09

Primeiro job

Transcrição

O script que o instrutor segue durante a aula é o seguinte:

```
# Configurando a chave privada criada no ambiente da VM no Jenkins
    cat ~/.ssh/id_rsa
    Credentials -> Jenkins -> Global Credentials -> Add Crendentials -> SSH Username with private ke
# Criando o primeiro  job que vai monitorar o repositorio
    Novo job -> jenkins-todo-list-principal -> Freestyle project:
    Esse job vai fazer o build do projeto e registrar a imagem no repositório.
    # Gerenciamento de código fonte:
        Git: git@github.com:rafaelvzago/treinamento-devops-alura.git [SSH]
        Credentials: git (github-ssh)
        Branch: master
    # Trigger de builds
        Pool SCM: * * * * *
    # Ambiente de build
        Delete workspace before build starts
    # Salvar
    # Validar o log em: Git Log de consulta periódica
```

[00:00] Bom pessoal, agora chegou uma parte legal, a gente vai criar nosso primeiro job no Jenkins e é muito simples de fazer. Antes de criar o job eu preciso fazer algumas configurações dentro do meu ambiente.

[00:11] Então a gente vai lá no nosso Jenkins e eu vou aqui em credenciais, vai pedir pra eu autenticar aqui e vamos autenticar de novo, dentro de credenciais eu vou clicar aqui em Jenkins, as credenciais globais, e eu vou adicionar uma credencial nova. Que tipo de credencial que eu vou adicionar? SSH com username e chave privada. Por que que eu preciso fazer isso? Pra que eu consiga interagir com meu repositório no GitHub.

[00:46] Então o que eu vou fazer? Eu vou criar o meu id aqui, eu vou chamar de github-ssh. Isso aqui é só um id dentro do Jenkins pra você referenciar em jobs futuros, e é legal a gente manter o padrão porque os outros jobs utilizam esse mesmo id. A descrição eu vou colocar a mesma. O usuário do GitHub é sempre git, não precisa ser seu usuário porque você vai conectar com chave.

[01:14] E aí eu vou ter que adicionar minha chave, como é que eu faço isso? Eu clico aqui pra entrar diretamente, clico pra adicionar a minha chave, e ele quer saber qual que é a minha chave. Então vamos voltar aqui no nosso terminal.

[01:26] Aqui no nosso terminal a gente tem, dentro do nosso diretório de SSH, o seguinte arquivo: id_rsa, ele não é um arquivo id_rsa.pub, aquela lá é a chave pública. Agora eu preciso da chave privada, por quê? É o Jenkins que vai interagir com o meu GitHub, assim como o nosso terminal interagiu com o GitHub agora é a vez do Jenkins.

[01:55] Então, pra isso, eu preciso copiar a minha chave privada pro Jenkins. Então eu seleciono aqui, lembrando que vocês nunca devem passar essa chave pra ninguém, e a gente adiciona aqui como texto.

[02:14] Agora que a gente adicionou não tem passphrases, vocês lembram que a gente não configurou o passphrase. A gente dá um ok, nossa credencial tá configurada, agora é a hora de criar o nosso primeiro job.

[02:26] Então a gente vem aqui em Jenkins, Novo job, e a gente vai dar um nome pra esse job. O nome que a gente vai usar pra esse job vai ser jenkins-todo-list-principal. Por que que a gente deu esse nome de principal? Porque a maioria das interações com o banco vão ser feitas através desse job, e esse job é do tipo Freestyle. A gente vai entender a diferença de cada um deles, do Freestyle pro Pipeline, por exemplo, mas dessa vez a gente vai escolher o Freestyle.

[03:05] Clicamos aqui no Freestyle, vamos dar um ok. Ele vai criar o primeiro job. O job vai ser executado de acordo com as configurações que a gente definir. Então, primeira coisa que a gente vai fazer, a gente vai avisar que é um projeto git e vai copiar a URL do git. Atentem-se pra colocar o git@github pra pegar o clone da configuração via SSH e não com Https, isso faz diferença na hora de commitar o seu código.

