Logging e Deploy

DEVinHouse

Parcerias para desenvolver a sua carreira

SENAI

<LAB365>

AGENDA

- Considerações iniciais
- Revisão do projeto
- Logging
- Deploy
- Material complementar

Configurando RollingFileAppender no Logback

- Vamos adicionar as configurações no logback-spring.xml

```
<!--Propriedade para pegar path do diretório para salvar os logs-->
<springProperty name="LOGS" source="logging.dir.path"/>
```

- Definimos o valor de **logging.dir.path** no arquivo application.properties
- Adicionaremos o bloco de configuração do RollingFileAppender

Configurando RollingFileAppender no Logback

```
<!-- Adicionamos o RollingFileAppender para gerar logs em arquivo -->
<appender name="RollingFile"
          class="ch.qos.logback.core.rolling.RollingFileAppender">
    <file>${LOGS}/spring-boot-logger.log</file>
    <encoder
            class="ch.qos.logback.classic.encoder.PatternLayoutEncoder">
        <Pattern>%d %p %C{1.} [%t] %m%n</Pattern>
    </encoder>

    <!-- Configuramos a política de geração dos arquivos de log -->
    <rollingPolicy
            class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
        <!-- rollover daily and when the file reaches 10 MegaBytes -->
        <fileNamePattern>${LOGS}/archived/spring-boot-logger-%d{yyyy-MM-dd}.%i.log
        </fileNamePattern>
        <timeBasedFileNamingAndTriggeringPolicy
                class="ch.qos.logback.core.rolling.SizeAndTimeBasedFNATP">
            <maxFileSize>10MB</maxFileSize>
        </timeBasedFileNamingAndTriggeringPolicy>
    </rollingPolicy>
</appender>
```

Configurando RollingFileAppender no Logback

```
<!--Configuramos o log de forma assincrona-->
<appender name="logglyAsync" class="ch.qos.logback.classic.AsyncAppender">
    <appender-ref ref="loggly"/>
</appender>

<!-- Qualquer log do nível de info para cima, vamos registrar as mensagens
 no appender do Loggly e RollingFile -->
<root level="info">
    <appender-ref ref="logglyAsync"/>
    <appender-ref ref="RollingFile" />
</root>
```

Deploy

O que é deploy?

- O verbo **deploy**, em inglês quer dizer **implantar**
- Em programação, seu sentido está intimamente relacionado à sua tradução literal: fazer um deploy, em termos práticos, significa colocar no ar alguma aplicação que teve seu desenvolvimento concluído
- Esta tarefa é extremamente comum dentro do escopo de trabalho dos programadores, embora seja muito comumente associada aos profissionais de infraestrutura, ou DevOps

Quais as formas de fazer deploy?

- Nos últimos anos as tecnologias voltadas à automatização deste processo evoluíram muito, tornando-os muito mais rápido e eficientes
- O que antes era feito somente de maneira manual, hoje pode ser feito através de ferramentas de fácil manuseio e com mínimo risco, auxiliando as aplicações a serem atualizadas de maneira muito mais segura

Estratégias de deploy

- **Rolling**
  - Esta estratégia consiste em fazer com que duas versões de uma mesma aplicação coexistem enquanto o deploy é executado
  - Na prática, este deploy é realizado substituindo totalmente o código fonte de uma aplicação por um novo código, que já contém as alterações implementadas
  - Este deploy é feito de modo gradual e a versão antiga só sairá do ar quando a nova estiver totalmente pronta

Estratégias de deploy

- **Blue-Green**
  - Esta estratégia tem a característica principal onde todos os testes de uma aplicação são feitos em um ambiente de produção espelhado, chamado *mirror*
  - Com essa estratégia, o **blue** representa o ambiente antigo, enquanto o **green** representa o ambiente atualizado
  - Para implementar o deploy nestes moldes, a aplicação é espelhada em produção, mas os usuários não têm acesso a ela enquanto estiver totalmente pronta. Quando o deploy é finalizado, um *load balancer* direciona o tráfego para o novo ambiente, enquanto o antigo é excluído

