



Desenvolvimento de Sistemas

Java

Java é uma das linguagens de programação mais populares em todo o mundo. Ela foi criada pela empresa Sun Microsystems, na qual James Gosling liderou uma equipe de pesquisadores em um esforço para criar uma nova linguagem voltada para a televisão interativa. Os princípios para a criação do Java foram simplicidade, robustez, portabilidade, independência de plataforma, segurança, alto desempenho, *multithread*, arquitetura neutra, orientação a objetos, interpretada e dinâmica. A criação da linguagem começou em 1991 e em pouco tempo o foco da equipe mudou para um novo nicho, a **World Wide Web** (rede mundial de computadores, em português).

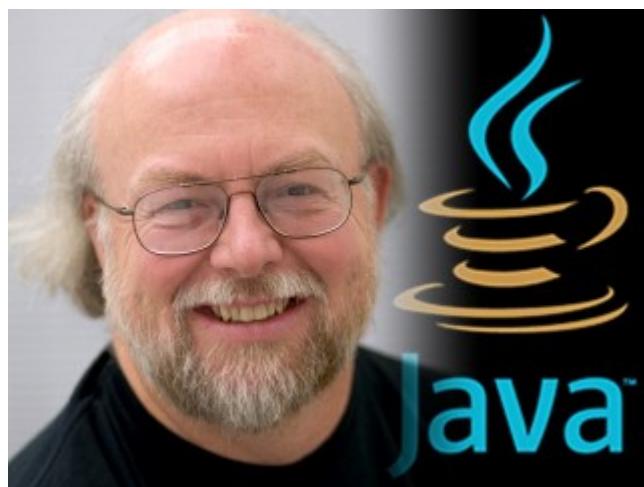


Figura 1 – James Gosling, criador do Java

Fonte: Deep (2012)

No final da década de 1990, a linguagem Java começou a crescer além da *web*, expandindo-se para outros dispositivos como celulares, computadores domésticos e até mesmo o computador de bordo dos veículos de exploração de

Marte da NASA (National Aeronautics and Space Administration). Devido a essa popularidade, a Sun criou diferentes variedades de Java para diferentes propósitos:

- ◆ Java SE (Java Standard Edition): indicada para o desenvolvimento de aplicativos *desktop*.
- ◆ Java ME (Java Micro Edition): indicada para dispositivos móveis e sistemas embutidos.
- ◆ Java EE (Java Enterprise Edition): indicada para o desenvolvimento de aplicações *web*.

Em 2010, a Oracle Corporation assumiu a gestão do Java quando adquiriu a Sun Microsystems. Desde então, a Oracle Corporation tem atualizado a linguagem de programação regularmente com a ajuda de uma comunidade dinâmica. Cada nova versão do Java traz muitos novos recursos e melhorias de desempenho, o que permite a evolução constante da linguagem.

Versões do Java

Muitas versões de Java foram lançadas até agora. A linguagem passou por várias mudanças desde o seu lançamento, com inúmeras adições de classes e pacotes à biblioteca padrão. Além das mudanças técnicas, as versões do Java também tiveram mudanças significativas em suas licenças e no seu suporte.

A versão estável atual do Java é o Java SE 17.

Versão	Data de lançamento
JDK Alfa e Beta	1995
JDK 1.0	23 de janeiro de 1996
JDK 1.1	19 de fevereiro de 1997
J2SE 1.2	8 de dezembro de 1998
J2SE 1.3	8 de maio de 2000
J2SE 1.4	6 de fevereiro de 2002
J2SE 5.0	30 de setembro de 2004
Java SE 6	11 de dezembro de 2006
Java SE 7	28 de julho de 2011
Java SE 8	18 de março de 2014
Java SE 9	21 de setembro de 2017
Java SE 10	20 de março de 2018
Java SE 11	Setembro de 2018
Java SE 12	Março de 2019
Java SE 13	Setembro de 2019
Java SE 14	Março de 2020
Java SE 15	Setembro de 2020
Java SE 16	Março de 2021
Java SE 17	Setembro de 2021

Tabela 1 – Versões do Java

Fonte: Senac EAD (2022)

Se você pesquisar pela versão 17 do Java, provavelmente encontrará, por exemplo, a versão **17.0.2**. Essa sequência de números separados por ponto é uma padronização que está presente em todas as versões do Java, em que:

- ◆ **17** representa a versão base.
- ◆ **0** representa mudanças de bibliotecas. Quando atualizado, significa que mais objetos e/ou métodos foram implementados.
- ◆ **2** representa a correção de *bugs*.
- ◆ Cada versão base do Java contém uma licença na qual é especificado o que se pode ou não fazer com a linguagem.

