**面试准备**

1. 技术

Java

Java基础

Java常用集合

1. Collection

----List

--------ArrayList，基于数组，随机访问效率高，线程不安全

--------LinkedList，基于链表，头尾来增删的效率高

--------Vector，队列，线程安全的

------------stack

----Set，不含重复元素。相当于只存储Key的Map。

1. Map

----HashMap，线程不安全，轻量级HashTable。

HashMap是由数组+链表的一个结构组成。

----HashTable，线程安全，是HashMap的前身。

----TreeMap

1. 注：ArrayList和LinkedList

ArrayList的随机访问更高，基于数组实现的ArrayList可直接定位到目标对象，而LinkedList需要从头Node或尾Node开始向后/向前遍历若干次才能定位到目标对象

LinkedList在头/尾节点执行插入/删除操作的效率比ArrayList要高

由于ArrayList每次扩容的容量是当前的1.5倍，所以LinkedList所占的内存空间要更小一些

数组存储区间是连续的，占用内存严重。数组的特点是：寻址容易，插入和删除困难；链表存储区间离散，占用内存比较宽松。链表的特点是：寻址困难，插入和删除容易。

二者的遍历效率接近，但需要注意，遍历LinkedList时应用iterator方式，不要用get(int)方式，否则效率会很低

异常

1. 结构

Throwable

----Error，是程序无法处理的错误，表示运行应用程序中较严重问题。

----Exception

--------运行时异常（非检查异常），Java编译器不会检查它，编译会通过。

--------非运行时异常（检查异常），IOException、SQLException等以及用户自定义的Exception异常

1. 异常举例
2. Error

StackOutMemoryError – 内存溢出

1. 运行时异常

OutMemoryException - 内存不足

NullPointerException - 空指针引用异常  
ClassCastException - 类型强制转换异常。

IndexOutOfBoundsException - 下标越界异常

ArithmeticException - 算术运算异常  
NumberFormatException - 数字格式异常

1. 非运行时异常：即，检查异常。

IOException、SQLException等以及用户自定义的Exception异常

垃圾回收机制

1. 解决的问题：

程序员容易忘记回收不再使用的对象，从而造成内存泄漏。

1. 解决方案：

java语言设计了个垃圾回收机制，自动回收。

1. 怎么判断需要回收：

GC通过确定对象是否被活动对象引用来确定是否收集该对象。

1. 主动回收：

执行 system.gc()方法。

注：内存泄漏（Memory Leak），是指程序中已动态分配的堆内存由于某种原因程序未释放或无法释放，造成系统内存的浪费，导致程序运行速度减慢甚至系统崩溃等严重后果。

事务管理

1、四个基本特性：原子性(Atomicity)、一致性(Consistency)、隔离性(Isolation)、持久性(Durability)

原子性是指事务个体；一致性是指事务操作过程中数据使用上；隔离性是指事务个体之间；持久性是指数据的存储。

2、七大传播特性：

必须的、支持的、强制的、绝不会出现、嵌套、必须启用新的、不支持的。

1. 英文：

Propagation：PROPAGATION，传播。

Required：REQUIRED，必须地。

Supports：SUPPORTS，支持。

Mandatory：MANDATORY，强制性的。

never： NEVER，绝不会

nested：NESTED，嵌套。

1. 须不需、该不该、有没有、新与此、在不在、挂与抛及需TransactionManager、嵌套。

| **传播行为** | **英文（简述）** | **含义** |
| --- | --- | --- |
| PROPAGATION\_REQUIRED | 必须地  须且无则在新 | 表示当前方法必须运行在事务中。如果当前事务存在，方法将会在该事务中运行。否则，会启动一个新的事务 |
| PROPAGATION\_SUPPORTS | 支持地  不需且有则在此 | 表示当前方法不需要事务上下文，但是如果存在当前事务的话，那么该方法会在这个事务中运行 |
| PROPAGATION\_MANDATORY | 强制使用  无则抛 | 表示该方法必须在事务中运行，如果当前事务不存在，则会抛出一个异常 |
| PROPAGATION\_REQUIRED\_NEW | 必须启新  1、须且新启而在  2、方法执行而事务挂起  3、TransactionManager来访问 | 表示当前方法必须运行在它自己的事务中。一个新的事务将被启动。如果存在当前事务，在该方法执行期间，当前事务会被挂起。如果使用JTATransactionManager的话，则需要访问TransactionManager |
| PROPAGATION\_NOT\_SUPPORTED | 不支持  1、不该且有则挂起  2、TransactionManager来访问 | 表示该方法不应该运行在事务中。如果存在当前事务，在该方法运行期间，当前事务将被挂起。如果使用JTATransactionManager的话，则需要访问TransactionManager |
| PROPAGATION\_NEVER | 绝不，未曾，不会发生。  不该且在则抛 | 表示当前方法不应该运行在事务上下文中。如果当前正有一个事务在运行，则会抛出异常 |
| PROPAGATION\_NESTED | 嵌套 | 表示如果当前已经存在一个事务，那么该方法将会在嵌套事务中运行。嵌套的事务可以独立于当前事务进行单独地提交或回滚。如果当前事务不存在，那么其行为与PROPAGATION\_REQUIRED一样。注意各厂商对这种传播行为的支持是有所差异的。可以参考资源管理器的文档来确认它们是否支持嵌套事务 |

