

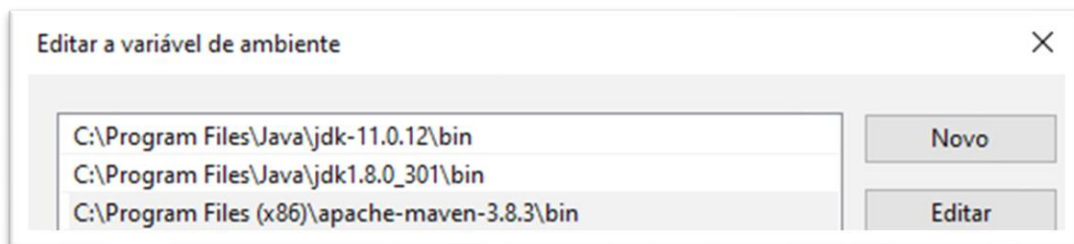
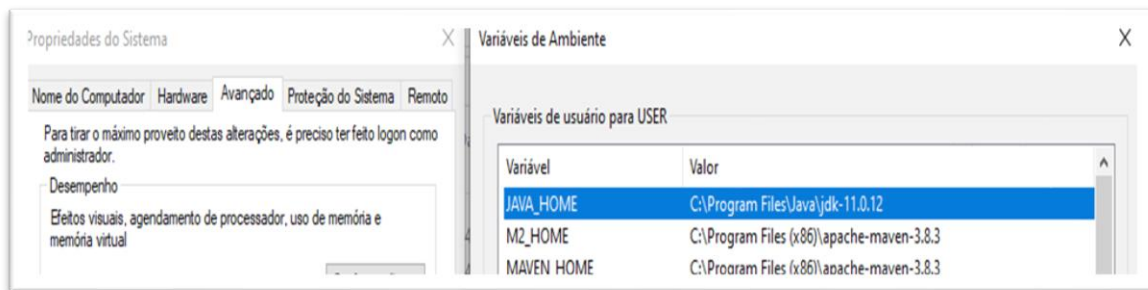
Jenkins X Spring

Jenkins: é um servidor de automação de código aberto. Ele ajuda a automatizar as partes do desenvolvimento de software relacionadas à construção, teste e implantação, facilitando a integração e entrega contínuas.

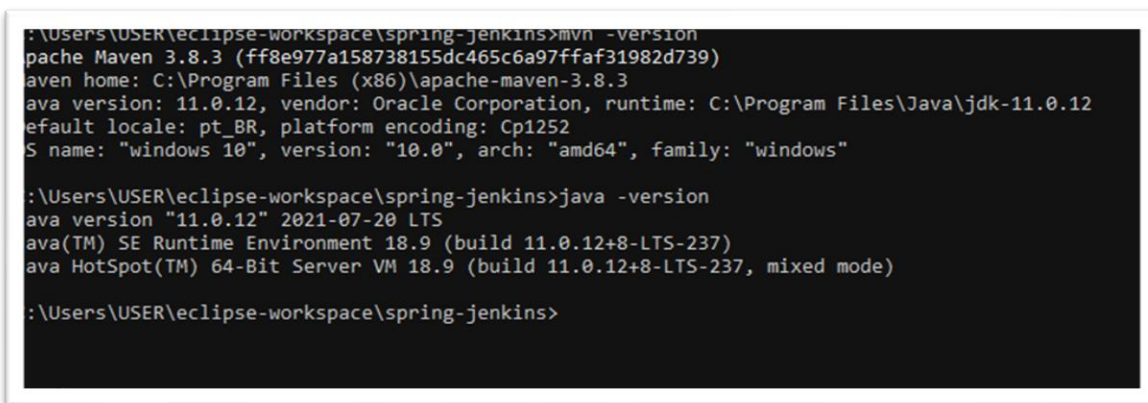
Spring: é um framework open source para a plataforma Java criado por Rod Johnson e descrito em seu livro "Expert One-on-One: JEE Design e Development".

