



# Certified Tech Developer

The Ultimate Degree

## Infraestrutura II

**Atividade obrigatória e individual**

**Dificuldade: média**

# Exercício: Integração Contínua (IC)

Vamos colocar em prática o que aprendemos nesta semana. E para isso, vamos trabalhar na plataforma Jenkins na criação e execução do processo de construção de uma simples aplicação Java.

## Verificando os pré-requisitos

Para poder realizar este exercício precisamos ter o Jenkins instalado e o código fonte de uma aplicação Java para realizar o processo de construção.

Para verificar se o Jenkins está funcionando corretamente, acessamos a rota web:

<http://localhost:8080/>



Welcome to Jenkins!



Sign in

☐ Keep me signed in

Caso não consigamos acessá-lo porque o serviço não está ativo, podemos iniciá-lo com o comando:

```
sudo systemctl start jenkins
```

Em seguida, verificamos se ele está realmente rodando:

```
usuario@digitalhouse: ~$ sudo systemctl status jenkins
jenkins.service - LSB: Start Jenkins at boot time

   Loaded: loaded (/etc/init.d/jenkins; generated)
   Active: active (exited)
```

O aplicativo Java que vamos executar é um aplicativo muito simples que calcula a série de Fibonacci com 10 valores. Está em um único arquivo, pois nosso objetivo é compilar corretamente em um processo básico de CI (*Integração Contínua*).

Vamos criar um arquivo em /tmp chamado "Fibonacci.java" e colar o seguinte código:

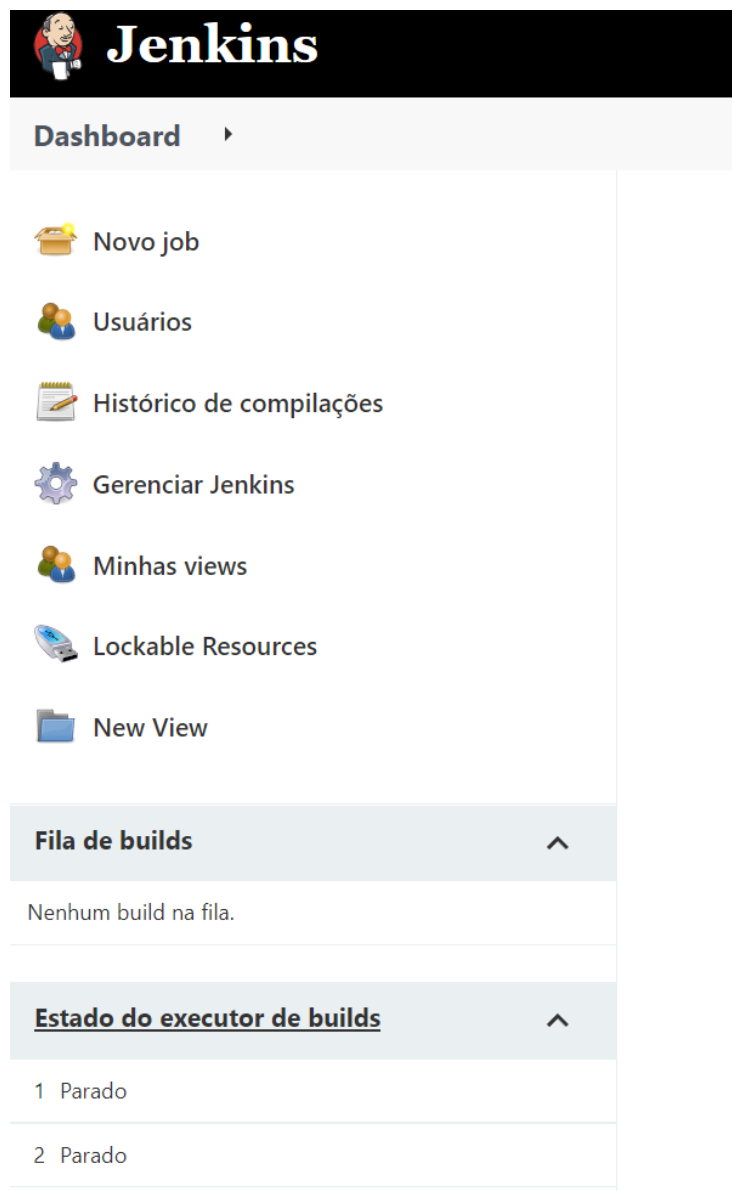
```
public class Fibonacci {
    public static void main(String[] args) {
        int number = 10;
        int a = 0;
        int b = 1;
        int sum = 0;
        System.out.println("Serie de Fibonacci na DH");
        System.out.print(a + " " + b + " ");
        for( int i = 2; i < number ; i++){
            sum = a + b;
            a = b;
            b = sum;
            System.out.print(sum + " ");
        }
    }
}
```

