一：面向对象：

1.面向对象编程（OOP）

Java 是一个支持并发、基于类和面向对象的计算机编程语言。下面列出了面向对象软件开发的优点：

- 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

封装给对象提供了隐藏内部特性和行为的能力，并对外提供访问方式。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。

继承

继承给对象提供了从父类获取属性和方法的能力，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。

多态

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。

抽象

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。

抽象和封装的不同点

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。

2.什么是 Java 虚拟机？为什么 Java 被称作是"平台无关的编程语言"？

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。

2.JDK 和 JRE 的区别是什么？

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。

3."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？

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。

4.是否可以在 static 环境中访问非 static 变量？

答案是不可以，因为 static 变量是属于类的，在类加载的时候就被初始化了，这时候非静态变量并没有加载，故静态变量不能访问。

5.Java 支持的数据类型有哪些？什么是自动拆装箱？

Java 语言支持的 8 中基本数据类型是：

- byte
- short
- int
- long
- float
- double
- boolean
- char

还包括引用类数据类型，如 数组，类 ，接口

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。

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

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。

7.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？

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。

8.Java 支持多继承么？

不支持，Java 不支持多继承，这也是 java 中的一个弊端。每个类都只能继承一个类，但是可以实现多个接口，接口避免了单继承的局限性。

9.接口和抽象类的区别是什么？

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
- 类如果要实现一个接口，它必须要实现接口声明的所有方法。但是，类可以不实现抽象类声明的所有方法，当然，在这种情况下，类也必须得声明成是抽象的。
- 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。
- 若无特殊需求优先使用接口

10.什么是值传递和引用传递？

对象被值传递，意味着传递了对象的一个副本。因此，就算是改变了对象副本，也不会影响源对象的值。

对象被引用传递，意味着传递的并不是实际的对象，而是对象的引用。因此，外部对引用对象所做的改变会反映到所有的对象上。

103）接口是什么？为什么要使用接口而不是直接使用具体类？
接口用于定义 API。它定义了类必须得遵循的规则。同时，它提供了一种抽象，因为客户端只使用接口，这样可以有多重实现，如 List 接口，你可以使用可随机访问的 ArrayList，也可以使用方便插入和删除的 LinkedList。接口中不允许写代码，以此来保证抽象，但是 Java 8 中你可以在接口声明静态的默认方法，这种方法是具体的。

Java 中，抽象类与接口之间有什么不同？(答案)

除了单例模式，你在生产环境中还用过什么设计模式？
这需要根据你的经验来回答。一般情况下，你可以说依赖注入，工厂模式，装饰模式或者观察者模式，随意选择你使用过的一种即可。不过你要准备回答接下的基于你选择的模式的问题。

你能解释一下里氏替换原则吗?(答案)

二：线程

11.进程和线程的区别是什么？

进程是执行着的应用程序，而线程是进程内部的一个执行单元，一个进程至少有一个线程，线程又叫做轻量级进程。

12.创建线程有几种不同的方式？你喜欢哪一种？为什么？

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，避免了 java 中单继承的局限性，同时又将线程任务与线程对象分离开来，实现了解耦。

同时，线程池也是非常高效的，线程池解决了线程生命周期开销问题和资源不足问题，将线程创建的代价分配到多个任务上，同时当开启线程时线程已经被创建，降低了线程的延迟，提升了资源利用的效率。

13.概括的解释下线程的几种可用状态。

线程在执行过程中，可以处于下面几种状态：

- 就绪(Runnable):线程准备运行，不一定立马就能开始执行。
- 运行中(Running)：进程正在执行线程的代码。
- 等待中(Waiting):线程处于阻塞的状态，等待外部的处理结束。
- 睡眠中(Sleeping)：线程被强制睡眠。
- I/O 阻塞(Blocked on I/O)：等待 I/O 操作完成。
- 同步阻塞(Blocked on Synchronization)：等待获取锁。
- 死亡(Dead)：线程完成了执行。

14.同步方法和同步代码块的区别是什么？

在 Java 语言中，每一个对象有一把锁。线程可以使用 synchronized 关键字来获取对象上的锁。synchronized 关键字可应用在方法级别(粗粒度锁)或者是代码块级别(细粒度锁)。

16.什么是死锁(deadlock)？如何避免死锁？

在多线程运行的情况下，由于多个线程因资源的相互竞争与互相的通信阻塞而导致多个线程处于相互等待的状态，若无外力干预将无限的持续下去就发生了死锁

1. 设置获取锁的顺序
2. 设置加锁时限
3. 死锁检测

14）Java 中 sleep 方法和 wait 方法的区别?
虽然两者都是用来暂停当前运行的线程，但是 sleep() 实际上只是短暂停顿，因为它不会释放锁，而 wait() 意味着条件等待，这就是为什么该方法要释放锁，因为只有这样，其他等待的线程才能在满足条件时获取到该锁。

三：集合

18.Java 集合类框架的基本接口有哪些？

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。

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

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。

20.什么是迭代器(Iterator)?

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。

21.Iterator 和 ListIterator 的区别是什么？

- Iterator 可用来遍历很多集合，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

3.Java 中的 HashMap 的工作原理是什么？

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)。

利用 key 的 hashCode 重新 hash 计算出当前对象的元素在数组中的下标
存储时，如果出现 hash 值相同的 key，此时有两种情况。(1)如果 key 相同，则覆盖原始值；(2)如果 key 不同（出现冲突），则将当前的 key-value 放入链表中
获取时，直接找到 hash 值对应的下标，在进一步判断 key 是否相同，从而找到对应值。
理解了以上过程就不难明白 HashMap 是如何解决 hash 冲突的问题，核心就是使用了数组的存储方式，然后将冲突的 key 的对象放入链表中，一旦发现冲突就在链表中做进一步的对比。

89.HashMap 中链表长度大于 8 时，会怎么样（优化 hashMap）

HashMap 在 JDK1.8 及以后的版本中引入了红黑树结构，若桶中链表元素个数大于等于 8 时，链表转换成树结构；若桶中链表元素个数小于等于 6 时，树结构还原成链表。因为红黑树的平均查找长度是 log(n)，长度为 8 的时候，平均查找长度为 3，如果继续使用链表，平均查找长度为 8/2=4，这才有转换为树的必要。链表长度如果是小于等于 6，6/2=3，虽然速度也很快的，但是转化为树结构和生成树的时间并不会太短。
还有选择 6 和 8，中间有个差值 7 可以有效防止链表和树频繁转换。假设一下，如果设计成链表个数超过 8 则链表转换成树结构，链表个数小于 8 则树结构转换成链表，如果一个 HashMap 不停的插入、删除元素，链表个数在 8 左右徘徊，就会频繁的发生树转链表、链表转树，效率会很低

6.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？

- 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。

27.ArrayList 和 LinkedList 有什么区别？

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。

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

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。

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

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。

32.Java 集合类框架的最佳实践有哪些？

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。

33.Enumeration 接口和 Iterator 接口的区别有哪些？

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。

34.HashSet 和 TreeSet 有什么区别？

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)。

35.Java 中 java.util.Date 与 java.sql.Date 有什么区别？

java.util.Date 是在除了 SQL 语句的情况下面使用的。java.sql.Date 是 java.util.Date 的子类，针对 SQL 语句使用的，它只包含日期而没有时间部分，对应数据库中的 Date 类型

49）poll() 方法和 remove() 方法的区别？
poll() 和 remove() 都是从队列中取出一个元素，但是 poll() 在获取元素失败的时候会返回空，但是 remove() 失败的时候会抛出异常。

51）ArrayList 与 LinkedList 的区别？(答案)
最明显的区别是 ArrrayList 底层的数据结构是数组，支持随机访问，而 LinkedList 的底层数据结构书链表，不支持随机访问。使用下标访问一个元素，ArrayList 的时间复杂度是 O(1)，而 LinkedList 是 O(n)。更多细节的讨论参见答案。

用哪两种方式来实现集合的排序？(答案)
你可以使用有序集合，如 TreeSet 或 TreeMap，你也可以使用有顺序的的集合，如 list，然后通过 Collections.sort() 来排序。

Java 中怎么打印数组？(answer 答案)
你可以使用 Arrays.toString() 和 Arrays.deepToString() 方法来打印数组。由于数组没有实现 toString() 方法，所以如果将数组传递给 System.out.println() 方法，将无法打印出数组的内容，但是 Arrays.toString() 可以打印每个元素。

Java 中的 LinkedList 是单向链表还是双向链表？(答案)
是双向链表，你可以检查 JDK 的源码。在 Eclipse，你可以使用快捷键 Ctrl + T，直接在编辑器中打开该类

55）Java 中的 TreeMap 是采用什么树实现的？(答案)
Java 中的 TreeMap 是使用红黑树实现的。

58）写一段代码在遍历 ArrayList 时移除一个元素？(答案)
该问题的关键在于面试者使用的是 ArrayList 的 remove() 还是 Iterator 的 remove()方法。这有一段示例代码，是使用正确的方式来实现在遍历的过程中移除元素，而不会出现 ConcurrentModificationException 异常的示例代码。

59）我们能自己写一个容器类，然后使用 for-each 循环码？
可以，你可以写一个自己的容器类。如果你想使用 Java 中增强的循环来遍历，你只需要实现 Iterable 接口。如果你实现 Collection 接口，默认就具有该属性。

ArrayList 和 HashMap 的默认大小是多数？(答案)

在 Java 7 中，ArrayList 的默认大小是 10 个元素，HashMap 的默认大小是 16 个元素（必须是 2 的幂）。这就是 Java 7 中 ArrayList 和 HashMap 类的代码片段：

61）有没有可能两个不相等的对象有有相同的 hashcode？
有可能，两个不相等的对象可能会有相同的 hashcode 值，这就是为什么在 hashmap 中会有冲突。相等 hashcode 值的规定只是说如果两个对象相等，必须有相同的 hashcode 值，但是没有关于不相等对象的任何规定。

2）两个相同的对象会有不同的的 hash code 吗？
不能，根据 hash code 的规定，这是不可能的。

63）我们可以在 hashcode() 中使用随机数字吗？(答案)
不行，因为对象的 hashcode 值必须是相同的。参见答案获取更多关于 Java 中重写 hashCode() 方法的知识。

Java 中，Comparator 与 Comparable 有什么不同？(答案)
Comparable 接口用于定义对象的自然顺序，而 comparator 通常用于定义用户定制的顺序。Comparable 总是只有一个，但是可以有多个 comparator 来定义对象的顺序。

65）为什么在重写 equals 方法的时候需要重写 hashCode 方法？(答案)
因为有强制的规范指定需要同时重写 hashcode 与 equal 是方法，许多容器类，如 HashMap、HashSet 都依赖于 hashcode 与 equals 的规定。

13、Set 里的元素是不能重复的，那么用什么方法来区分重复与否呢?是用==还是 equals()?它们有何区别?
Set 里的元素是不能重复的，元素重复与否是使用 equals()方法进行判断的。
equals()和==方法决定引用值是否指向同一对象 equals()在类中被覆盖，为的是当两个分离的对象的内容和类型相配的话，返回真值。
14、两个对象值相同(x.equals(y) == true)，但却可有不同的 hash code，这句话对不对?
对。
如果对象要保存在 HashSet 或 HashMap 中，它们的 equals 相等，那么，它们的 hashcode 值就必须相等。
如果不是要保存在 HashSet 或 HashMap，则与 hashcode 没有什么关系了，这时候 hashcode 不等是可以的，例如 arrayList 存储的对象就不用实现 hashcode，当然，我们没有理由不实现，通常都会去实现的。

四：异常

43.Java 中的两种异常类型是什么？他们有什么区别？

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。

44.Java 中 Exception 和 Error 有什么区别？

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。

45.throw 和 throws 有什么区别？

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。

46.异常处理的时候，finally 代码块的重要性是什么？

无论是否抛出异常，finally 代码块总是会被执行。就算是没有 catch 语句同时又抛出异常的情况下，finally 代码块仍然会被执行。最后要说的是，finally 代码块主要用来释放资源，比如：I/O 缓冲区，数据库连接。