3、五大隔离级别

|  |  |  |
| --- | --- | --- |
| 参数 | 英文（注释） | 详细说明 |
| ISOLATION\_DEFAULT | 默认 | 使用后端数据库默认的隔离级别 |
| ISOLATION\_READ\_UNCOMMITTED | 未提交  脏读（回滚） | 最低的隔离级别，允许读取尚未提交的数据变更，可能会导致脏读、幻读或不可重复读 |
| ISOLATION\_READ\_COMMITTED | 已提交  不可重复读（数据行） | 允许读取并发事务已经提交的数据，可以阻止脏读，但是幻读或不可重复读仍有可能发生 |
| ISOLATION\_REPEATABLE\_READ | 重读  幻读（事务间并行，增删与条件） | 对同一字段的多次读取结果都是一致的，除非数据是被本身事务自己所修改，可以阻止脏读和不可重复读，但幻读仍有可能发生 |
| ISOLATION\_SERIALIZABLE | 序列化  脏读、不可重复读和幻读全解决 | 最高的隔离级别，完全服从ACID的隔离级别，确保阻止脏读、不可重复读以及幻读，也是最慢的事务隔离级别，因为它通常是通过完全锁定事务相关的数据库表来实现的 |

注解

注解是IoC容器各种扫描并搜集的设定。

1、声明Bean类：

@Mapper @Repository（持久层）、@Service(业务层)、@Controller (控制层)和 @Componen[t](https://www.cnblogs.com/lxcmyf/p/5674399.html)（类路径扫描下类上标注自动配置bean）和@Bean（更广更精准）

2、注入Bean类：

@Autowired 和@Resource。

3、Java配置类：

@Configuration 声明当前类为配置类，相当于xml形式的Spring配置（类上）

@Bean 注解在方法上，声明当前方法的返回值为一个bean，替代xml中的方式（类或方法上）

@Configuration 声明当前类为配置类，其中内部组合了@Component注解，表明这个类是一个bean（类上）

@ComponentScan 用于对Component进行扫描，相当于xml中的（类上）

4、其他：

@EnableScheduling 在配置类上使用，开启计划任务的支持（类上）

@EnableTransactionManagement 开启注解式事务的支持

HTTP协议，GET和POST 的区别

1. 英文：

Get：接收，挣得，主观地自己动手；Post：邮寄，投递，第三方代理。

1. 方法：

Http定义了与服务器交互的不同方法，最基本的方法有4种，分别是GET，POST，PUT，DELETE。

1. 原理：
   1. 简述：URL后面的get与post是URL的一部分，对应查与更新。

URL全称是资源描述符，我们可以这样认为：一个URL地址，它用于描述一个网络上的资源，而HTTP中的GET，POST，PUT，DELETE就对应着对这个资源的查，改，增，删4个操作。到这里，大家应该有个大概的了解了，GET一般用于获取/查询资源信息，而POST一般用于更新资源信息。

* 1. 其他特点

根据HTTP规范，GET用于信息获取，而且应该是安全的和幂等的。

1. .所谓安全的意味着该操作用于获取信息而非修改信息。
2. .幂等的意味着对同一URL的多个请求应该返回同样的结果。

根据HTTP规范，POST表示可能修改变服务器上的资源的请求。

1. GET和POST的区别：说完原理性的问题，我们再从表面现像上面看看GET和POST的区别
2. 请求数据的位置

GET请求的数据会附在URL之后（就是把数据放置在HTTP协议头中），以?分割URL和传输数据，参数之间以&相连。POST把提交的数据则放置在是HTTP包的包体中。