Para que a api se execute corretamente primeiro devemos configurar as variáveis de ambiente com JDK y MAVEN. Jdk se encarga de interpretar y compilar a linguagem java enquanto que maven maneja as dependências que se configuram em el arquivo POM (é a peça fundamental de um projeto do Apache Maven. Um POM possui as informações básicas de um projeto, bem como as diretivas de como o artefato final deste projeto deve ser construído)

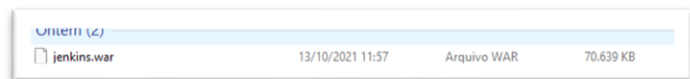
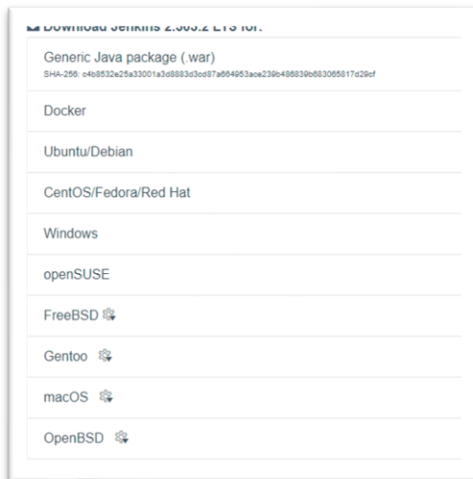
Se configura JAVA_HOME e MAVEN_HOME com os arquivos baixados de oracle



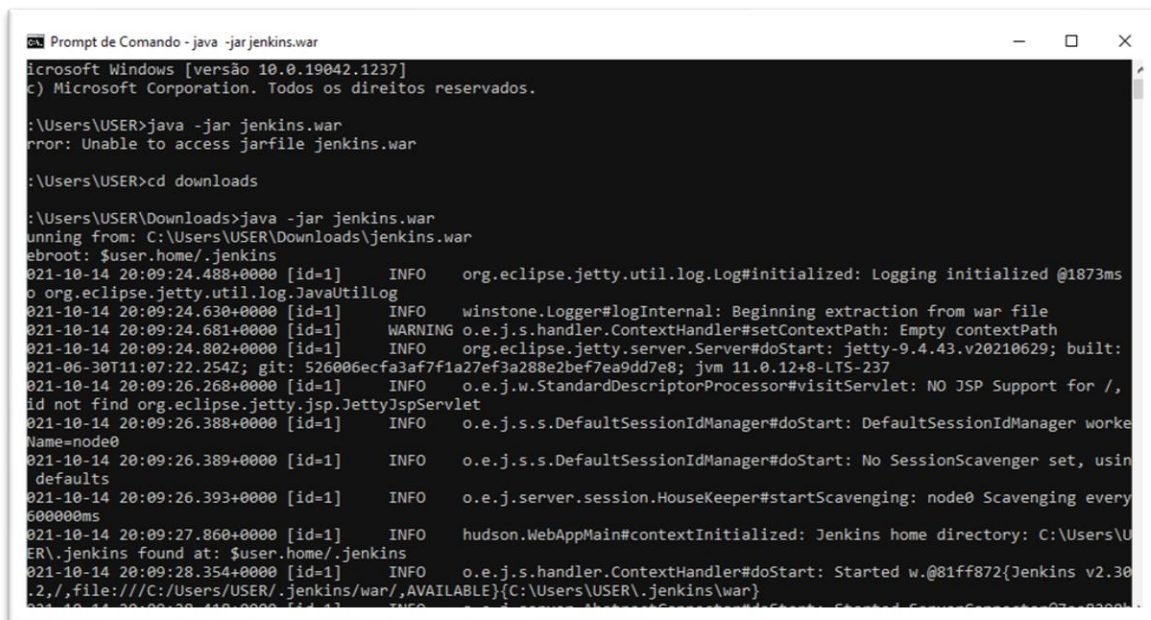
Verificamos que se haja instalado corretamente mediante a linha de comando de Windows



Descarregamos Jenkins da página oficial, existem várias opções, o instalador para Windows y o executável de java .war (Jenkins e uma api desenvolvida em java)