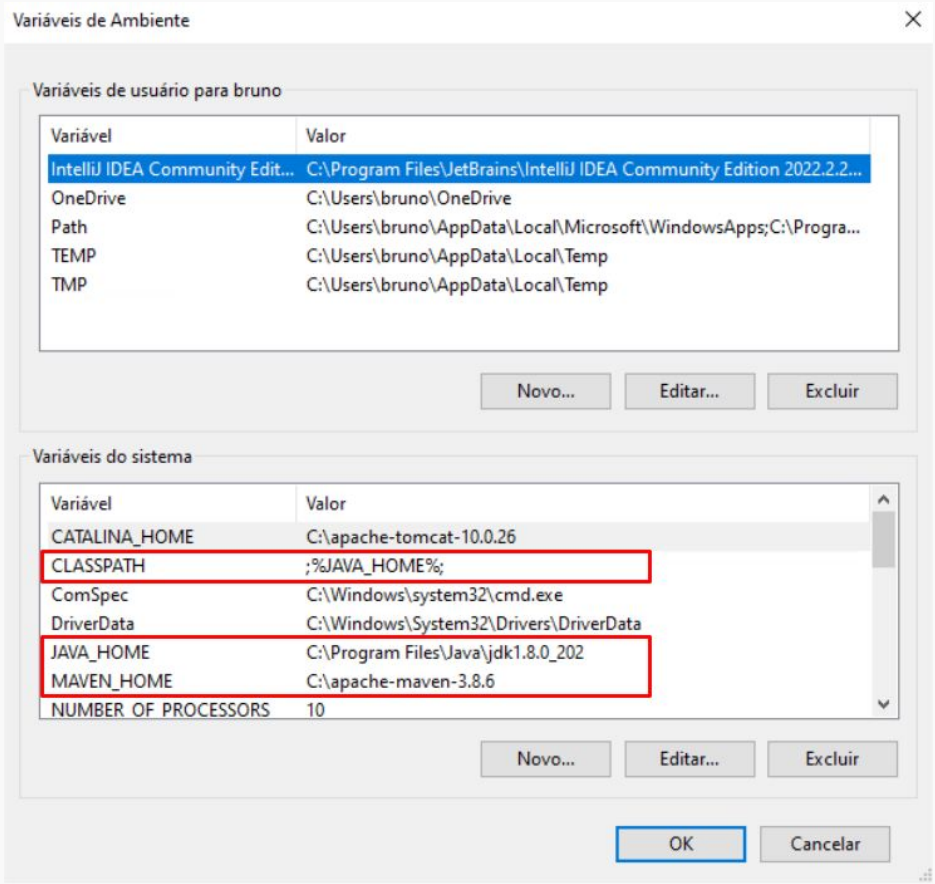
Estratégias de deploy

- **Canary**
  - Nesta estratégia os teste da nova versão são realizados na prática pelos seus usuários finais
  - Está é a mais complexa estratégia de deploy porque consiste em liberar a nova versão de um software, site ou aplicativo apenas para uma parte das pessoas que utilizam para que, em seguida, sejam feitas análises de seus comportamentos para a identificação de erros e falhas
  - Com Canary, a nova versão também coexiste com a antiga, com a diferença de que aqui duas versões estão em um ambiente de produção acessível aos usuários finais

Empacotando um projeto Spring Boot com Maven

- Baixar o Maven https://maven.apache.org/download.cgi
- Instalar o Maven no computador, pois o Maven que utilizamos atualmente é interno da IDE de desenvolvimento
- Descompacte o arquivo baixado na unidade C:\
- Configurar as variáveis de ambiente
  - MAVEN_HOME apontando para o diretório do Maven
  - JAVA_HOME apontando para o diretório de instalação do JDK
  - CLASSPATH definindo o valor ;%JAVA_HOME%;
- Configurar a variável Path incluindo o diretório bin do Maven
  - %MAVEN_HOME%\bin

Configurando as variáveis de ambiente



Como empacotar um projeto Spring

- Empacotando uma aplicação com Maven
  - **mvn clean package (limpa e empacota o projet)**
  - **mvn package (somente empacota)**
- Executando a aplicação
  - **java -jar agenda-clamed-1.0-SNAPSHOT.jar**

INTERVALO DE AULA

DEV!

Finalizamos o nosso primeiro período de hoje. Que tal descansar um pouco?!

Nos vemos em 20 minutos.

Início: 20:20
Retorno: 20:40

AVALIAÇÃO DOCENTE

O que você está achando das minhas aulas neste conteúdo?

Clique aqui ou escaneie o QRCode ao lado para avaliar minha aula.

Sinta-se à vontade para fornecer uma avaliação sempre que achar necessário.

DEVinHouse

Parcerias para desenvolver a sua carreira

OBRIGADO!

SENAI

<LAB365>