Monoliths to Microservices
Cloud Native
Cloud Native

Independently Deployable

Deploying Spring Boot Applications

Spring Boot applications can be deployed into production systems with various methods. In this article, we will go through step by step deployment of Spring Boot applications via the following 5 methods:
Cloud Native

Service Registry Integration
Cloud Native

Distributed Configuration
Route Configuration: YAML

```yaml
spring:
  cloud:
    gateway:
      - id: foo_route
        uri: lb://foo
        predicates:
          - Host=**.foo.org
          - Path=/headers
          - Method=GET
          - Header=X-Request-Id, \d+
          - Query=foo, ba.
          - Query=baz
          - Cookie=chocolate, ch.p
          - After=1900-01-20T17:42:47.789-07:00[America/Denver]
        filters:
          - AddRequestHeader=X-Request-Foo, Bar
          - AddResponseHeader=X-Response-Foo, Bar
          - Hystrix=foo
          - SecureHeaders
          - RewritePath=/foo/(?<segment>.*), /${segment}
```

Unless otherwise indicated, these slides are © 2013-2018 Pivotal Software, Inc. and licensed under a Creative Commons Attribution-NonCommercial license: http://creativecommons.org/licenses/by-nc/3.0/
Cloud Native

Fault Tolerance
Teams Love It
Teams Love It
Reactor Netty: Non-Blocking
Route Configuration: Java

```java
@Bean
public RouteLocator customRouteLocator(RouteLocatorBuilder builder) {
    return builder.routes()
        .route("bypath", r -> r.path("/get").or().host("*.get.org").uri("http://httpbin.org"))
        .route("host_segment", r -> r.host("{sub}.yourhost.org")
            .filters(f -> f.addRequestHeader("X-Foo", "Bar-{sub}")).uri("http://httpbin.org"))
        .route("hystrix", r -> r.host("**.hystrix.org")
            .filters(f -> f.hystrix(config ->
                config.setName("mycmd").setFallbackUri("forward:/myfallback")))
            .uri("http://httpbin.org"))
        .route("rewrite", r -> r.host("**.rewrite.org")
            .filters(f -> f.rewritePath("/foo/(?<segment>.*")", "/${segment}"))
            .uri("http://httpbin.org"))
        .route("websockets", r -> r.path("/echo")
            .uri("ws://localhost:9000"))
        .build();
}
```
Route Configuration: Kotlin

```kotlin
val routeLocator = builder.routes {
    route(id = "test") {
        host("**.abc.org") and path("/image/png")
        filters {
            addResponseHeader("X-TestHeader", "foobar")
        }
        uri("http://httpbin.org:80")
    }
    route(id = "test2") {
        path("/image/webp") or path("/image/anotherone")
        filters {
            addResponseHeader("X-AnotherHeader", "baz")
            addResponseHeader("X-AnotherHeader-2", "baz-2")
        }
        uri("https://httpbin.org:443")
    }
}
```
Inside the Gateway

Client → Predicates → Pre-Filters → Global Filters → Post-Filters → Downstream Service
Inside the Gateway

Predicates

- Path
- Host
- Date/Time
- Method
- Headers
- Query Params
- Cookies
- Cookies
- Read Body
  ....
Inside the Gateway

Pre-Filters

Headers
Path
Rate Limiting
Cookies
Hystrix
Modify Body
...
Inside the Gateway

Gateway

Global Filters

Netty Router
Web Sockets
Load Balancer
Metrics
...

Client

Predicates

Handler Mapping

Handler

Post-Filters

Downstream Service
Inside the Gateway

Gateway

Handler Mapping

Web Handler

Client

Predicates

Pre-Filters

Global

Post Filters

Set status

Headers

Cookies

...
Custom Filters

```java
public class SetStatusGatewayFilterFactory implements GatewayFilter {
    public GatewayFilter apply(Config config) {
        HttpStatusHolder statusHolder = HttpStatusHolder.parse(config.status);

        return chain.filter(exchange)
            .then(Mono.fromRunnable(() -> {
                setResponseStatus(exchange, statusHolder);
            }));
    }
}
```
Demo!
Teams Love It

Management Console with:
Links to API Docs, Metrics & Tracing.
API Discoverability
<table>
<thead>
<tr>
<th>Details</th>
<th>Inbound / Outbound</th>
<th>Filters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host example to HTTP</td>
<td>HTTP → HTTP</td>
<td>AddResponseHeader X-Response-Defax</td>
</tr>
<tr>
<td>Host example to HTTP</td>
<td>HTTP → HTTP</td>
<td>PrefixPath prefix: /httpbin</td>
</tr>
<tr>
<td>Host example to HTTP</td>
<td>HTTP → HTTP</td>
<td></td>
</tr>
<tr>
<td>Host example to HTTP</td>
<td>HTTP → HTTP</td>
<td></td>
</tr>
<tr>
<td>Host example to HTTP</td>
<td>HTTP → HTTP</td>
<td></td>
</tr>
<tr>
<td>Host example to HTTP</td>
<td>HTTP → HTTP</td>
<td></td>
</tr>
</tbody>
</table>

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
Teams Love It

Releases, Not Tickets
Teams Love It

Polyglot Clients and Services
Security Loves It
Security Loves It

Single Sign On (SSO)
Security Loves It

Cross-Cutting Security
Operators Love It
Operators Love It

Deconstructing Monoliths
Operators Love It

Exports Metrics To Existing Systems
Micrometer Support
Operators Love It

Distributed Tracing Integration
Operators Love It

Request Rate Limiting
Operators Love It

Customized Actuator Endpoints
Operators Love It
Spring Cloud Gateway

The Cloud-Native Gateway Enterprises Love
Questions?

https://github.com/spring-cloud/spring-cloud-gateway

https://github.com/spring-cloud-samples/spring-cloud-gateway-sample

http://slides.com/spencer/spring-cloud-gateway

https://github.com/ryanjbaxter/gateway-s1p-2018

Spencer Gibb - @spencerbgibb
Ryan Baxter @ryanjbaxter
Stephane Maldini - @smaldini