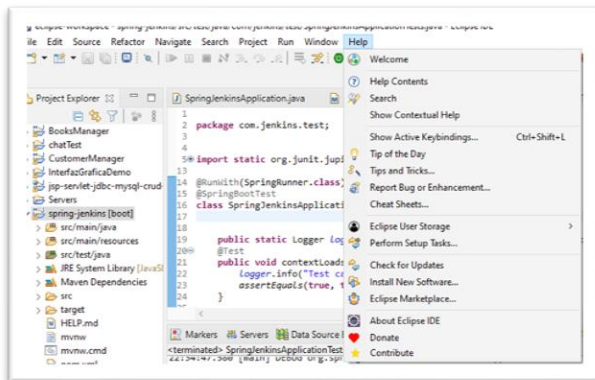


Para executar Jenkins e importante desativar qualquer antivírus que possa identificar Jenkins como um arquivo malicioso y executamos em a línea de comando na pasta descarregada

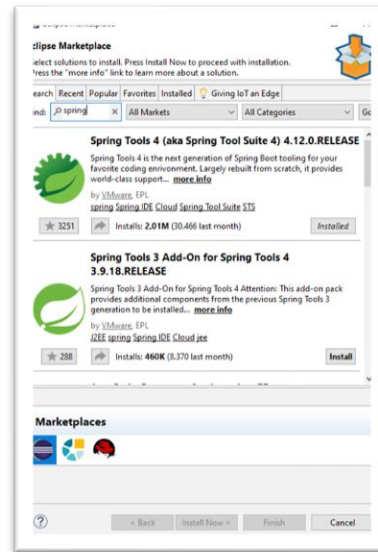


Vamos a testar uma api java-spring dentro do ambiente de desenvolvimento de eclipse, existem várias opções uma de elas e o próprio ambiente de spring-boot, initializer de spring.io o intellij. as instruções de como configurar eclipse se encontram a continuação y as dependências dependeram de cada projeto que se deseje testar

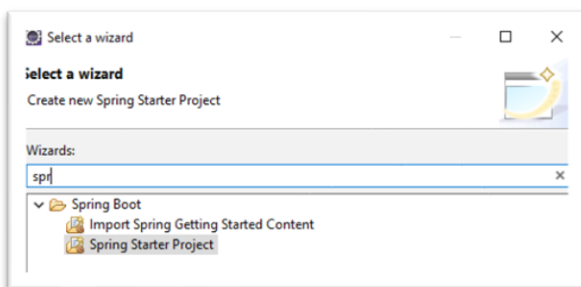
1)



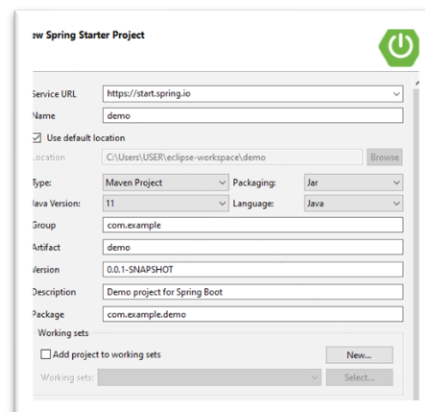
2)



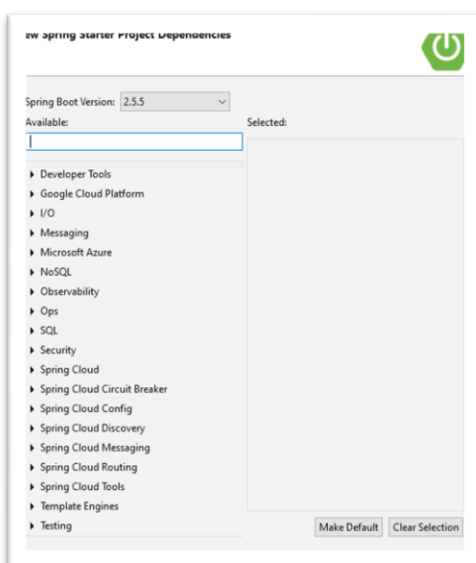
3)



