09

Deploy da aplicação no Tomcat

Transcrição

Para enviar o arquivo loja.war (https://s3.amazonaws.com/caelum-online-public/aws-ec2/downloads/loja.war) (caso você ainda não tenha baixado, pode encontrá-lo aqui (https://s3.amazonaws.com/caelum-online-public/aws-ec2/downloads/loja.war)) usaremos o comando `scp` (*secure copy*) que usa o protocolo SSH para permitir trocas de arquivos entre dois locais remotos de forma segura.

Para enviar o arquivo precisaremos do mesmo ***keypair*** usado para acessar a máquina via SSH. Portanto, acesse o diretório onde se encontra seu arquivo `loja.pem` e entre com o comando:

```
scp -i "loja.pem" <diretorio-do-arquivo>/loja.war ubuntu@ec2-54-94-172-57.sa-east-1.compute.amazonaws.com:~/
```

No último parâmetro do comando (`ubuntu@ec2-54-94-172-57.sa-east-1.compute.amazonaws.com:~/`) dizemos o nome do usuário da maquina remota (**ubuntu**), o endereço físico (**ec2-54-94-172-57.sa-east-1.compute.amazonaws.com**) e o local da máquina remoto onde queremos que o arquivo seja salvo (**~/** diretório raiz). Um outro exemplo de uso dessa mesma notação é quando precisamos fazer o *clone* de um repositório remoto para nossa máquina com `git` . Onde usamos: `git@github.com:usuario-github/nome-do-repositorio.git` .

Fazendo deploy da aplicação no Tomcat

Voltando para nossa instância, ao fazer `ls` no diretório raiz podemos ver o arquivo que transferimos (`loja.war`)

```
ubuntu@ip-172-31-17-202:~$ ls
loja.war
ubuntu@ip-172-31-17-202:~$
```

O arquivo `.war` deve ficar dentro da pasta `webapps` do Tomcat, vamos então mover o arquivo `loja.war` do diretório raiz para o diretório `/var/lib/tomcat8/webapps` com o comando `mv` :

```
sudo mv loja.war /var/lib/tomcat8/webapps
```

Você deve está se perguntando agora: *como a aplicação vai reconhecer meu MySQL se eu não passei nenhuma informação?* Acontece que o código da aplicação já estamos configurando a aplicação para encontrar um banco de dados na mesma máquina onde está rodando o Tomcat (*localhost*). *Mas se por algum motivo eu precisar trocar a máquina onde está nosso banco, vou precisar alterar meu código para apontar para esse novo endereço? Quando estou em desenvolvimento, quero manter localhost mas em produção quero mudar, como deixar isso mais flexível sem precisar alterar a todo momento?*

Conhecendo as variáveis de ambiente

É muito comum deixarmos essas configurações relativas a ambiente configuradas nas chamadas `variáveis de ambiente`. Elas são pares de **chave-valor** que ficam salvos em um determinado ambiente e podem ser recuperados pela aplicação. O que significa que podemos criar uma variável de ambiente chamada `url` na máquina de produção com o valor que aponta para a *url* de produção e na máquina de desenvolvimento criar a mesma variável apontando para `localhost`.

Por curiosidade, colocarei o código essa configuração é feita na nossa aplicação (**lembrando que você não precisa conhecer *Java* nem tão pouco entender o código abaixo para assistir esse treinamento**):

```
@Bean
public DataSource getDataSource() {
    String url = System.getenv("url");
    String password = System.getenv("senha");

    if ( url == null ) url = "localhost";

        DriverManagerDataSource dataSource = new DriverManagerDataSource();

        dataSource.setDriverClassName("com.mysql.jdbc.Driver");
        dataSource.setUrl("jdbc:mysql://" + url + "/projeto_jpa");
        dataSource.setUsername("root");
        dataSource.setPassword(password == null ? "" : password);

        return dataSource;
}
```