47.finally 代码块和 finalize()方法有什么区别？

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

五：JDBC

72.什么是 JDBC？

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。

73.解释下驱动(Driver)在 JDBC 中的角色。

JDBC 是一套规范(通俗的说就是一套接口),它本身是没有针对具体数据库操作提供实现的，所以谁要让自家的数据库适配 JDBC,就实现这套接口即可。　　一般叫实现 JDBC 规范的 jar 包为 JDBC 驱动(我们通过 JDBC 接口执行数据库操作命令,具体让数据库操作的是驱动程序

74.Class.forName()方法有什么作用？

这个方法用来载入跟数据库建立连接的驱动。

75.PreparedStatement 比 Statement 有什么优势？

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。

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

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：prepareCall

77.数据库连接池是什么意思？

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。

六：Java 基础：

23）Java 中 ++ 操作符是线程安全的吗？(答案)
不是线程安全的操作。它涉及到多个指令，如读取变量值，增加，然后存储回内存，这个过程可能会出现多个线程交差。

我能在不进行强制转换的情况下将一个 double 值赋值给 long 类型的变量吗？(答案)
不行，你不能在没有强制类型转换的前提下将一个 double 值赋值给 long 类型的变量，因为 double 类型的范围比 long 类型更广，所以必须要进行强制转换。

24）a = a + b 与 a += b 的区别(答案)
+= 隐式的将加操作的结果类型强制转换为持有结果的类型。如果两这个整型相加，如 byte、short 或者 int，首先会将它们提升到 int 类型，然后在执行加法操作。如果加法操作的结果比 a 的最大值要大，则 a+b 会出现编译错误，但是 a += b 没问题，如下：

26）3*0.1 == 0.3 将会返回什么？true 还是 false？(答案)
false，因为有些浮点数不能完全精确的表示出来。

27）int 和 Integer 哪个会占用更多的内存？(答案)
Integer 对象会占用更多的内存。Integer 是一个对象，需要存储对象的元数据。但是 int 是一个原始类型的数据，所以占用的空间更少。

为什么 Java 中的 String 是不可变的（Immutable）？(answer 答案)
Java 中的 String 不可变是因为 Java 的设计者认为字符串使用非常频繁，将字符串设置为不可变可以允许多个客户端之间共享相同的字符串。更详细的内容参见答案。

Java 中的构造器链是什么？(answer 答案)
当你从一个构造器中调用另一个构造器，就是 Java 中的构造器链。这种情况只在重载了类的构造器的时候才会出现。

"a==b"和"a.equals(b)"有什么区别？(答案)
如果 a 和 b 都是对象，则 a==b 是比较两个对象的引用，只有当 a 和 b 指向的是堆中的同一个对象才会返回 true，而 a.equals(b) 是进行逻辑比较，所以通常需要重写该方法来提供逻辑一致性的比较。例如，String 类重写 equals() 方法，所以可以用于两个不同对象，但是包含的字母相同的比较。

a.hashCode() 有什么用？与 a.equals(b) 有什么关系？(答案)
hashCode() 方法是相应对象整型的 hash 值。它常用于基于 hash 的集合类，如 Hashtable、HashMap、LinkedHashMap 等等。它与 equals() 方法关系特别紧密。根据 Java 规范，两个使用 equal() 方法来判断相等的对象，必须具有相同的 hash code。

final、finalize 和 finally 的不同之处？(答案)
final 是一个修饰符，可以修饰变量、方法和类。如果 final 修饰变量，意味着该变量的值在初始化后不能被改变。finalize 方法是在对象被回收之前调用的方法，给对象自己最后一个复活的机会，但是什么时候调用 finalize 没有保证。finally 是一个关键字，与 try 和 catch 一起用于异常的处理。finally 块一定会被执行，无论在 try 块中是否有发生异常。

Java 实践问题:

76）Java 中，编写多线程程序的时候你会遵循哪些最佳实践？(答案)
a）给线程命名，这样可以帮助调试。
b）最小化同步的范围，而不是将整个方法同步，只对关键部分做同步。
c) 尽量使用 Runnable 接口创建线程，实现线程对象与线程任务的分离

d)当线程任务较多时我们可以使用线程池来实现资源的优化。

77）说出几点 Java 中使用 Collections 的最佳实践(答案)
这是我在使用 Java 中 Collectionc 类的一些最佳实践:
a）使用正确的集合类，例如，如果不需要同步列表，使用 ArrayList 而不是 Vector。

b）优先使用并发集合，而不是对集合进行同步。并发集合提供更好的可扩展性。
c）使用接口代表和访问集合，如使用 List 存储 ArrayList，使用 Map 存储 HashMap 等等。
d）使用迭代器来循环集合。
e）使用集合的时候使用泛型。

F）根据不同操作先用合理的集合如 ArrayList 与 LinkedList。

9）说出 5 条 IO 的最佳实践(答案)
IO 对 Java 应用的性能非常重要。理想情况下，你不应该在你应用的关键路径上避免 IO 操作。下面是一些你应该遵循的 Java IO 最佳实践:
a）使用有缓冲区的 IO 类，而不要单独读取字节或字符。
b）使用 NIO 和 NIO2
c）在 finally 块中关闭流，或者使用 try-with-resource 语句。
d）使用内存映射文件获取更快的 IO。

80）列出 5 个应该遵循的 JDBC 最佳实践(答案)
有很多的最佳实践，你可以根据你的喜好来例举。下面是一些更通用的原则:
a）使用批量的操作来插入和更新数据
b）使用 PreparedStatement 来避免 SQL 异常，并提高性能。
c）使用数据库连接池
d）通过列名来获取结果集，不要使用列的下标来获取。