[03:43] Aí a gente colocou aqui o nosso GitHub, se eu dar um tab ele vai falhar, ele vai falar "olha, eu não consigo conectar", por que? Porque esse repositório, do jeito que eu coloquei pra acessar, ele precisa ter as credenciais de SSH. Então eu venho aqui e seleciono github.ssh, automaticamente ele já fez a conexão e já entendeu que tá funcionando. Ou seja, agora nosso job já é capaz de ler e interagir com o GitHub, com nosso repositório.

[04:14] Algumas coisinhas que a gente vai ter que fazer: dentro do ambiente de build a gente vai marcar pra deletar o workspace antes do projeto iniciar, a gente faz isso pra evitar que fique alguma sujeira de código porque, na verdade, o workspace nada mais é do que um diretório dentro do Jenkins, então eu preciso limpar esse cara antes de fazer a configuração.

[04:38] Outra coisa que a gente vai fazer pra ele é o seguinte: ele vai consultar o git periodicamente. Primeiro vamos configurar, depois eu vou explicar pra vocês a diferença. Pra quem já mexeu com o crontab é a mesma sintaxe. Então eu vou colocar aqui primeiro, pra que ele verifique a cada minuto, a cada hora, a cada dia da semana, a cada mês e a cada dia do mês.

[05:03] Vai uma ressalva aqui, não significa que seu eu colocar mais passos pra baixo pra fazer um build, alguma coisa que impacte mais o código, que ele vá fazer isso a todo minuto. Nessa agenda que eu coloquei ele vai verificar a cada minuto se o meu repositório tem alguma alteração. Vamos ver isso aí como vai ficar.

[05:27] A gente vai dar um salvar, a gente vai esperar um pouquinho, ele vai disparar a primeira vez, ele vai consultar o repositório automaticamente, ele faz isso todo minuto. Vamos esperar um pouquinho que aí ele já vai aparecer.

[05:46] Olha lá, apareceu aqui pra gente, primeiro processo pendente, e ele começou a executar. Ele tem a hora, isso aqui é a hora do nosso Vagrant, da nossa máquina virtual, tá diferente da minha, mas não tem problema. Aqui ele executou com sucesso, eu vou clicar aqui nesse cara.

[06:05] Olha a saída do console, esse aqui é o nosso workspace, ele conectou no nosso repositório e viu que tem um código lá. Como não tinha nada, ele já sabe exatamente a partir de onde ele vai trabalhar, ou seja, a partir de agora todas as vezes que nós mudarmos o código dentro do GitHub, ele vai ficar checando e vai startar daqui para frente os builds que a gente pedir.

[06:34] Pessoal, antes da próxima aula tem mais uma informação que eu queria passar pra vocês: depois que a gente fez toda a instalação, tanto do ambiente quanto do Jenkins, a gente tem que fazer dois passos a mais agora, que seria o que? Adicionar tanto o meu usuário, no caso o usuário Vagrant ou o usuário de vocês, na máquina de vocês, e o usuário do Jenkins no grupo do Docker.

[07:01] O Docker foi instalado automaticamente, a gente vai falar deles mais para frente, mas é um passo adicional agora que vai fazer toda a diferença nos nossos jobs. E os comandos são muito simples. O primeiro comando a gente vai adicionar no grupo do Docker o usuário, o user, essa variável $USER é o próprio usuário que eu tô logado, e o usuário do Jenkins. Só pra gente evitar problemas de permissão na hora de rodar os containers.

[07:32] Pra efetivar todas as nossas alterações, a gente vai sair do nosso Vagrant e vai dar um vagrant reload. Assim as seções vão ser recarregadas com as permissões corretas e eu não vou mais precisar usar sudo pra rodar nenhum container.

[07:50] Então, agora, qual que é a próxima aula? Nós vamos entender passo a passo como que o build dessa aplicação é feito, pra que no futuro a gente consiga automatizar utilizando Jenkins e as nossas ferramentas que estão instaladas aqui.