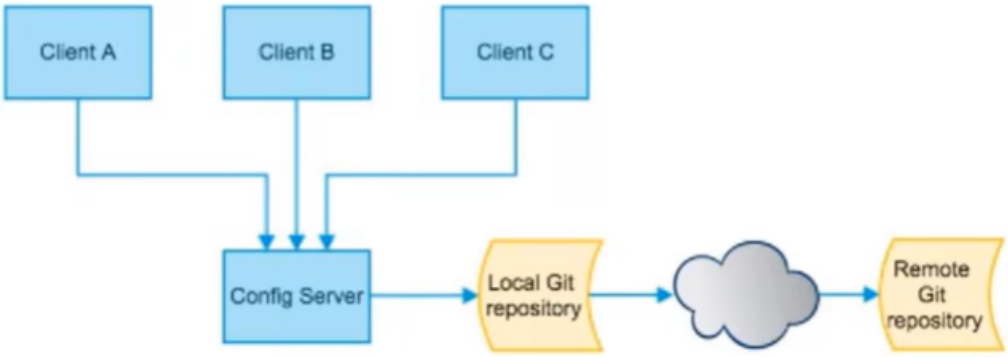
Spring Cloud Config分布式配置

概念

分布式系统面临的–配置文件问题

- 微服务意味着要将单体应用中的业务拆分成一个个子服务，每个服务的粒度相对较小，因此系统中会出现大量的服务，由于每个服务都需要必要的配置信息才能运行，所以一套集中式的，动态的配置管理设施是必不可少的。spring cloud提供了configServer来解决这个问题，我们每一个微服务自己带着一个application.properties，那上百个的配置文件修改起来，令人头疼！



什么是Spring Cloud config分布式配置中心?

spring cloud config 为微服务架构中的微服务提供集中化的外部支持，配置服务器为各个不同微服务应用的所有环节提供了一个**中心化的外部配置。**

spring cloud config 分为**服务端**和**客户端**两部分。

服务端也称为 **分布式配置中心**，它是一个独立的微服务应用，用来连接配置服务器并为客户端提供获取配置信息，加密，解密信息等访问接口。

客户端则是**通过指定的配置中心来管理应用资源，以及与业务相关的配置内容，并在启动的时候从配置中心获取和加载配置信息。**配置服务器默认采用git来存储配置信息，这样就有助于对环境配置进行版本管理。并且可用通过git客户端工具来方便的管理和访问配置内容。

spring cloud config 分布式配置中心能干嘛?

- 集中式管理配置文件
- 不同环境，不同配置，动态化的配置更新，分环境部署，比如 /dev /test /prod /beta /release
- 运行期间动态调整配置，不再需要在每个服务部署的机器上编写配置文件，服务会向配置中心统一拉取配置自己的信息
- 当配置发生变动时，服务不需要重启，即可感知到配置的变化，并应用新的配置
- 将配置信息以REST接口的形式暴露

spring cloud config 分布式配置中心与GitHub整合

由于spring cloud config 默认使用git来存储配置文件 (也有其他方式，比如自持SVN 和本地文件)，但是最推荐的还是git ， 而且使用的是 http / https 访问的形式。

Git环境搭建

安装Git

Git-2.40.0-64-bit

登录Gitee，注册账号

网址：https://gitee.com/

配置Git

参考网址：https://www.cnblogs.com/xiong950413/p/16498644.html

克隆远程仓库到本地

- 在Gitee上新建仓库

- 将Gitee上刚刚创建的仓库克隆到本地

  命令：git clone url

上传application.properties文件到远程仓库

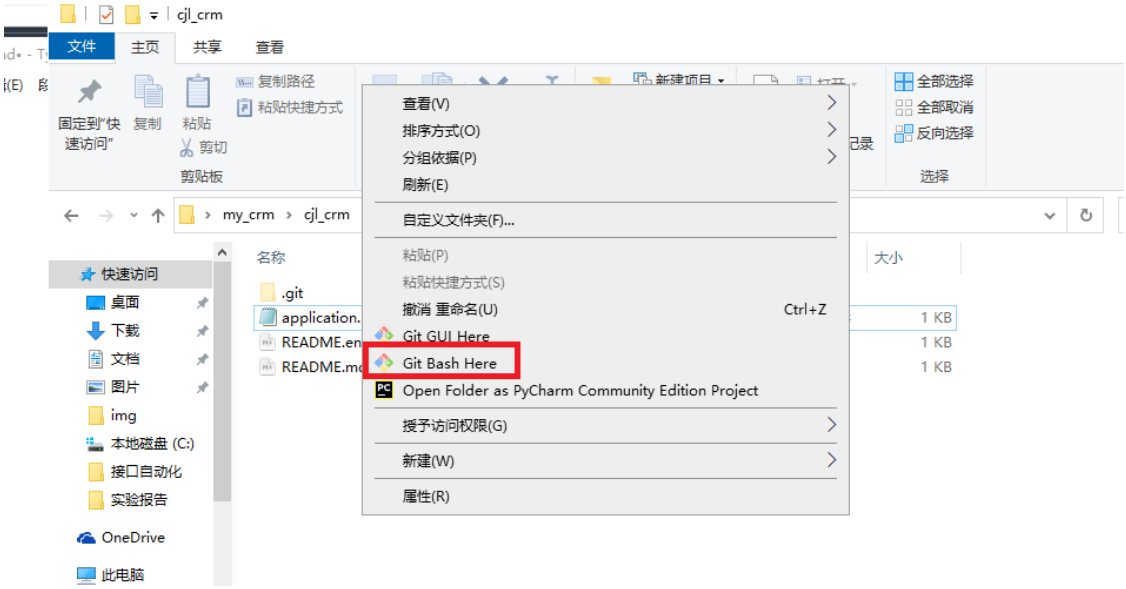
- 本地编写application.properties文件

```
spring.profiles.active:dev

spring.profiles:dev
spring.application.name:springcloud-config-dev

spring.profiles:test
spring.application.name:springcloud-config-test
```

- 输入命令上传文件




```
$ git add .
$ git commit -m "add profiles"
$ git pull origin master
$ git push -u origin  master
```

入门案例

- 创建module：springcloud-config-server-3344
- 添加依赖

```
<dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>

        <dependency>
            <groupId>org.springframework.cloud</groupId>
            <artifactId>spring-cloud-starter-eureka-server</artifactId>
            <version>1.4.6.RELEASE</version>
        </dependency>

        <dependency>
            <groupId>org.springframework.cloud</groupId>
            <artifactId>spring-cloud-config-server</artifactId>
            <version>2.2.1.RELEASE</version>
        </dependency>

</dependencies>
```

- 创建配置文件

```
server.port=3344

spring.application.name=spring-config-server

spring.cloud.config.server.git.uri=https://gitee.com/chenjunlan001/cjl_crm.git
```

- 创建微服务启动类

```
package com.cjl.springcloud;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.cloud.netflix.zuul.EnableZuulProxy;

@SpringBootApplication
@EnableZuulProxy
public class ZuulApplication_9527 {
    public static void main(String[] args) {
        SpringApplication.run(ZuulApplication_9527.class, args);
    }
}
```

- 如果出现报错，在application.properties中配置git的用户名和密码

```
spring.cloud.config.server.git.username=
spring.cloud.config.server.git.password=
```