DigitalHouse>

Certified Tech Developer
The Ultimate Degree

Infraestrutura II

Criando nosso primeiro Pipeline

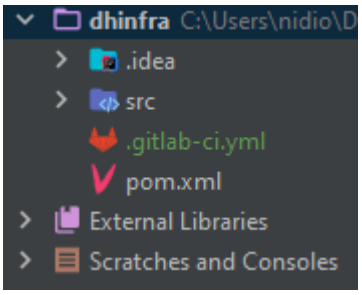
Tendo configurado nossa conta no GitLab e carregado nosso primeiro código, hoje vamos fazer um pipeline com 2 estágios.

Objetivo final da prática

- Fazer um pipeline.
- Modifique parte do nosso código para ver como isso afeta o pipeline
- Adicione um estágio ao nosso pipeline

Vamos trabalhar!

O local para definir um pipeline no GitLab é o arquivo .gitlab-ci.yml, que deve estar localizado na raiz do nosso projeto.
Para isso, abrimos um arquivo com o nome **.gitlab-ci.yml** em **nosso computador** com o editor de texto de nossa escolha.



1. Criando os estágios
   A definição do pipeline se dá em stage e jobs, estes pertencem aos estágios, ou seja, os primeiros estágios são os de nível hierárquico mais alto.

   A execução das etapas é de acordo com como as definimos em nosso arquivo, por convenção para esta primeira prática, vamos construir duas etapas
   - build
   - test

   Adicionamos as linhas em nosso arquivo .gitlab-ci.yml, lembre-se sempre de respeitar a indentação

```
stages:
    - build
    - test
```

   Embora esta definição esteja correta, basicamente este pipeline não fará nada, pois os estágios não possuem jobs, que é onde os comandos são executados.

2. Adicionamos os jobs
   Os jobs são definidos no arquivo e sua pseudo estrutura é

   job_name
      stage: “etapa ao qual pertence”
      script:
        - “comandos a serem executados no job”
        - echo “hello pipeline”

Primeiro vamos definir um job para nosso estágio "build", que compilará o código Java que temos; para isso usaremos o Maven como compilador, o comando para compilar será “mvn compile”.
O código do nosso **.gitlab-ci.yml** com o job adicionado deve ser:

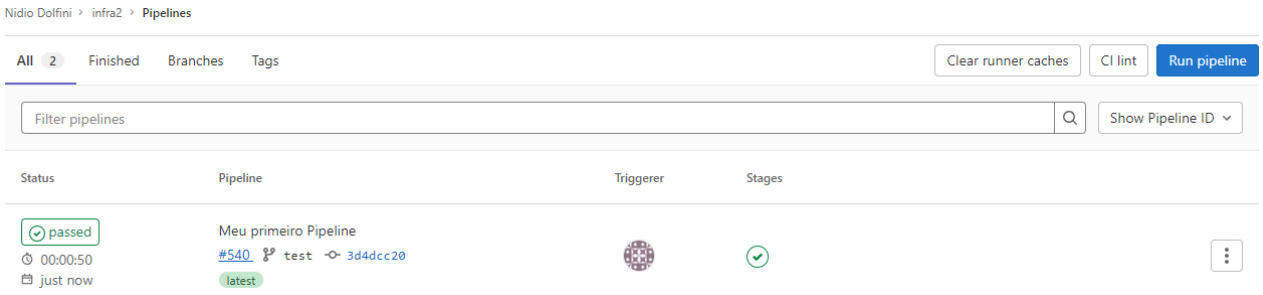
```
image: maven:3.6.3
stages:
    - build
    - test

build_job:
  stage: build
  script:
    - echo "Maven compile started"
    - "mvn compile"
```

Até agora, nós definimos o pipeline, mas o GitLab não está ciente das alterações, é hora de fazê-lo e para isso devemos implementar essas mudanças que ainda estão em nosso computador para o repositório, é por isso que devemos enviar este novo arquivo, executando em **nosso computador**:

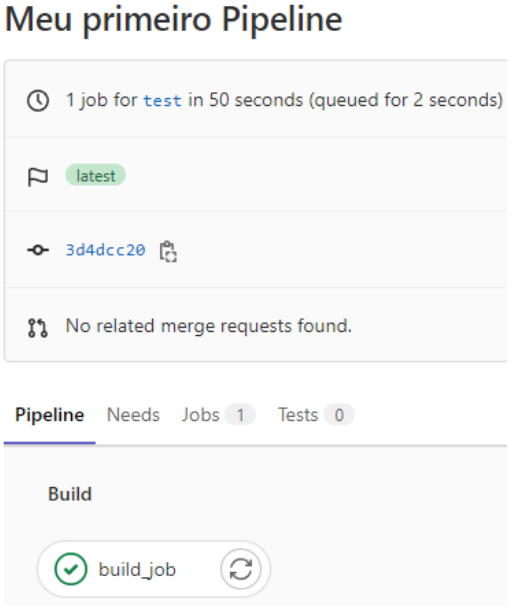
```
git add .
git commit -m "Meu primeiro pipeline"
git push
```