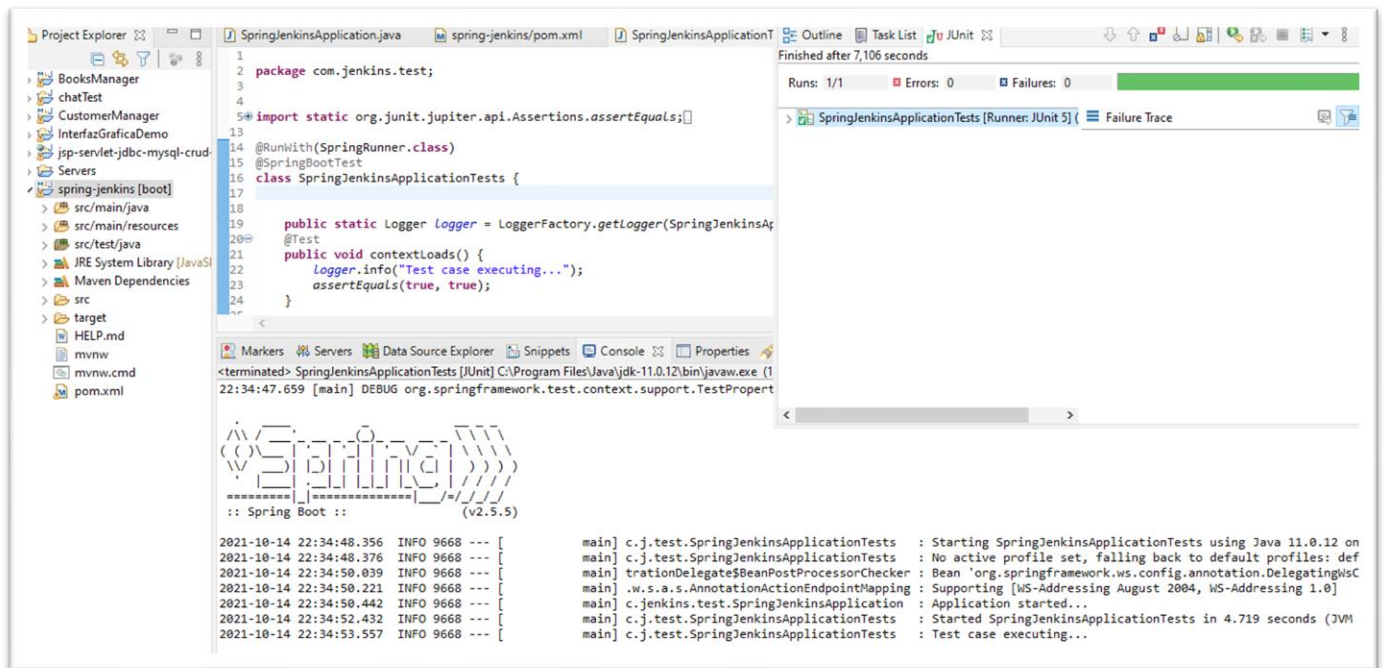
4)



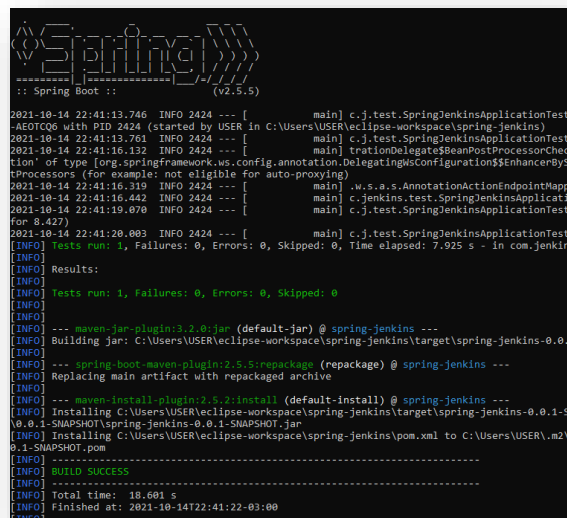
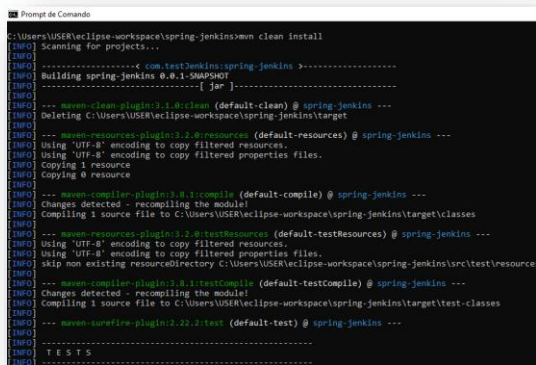
5)