81）说出几条 Java 中方法重载的最佳实践？(答案)
下面有几条可以遵循的方法重载的最佳实践来避免造成自动装箱的混乱。
a）不要重载这样的方法：一个方法接收 int 参数，而另个方法接收 Integer 参数。
b）不要重载参数数量一致，而只是参数顺序不同的方法。
c）如果重载的方法参数个数多于 5 个，采用可变参数。

Java 中，DOM 和 SAX 解析器有什么不同？(答案)
DOM 解析器将整个 XML 文档加载到内存来创建一棵 DOM 模型树，这样可以更快的查找节点和修改 XML 结构，而 SAX 解析器是一个基于事件的解析器，不会将整个 XML 文档加载到内存。由于这个原因，DOM 比 SAX 更快，也要求更多的内存，不适合于解析大 XML 文件。

简述堆和栈的区别

VM 中堆和栈属于不同的内存区域，使用目的也不同。栈常用于保存方法帧和局部变量，而对象总是在堆上分配。栈通常都比堆小，也不会在多个线程之间共享，而堆被整个 JVM 的所有线程共享。

简述 JVM 内存分配

1. 基本数据类型比变量和对象的引用都是在栈分配的
2. 堆内存用来存放由 new 创建的对象和数组

3. 类变量（static 修饰的变量），程序在一加载的时候就在堆中为类变量分配内存，堆中的内存地址存放在栈中

4. 实例变量：当你使用 java 关键字 new 的时候，系统在堆中开辟并不一定是连续的空间分配给变量，是根据零散的堆内存地址，通过哈希算法换算为一长串数字以表征这个变量在堆中的"物理位置",实例变量的生命周期--当实例变量的引用丢失后，将被 GC（垃圾回收器）列入可回收“名单”中，但并不是马上就释放堆中内存

5. 局部变量: 由声明在某方法，或某代码段里（比如 for 循环），执行到它的时候在栈中开辟内存，当局部变量一但脱离作用域，内存立即释放

JVM 特性

平台无关性. Java 语言的一个非常重要的特点就是与平台的无关性。而使用 Java 虚拟机是实现这一特点的关键。一般的高级语言如果要在不同的平台上运行，至少需要编译成不同的目标代码。而引入 Java 语言虚拟机后，Java 语言在不同平台上运行时不需要重新编译。Java 语言使用模式 Java 虚拟机屏蔽了与具体平台相关的信息，使得 Java 语言编译程序只需生成在 Java 虚拟机上运行的目标代码（字节码），就可以在多种平台上不加修改地运行。Java 虚拟机在执行字节码时，把字节码解释成具体平台上的机器指令执行。

简述堆和栈的区别

JVM 中堆和栈属于不同的内存区域，使用目的也不同。栈常用于保存方法帧和局部变量，而对象总是在堆上分配。栈通常都比堆小，也不会在多个线程之间共享，而堆被整个 JVM 的所有线程共享。

XML 解析的几种方式和特点

DOM,SAX 三种解析方式:

DOM:消耗内存：先把 xml 文档都读到内存中，然后再用 DOM API 来访问树形结构，并获取数据。这个写起来很简单，但是很消耗内存。要是数据过大，手机不够牛逼，可能手机直接死机

SAX:解析效率高，占用内存少，基于事件驱动的：更加简单地说就是对文档进行顺序扫描，当扫描到文档(document)开始与结束、元素(element)开始与结束、文档(document)结束等地方时通知事件处理函数，由事件处理函数做相应动作，然后继续同样的扫描，直至文档结束。

JDK 1.8 特性

java 8 在 Java 历史上是一个开创新的版本，下面 JDK 8 中 5 个主要的特性： Lambda 表达式，允许像对象一样传递匿名函数 Stream API，充分利用现代多核 CPU，可以写出很简洁的代码 Date 与 Time API，最终，有一个稳定、简单的日期和时间库可供你使用 扩展方法，现在，接口中可以有静态、默认方法。 重复注解，现在你可以将相同的注解在同一类型上使用多次。

MySql 数据库：

2.数据库事务的四个特性及含义

数据库事务 transanction 正确执行的四个基本要素。ACID,原子性(Atomicity)、一致性(Correspondence)、隔离性(Isolation)、持久性(Durability)。

原子性:整个事务中的所有操作，要么全部完成，要么全部不完成，不可能停滞在中间某个环节。事务在执行过程中发生错误，会被回滚（Rollback）到事务开始前的状态，就像这个事务从来没有执行过一样。

一致性:在事务开始之前和事务结束以后，数据库的完整性约束没有被破坏。

隔离性:隔离状态执行事务，使它们好像是系统在给定时间内执行的唯一操作。如果有两个事务，运行在相同的时间内，执行 相同的功能，事务的隔离性将确保每一事务在系统中认为只有该事务在使用系统。这种属性有时称为串行化，为了防止事务操作间的混淆，必须串行化或序列化请 求，使得在同一时间仅有一个请求用于同一数据。

持久性:在事务完成以后，该事务所对数据库所作的更改便持久的保存在数据库之中，并不会被回滚。

2.数据库隔离级别

```
脏读：事务 B 读取事务 A 还没有提交的数据

不可重复读：两次事务读的数据不一致

幻读:事务 A 修改了数据，事务 B 也修改了数据，这时在事务 A 看来，明明修改了数据，咋不一样
```

3.MYSQL 的两种存储引擎区别（事务、锁级别等等），各自的适用场景

Myisam 存储引擎:如果表对事物要求不高,同时是以查询和添加为主的,我们考虑使用 myisam 存储引擎,比如论坛中的发帖表,回复表

Innodb 存储引擎:对事物要求高,保存的数据都是重要数据.我们建议使用 innodb,比如订单表,账号表

Myisam 和 innodb 的区别:

1. 事务安全,myisam 不支持使用,而 innodb 支持
2. 查询和添加速度,myisam 不用支持事务就不用考虑同步锁,查找和添加的速度快
3. 支持全文索引,myisam 支持,innodb 不支持
4. 锁机制:myisam 支持表锁,innodb 支持行锁
5. 外键:myisam 不支持外键,innodb 支持外键