3. **Pipeline em execução!**

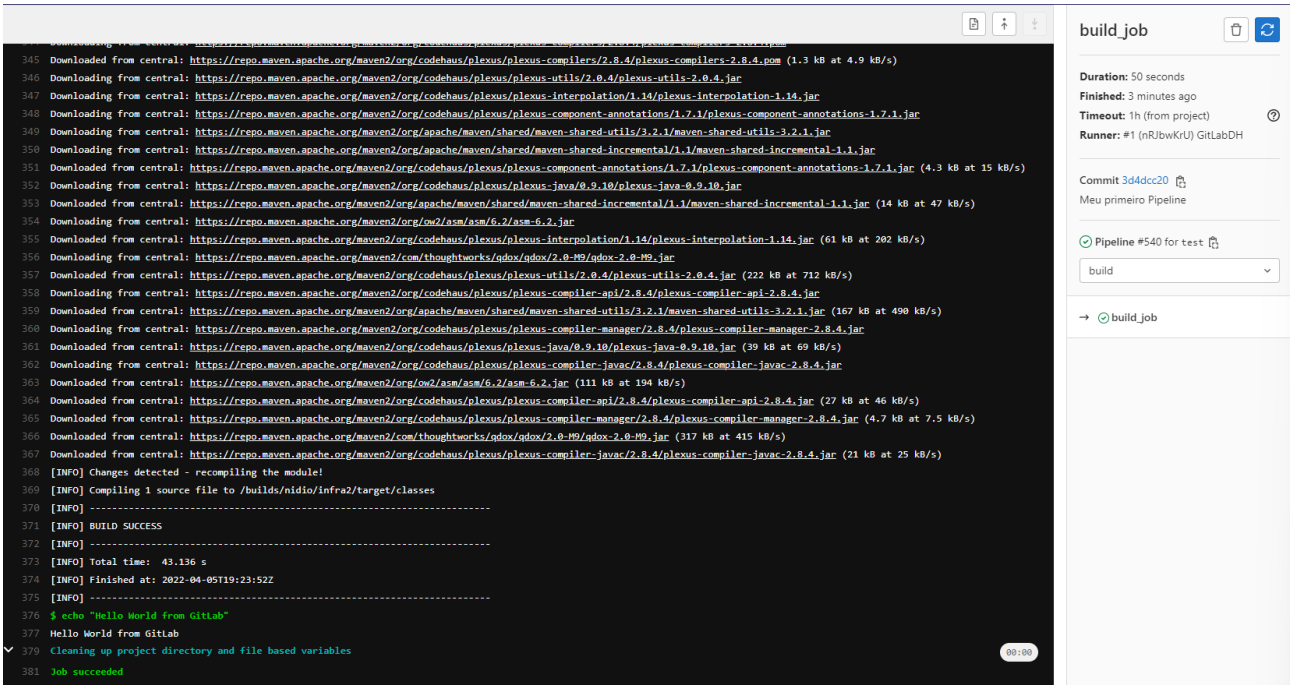
   O GitLab, ao receber as alterações no código, detecta automaticamente a presença do arquivo .gitlab-ci.yml e se sua estrutura for válida, **EXECUTA** os passos definidos nele, por isso agora vamos ao nosso **GitLab** CI/CD Pipelines, veremos a lista de Pipelines executados e haverá o correspondente ao último commit.



Podemos ver detalhes de sua execução?
Sim, digitamos o número (neste caso de exemplo "#540") lá veremos os estágios que foram executados



Aqui vemos que apenas o estágio BUILD foi executado, e seu trabalho "build_job", se entrarmos observamos todos os detalhes da execução. Clique no botão "build_job".



4. E em que momento o Pipeline é executado novamente?
   Estando vinculado ao repositório de código, qualquer alteração no meu código-fonte (não apenas no .gitlab-ci.yml) fará com que o pipeline seja executado novamente.

   Vamos ao trabalho: faça uma alteração em algum arquivo do código fonte (por exemplo a mensagem mostrada em `src/main/java/com/dhinfra/app/App.java`) e ao realizar o push observe o comportamento do pipeline.

5. Um passo mais!
   Em nosso arquivo .gitlab-ci.yml definimos os dois estágios (build e test), mas fizemos apenas um job para o build, é hora de criar um job para o estágio de teste, daremos apenas um dica, o comando para rodar o teste no Maven é "mvn test", **você tem coragem de adicionar o job correspondente?**