Um exemplo de um teste executado dentro de eclipse

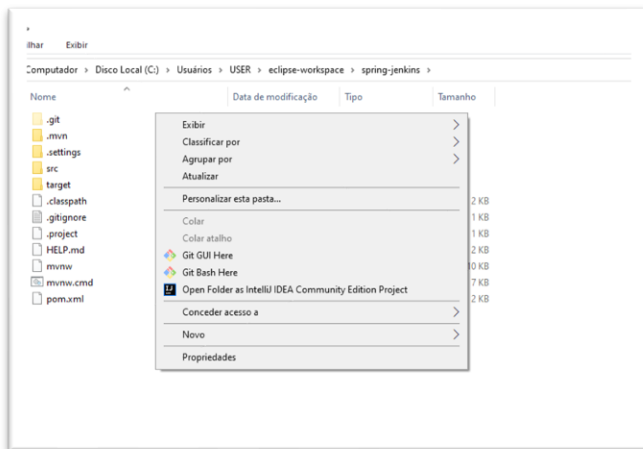


E também compilamos mediante o comando *mvn clean install*, donde se verifica que todo este funcionando corretamente dentro das configurações do sistema atual

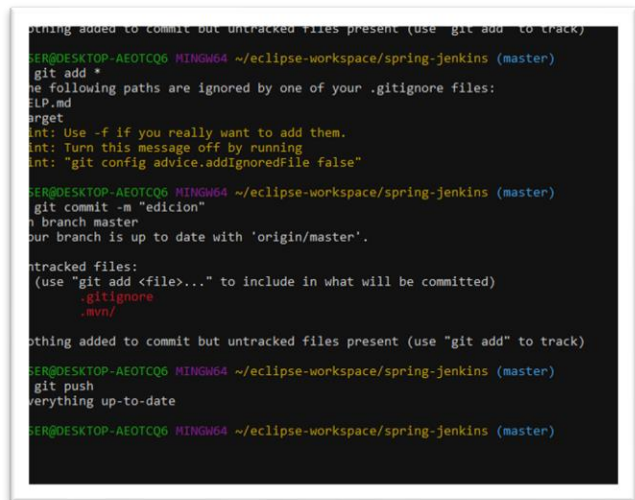


Jenkins trabalha em conjunto com um repositório em este caso se usara github y se subira a informação usando git a carpeta donde se encontra a api Spring

