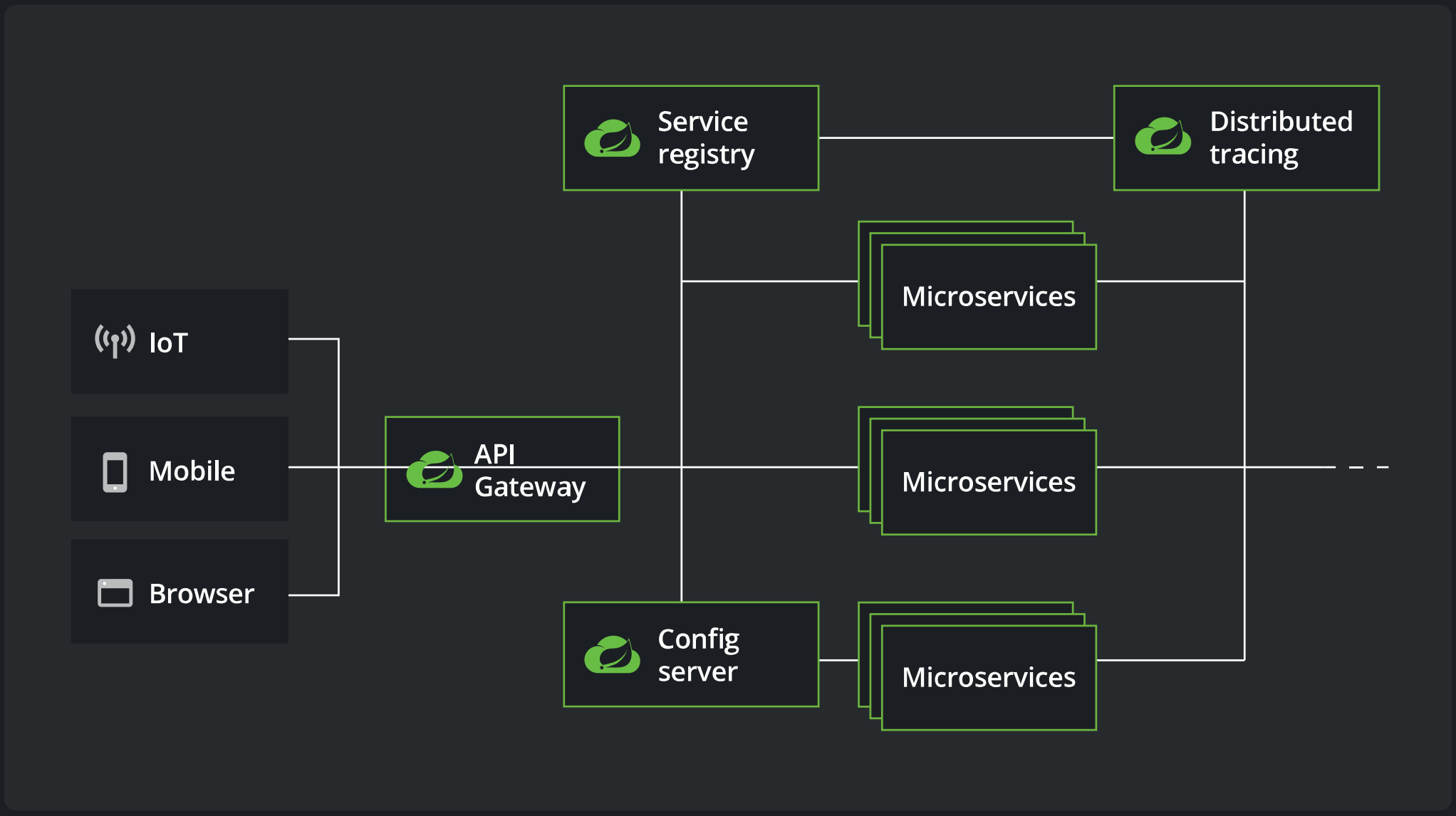
Cloud

Developing distributed systems can be challenging. Complexity is moved from the application layer to the network layer and demands greater interaction between services. Making your code ‘cloud-native’ means dealing with 12-factor issues such as external configuration, statelessness, logging, and connecting to backing services. The Spring Cloud suite of projects contains many of the services you need to make your applications run in the cloud.

Spring Cloud architecture highlights



Service discovery

In the cloud, applications can't always know the exact location of other services. A service registry, such as Netflix Eureka, or a sidecar solution, such as HashiCorp Consul, can help. Spring Cloud provides `DiscoveryClient` implementations for popular registries such as

Eureka, Consul, Zookeeper, and even Kubernetes' built-in system. There's also a Spring Cloud Load Balancer to help you distribute the load carefully among your service instances.

Get started with this simple guide

API gateway

With so many clients and servers in play, it's often helpful to include an API gateway in your cloud architecture. A gateway can take care of securing and routing messages, hiding services, throttling load, and many other useful things. Spring Cloud Gateway gives you precise control of your API layer, integrating Spring Cloud service discovery and client-side load-balancing solutions to simplify configuration and maintenance.

Getting Started with Spring Cloud Gateway

Cloud configuration

In the cloud, configuration can't simply be embedded inside the application. The configuration has to be flexible enough to cope with multiple applications, environments, and service instances, as well as deal with dynamic changes without downtime. Spring Cloud Config is designed to ease these burdens and offers integration with version control systems like Git to help you keep your configuration safe.

Try it now

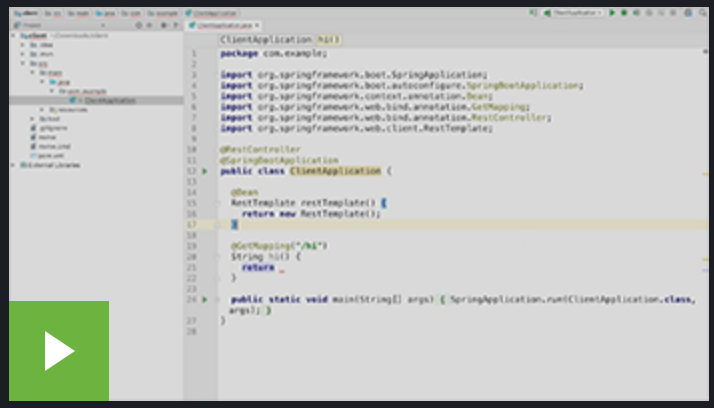
Circuit breakers

Distributed systems can be unreliable. Requests might encounter timeouts or fail completely. A circuit breaker can help mitigate these issues, and Spring Cloud Circuit Breaker gives you the choice of three popular options: Resilience4J, Sentinel, or Hystrix.

Try this guide to get started

Tracing

Debugging distributed applications can be complex and take a long time. For any given failure, you might need to piece together traces of information from several independent services. Spring Cloud Sleuth can instrument your applications in a predictable and repeatable way. And when used in conjunction with Zipkin, you can zero in on any latency problems you might have.



VIDEO

Spring Tips: Zipkin and Distributed Tracing

Testing

In the cloud, you get extra points for having reliable, trustworthy, stable APIs—but getting there can be a journey. Contract-based testing is one technique that high-performing teams often use to stay on track. It helps by formalizing the content of APIs and building tests around them to ensure code remains in check.

Spring Cloud Contract provides contract-based testing support for REST and messaging-based APIs with contracts written in Groovy, Java, or Kotlin.

Try this guide to get started