6.索引的优缺点，什么时候使用索引，什么时候不能使用索引

索引最大的好处是提高查询速度，
缺点是更新数据时效率低，因为要同时更新索引
对数据进行频繁查询进建立索引，如果要频繁更改数据不建议使用索引。

11.Sql 的优化

1.sql 尽量使用索引,而且查询要走索引

2.对 sql 语句优化

子查询变成 left join
limit 分布优化，先利用 ID 定位，再分页
or 条件优化，多个 or 条件可以用 union all 对结果进行合并（union all 结果可能重复）
不必要的排序
where 代替 having,having 检索完所有记录，才进行过滤
避免嵌套查询
对多个字段进行等值查询时，联合索引

12.索引最左前缀问题

如果对三个字段建立联合索引，如果第二个字段没有使用索引，第三个字段也使用不到索引了

16.varchar 和 char 的使用场景

17.数据库连接池的作用

维护一定数量的连接，减少创建连接的时间

更快的响应时间
统一的管理

20.数据库三范式

1NF：原子性：字段不可再分

2NF：非主键字段完全依赖于主键字段

3NF：非主键属性无传递依赖

21.关系型数据库和非关系型数据库区别

关系型数据库

优点

1、容易理解：二维表结构是非常贴近逻辑世界一个概念，关系模型相对网状、层次等其他模型来说更容易理解；

2、使用方便：通用的 SQL 语言使得操作关系型数据库非常方便；

3、易于维护：丰富的完整性(实体完整性、参照完整性和用户定义的完整性)大大减低了数据冗余和数据不一致的概率；

4、支持 SQL，可用于复杂的查询。

5.支持事务

缺点

1、为了维护一致性所付出的巨大代价就是其读写性能比较差；

2、固定的表结构；

3、不支持高并发读写需求；

4、不支持海量数据的高效率读写

非关系型数据库

```
1、使用键值对存储数据；
2、分布式；
优点
无需经过 sql 层的解析，读写性能很高
基于键值对，数据没有耦合性，容易扩展
存储数据的格式：nosql 的存储格式是 key,value 形式
缺点
不提供 sql 支持
```

22.数据库中 join 的 inner join, outer join, cross join

1.以 A，B 两张表为例

A left join B：选出 A 的所有记录，B 表中没有的以 null 代替 right join 同理

2.inner join ： A,B 的所有记录都选出，没有的记录以 null 代替

3.cross join (笛卡尔积) A 中的每一条记录和 B 中的每一条记录生成一条记录

例如 A 中有 4 条，B 中有 4 条，cross join 就有 16 条记录

3.视图的作用，视图可以更改么？

视图是虚拟的表，与包含数据的表不一样，视图只包含使用时动态检索数据的查询；不包含任何列或数据。使用视图可以简化复杂的 sql 操作，隐藏具体的细节，保护数据；视图创建

后，可以使用与表相同的方式利用它们。
视图不能被索引，也不能有关联的触发器或默认值，如果视图本身内有 order by 则对视图再次 order by 将被覆盖。
创建视图：create view XXX as XXXXXXXXXXXXX;
对于某些视图比如未使用联结子查询分组聚集函数 Distinct Union 等，是可以对其更新的，对视图的更新将对基表进行更新；但是视图主要用于简化检索，保护数据，并不用于更新，而且大部分视图都不可以更新。

26.MYSQL 和 ORACLE 的分页语句

为什么要分页?很多数据不能完全展示出来,需要进行分段显示

mysql:是使用关键字 limit 来进行分页的.LIMIT [offset,] rows:offset 指定要返回的第一行的偏移量(也就是从哪个索引开始)，rows 第二个指定返回行的最大数目。初始行的偏移量是 0(不是 1)

oracle:一般是使用 rownum 加 select 嵌套查询

27.触发器的使用场景?

触发器:触发器需要有触发条件,当条件满足后,做什么操作

应用场景:某些社交软件的日志更新,会通知好友; 一些论坛中,当插入新帖时,会更改当前帖子总数以及最后发帖时间.

65.选择合适的索引

索引是帮助 DBMS 高效获取数据的数据结构.

分类:普通索引,唯一索引,主键索引,全文索引

1.普通索引:允许重复的值出现

2.唯一索引:除了不能有重复的记录外,其它和普通索引一样.(用户名;用户身份证;手机号)

3.主键索引:是随着设定主键而创建的;也就是把某个列设为主键的时候,数据库就会给该列创建索引;唯一且没有 null 值

4.全文索引:用来对表中文本域(char,varchar,text)进行索引,全文索引针对 myisam

66.使用索引的一些技巧

索引弊端:

1. 占用磁盘空间.

2. 对 dml(插入,修改.删除)操作有影响,变慢

使用场景:

1. 肯定在 where 条件经常使用,如果不做查询就没有意义

2. 该字段的内容不是唯一的几个值(sex).

3. 字段内容不是频繁变化

具体技巧:

1. 对于创建的多列索引(复合索引),不是使用的第一部分就不会使用索引(最左匹配)

2. 对于使用 like 查询,查询如果是”%aaa”不会使用到索引,而”aaa%”会使用到索引

3. 如果条件中有 or,有条件没有使用索引,即使其中有条件带索引,也不会使用.简单来说,就是要求使用的所有字段,都必须单独使用时才能使用索引.

4. 如果列类型是字符串,拿一定要在条件中将数据使用引号引用起来,否则索引失效

5. 如果 mysql 估计使用全表扫描要比索引快,则不适用索引.例子:表里只有一条记录

67.数据库优化之分表

分表分为水平分表(按行)和垂直分表(按列)

水平:在实际操作中,mysql 表数据一般达到百万级别,查询效率会很低,容易造成表锁,甚至堆积很多连接,直接挂掉.水平分表能够很大程度的减少这些压力.