1. 请求数据大小的限制

"GET方式提交的数据最多只能是1024（特定的浏览器及服务器对URL它的限制）字节，理论上POST没有限制（起限制作用的是服务器的处理程序的处理能力）。

1. 安全性

POST的安全性要比GET的安全性高。

框架

Spring

1、原理：IOC、依赖注入和AOP

核心：bean工厂；

1. IoC:  
   概念：控制权由对象本身转向容器；由容器根据配置文件去创建实例并创建各个实例之间的依赖关系

控制权，是指对象的实例创建权与实例之间的依赖关系的控制权。

1. 依赖注入：

把有依赖关系的类放到容器中，解析出这些类的实例，就是依赖注入

1. AOP

AOP的主要原理：Spring AOP使用的是动态代理。所谓的动态代理，就是说AOP框架不会去修改字节码，而是在内存中临时为方法生成一个AOP对象，这个AOP对象包含了目标对象的全部方法，并且在特定的切点做了增强处理，并回调原对象的方法。

2、注：Spring目的

就是让对象与对象（模块与模块）之间的关系没有通过代码来关联，都是通过配置类说明 管理的（Spring根据这些配置 内部通过反射去动态的组装对象）  
切记：Spring是一个容器，凡是在容器里的对象才会有Spring所提供的这些服务和功能。

3、综述：

内部最核心的就是IOC了， 动态注入，让一个对象的创建不用new了，可以自动的生产，这其实就是利用java里的反射 反射其实就是在运行时动态的去创建、调用对象，Spring就是在运行时，跟xml Spring的配置 文件来动态的创建对象，和调用对象里的方法的 。  
Spring还有一个核心就是AOP这个就是面向切面编程，可以为某一类对象 进行监督和控制（也就是 在调用这类对象的具体方法的前后去调用你指定的模块）  
从而达到对一个模块扩充的功能。这些都是通过 配置类达到的。

###### Springboot

1、什么事springboot？（简单介绍）

Spring Boot 是 Spring 开源组织下的子项目，是 Spring 组件一站式处理方案，主要是简化了使用 Spring 的难度，简省了繁重的配置，提供了各种启动器，开发者能快速上手。

2、为什么要用 Spring Boot？（优点）

1）Spring Boot 优点非常多，如：

独立运行

简化配置（项目中，手动的配置文件与配置类）

自动配置（springboot的约定，已约定好的配置）

无代码生成和XML配置

应用监控

上手容易

-快速创建独立运行的spring项目与主流框架集成   
-使用嵌入式的servlet容器（内置了 Tomcat/ Jetty 等容器），应用无需打包成war包   
-starters自动依赖与版本控制   
-大量的自动配置，简化开发，也可修改默认值   
-准生产环境的运行应用监控   
-与云计算的天然集成

Spring Boot 集这么多优点于一身，还有理由不使用它呢？

注：约定优于或大于配置，约定指的是“自动配置”，配置指的是“简化配置”

2）简单讲

敏捷开发

内置tomcat

无需xml配置

3、Spring Boot 的核心配置文件有哪几个？它们的区别是什么？

Spring Boot 的核心配置文件是 application 和 bootstrap 配置文件。

application 配置文件这个容易了解，主要用于 Spring Boot 项目的自动化配置。

bootstrap 配置文件有以下几个应用场景。

使用 Spring Cloud Config 配置中心时，这时需要在 bootstrap 配置文件中增加连接到配置中心的配置属性来加载外部配置中心的配置信息；

少量固定的不能被覆盖的属性；

少量加密/解密的场景；

4、Spring Boot 的配置文件有哪几种格式？它们有什么区别？

.properties 和 .yml，它们的区别主要是书写格式不同。

1).properties

IMG_256

2).yml



另外，

.yml 格式不支持@PropertySource注解导入配置；

格式严格，不能多余的空格且首行比上个冒号多空两个空格。

5、Spring Boot 的核心注解是哪个？它主要由哪几个注解组成的？

启动类上面的注解是@SpringBootApplication，它也是 Spring Boot 的核心注解，主要组合包含了以下 3 个注解：

* 1. @SpringBootConfiguration

组合了 @Configuration 注解，实现配置文件的功能。

* 1. @EnableAutoConfiguration

打开自动配置的功能，也可以关闭某个自动配置的选项，如关闭数据源自动配置功能： @SpringBootApplication(exclude = { DataSourceAutoConfiguration.class })。

