# Health Check

Uma característica bastante importante que as aplicações devem implementar é o Health Check.

Nossas aplicações normalmente rodam em um ambiente gerenciado, ou seja, uma plataforma que é capaz de detectar se algo não está funcionando conforme o esperado, um crash de aplicação ou carga demasiada. Depois de notar essas falhas, as plataformas são capazes de tomar algumas ações para tentar remediar a aplicação, reiniciar ou mesmo desligar a aplicação e lançá-la novamente em outro servidor.

Para que isso seja possível precisamos expor alguma operação em nossa aplicação para que a plataforma possa utilizar e verificar se tudo está funcionando perfeitamente. Temos algumas alternativas para isso:

- Quando nossa aplicação expõe uma API REST podemos usar um endpoint *HTTP GET* para esse fim. Esse endpoint deve verificar se o que a aplicação precisa para funcionar está 100% operacional. Isso pode incluir banco de dados e serviços de mensageria, por exemplo.
- Se nossa aplicação é de processamento em lote ou batch, podemos expor uma funcionalidade via linha de comando, seguindo os mesmos princípios, ou seja, validando conexões externas das nossas aplicações.

## Readiness Check ou Liveness Check

Alguns ambientes gerenciados utilizam dois conceitos diferentes de health check para determinar a saúde das nossas aplicações e ações a serem tomadas caso algo de errado aconteça. Vejamos os exemplos:

- Geralmente nossas aplicações consomem um determinado tempo para estarem aptas a receber um fluxo de trabalho. Esse processo de preparação pode compreender a configuração de uma fila, conexão com um banco de dados ou um determinado pré-processamento. Durante esse intervalo de tempo é recomendado que nenhuma requisição seja redirecionada para a nossa aplicação em processo de preparação. Essa verificação é chamada de Readiness Check e caso a plataforma verifique que a aplicação ainda não está pronta para

utilização, não enviará tráfego para ela. Uma vez que a plataforma executa a verificação e nota que o status retornado foi de sucesso, as requisições começam a ser redirecionadas para esta nova instância de serviço.

- Agora imagine que nossa aplicação está sendo executada normalmente, mas por algum motivo deixou de ser capaz de processar as requisições que chegam até ela (problemas de vazamento de memória, deadlock, etc) e está presa em um estado de falha. Utilizando o Liveness Check a plataforma identifica que a aplicação não é capaz de processar as requisições e reiniciará o container da aplicação na tentativa de restaurá-la e permitir a retomada do seu funcionamento normal.

Uma vez que nossa aplicação expõe uma API REST podemos usar um endpoints *HTTP GET* para estes fins. Esses endpoints devem permitir o Readiness Check e Liveness Check, ou seja, verificar se a aplicação está apta a receber fluxo de trabalho ou requisições e se as requisições estão sendo processadas, respectivamente.

Processamentos em lote também podem implementar este conceito expondo operações de linha de comando, dessa forma, a plataforma cujo serviço está instalado é capaz de identificar se realmente o serviço está apto a começar o trabalho.

## Como configurar o Spring Boot Actuator?

O Spring Boot Actuator inclui vários recursos adicionais para ajudá-lo a monitorar e gerenciar sua aplicação, como por exemplo:

- Endpoint para monitoramento da saúde da aplicação (Health Check).
- Endpoint para expor métricas da aplicação.
- Endpoint para expor as propriedades da sua aplicação.

Super legal né!? Quer saber quais são os outros endpoints, acesse o link!

Vamos configurar?

1º Precisamos adicionar a seguinte dependência em seu arquivo pom.xml:

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

2º Precisamos verificar se está tudo configurado, para isso abra em seu navegador o seguinte endereço:

```
http://localhost:8080/actuator
```

Neste endpoint irá listar todos os endpoints configurados, como por exemplo:

```
# Endpoint para monitoramento da saúde da aplicação (Health Check)

http://localhost:8080/actuator/health
```

Eba! Está tudo OK!

Sim, mas antes de continuar com sua tarefa, aconselhamos dar uma lida no tópico abaixo, sobre segurança!