垂直:如果一张表中某个字段值非常多(长文本,二进制等),而且只有在很少的情况下会查询,比如商品的详情描述,这时候就可以把字段单个放到一个表,通过外键与原表关联起来

水平分表策略:

1. 按时间分表:这种分表方式有一定的局限性,当数据有较强的时效性.如微博发布纪录,微信消息纪录等,这种数据很少会有用户查询几个月前的数据,这时可以按月分表

2. 按区间范围分表:一般在有严格的自增 id 需求上,如按照 user_id 水平分表

3. Hash 分表(用的多):通过一个原始目标的 id 或者名称通过一定的 hash 算法计算出数据库存储表的表名,然后访问相应的表.

68.数据库的读写分离

一台数据库支持的最大并发连接数是有限的,如果用户并发访问太多,一台服务器满足不了要求时,可以集群处理.mysql 的集群处理技术最常用的是读写分离,

1.主从同步:数据库最终会把数据持久化到磁盘,如果集群必须确保每个数据库服务器的数据时一致的.能改变数据库数据的操作都往主数据库去写,而其他的数据库从主数据库上同步数据

2.读写分离:使用负载均衡来实现写的操作都往主数据.而读的操作都往从数据库去

Servlet

91.什么是 Servlet?

Java Servlet API 是 Servlet 容器(tomcat)和 servlet 之间的接口，它定义了 serlvet 的各种方法，还定义了 Servlet 容器传送给 Servlet 的对象类，其中最重要的就是 ServletRequest 和 ServletResponse。所以说我们在编写 servlet 时，需要实现 Servlet 接口，按照其规范进行操作

92.说一下 Servlet 的体系结构。

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。

94.GenericServlet 和 HttpServlet 有什么区别？

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。

95.解释下 Servlet 的生命周期。

用户第一次访问 Servlet 的时候,服务器会创建一个 Servlet 的实例,那么 Servlet 中 init 方法就会执行.任何一次请求服务器都会创建一个新的线程访问 Servlet 中的 service 的方法.在 service 方法内部根据请求的方式的不同调用 doXXX 的方法.(get 请求调用 doGet,post 请求调用 doPost).当 Servlet 中服务器中移除掉,或者关闭服务器,Servlet 的实例就会被销毁,那么 destroy 方法就会执行.

96.doGet()方法和 doPost()方法有什么区别？

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。

99.什么是 Servlet 链(Servlet Chaining)？

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。

100.如何知道是哪一个客户端的机器正在请求你的 Servlet？

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。

101.HTTP 响应的结构是怎么样的？

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

102.什么是 cookie？session 和 cookie 有什么区别？

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。

- Session 和 cookie 都是会话(session)跟踪技术.cookie 通过在客户端记录信息确定用户身份,而 session 是通过在服务器端记录信息确定用户身份.但是 session 的实现依赖于 cookie 机制来保存 JESESSIONID(session 的唯一标识,需要存在客户端)
- 区别:
- 1. cookie 的数据存储在客户端,session 的数据存储在服务器上
- 1. cookie 不是很安全,别人可以通过分析存放在本地的 cookie 并进行 cookie 欺骗,考虑到安全应该使用 session
- 1. session 会在一定时间内保存在服务器上,当访问增多时,会影响服务器的性能.考虑到服务器性能,应当使用 cookie.
- 1. 单个 cookie 保存数据不能超过 4k,很多浏览器显示一个站点最多保存 20 个 cookie
- 2. 将重要信息保存在 session 中(登陆),将其他需要保留的信心存放在 cookie 中(购物车,cookie 是可以在客户端禁用的,这时候要使用 cookie+数据库的方式实现购物车,当 cookie 中不能取出数据,就从数据库中取)

105.sendRedirect()和 forward()方法有什么区别？

URL 重定向的特点:

1):浏览器地址栏路径发送变化

2):只发送了两个请求.

3):因为是不同的请求,所以不能共享请求中的数据.

4):可以跨域访问资源.

5):不可以访问 WEB-INF 中的资源.

请求转发的特点:

1):浏览器地址栏路径没变

2):只发送了一个请求.

3):共享同一个请求,在请求敏感词享数据.

4):只能访问当前应用中的资源,不能跨域跳转.

5):可以访问 WEB-INF 中的资源.

重定向(redirect)其实是两次 request,第一次，客户端 request,A 服务器响应，并 response 回来，告诉浏览器，你应该去 B。这个时候 IE 可以看到地址变了，而且历史的回退按钮也亮了。重定向可以访问自己 web 应用以外的资源。在重定向的过程中，传输的信息会被丢失。

请求转发(forward)是服务器内部把对一个 request/response 的处理权，移交给另外一个.对于客户端而言，它只知道自己最早请求的那个 A，而不知道中间的 B，甚至 C、D。传输的信息不会丢失。

Session：跨多个页面请求或访问网站时识别用户以及存储有关用户信息的方式。
Cookie ： 是存储在客户端计算机上的文本文件，并保留了各种跟踪信息。

14.HTTP GET POST 请求的区别

1、GET 请求，请求的数据会附加在 URL 之后，以?分割 URL 和传输数据，多个参数用&连接。URL 的编码格式采用的是 ASCII 编码，而不是 uniclde，即是说所有的非 ASCII 字符都要编码之后再传输。

POST 请求：POST 请求会把请求的数据放置在 HTTP 请求包的包体中

因此，GET 请求的数据会暴露在地址栏中，而 POST 请求则不会。

2、传输数据的大小

在 HTTP 规范中，没有对 URL 的长度和传输的数据大小进行限制。但是在实际开发过程中，对于 GET，特定的浏览器和服务器对 URL 的长度有限制。因此，在使用 GET 请求时，传输数据会受到 URL 长度的限制。

对于 POST，由于不是 URL 传值，理论上是不会受限制的，但是实际上各个服务器会规定对 POST 提交数据大小进行限制。

3、安全性

POST 的安全性比 GET 的高。这里的安全是指真正的安全，而不同于上面 GET 提到的安全方法中的安全，上面提到的安全仅仅是不修改服务器的数据。比如，在进行登录操作，通过 GET 请求，用户名和密码都会暴露再 URL 上，因为登录页面有可能被浏览器缓存以及其他人查看浏览器的历史记录的原因，此时的用户名和密码就很容易被他人拿到了。

15.说说你对 SERVLET 的理解

就是一个运行在 WEB 服务器上的小的 Java 程序,用来接收和响应从客户端发送过来的请求,通常使用 HTTP 协议.