Assim que a Oracle se tornou proprietária do Java, as versões sucessoras do Java 8 passaram a ter mudanças significativas em suas licenças, solicitando pagamentos que antes não eram necessários. Isso fez com que muitos programadores se mantivessem na versão 8 por muitos anos e também implicou a criação de disponibilização do **OpenJDK**, uma ferramenta de desenvolvimento análoga às ferramentas padrão do Java, mas mantida por uma comunidade independente de programadores.

Com a chegada do **Java 17**, tem-se uma nova licença: a “Oracle No-Fee Terms and Conditions” – NFTC (ou “Termos e Condições Sem Taxa da Oracle”, em português). Essa licença permitiu o uso gratuito para todos os usuários, mesmo sendo uso comercial e de produção, algo muito semelhante à licença do Java 8, e essa licença deve fazer parte das futuras versões do Java também.

Outra questão que costuma confundir bastante quem está começando a estudar Java é a diferença entre os termos JVM, JRE e JDK.

JVM (Máquina Virtual Java)

A primeira coisa que você precisa ter em mente é que o Java é muito conhecido pelo conceito de multiplataforma e esse é o principal motivo do seu grande sucesso. A JVM (Java Virtual Machine) é um programa cuja finalidade é

executar programas Java. A JVM tem duas funções principais: permitir que programas Java sejam executados em qualquer dispositivo ou sistema operacional (conhecido como princípio *write once, run anywhere* ou, em português, “gravar uma vez, executar em qualquer lugar”) e gerenciar e otimizar a memória do programa.

Quando o Java foi lançado em 1995, todos os programas de computador eram escritos em um sistema operacional específico e a memória do programa era gerenciada pelo desenvolvedor do *software*. Então, a JVM foi uma inovação para a época.

JRE (Ambiente de Execução Java)

O JRE contém uma JVM, os pacotes básicos do Java API Core e todas as ferramentas necessárias para a **execução** de programas Java. O Java presente na maioria das máquinas é uma versão do JRE e é o mínimo necessário para a execução de programas Java.

O JDK (Java Development Kit) é uma coleção de ferramentas para o desenvolvimento de aplicações Java. Ele inclui as APIs (**application programming interfaces**, ou interfaces de programação de aplicação, em português) Java para o desenvolvimento de aplicações, um compilador, um depurador e o próprio JRE para executar as aplicações desenvolvidas.

