一、什么是微服务

1、微服务的由来

微服务最早由Martin Fowler与James Lewis于2014年共同提出，微服务架构风格是一种使用一套小服务来开发单个应用的方式途径，每个服务运行在自己的进程中，并使用轻量级机制通信，通常是HTTP API，这些服务基于业务能力构建，并能够通过自动化部署机制来独立部署，这些服务使用不同的编程语言实现，以及不同数据存储技术，并保持最低限度的集中式管理。

2、为什么需要微服务

在传统的IT行业软件大多都是各种独立系统的堆砌，这些系统的问题总结来说就是扩展性差，可靠性不高，维护成本高。到后面引入了SOA服务化，但是，由于 SOA 早期均使用了总线模式，这种总线模式是与某种技术栈强绑定的，比如：J2EE。这导致很多企业的遗留系统很难对接，切换时间太长，成本太高，新系统稳定性的收敛也需要一些时间。

3、微服务与单体架构区别

（1）单体架构所有的模块全都耦合在一块，代码量大，维护困难。

微服务每个模块就相当于一个单独的项目，代码量明显减少，遇到问题也相对来说比较好解决。

（2）单体架构所有的模块都共用一个数据库，存储方式比较单一。

微服务每个模块都可以使用不同的存储方式（比如有的用redis，有的用mysql等），数据库也是单个模块对应自己的数据库。

（3）单体架构所有的模块开发所使用的技术一样。

微服务每个模块都可以使用不同的开发技术，开发模式更灵活。

4、微服务本质

（1）微服务，关键其实不仅仅是微服务本身，而是系统要提供一套基础的架构，这种架构使得微服务可以独立的部署、运行、升级，不仅如此，这个系统架构还让微服务与微服务之间在结构上“松耦合”，而在功能上则表现为一个统一的整体。这种所谓的“统一的整体”表现出来的是统

一风格的界面，统一的权限管理，统一的安全策略，统一的上线过程，统一的日志和审计方法，统一的调度方式，统一的访问入口等等。

（2）微服务的目的是有效的拆分应用，实现敏捷开发和部署 。

（3）微服务提倡的理念团队间应该是 inter-operate, not integrate 。inter-operate是定义好系统的边界和接口，在一个团队内全栈，让团队自治，原因就是因为如果团队按照这样的方式组建，将沟通的成本维持在系统内部，每个子系统就会更加内聚，彼此的依赖耦合能变弱，跨系统的沟通成本也就能降低。

5、什么样的项目适合微服务

微服务可以按照业务功能本身的独立性来划分，如果系统提供的业务是非常底层的，如：操作系统内核、存储系统、网络系统、数据库系统等等，这类系统都偏底层，功能和功能之间有着紧密的配合关系，如果强制拆分为较小的服务单元，会让集成工作量急剧上升，并且这种人为的切割无法带来业务上的真正的隔离，所以无法做到独立部署和运行，也就不适合做成微服务了。

6、微服务开发框架

目前微服务的开发框架，最常用的有以下四个：

Spring Cloud：http://projects.spring.io/spring-cloud（现在非常流行的微服务架构）

Dubbo：http：//dubbo.io

Dropwizard：http://www.dropwizard.io （关注单个微服务的开发）

Consul、etcd&etc.（微服务的模块）

7、什么是Spring Cloud

Spring Cloud是一系列框架的集合。它利用Spring Boot的开发便利性简化了分布式系统基础设施的开发，如服务发现、服务注册、配置中心、消息总线、负载均衡、 熔断器、数据监控等，都可以用Spring Boot的开发风格做到一键启动和部署。Spring并没有重复制造轮子，它只是将目前各家公司开发的比较成熟、经得起实际考验的服务框架组合起来，通过SpringBoot风格进行再封装屏蔽掉了复杂的配置和实现原理，最终给开发者留出了一套简单易懂、易部署和易维护的分布式系统开发工具包

8、Spring Cloud和Spring Boot是什么关系

Spring Boot 是 Spring 的一套快速配置脚手架，可以基于Spring Boot 快速开发单个微服务，Spring Cloud是一个基于Spring Boot实现的开发工具；Spring Boot专注于快速、方便集成的单个微服务个体，Spring Cloud关注全局的服务治理框架； Spring Boot使用了默认大于配置的理念，很多集成方案已经帮你选择好了，能不配置就不配置，Spring Cloud很大的一部分是基于Spring Boot来实现，必须基于Spring Boot开发。可以单独使用Spring Boot开发项目，但是Spring Cloud离不开 Spring Boot。

9、Spring Cloud相关基础服务组件

服务发现——Netflix Eureka （Nacos）

服务调用——Netflix Feign

熔断器——Netflix Hystrix

服务网关——Spring Cloud GateWay

分布式配置——Spring Cloud Config （Nacos）

消息总线 —— Spring Cloud Bus （Nacos）

10、Spring Cloud的版本

Spring Cloud并没有熟悉的数字版本号，而是对应一个开发代号。

Cloud代号	Boot版本(train)	Boot版本(tested)	lifecycle
Angle	1.2.x	incompatible with 1.3	EOL in July 2017
Brixton	1.3.x	1.4.x	2017-07卒
Camden	1.4.x	1.5.x	-
Dalston	1.5.x	not expected 2.x	-
Edgware	1.5.x	not expected 2.x	-
Finchley	2.0.x	not expected 1.5.x	-
Greenwich	**2.1.x**		
Hoxton	2.2.x		

开发代号看似没有什么规律，但实际上首字母是有顺序的，比如：Dalston版本，我们可以简称 D 版本，对应的 Edgware 版本我们可以简称 E 版本。

小版本

Spring Cloud 小版本分为:

SNAPSHOT： 快照版本，随时可能修改

M： MileStone，M1表示第1个里程碑版本，一般同时标注PRE，表示预览版版。

SR： Service Release，SR1表示第1个正式版本，一般同时标注GA：(GenerallyAvailable),表示稳定版本。