## Como implementar um Health Check utilizando Spring Boot Actuator?

Neste tutorial vamos aprender como fazer nosso próprio Health Check caso for necessário:

1º Precisamos criar nossa classe que irá representar o Health Check desejado, conforme código abaixo:

```
public class MeuHealthCheck {
    // Código omitido
}
```

2º Precisamos implementar a interface HealthIndicator que o Spring Boot Actuator nos provê, conforme código abaixo:

```
public class MeuHealthCheck implements HealthIndicator {

    @Override
    public Health health() {
        // TODO Precisamos implementar o método
        return null;
    }

}
```

Ao implementar essa interface, precisamos implementar o método health no qual é necessário retornar o objeto Health.

Este objeto é bastante simples, você precisa passar qual status do Health Check, como por exemplo

- DOWN: Status indicando que o componente ou subsistema sofreu uma falha inesperada.
- OUT_OF_SERVICE: Status indicando que o componente ou subsistema foi retirado de serviço e não deve ser usado.
- UNKNOWN: Status indicando que o componente ou subsistema está em um estado desconhecido.
- UP: Status indicando que o componente ou subsistema está funcionando conforme o esperado.

Caso necessário, você pode passar detalhes do seu Health Check, como por exemplo:

- Versão
- Descrição
- IP

Muito legal né? Vamos implementar?

```
public class MeuHealthCheck implements HealthIndicator {

    @Override
    public Health health() {
        Map<String, Object> details = new HashMap<>();
        details.put("versão", "1.2.3");
        details.put("descrição", "Meu primeiro Health Check customizado!");
        details.put("endereço", "127.0.0.1");

        return Health.status(Status.UP).withDetails(details).build();
    }

}
```

E para finalizar, precisamos dizer para o Spring Boot considerar essa implementação, para isto, precisamos adicionar a anotação @Component , conforme código abaixo:

```
@Component
public class MeuHealthCheck implements HealthIndicator {
    // Código omitido
}
```

Pronto! Criamos nosso primeiro Health Check customizado utilizando Spring Boot Actuator!

Para testar, basta abrir seu navegador e chamar o endereço http://localhost:8080/actuator/health !

## Dicas

- Incorpore o padrão de Health Check nos seus serviços, de maneira que todo serviço seja produzido já o implemente.
- Tente minimizar o tempo de preparação da sua aplicação sempre que possível. Esse tempo pode impactar sua regra de auto-scaling e alta-disponibilidade da sua aplicação, pois durante essa fase sua aplicação não pode receber fluxo de trabalho.
- É importante termos um trabalho de ajuste fino nessas configurações, pois elas impactam diretamente suas regras de alta-disponibilidade e auto-scaling
- Não deixe pública sua API, alinhe sempre com sua equipe as melhores práticas, como por exemplo:
- Adicionar autenticação
- Adicionar autorização
- Não negligencie as informações que você está expondo sobre a sua infraestrutura.

## Informações de Suporte

- O Spring Boot oferece uma maneira eficiente de lidar com Health Check. Aqui você pode descobrir como.
- Se você quer descobrir como o Kubernetes utiliza seu Health Check para garantir que sua aplicação esteja viva, esse link vai resolver o seu problema.
- Este padrão é bastante comum na arquitetura de Microsserviços. Se você deseja ver uma descrição detalhada, esse link é uma boa opção.

- Se você quer descobrir como o Kubernetes utiliza seu Readiness Check para garantir que sua aplicação esteja apta para receber fluxo de trabalho, esse link vai resolver o seu problema.
- Se você usa o Spring e quer saber como o framework suporta readiness check acesse aqui!
- Talvez você esteja pensando sobre segurança no Spring Boot Actuator? Aqui tem uma explicação do que entendemos que você deve considerar!
- Quer saber mais sobre Spring Boot Actuator? Acesse o link!
- Quer saber mais sobre o The Twelve-Factor App? Acesse o link!
- Gostaria de saber mais sobre Health Check no Spring Boot Actuator? Acesse o link!