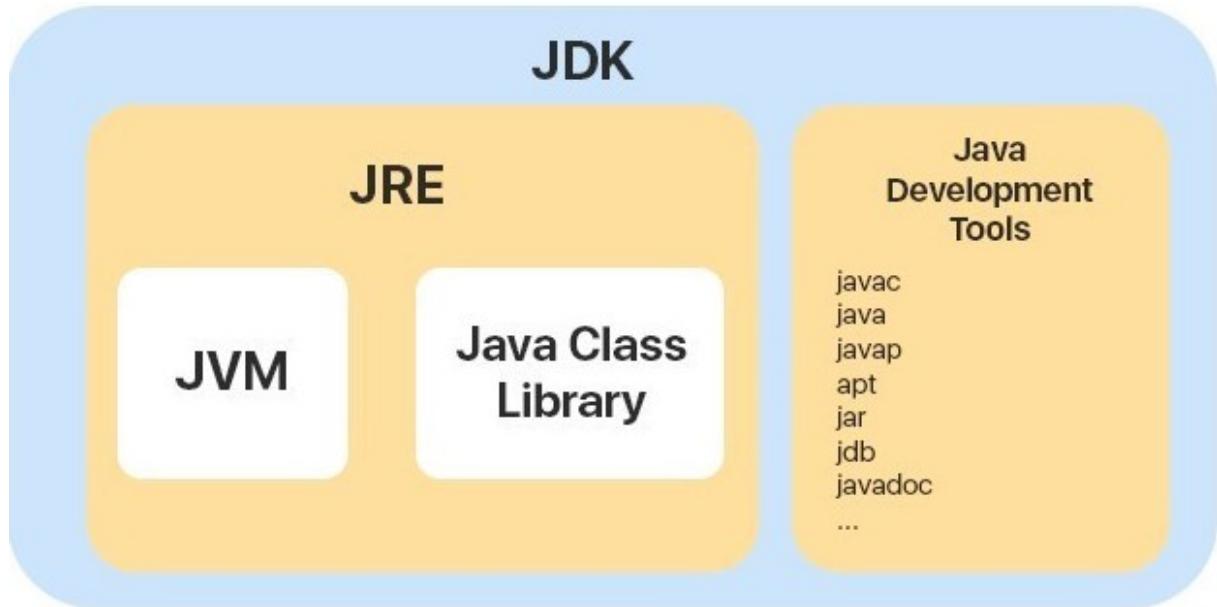


Figura 2 – Diferença entre JDK, JRE, JVM e Java Class Library

Fonte: Andrew; Dora (2019)

Na prática, a interação da JVM, do JRE e do JDK ocorre da seguinte maneira:

1. Você escreve o código do seu projeto usando um IDE (ambiente de desenvolvimento integrado).
2. Você utiliza o JDK para compilar o seu código e gerar um programa.
3. Para executar o seu programa, a JVM lê o seu código compilado e as bibliotecas necessárias que estão incluídas no JRE.
4. O seu programa é aberto e executado.

Com isso, conclui-se que, se o seu objetivo é desenvolver aplicações Java, é essencial que ter o JDK instalado no seu computador. Agora, se o seu objetivo é apenas executar uma aplicação Java, então você precisará apenas do JRE. Em outras palavras, o JDK é destinado aos programadores Java, enquanto o JRE é destinado aos usuários de *softwares* desenvolvidos em Java.

Agora que você entendeu o que significam as siglas do mundo Java, que tal instalar e configurar o ambiente de desenvolvimento para o Java?

Ambiente de desenvolvimento

Um ambiente de desenvolvimento é o conjunto de ferramentas utilizado no desenvolvimento de sistemas. Geralmente, esse ambiente é composto de algum *software* específico da linguagem de programação que será utilizada para programar e de um IDE, que conterá um editor de texto interno para escrever os códigos e também outras ferramentas para facilitar o processo de desenvolvimento de *software*.

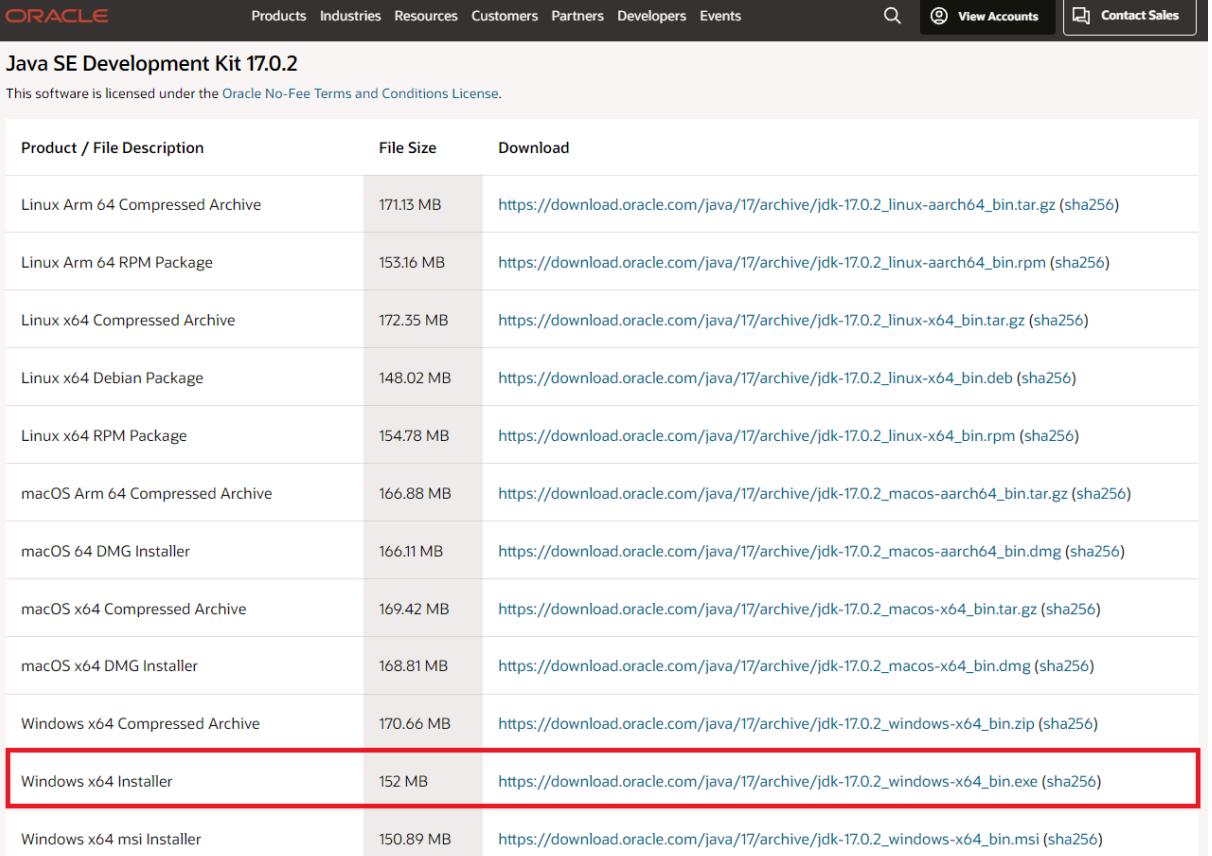
Pensando na linguagem de programação Java, você precisará de uma versão do JDK e um IDE. Portanto, este estudo utilizará as versões mais recentes, o **Java Development Kit 17**, lançado em setembro de 2021, e o **Apache NetBeans IDE 13**, lançado em 4 de março de 2022.

