1.什么是微服务?

单个轻量级服务一般为一个单独微服务，**微服务讲究的是 专注某个功能的实现**，比如登录系统只专注于用户登录方面功能的实现，**讲究的是职责单一，开箱即用，可以独立运行。**微服务架构系统是一个分布式的系统，**按照业务进行划分服务单元模块**，解决单个系统的不足，满足越来越复杂的业务需求。

马丁福勒（Martin Fowler）：就目前而言，对于微服务业界并没有一个统一的、标准的定义。但通常而言，**微服务架构是一种架构模式或者说是架构风格，它提倡将单一应用程序划分成一组小的服务。**每个服务运行在其独立的自己的进程中，服务之间相互配合、相互协调，为用户提供最终价值。**服务之间采用轻量级通信。每个服务都围绕具体业务进行构建，**并能够独立部署到生产环境等。另外应尽量避免统一的、集中的服务管理机制。

通俗的来讲：

> **微服务就是一个独立的职责单一的服务应用程序。**在 intellij idea 工具里面就是用maven开发的一个个独立的module，具体就是使用springboot 开发的一个小的模块，处理单一专业的业务逻辑，一个模块只做一个事情。

微服务强调的是服务大小，关注的是某一个点，具体解决某一个问题/落地对应的一个服务应用，可以看做是idea 里面一个 module。

比如你去医院：你的牙齿不舒服，那么你就去牙科。你的头疼，那么你就去脑科。一个个的科室，就是一个微服务，一个功能就是一个服务。

业界大牛 马丁福勒（Martin Fowler）讲解：

> https://martinfowler.com/bliki/

看不懂英文，这里有中文博客翻译的：

> https://blog.csdn.net/u013970991/article/details/53333921

参考：【120期】面试官：谈谈什么是微服务?

2.微服务之间如何独立通讯的?

同步通信：dobbo通过 RPC 远程过程调用、springcloud通过 REST 接口json调用 等。

异步：消息队列，如：RabbitMq、ActiveMq、Kafka 等。

3.SpringCloud 和 Dubbo 有哪些区别?

首先，他们都是分布式管理框架。

dubbo 是二进制传输，占用带宽会少一点。SpringCloud是http 传输，带宽会多一点，同时使用http协议一般会使用JSON报文，消耗会更大。

dubbo 开发难度较大，所依赖的 jar 包有很多问题大型工程无法解决。SpringCloud 对第三方的继承可以一键式生成，天然集成。

SpringCloud 接口协议约定比较松散，需要强有力的行政措施来限制接口无序升级。

最大的区别:

Spring Cloud抛弃了Dubbo 的RPC通信，采用的是基于HTTP的REST方式。

严格来说，这两种方式各有优劣。虽然在一定程度上来说，后者牺牲了服务调用的性能，但也避免了上面提到的原生RPC带来的问题。而且REST相比RPC更为灵活，服务提供方和调用方的依赖只依靠一纸契约，不存在代码级别的强依赖，这在强调快速演化的微服务环境下，显得更为合适。

	Dubbo	Spring Cloud
服务注册中心	Zookeeper	Spring Cloud Netflix Eureka
服务调用方式	RPC	REST API
服务监控	Dubbo-monitor	Spring Boot Admin
断路器	不完善	Spring Cloud Netflix Hystrix
服务网关	无	Spring Cloud Netflix Zuul
分布式配置	无	Spring Cloud Config
服务跟踪	无	Spring Cloud Sleuth
消息总线	无	Spring Cloud Bus
数据流	无	Spring Cloud Stream
批量任务	无	Spring Cloud Task

4.SpringBoot 和 SpringCloud 之间关系?

SpringBoot：专注于快速方便的开发单个个体微服务（关注微观）；SpringCloud：关注全局的微服务协调治理框架，将SpringBoot开发的一个个单体微服务组合并管理起来（关注宏观）；

SpringBoot可以离开SpringCloud独立使用，但是SpringCloud不可以离开SpringBoot，属于依赖关系。

参考：

> https://blog.csdn.net/qq_41497111/article/details/91042405

5.什么是熔断？什么是服务降级?

服务熔断的作用类似于我们家用的保险丝，当某服务出现不可用或响应超时的情况时，为了防止整个系统出现雪崩，暂时停止对该服务的调用。

服务降级是从整个系统的负荷情况出发和考虑的，对某些负荷会比较高的情况，为了预防某些功能（业务场景）出现负荷过载或者响应慢的情况，在其内部暂时舍弃对一些非核心的接口和数据的请求，而直接返回一个提前准备好的fallback（退路）错误处理信息。这样，虽然提供的是一个有损的服务，但却保证了整个系统的稳定性和可用性。

参考：【121期】面试官：什么是熔断？什么是服务降级？

6.微服务的优缺点是什么？说下你在项目中碰到的坑。

优点：松耦合，聚焦单一业务功能，无关开发语言，团队规模降低。在开发中，不需要了解多有业务，只专注于当前功能，便利集中，功能小而精。微服务一个功能受损，对其他功能影响并不是太大，可以快速定位问题。微服务只专注于当前业务逻辑代码，不会和 html、css 或其他界面进行混合。可以灵活搭配技术，独立性比较舒服。

缺点：随着服务数量增加，管理复杂，部署复杂，服务器需要增多，服务通信和调用压力增大，运维工程师压力增大，人力资源增多，系统依赖增强，数据一致性，性能监控。

7.eureka和zookeeper都可以提供服务注册与发现的功能，请说说两个的区别?

- zookeeper 是CP原则，强一致性和分区容错性。
- eureka 是AP 原则 可用性和分区容错性。
- zookeeper当主节点故障时，zk会在剩余节点重新选择主节点，耗时过长，虽然最终能够恢复，但是选取主节点期间会导致服务不可用，这是不能容忍的。
- eureka各个节点是平等的，一个节点挂掉，其他节点仍会正常保证服务。

8.你所知道微服务的技术栈有哪些？列举一二。

微服务条目	落地技术
服务开发	SpringBoot、Spring、SpringMVC
服务配置与管理	Netfix公司的Archaius、阿里的Diamond等
服务注册与发现	Eurka、Consul、Zookeeper等
服务调用	Rest（服务通信）、RPC（Dubbo）、GRpc
服务熔断器	Hystrix、Envoy等
负载均衡	Nginx、Ribbon等
服务接口调用（客户端简化工具）	Fegin等
消息队列	Kafka、RabbitMQ、ActiveMQ等
服务配置中心管理	SpringCloudConfig、Chef等
服务路由（API网关）	Zuul等
服务监控	Zabbix，Nagios，Metrics，Spectator等
全链路追踪	Zipkin，Brave，Dapper等
服务部署	Docker，OpenStack，Kubernetes等
数据流操作开发包	SpringCloud Stream（封装与Redis，Rabbit，kafka等发送接收消息）
事件消息总线	Spring Cloud Bus

9.什么是微服务架构?

在前面你理解什么是微服务，那么对于微服务架构基本上就已经理解了。

微服务架构 就是 对微服务进行管理整合应用的。微服务架构 依赖于 微服务，是在微服务基础之上的。

例如：上面已经列举了什么是微服务。在医院里，每一个科室都是一个独立的微服务，那么 这个医院就是 一个大型的微服务架构，就类似 院长 可以 对下面的 科室进行管理。微服务架构主要就是这种功能。