使用:1、编写一个 Java 类，实现 servlet 接口。

2、把开发好的 Java 类部署到 web 服务器中。

按照一种约定俗成的称呼习惯，通常我们也把实现了 servlet 接口的 java 程序，称之为 Servlet

9.jsp 内置对象和四大作用域和页面传值

内置对象名　类型

request　HttpServletRequest

response HttpServletResponse

config　ServletConfig

application ServletContext

session　HttpSession

exception Throwable

page　Object(this)

out JspWriter

pageContext PageContext

域 范围

page 域　只能在当前 jsp 页面使用

request 域　只能在同一个请求中使用

session 域　只能在同一个会话(session 对象)中使用

context 域　只能在同一个 web 应用中使用

18.JSP 和 SERVLET 的相同点和不同点

联系：

JSP 是 Servlet 技术的扩展，本质上是 Servlet 的简易方式，更强调应用的外表表达。

JSP 编译后是”类 servlet”。

不同点：

- Servlet 的应用逻辑是在 Java 文件中，并且完全从表示层中的 HTML 里分离开来。

- JSP 的情况是 Java 和 HTML 可以组合成一个扩展名为.jsp 的文件。做界面展示比较方便,而嵌入逻辑复杂.

- JSP 侧重于视图，Servlet 主要用于控制逻辑

21.MVC 模式和 MVC 各部分的实现

MVC 的全名是 Model View Controller，是模型(model)一视图(view)一控制器(controller)的缩写，是一种软件设计典范.低耦合(m 和 v 的分离)

struts2 的 mvc:jsp->StrutsPrepareAndExecuteFilter(前端控制器,核心控制器)+action->javabean->result

JSP

107.什么是 JSP 页面？

JSP 全称 Java Server Pages，是一种动态网页开发技术。它使用 JSP 标签在 HTML 网页中插入 Java 代码。JSP 是一种 Java servlet，主要用于实现 Java web 应用程序的用户界面部分。网页开发者们通过结合 HTML 代码、XHTML 代码、XML 元素以及嵌入 JSP 操作和命令来编写 JSP。

109.JSP 有什么优点？

下面列出了使用 JSP 的优点：

- JSP 页面是被动态编译成 Servlet 的，因此，开发者可以很容易的更新展现代码。
- JSP 页面可以被预编译。
- JSP 页面可以很容易的和静态模板结合，包括：HTML 或者 XML，也可以很容易的和产生动态内容的代码结合起来。
- 开发者可以提供让页面设计者以类 XML 格式来访问的自定义的 JSP 标签库。
- 开发者可以在组件层做逻辑上的改变，而不需要编辑单独使用了应用层逻辑的页面。

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

jsp 和 servlet 的区别和联系：
1. jsp 经编译后就变成了 Servlet.
(JSP 的本质就是 Servlet，JVM 只能识别 java 的类，不能识别 JSP 的代码,Web 容器将 JSP 的代码编译成 JVM 能够识别的 java 类)
2. jsp 更擅长表现于页面显示,servlet 更擅长于逻辑控制.
3. Servlet 中没有内置对象，Jsp 中的内置对象都是必须通过 HttpServletRequest 对象，HttpServletResponse 对象以及 HttpServlet 对象得到.
Jsp 是 Servlet 的一种简化，使用 Jsp 只需要完成程序员需要输出到客户端的内容，Jsp 中的 Java 脚本如何镶嵌到一个类中，由 Jsp 容器完成。
而 Servlet 则是个完整的 Java 类，这个类的 Service 方法用于生成对客户端的响应。

21、解释内存中的栈(stack)、堆(heap)和方法区(method area)的用法。

答：通常我们定义一个基本数据类型的变量，一个对象的引用，还有就是函数调用的现场保存都使用 JVM 中的栈空间;而通过 new 关键字和构造器创建的对象则放在堆空间，堆是垃圾收集器管理的主要区域，由于现在的垃圾收集器都采用分代收集算法，所以堆空间还可以细分为新生代和老生代，再具体一点可以分为 Eden、Survivor(又可分为 From Survivor 和 To Survivor)、Tenured;方法区和堆都是各个线程共享的内存区域，用于存储已经被 JVM 加载的类信息、常量、静态变量、JIT 编译器编译后的代码等数据;程序中的字面量(literal)如直接书写的 100、"hello"和常量都是放在常量池中，常量池是方法区的一部分，。栈空间操作起来最快但是栈很小，通常大量的对象都是放在堆空间，栈和堆的大小都可以通过 JVM 的启动参数来进行调整，栈空间用光了会引发 StackOverflowError，而堆和常量池空间不足则会引发 OutOfMemoryError。

框架

1.你如何理解 Spring?

具体来说 Spring 是一个轻量级的容器，用于管理业务相关对象的。核心功能主要为：IOC,AOP

IOC：控制反转，将对象的创建过程交给容器，让容器管理对象的生命周期如创建，初始化，销毁等。

AOP：面向切面编程，对关注点进行模块化，通过对某一功能点进行编程，比如记录日志，有很多个类都需要记录日志的方法，则创建记录日志的代理方法，需要调用该功能是只需要调用代理方法，这就是 AOP。

2.spring 配置 bean 实例化有哪些方式?