As versões mais recentes do **Java Development Kit** e do **Apache NetBeans IDE** podem variar, de acordo com a data em que você estiver lendo esse conteúdo.

JDK 17 – Instalação e configuração (Windows)

O primeiro passo para instalar o Java Development Kit 17 no Windows é baixar o instalador por meio do *site* oficial da Oracle. A versão mais recente disponível é a 17.0.2. Logo, essa é a versão que será utilizada neste material. Procure por “Java SE 17 Archive Downloads” para acessar a página com opções de *download* dessa versão.

Para fazer o *download* do instalador, clique no *link* de *download* do **Windows x64 Installer**.



The screenshot shows the Oracle Java SE Development Kit 17.0.2 download page. At the top, there's a navigation bar with links for Products, Industries, Resources, Customers, Partners, Developers, Events, a search icon, and buttons for View Accounts and Contact Sales. Below the navigation bar, the title "Java SE Development Kit 17.0.2" is displayed, followed by a note about licensing under the Oracle No-Fee Terms and Conditions License.

The main content is a table with three columns: Product / File Description, File Size, and Download. The table lists various download options:

Product / File Description	File Size	Download
Linux Arm 64 Compressed Archive	171.13 MB	https://download.oracle.com/java/17/archive/jdk-17.0.2_linux-aarch64_bin.tar.gz (sha256)
Linux Arm 64 RPM Package	153.16 MB	https://download.oracle.com/java/17/archive/jdk-17.0.2_linux-aarch64_bin.rpm (sha256)
Linux x64 Compressed Archive	172.35 MB	https://download.oracle.com/java/17/archive/jdk-17.0.2_linux-x64_bin.tar.gz (sha256)
Linux x64 Debian Package	148.02 MB	https://download.oracle.com/java/17/archive/jdk-17.0.2_linux-x64_bin.deb (sha256)
Linux x64 RPM Package	154.78 MB	https://download.oracle.com/java/17/archive/jdk-17.0.2_linux-x64_bin.rpm (sha256)
macOS Arm 64 Compressed Archive	166.88 MB	https://download.oracle.com/java/17/archive/jdk-17.0.2_macos-aarch64_bin.tar.gz (sha256)
macOS 64 DMG Installer	166.11 MB	https://download.oracle.com/java/17/archive/jdk-17.0.2_macos-aarch64_bin.dmg (sha256)
macOS x64 Compressed Archive	169.42 MB	https://download.oracle.com/java/17/archive/jdk-17.0.2_macos-x64_bin.tar.gz (sha256)
macOS x64 DMG Installer	168.81 MB	https://download.oracle.com/java/17/archive/jdk-17.0.2_macos-x64_bin.dmg (sha256)
Windows x64 Compressed Archive	170.66 MB	https://download.oracle.com/java/17/archive/jdk-17.0.2_windows-x64_bin.zip (sha256)
Windows x64 Installer	152 MB	https://download.oracle.com/java/17/archive/jdk-17.0.2_windows-x64_bin.exe (sha256)
Windows x64 msi Installer	150.89 MB	https://download.oracle.com/java/17/archive/jdk-17.0.2_windows-x64_bin.msi (sha256)

