为会产生一个新的请求，而转发(forwarding)以后，之前请求作用域范围以内的对象还是能访问的。一般认为 sendRedirect()比 forward()要慢。

什么是 URL 编码和 URL 解码？

参考答案

URL 编码是负责把 URL 里面的空格和其他的特殊字符替换成对应的十六进制表示，反之就是解码。

JSP 请求是如何被处理的？

参考答案

浏览器首先要请求一个以.jsp 扩展名结尾的页面，发起 JSP 请求，然后，Web 服务器读取这个请求，使用 JSP 编译器把 JSP 页面转化成一个 Servlet 类。需要注意的是，只有当第 一次请求页面或者是 JSP 文件发生改变的时候 JSP 文件才会被编译，然后服务器调用servlet 类，处理浏览器的请求。一旦请求执行结束，servlet 会把响应发送给客户端。这里 看下如何在 JSP 中获取请求参数。

什么是 JSP 指令(Directive)？JSP 中有哪些不同类型的指令？

参考答案

Directive 是当 JSP 页面被编译成 Servlet 的时候，JSP 引擎要处理的指令。Directive 用来设置页面级别的指令，从外部文件插入数据，指定自定义的标签库。Directive 是定义在 <%@ 和 %>之间的。下面列出了不同类型的 Directive：
包含指令(Include directive)：用来包含文件和合并文件内容到当前的页面。
页面指令(Page directive)：用来定义 JSP 页面中特定的属性，比如错误页面和缓冲区。
Taglib 指令： 用来声明页面中使用的自定义的标签库。

什么是 JSP 动作(JSP action)？

参考答案

JSP 动作以 XML 语法的结构来控制 Servlet 引擎的行为。当 JSP 页面被请求的时候，JSP 动作会被执行。它们可以被动态的插入到文件中，重用 JavaBean 组件，转发用户到其他的 页面，或者是给 Java 插件产生 HTML 代码。下面列出了可用的动作：
jsp:include-当 JSP 页面被请求的时候包含一个文件。
jsp:useBean-找出或者是初始化 Javabean。
jsp:setProperty-设置 JavaBean 的属性。
jsp:getProperty-获取 JavaBean 的属性。
jsp:forward-把请求转发到新的页面。
jsp:plugin-产生特定浏览器的代码。

什么是表达式(Expression)？

参考答案

【列表很长，可以分上、中、下发布】
JSP 表达式是 Web 服务器把脚本语言表达式的值转化成一个 String 对象，插入到返回给客户端的数据流中。表达式是在<%=和%>这两个标签之间定义的。

隐含对象是什么意思？有哪些隐含对象？

参考答案

JSP 隐含对象是页面中的一些 Java 对象，JSP 容器让这些 Java 对象可以为开发者所使用。开发者不用明确的声明就可以直接使用他们。JSP 隐含对象也叫做预定义变量。下面列出 了 JSP 页面中的隐含对象：
application
page

request
response
session
exception
out
config
pageContext

面向对象软件开发的优点有哪些？

参考答案

代码开发模块化，更易维护和修改。
代码复用。
增强代码的可靠性和灵活性。
增加代码的可理解性。
面向对象编程有很多重要的特性，比如：封装，继承，多态和抽象。下面的章节我们会逐个 分析这些特性。

封装的定义和好处有哪些？

参考答案

封装给对象提供了隐藏内部特性和行为的能力。对象提供一些能被其他对象访问的方法来改变它内部的数据。在 Java 当中，有 3 种修饰符：public，private 和 protected。每一种修饰符给其他的位于同一个包或者不同包下面对象赋予了不同的访问权限。
下面列出了使用封装的一些好处：
通过隐藏对象的属性来保护对象内部的状态。
提高了代码的可用性和可维护性，因为对象的行为可以被单独的改变或者是扩展。
禁止对象之间的不良交互提高模块化。
参考这个文档获取更多关于封装的细节和示例。

多态的定义？

参考答案

多态是编程语言给不同的底层数据类型做相同的接口展示的一种能力。一个多态类型上的操作可以应用到其他类型的值上面。

继承的定义？

参考答案

继承给对象提供了从基类获取字段和方法的能力。继承提供了代码的重用行，也可以在不修改类的情况下给现存的类添加新特性。

抽象的定义？抽象和封装的不同点？

参考答案

抽象是把想法从具体的实例中分离出来的步骤，因此，要根据他们的功能而不是实现细节来创建类。Java 支持创建只暴漏接口而不包含方法实现的抽象的类。这种抽象技术的主要目

的是把类的行为和实现细节分离开。
抽象和封装是互补的概念。一方面，抽象关注对象的行为。另一方面，封装关注对象行为的细节。一般是通过隐藏对象内部状态信息做到封装，因此，封装可以看成是用来提供抽象的 一种策略。

更多请关注微信公众号：Java技术栈