1）使用类构造器实例化(默认无参数)

```
<bean id="bean1" class="cn.itcast.spring.b_instance.Bean1"></bean>
```

2）使用静态工厂方法实例化(简单工厂模式)

```
//下面这段配置的含义：调用 Bean2Factory 的 getBean2 方法得到 bean2
<bean id="bean2" class="cn.itcast.spring.b_instance.Bean2Factory" factory-method="getBean2"></bean>
```

3）使用实例工厂方法实例化(工厂方法模式)

```
//先创建工厂实例 bean3Facory，再通过工厂实例创建目标 bean 实例
<bean id="bean3Factory" class="cn.itcast.spring.b_instance.Bean3Factory"></bean>
<bean id="bean3" factory-bean="bean3Factory" factory-method="getBean3"></bean>
```

3.介绍一下 Spring 的事物管理

事务就是我们实现一个操作的一组逻辑，这组逻辑里面的所有操作要么都成功要么，都失败，即事务的原子性。这样可以防止出现脏数据，保证数据库数据的完整性

开发中为了避免这种情况一般都会进行事务管理。Spring 中也有自己的事务管理机制，一般是使用 TransactionMananger 进行管 理，可以通过 Spring 的注入来完成此功能。
Spring 支持如下两种方式的事务管理：
编程式事务管理：这意味着你可以通过编程的方式管理事务，这种方式带来了很大的灵活性，但很难维护。
声明式事务管理：这种方式意味着你可以将事务管理和业务代码分离。你只需要通过注解或者 XML 配置管理事务。

一般选择声明式事务管理，因为这种方式和应用程序的关联较少。

5.讲述 SpringMVC 工作流程

1、用户发送请求至前端控制器 DispatcherServlet

2、DispatcherServlet 收到请求调用 HandlerMapping 处理器映射器。

3、处理器映射器找到具体的处理器，生成处理器对象及处理器拦截器(如果有则生成)一并返回给 DispatcherServlet。

4、DispatcherServlet 调用 HandlerAdapter 处理器适配器

5、HandlerAdapter 经过适配调用具体的处理器(Controller，也叫后端控制器)。

6、Controller 执行完成返回 ModelAndView

7、HandlerAdapter 将 controller 执行结果 ModelAndView 返回给 DispatcherServlet

8、DispatcherServlet 将 ModelAndView 传给 ViewReslover 视图解析器

9、ViewReslover 解析后返回具体 View

10、DispatcherServlet 根据 View 进行渲染视图（即将模型数据填充至视图中）。

11、DispatcherServlet 响应用户

7.Spring 中用到哪些设计模式?

1.工厂模式（BeanFactory 中）

2.单例模式（Spring 中默认 bean 为单例）

3.适配器模式（HandlerAdater）

4.装饰者模式

5.代理模式（AOP 中用到 JDK 动态代理）

6.观察者模式（listener 的实现，例如 ApplicationListener）

7.策略模式（定义一系列的算法，把它们一个个的封装起来，并且使它们可以相互替换。在实例化对象时用到）

8.模板模式（jdbcTemplate）

(1) Spring 在 SSM 起什么作用

Spring 是一个轻量级框架，也是一个容器，Spring 实质上讲就是一个 Bean 工厂，主要用来管理 Bean 的生命周期和框架集成。有 IOC 控制反转，DI 依赖注入，控制反转是把 dao 依赖注入到 servic 层，然后 service 层反转给 action 层，Spring 的顶层容器为 BeanFactory，常用的 ApplicationContext 为它的子接口，实现了工厂模式，Spring 还提供了 AOP 的支持，方便在切面级开发，

(5)Spring 的配置文件有哪些内容?

- 开启事务注解驱动
- 事务管理器
- 开启注解功能，并配置扫描包
- 配置数据源
- 配置 SQL 会话工厂、别名、映射文件
- 不用编写 DAO 层的实现类（代理模式）

(6)说下 Spring 的注解

- @Controller
- @Service
- @Component
- @RequestMapping
- @Resource、@Autowired
- @ResponseBody
- @Transactional

(8)Spring DI 的几种方式

配置文件的形式:

- (1)构造器注入：通过构造方法初始化
- <constructor-arg name="dao"</constructor-arg>
- (2)setter 注入：通过 setter 方法初始化注入
- <property name="dao" ref="dao2"></property>
- 注意：在实际开发中常用 setter 注入。

注解的形式

什么是 MyBatis 的接口绑定,有什么好处

接口映射就是在 MyBatis 中任意定义接口,然后把接口里面的方法和 SQL 语句绑定,我们直接调用接口方法就可以,这样比起原来了 SqlSession 提供的方法我们可以有更加灵活的选择和设置，同时也为之后与 spring 的整合做铺垫

(5)讲下 MyBatis 的缓存

MyBatis 的缓存分为一级缓存和二级缓存,一级缓存放在 session 里面,默认就有,二级缓存放在它的命名空间里,默认是打开的,使用二级缓存属性类需要实现 Serializable 序列化接口(可用来保存对象的状态),可在它的映射文件中配置<cache/>

(6)MyBatis(IBatis)的好处是什么

ibatis 把 sql 语句从 Java 源程序中独立出来，放在单独的 XML 文件中编写，给程序的维护带来了很大便利。

ibatis 封装了底层 JDBC API 的调用细节，并能自动将结果集转换成 JavaBean 对象，大大简化了 Java 数据库编程的重复工作。

因为 Ibatis 需要程序员自己去编写 sql 语句，程序员可以结合数据库自身的特点灵活控制 sql 语句，因此能够实现更高的查询效率，能够完成复杂查询。

40 谈谈对 mybatis 中的 sqlSession、sqlSessionFactoryBuild 和 sqlSessionFactory 的理解。

sqlSession:封装了对数据 增删改查的方法

sqlSession 是通过 sqlSessionFactory 创建的

.sqlSessionFactory 是通过 sqlSessionFactoryBuild 创建的

sqlSessionFactoryBuild 是创建 sqlSessionFactory 时使用的.一旦创建成功后就不需要 sqlSessionFactoryBuild 的,因为 sqlSession 是通过 sqlSessionFactory 创建的,可以可以当做工具类使用

sqlSessionFactory 是一个接口，类里重载了 opensession 的不同的方法使用范围是在整个运行范围内,一旦创建可以重复使用.可以当做单实例对象来管理

sqlSession 是面向用户的一个操作数据库的接口 每个线程都应该有自己的 sqlSession 并且 sqlSession 不可以共享. 线程是不安全的,打开一个 sqlSession 用完之后就要关闭