Figura 3 – Site oficial do Java JDK 17

Fonte: Oracle (c2022)

Depois de fazer o *download* do instalador, abra-o para iniciar o processo de instalação do JDK 17. Na primeira tela, você terá uma mensagem de boas-vindas e não será necessário realizar nenhuma ação. Em seguida, clique no botão **Next** para continuar.

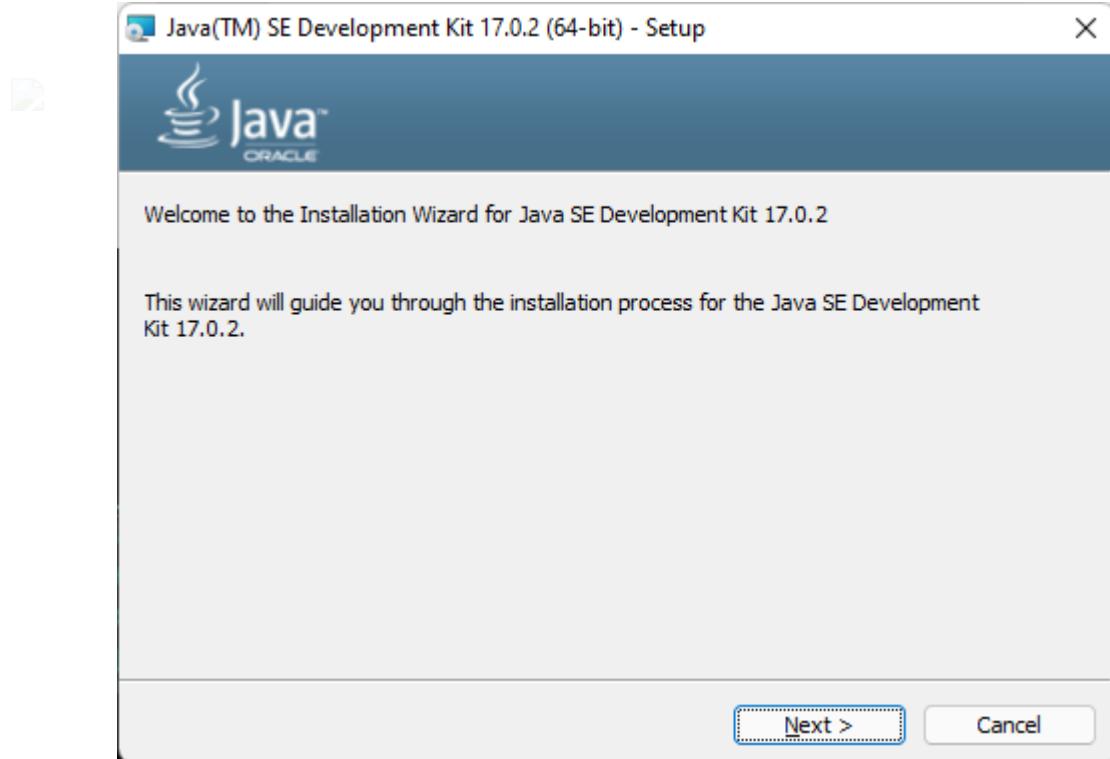


Figura 4 – Tela de boas-vindas do instalador do Java Development Kit 17

Fonte: Java Development Kit 17 (2022)

Na tela seguinte, você deve informar ao instalador onde o JDK deve ser instalado. Por padrão, o local informado é o **C:\Program Files\Java\jdk-17.0.2** e recomenda-se que esse caminho não seja alterado, para facilitar a localização do JDK por outras ferramentas. Sendo assim, não é necessário fazer nenhuma configuração nessa tela. Clique em **Next** para continuar.