Nosso requisito final é ter as ferramentas de construção Java instaladas. Isso é feito com o seguinte comando de terminal:

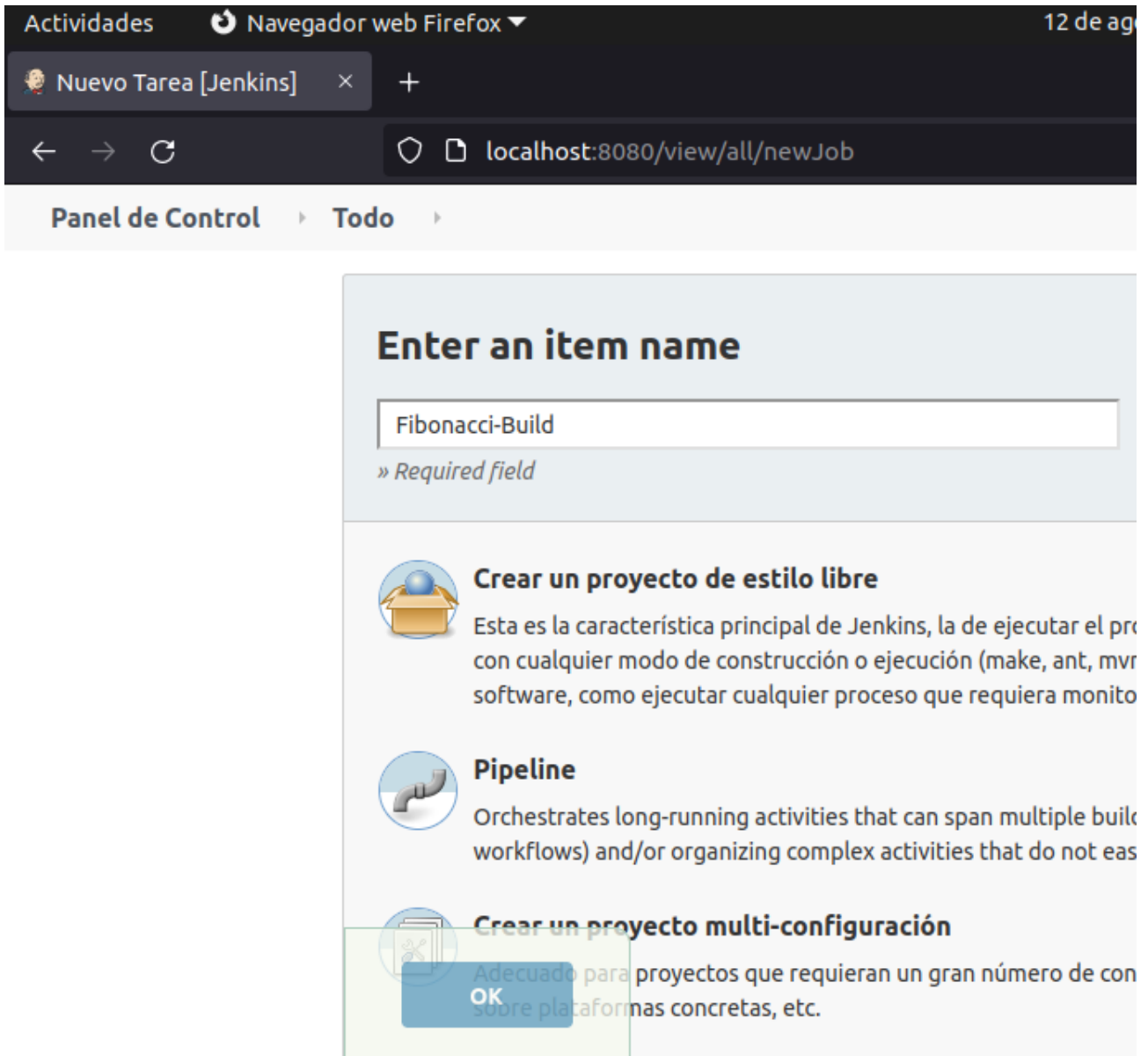
```
sudo apt install openjdk-11-jdk
```

## Criando o Job (tarefas no Jenkins)

Para criar qualquer tarefa ou Job no Jenkins vamos em “Novo Job”:



Digitamos um nome e escolhemos o tipo de projeto. Neste caso, vamos selecionar “Criar um projeto freestyle”:



Actividades Navegador web Firefox 12 de ag

Nuevo Tarea [Jenkins] +




localhost:8080/view/all/newJob

Panel de Control ▸ Todo ▸

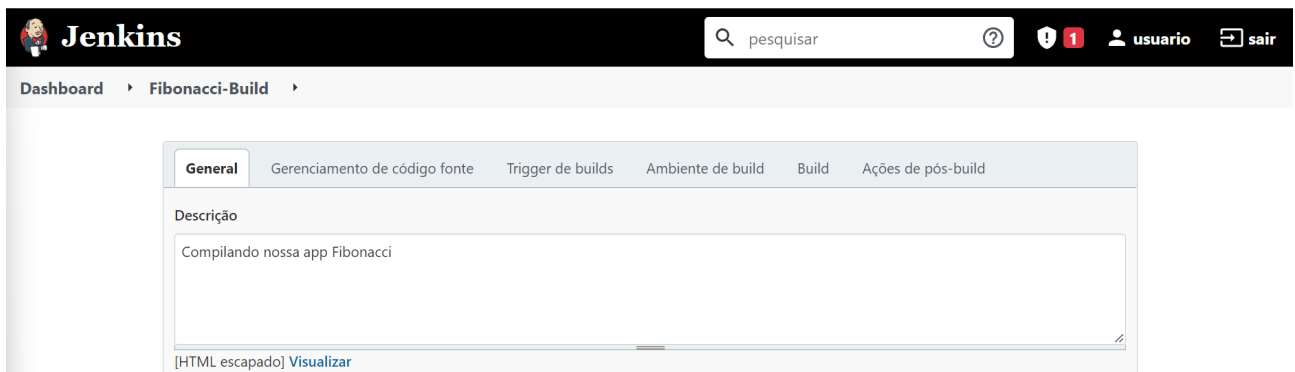
### Enter an item name

Fibonacci-Build

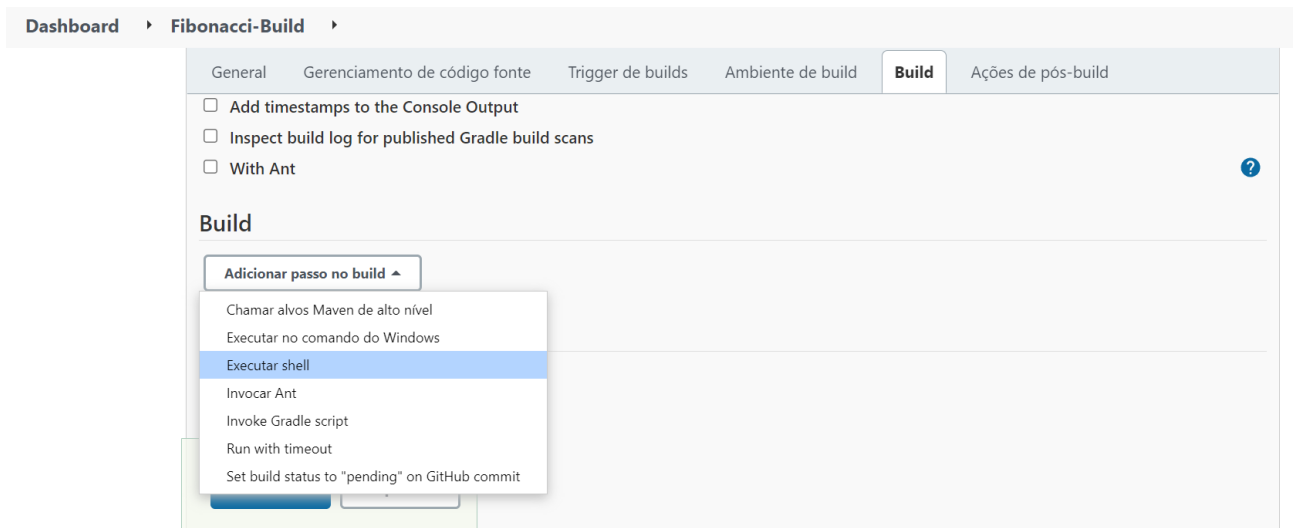
» Required field

-  **Crear un proyecto de estilo libre**  
Esta es la característica principal de Jenkins, la de ejecutar el pro con cualquier modo de construcción o ejecución (make, ant, mvr software, como ejecutar cualquier proceso que requiera monito
-  **Pipeline**  
Orchestrates long-running activities that can span multiple build workflows) and/or organizing complex activities that do not eas
-  **Crear un proyecto multi-configuración**  
Adecuado para proyectos que requieran un gran número de con sobre plataformas concretas, etc.

É importante escrever uma descrição do que estamos fazendo para que nossa equipe e quem vê nosso trabalho saiba exatamente o que cada Job faz:



Vamos para o “Build ” e adicionamos uma nova etapa. Nas opções exibidas selecionamos “Executar shell”:



Para terminar de configurar nossa compilação, vamos escrever os comandos que são executados nesta tarefa:

```
cd /tmp  
  
javac Fibonacci.java  
  
java Fibonacci
```

Antes de salvar, vamos review Deixe nosso espaço de trabalho Jenkins assim:

Dashboard > Fibonacci-Build >

General Gerenciamento de código fonte Trigger de builds Ambiente de build **Build** Ações de pós-build

Comando

```
cd /tmp
javac Fibonacci.java
java Fibonacci
```

[Veja a lista de variáveis de ambiente disponíveis](#)

Avançado...

Adicionar passo no build ▾

Ações de pós-build

Adicionar ação de pós-build ▾

Salvar Aplicar


Toda vez que fizermos uma alteração em nosso código, basta executar "Build Now" para compilar a aplicação:





**Jenkins**


Dashboard > Fibonacci-Build >


 Voltar para o Dashboard

 Situação


 Alterações


 Workspace

 Construir agora

 Configurar

 Excluir Projeto

 Rename

 Histórico de builds Tendência ^

## Projeto Fibonacci-Build

Compilando nossa app Fibonacci

 Workspace

 Mudanças recentes

## Links permanentes

Quando executarmos(Construir agora) poderemos ver a saída e que nosso código foi compilado e executado corretamente.

## Saída do console

```
Iniciado pelo usuário usuario
Running as SYSTEM
Construindo no workspace /var/lib/jenkins/workspace/Fibonacci-Build
[Fibonacci-Build] $ /bin/sh -xe /tmp/jenkins6543109074741137298.sh
+ cd /tmp
+ javac Fibonacci.java
+ java Fibonacci
Serie de Fibonacci na DH
0 1 1 2 3 5 8 13 21 34 Finished: SUCCESS
```

Se, por exemplo, modificarmos o valor da variável "number" para 20, apenas executando o trabalho novamente podemos ver as alterações feitas:

```
public class Fibonacci {
    public static void main(String[] args) {
        .....int number = 20;
```

Ao executar e ver a saída veremos como a extensão da série Fibonacci mudou:

## Saída do console

```
Iniciado pelo usuário usuario
Running as SYSTEM
Construindo no workspace /var/lib/jenkins/workspace/Fibonacci-Build
[Fibonacci-Build] $ /bin/sh -xe /tmp/jenkins362746045079515913.sh
+ cd /tmp
+ javac Fibonacci.java
+ java Fibonacci
Serie de Fibonacci na DH
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181 Finished: SUCCESS
```

## Conclusão

Agora que sabemos como compilar por linha de comando e como executá-lo no Jenkins... existe forma ainda mais prática? E se, por exemplo, temos nosso código no GitHub, o que precisamos para conectá-lo? Também podemos pensar assim: o trabalho pode ser executado automaticamente quando modifico meu código no GitHub?

Deixamos-lhe uma pista! Você pode pesquisar os plugins disponíveis neste URL:

<https://plugins.jenkins.io/>