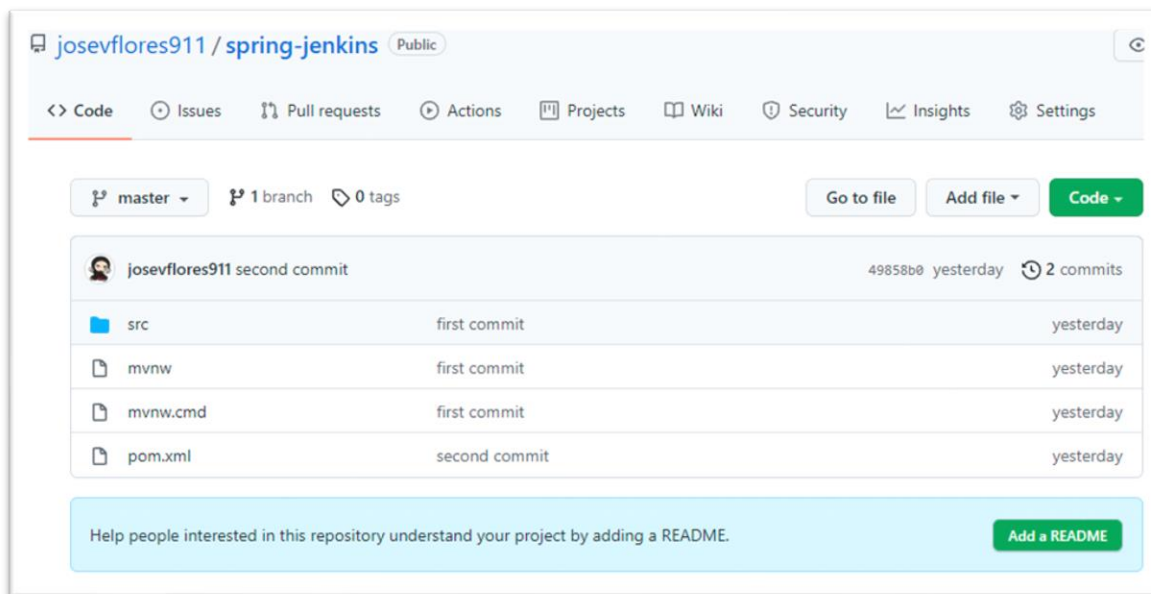
1)



2)



3)



Criamos um novo projeto

Nome do projeto

> This field cannot be empty, please enter a valid name

Construir um projeto de software free-style
Este é a melhor opção para a maioria dos casos. Se você quiser construir seu projeto a partir de um sistema de controle de versão (SVN, Git, etc.) ou se quiser usar um sistema de controle de versão para outros fins (como build de software).

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style/job type.

Construir projeto de múltiplas configurações
Aprimore os projetos que necessitam de grande número de diferentes configurações, como teste em múltiplos ambientes, builds para plataformas específicas, etc.

Folder
Creates a container that stores related items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

Multibranch Pipeline
Creates a set of Pipeline projects according to detected branches in one SCM repository.

Organization Folder
Creates a set of multibranch project subfolders by scanning for repositories.

If you want to create a new item from other existing, you can use this option:

Copy from

OK

Configuramos com a informação de nosso repositório

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

Project url

Display name

☐ This build requires lockable resources
☐ Throttle builds
☐ Desabilitar builds
☐ Execute os builds se necessário

Gerenciamento de código fonte
☐ Nenhum
☒ Git

Repositories

Avançado...

Selecionamos a frequência com a que será feita o teste

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

Agenda

⚠️ Você realmente quis dizer "a todo minuto" quando diz "* * * * *"? Talvez você quisesse dizer "H * * * * *" para verificar uma vez por hora
Deveria ter executado em quinta-feira, 14 de outubro de 2021 23:19:04 Horário Padrão de Brasília; deverá executar novamente em quinta-feira, 14 de outubro de 2021 23:19:04 Horário Padrão de Brasília.

☐ Ignorar ações de pós-commit
☐ GitHub hook trigger for GITScm polling

GeneralGerenciamento de código fonteTrigger de buildsAmbiente de buildBuildAções de pós-build

Chamar alvos Maven de alto nível

Versão do Maven

MAVEN_HOME

Goals

Avançado...

Adicionar passo no build

Ações de pós-build

Notificação de E-mail

Destinatários

JDK

JDK instalações

Adicionar JDK

JDK

Nome

JAVA_HOME

JAVA_HOME

C:\Program Files\Java\jdk-11.0.12

☐ Instalar automaticamente

Adicionar JDK

Lista de JDK instalações nesse sistema

Git

Git installations

Git

Name

Default

Path to Git executable

C:\Program Files\Git\bin\git.exe

Maven

Maven instalações

Adicionar Maven

Maven

Nome

MAVEN_HOME

MAVEN_HOME

C:\Program Files (x86)\apache-maven-3.8.3

☐ Instalar automaticamente

Adicionar Maven

Lista de Maven instalações nesse sistema

Excluir Maven

Jenkins sugere java 11 para su ejecucion pero internamente permite executar diversas versões de java hasta el 9 (al momento de editar este tutorial)