* 1. @ComponentScan

Spring组件扫描。

注：使用@SpringBootApplication 注解来开启 Spring Boot 的各项能力，如自动配置、组件扫描等

6、开启 Spring Boot 特性有哪几种方式？

1）继承spring-boot-starter-parent项目

2）导入spring-boot-dependencies项目依赖

7、运行 Spring Boot 有哪几种方式？

1）打包用命令或者者放到容器中运行

2）用 Maven/ Gradle 插件运行

3）直接执行 main 方法运行

8、Spring Boot 自动配置原理是什么？

注解 @EnableAutoConfiguration, @Configuration, @ConditionalOnClass 就是自动配置的核心，首先它得是一个配置文件，其次根据类路径下能否有这个类去自动配置。

9、你如何了解 Spring Boot 中的 Starters？

以spring-boot-starter-XXX命名的，代表了一个特定的应用类型；且第三方的启动器不能以spring-boot开头命名，它们都被Spring Boot官方保留。

Starters可以了解为启动器，它包含了一系列可以集成到应用里面的依赖包，你可以一站式集成 Spring 及其余技术，而不需要四处找示例代码和依赖包。如你想使用 Spring JPA 访问数据库，只需加入 spring-boot-starter-data-jpa 启动器依赖就能使用了。

Starters包含了许多项目中需要用到的依赖，它们能快速持续的运行，都是一系列得到支持的管理传递性依赖。

10、如何在 Spring Boot 启动的时候加载内容，如何运行少量特定的代码？

容器启动的时候执行一些内容，比如读取配置文件，数据库连接之类的。可以实现接口 ApplicationRunner 或者 CommandLineRunner，这两个接口实现方式一样，它们都只提供了一个 run 方法。

而，SpringApplicationBuilder更灵活但需要更多的代码。

11、Spring Boot 有哪几种读取配置的方式？

Spring Boot 可以通过 @PropertySource、@Value、@Environment,、@ConfigurationProperties 来绑定变量。

12、Spring Boot 支持哪些日志框架？推荐和默认的日志框架是哪个？

Spring Boot 支持 Java Util Logging, Log4j2, Lockback 作为日志框架，假如你使用 Starters 启动器，Spring Boot 将使用 Logback 作为默认日志框架。

13、pringBoot 实现热部署有哪几种方式？

主要有Spring Loaded和Spring-boot-devtools两种方式

14、Spring Boot 配置加载顺序：优先的被覆盖，后面的被使用。

1）配置方式：在 Spring Boot 里面，可以使用以下几种方式来加载配置。

1. properties文件（默认）；
2. YAML文件（默认）；
3. 系统环境变量；
4. 命令行参数；

等等……

2）加载顺序：配置文件的目录扫描顺序如下

1. 工程根目录:./config/
2. 工程根目录：./
3. classpath:/config/
4. classpath:/

注：加载的优先级顺序是从上向下加载，并且所有的文件都会被加载。

3）覆盖：后来者被用。

高优先级的内容会覆低优先级的内容，形成互补配置。举几个常用例子，优先级从高到低如下：

* 1. 操作系统环境变量参数
  2. application-{profile}.properties（YAML）
  3. application.properties
  4. `@Configuration`配置文件上 `@PropertySource` 注解加载的参数
  5. 默认参数（通过 `SpringApplication.setDefaultProperties` 指定）

注：优先级越高越被采用。

4）指定默认

* + 也可以通过指定配置spring.config.location来改变默认配置。
  + 或者在打包后，我们可以通过指令java -jar xxxx.jar --spring. config.location=D:/kawa/application.yml来加载外部的配置

###### Springboot开发

1、优点简述

敏捷开发

内置tomcat

无需xml配置

2、搭建

添加依赖包：利用了maven的继承和依赖特性！

parent，把基本包都已经添加了

web，添加了web应用所需包

3、创建服务

1）服务即是被作为请求的接口

2）@restController=@Controller+@RespondBody，返回josn字符串

3）添加注解

添加@SpringbootApplication或者@SpringBootConfigeration+@EnableAutoConfigeration+@ComponentScan便可运行了

有此注解便可注入到springboot容器，添加到tomcat，加载运行springboot

4）写springcloud：一套简单易懂、易部署和易维护的分布式系统开发工具包。

基本上是在写springmvc，其它是分布式系统的创建、（自动）部署和维护。