06

Consolidando o seu conhecimento

Chegou a hora de você seguir todos os passos realizados por mim durante esta aula. Caso já tenha feito, excelente. Se ainda não, é importante que você execute o que foi visto nos vídeos para poder continuar com a próxima aula.

1) Faça a integração com o Slack. Você pode se guiar pelo script que o instrutor seguiu durante a aula:

```
# Criar app no slack: alura-jenkins.slack.com
    URL básico: <Url do Jenkins app no seu canal do Slack>
    Token de integração: <Token do Jenkins app no seu canal do Slack>

# Instalar o plugin do slack: Gerenciar Jenkins > Gerenciar Plugins > Disponíveis: Slack Notificatic
        # Configurar no jenkins: Gerenciar Jenkins > Configuraçao o sistema > Global Slack Notifier
            # Slack compatible app URL (optional): <Url do Jenkins app no seu canal do Slack>
            # Integration Token Credential ID : ADD > Jenkins > Secret Text
                # Secret: <Token do Jenkins app no seu canal do Slack>
                # ID: slack-token
            # Channel or Slack ID: pipeline-todolist

# As notificações vão funcionar da seguinte maneira:
Job: todo-list-desenvolvimento será feito pelo Jenkinsfile (Próximas aulas)
Job: todo-list-producao: Ações de pós-build > Slack Notifications: Notify Success e Notify Every Fai
```

2) Crie o *job* para o ambiente de desenvolvimento. Você pode se guiar pelo script que o instrutor seguiu durante a aula:

```
# Novo Job: todo-list-desenvolvimento:
    # Tipo: Pipeline
    # Este build é parametrizado com 2 Builds de Strings:
        Nome: image
        Valor padrão: - Vazio, pois o valor sera recebido do job anterior.

        Nome: DOCKER_HOST
        Valor padrão: tcp://127.0.0.1:2376

    pipeline {

        agent any

        stages {
            stage('Oi Mundo Pipeline como Codigo') {
                steps {
                    sh 'echo "Oi Mundo"'
                }
            }
        }
    }

    pipeline {
        environment {
            dockerImage = "${image}"
        }
```

```
        agent any

        stages {
            stage('Carregando o ENV de desenvolvimento') {
                steps {
                    configFileProvider([configFile(fileId: '<id do seu arquivo de desenvolvimento>',
                        sh 'cat $env > .env'
                    }
                }
            }
            stage('Derrubando o container antigo') {
                steps {
                    script {
                        try {
                            sh 'docker rm -f django-todolist-dev1'
                        } catch (Exception e) {
                            sh "echo $e"
                        }
                    }
                }
            }
            stage('Subindo o container novo') {
                steps {
                    script {
                        try {
                            sh 'docker run -d -p 81:8000 -v /var/run/mysqld/mysqld.sock:/var/run/mys
                        } catch (Exception e) {
                            slackSend (color: 'error', message: "[ FALHA ] Não foi possivel subir o
                            sh "echo $e"
                            currentBuild.result = 'ABORTED'
                            error('Erro')
                        }
                    }
                }
            }
            stage('Notificando o usuario') {
                steps {
                    slackSend (color: 'good', message: '[ Sucesso ] O novo build esta disponivel em:
                }
            }
        }
    }

# todo-list-principal
    # Definir post build: image=$image
```