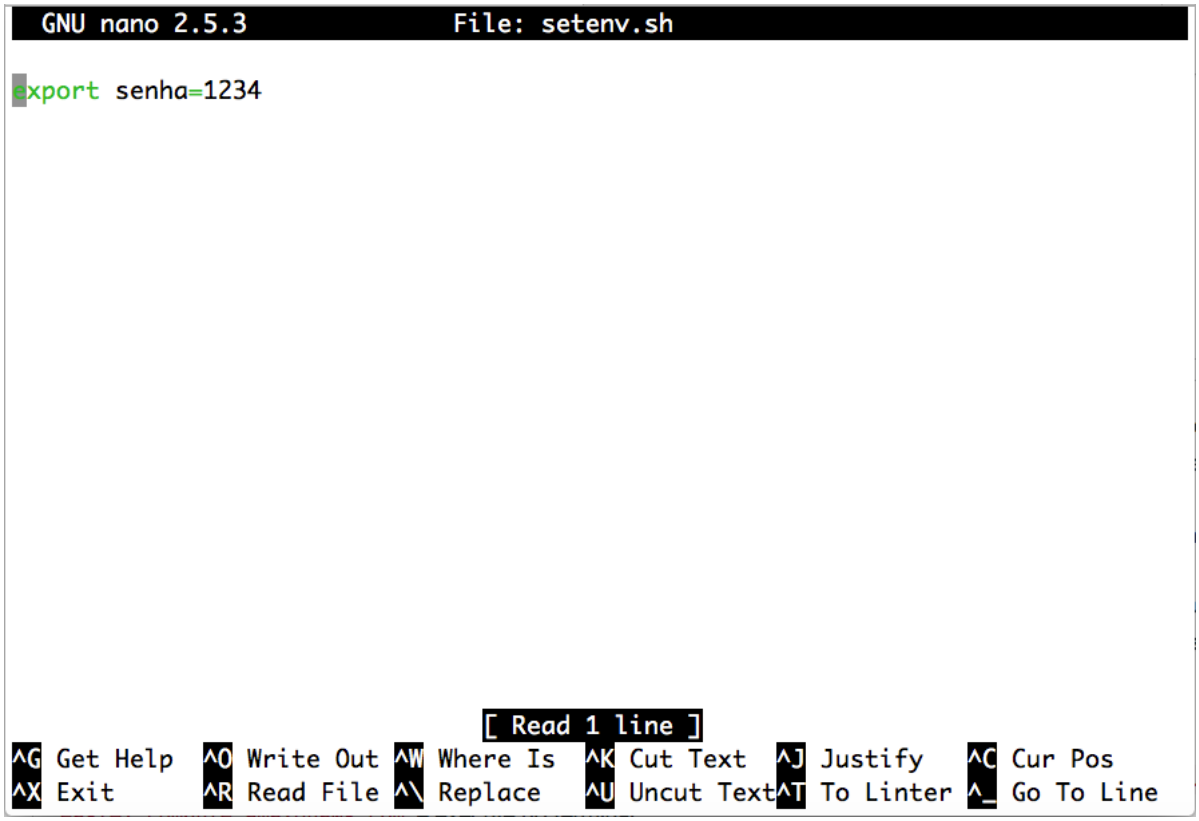
No nosso caso queremos acessar um MySQL que está instalado na mesma máquina do Tomcat e como o código acima considera **localhost** como valor padrão caso não exista a variável `url`, precisaremos apenas configurar a variável de ambiente para passar a senha cadastrada no banco de dados.

Com Tomcat podemos fazer isso criando um arquivo chamado `setenv.sh` no diretório `/usr/share/tomcat8/bin` e para isso, faremos:

```
cd /usr/share/tomcat8/bin
sudo nano setenv.sh
```

No arquivos vamos exportar a variável `senha` com o valor definido na instalação do MySQL.

```
export senha=1234
```



Acessando a aplicação na instância

Para terminarmos, vamos reiniciar o Tomcat:

```
sudo service tomcat8 stop

sudo service tomcat8 start
```

E acessar a aplicação através da URL da instância. Após de alguns segundos de carregamento inicial por conta da criação das tabelas, você conseguirá ver a aplicação: