COMO CRIAR ARQUIVOS DE OCNFIGURAÇÃO E DEFINIR OS PROFILES COM O PADRÃO YAML (.YML):

1. CRIAÇÃO DOS ARRQUIVOS .YML
   1. Neste projeto, na pasta src>java>resources foram criados os arquivos **application-dev.yml** e **application-prod.yml** para indicar os perfis de desenvolvimento e produção respectivamente. No arquivo **application.properties** são adicionadas configurações padrão ou comuns aos dois perfis.

**OBS:** Nota-se que podem haver tanto arquivos de configuração .properties, quanto .yml no projeto, podendo até existir os dois padrões ao mesmo tempo.

1. É interessante notar a diferença entre a escrita dos arquivos .properties e .yml. Veja:

Application-prod.properties:

#PROD environment  
app.message=This is the property file to $(spring.application.name) specific to PROD  
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver  
spring.datasource.url= <MYSQL\_JDBC>  
spring.datasource.username=<USERNAME>  
spring-datasource.password=<PASSWORD>

Application-dev.yml:

*#DEV environment***app**:  
 **message**: This is the property file to ${spring.application.name} specific to DEV  
**spring**:  
 **datasource**:  
 **driver-class-name**: org.h2.driver  
 **url**: jdbc:h2:mem:db:DB\_CLOSE\_DELAY=-1  
 **username**: sa  
 **password**: sa

1. É possível ainda passar parâmetros de configuração durante a execução da linha de comando. Exemplo: mvn spring-boot: run -Dserver.port=9090 -> podemos indicar, com o -D que vamos passar parâmetros e server.port=9090 é o parâmetro indicado, de forma que o servidor subirá na porta 9090, não importando qual porta está configurada nos arquivos .properties ou .yml.
2. PEGANDO VALORES DE VARIÁVEIS DE AMBIENTE
3. É possível obter valores de variáveis de ambiente, utilizando a anotação @Value(${<variável>: <valor\_padrao>}). Essa anotação significa que:

<variável> é o nome de uma variável de ambiente: exemplo: JAVA\_HOME, JRE\_HOME.

<valor\_padrão> é opcional. Pode ser definido um valor padrão, casos a variável de ambiente não esteja definida.

No exemplo, foi criada uma variável de ambiente chamada DB\_ADDRESS, com o valor jdbc:cj:mysql:server:db , por da inserção do comando abaixo no shell:

* 1. export DB\_ADDRESS=jdbc:cj:mysql:server:db

Para indicar que uma variável da aplicação vai receber o valor da variável de ambiente, foi inserido o seguinte código na classe AppController:

**@Value*(*"${ DB\_ADDRESS: NENHUM }"*)*   
private String dbAddress;**