

# 服务配置



扫码试看/订阅  
《玩转 Spring 全家桶》

# 基于 Git 的配置中心

# Spring Cloud Config Server

## 目的

- 提供针对外置配置的 HTTP API 默认localhost:8888

## 依赖

- spring-cloud-config-server
- @EnableConfigServer
- 支持 Git / SVN / Vault / JDBC ...

# 使用 Git 作为后端存储

## 配置

- `MultipleJGitEnvironmentProperties`
  - `spring.cloud.config.server.git.uri` 告诉config server git的URI是什么

## 配置文件的要素

- `{application}`, 即客户端的 `spring.application.name`
- `{profile}`, 即客户端的 `spring.profiles.active` 不同Profile逗号分割
- `{label}`, 配置文件的特定标签, 默认 master

# 使用 Git 作为后端存储

## 访问配置内容

- HTTP 请求
  - GET `/ {application} / {profile} [ / {label} ]` 获得不同branch下面
  - GET `/ {application} - {profile} . yml`
  - GET `/ {label} / {application} - {profile} . yml`
  - GET `/ {application} - {profile} . properties`
  - GET `/ {label} / {application} - {profile} . properties`

**“Talk is cheap, show me the code.”**

*Chapter 14 / config-server*

# Spring Cloud Config Client

## 依赖

- `spring-cloud-starter-config`

## 发现配置中心

- `bootstrap.properties` | `yml`
- `spring.cloud.config.fail-fast=true`
- 通过配置
  - `spring.cloud.config.uri=http://localhost:8888`



# Spring Cloud Config Client

## 发现配置中心

- bootstrap.properties | yml
- 通过服务发现
  - `spring.cloud.config.discovery.enabled=true`
  - `spring.cloud.config.discovery.service-id=configserver`

## 配置刷新

- @RefreshScope
- Endpoint - /actuator/refresh

**“Talk is cheap, show me the code.”**

*Chapter 14 / git-config-waiter-service*

# 基于 Zookeeper 的配置中心

# Spring Cloud Zookeeper Config

## 依赖

- `spring-cloud-starter-zookeeper-config`
- 注意 Zookeeper 版本

## 启用

- `bootstrap.properties | yml`
  - `spring.cloud.zookeeper.connect-string=localhost:2181`
  - `spring.cloud.zookeeper.config.enabled=true`

# Zookeeper 中的数据怎么存

## 配置项

- `/config/应用名,profile/key=value`
- `/config/application,profile/key=value`

## 如何定制

- `spring.cloud.zookeeper.config.root=config`
- `spring.cloud.zookeeper.config.default-context=application`
- `spring.cloud.zookeeper.config.profile-separator=','`

**“Talk is cheap, show me the code.”**

*Chapter 14 / zk-config-waiter-service*

# 深入理解 Spring Cloud 的配置抽象

# Spring Cloud Config

## 目标

- 在分布式系统中，提供外置配置支持

## 实现

- 类似于 Spring 应用中的 Environment 与 PropertySource
- 在上下文中增加 Spring Cloud Config 的 PropertySource



# Spring Cloud Config 的 PropertySource

## PropertySource

- Spring Cloud Config Client - CompositePropertySource
- Zookeeper - ZookeeperPropertySource
- Consul - ConsulPropertySource / ConsulFilesPropertySource

## PropertySourceLocator

- 通过 PropertySourceLocator 提供 PropertySource

# Spring Cloud Config Server

## EnvironmentRepository

- Git / SVN / Vault / JDBC ...

## 功能特性

- SSH、代理访问、配置加密 ...

## 配置刷新

- /actuator/refresh
- Spring Cloud Bus - RefreshRemoteApplicationEvent

# Spring Cloud Config Zookeeper

## **ZookeeperConfigBootstrapConfiguration**

- 注册 ZookeeperPropertySourceLocator
- 提供 ZookeeperPropertySource

## **ZookeeperConfigAutoConfiguration**

- 注册 ConfigWatcher

# 配置的组合顺序

以 yml 为例

- 应用名-profile.yml
- 应用名.yml
- application-profile.yml
- application.yml

# 基于 Consul 的配置中心

# Spring Cloud Consul Config

## 依赖

- `spring-cloud-starter-consul-config`

## 启用

- `bootstrap.properties | yml`
  - `spring.cloud.consul.host=localhost`
  - `spring.cloud.consul.port=8500`
  - `spring.cloud.consul.config.enabled=true`

# Consul 中的数据怎么存

## 配置项

- `spring.cloud.consul.config.format=`  
`KEY_VALUE | YAML | PROPERTIES | FILES`
- `/config/应用名,profile/data`
- `/config/application,profile/data`

# Consul 中的数据怎么存

## 如何定制

- `spring.cloud.consul.config.data-key=data`
- `spring.cloud.consul.config.root=config`
- `spring.cloud.consul.config.default-context=application`
- `spring.cloud.consul.config.profile-separator=','`



# 配置项变更

## 自动刷新配置

- `spring.cloud.consul.config.watch.enabled=true`
- `spring.cloud.consul.config.watch.delay=1000`

## 实现原理

- 单线程 `ThreadPoolTaskScheduler`
- `ConsulConfigAutoConfiguration.CONFIG_WATCH_TASK_SCHEDULER_NAME`

**“Talk is cheap, show me the code.”**

*Chapter 14 / consul-config-waiter-service*

# 基于 Nacos 的配置中心

# Spring Cloud Alibaba Nacos Config

## 依赖

- `spring-cloud-starter-alibaba-nacos-config`
- `spring-cloud-alibaba-dependencies:0.9.0`
  - 注意 Spring Cloud 与 Spring Boot 的对应版本

## 启用

- `bootstrap.properties | yml`
  - `spring.cloud.nacos.config.server-addr=127.0.0.1:8848`
  - `spring.cloud.nacos.config.enabled=true`

# Nacos 中的数据怎么存

## 配置项

- dataId
  - `${prefix}-${spring.profile.active}.${file-extension}`
  - `spring.cloud.nacos.config.prefix`
  - `spring.cloud.nacos.config.file-extension`
- `spring.cloud.nacos.config.group`

**“Talk is cheap, show me the code.”**

*Chapter 14 / nacos-config-waiter-service*

# SpringBucks 实战项目进度小结

# 本章小结

## 几种不同的配置中心

- Spring Cloud Config Server
  - Git / SVN / RDBMS / Vault
- Zookeeper
- Consul
- Nacos



# SpringBucks 进度小结

## **waiter-service**

- 增加了订单金额与折扣
- 增加了 Waiter 名称
- 使用了不同的配置中心
  - Spring Cloud Config Client
  - 使用 Zookeeper
  - 使用 Consul
  - 使用 Nacos

# 携程 Apollo

## 官方地址

- <https://github.com/ctripcorp/apollo>

## 特性

- 统一管理不同环境、不同集群的配置
- 配置修改实时生效（热发布）
- 版本发布管理
- 灰度发布
- 权限管理、发布审核、操作审计
- 客户端配置信息监控
- 提供开放平台API



扫码试看/订阅  
《玩转 Spring 全家桶》