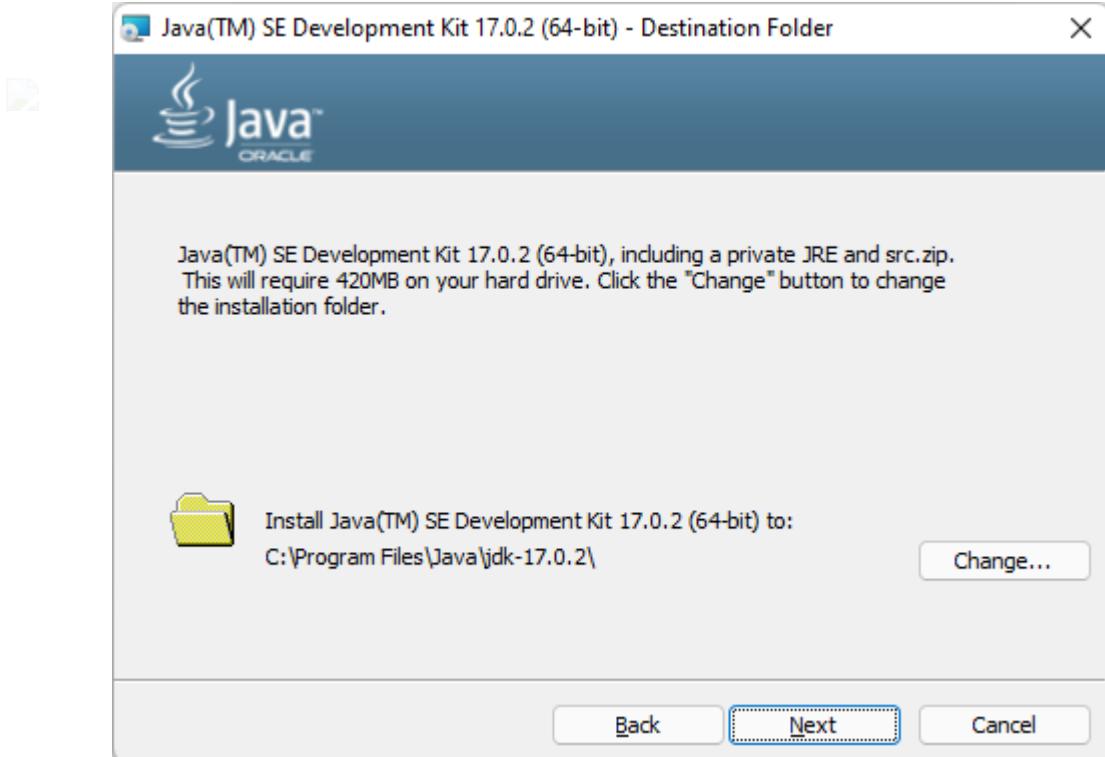


Figura 5 – Tela de local de instalação do instalador do Java Development Kit 17

Fonte: Java Development Kit 17 (2022)

Agora, o JDK começará o processo de instalação no seu computador. Aguarde.



Figura 6 – Tela de andamento do instalador do Java Development Kit 17

Fonte: Java Development Kit 17 (2022)

Quando a instalação for concluída, você será levado a uma nova tela do instalador, que informará que o JDK foi instalado com sucesso. Para finalizar a instalação, clique em **Close**.



Figura 7 – Tela de conclusão do instalador do Java Development Kit 17

Fonte: Java Development Kit 17 (2022)

Variáveis de ambiente

Quando um programa é executado, ele pode precisar de informações sobre o ambiente em que ele está sendo executado. Essas informações são passadas via variáveis de ambiente. Na prática, uma variável de ambiente é um atalho para um

valor. Na maioria dos casos, é preciso adicionar esses valores para que alguns programas consigam encontrar o que precisam para serem executados corretamente.

Por exemplo, muitos programas precisam saber onde o binário do Java está localizado. Mas encontrá-lo não é algo tão simples, pois esse local pode variar dependendo do sistema operacional e até mesmo da versão do Java. Logo, a forma mais simples para esse programa acessar o binário do Java é por meio do valor informado na variável de ambiente.

No caso do Java, é preciso configurar manualmente a variável de ambiente para que outros programas consigam localizá-lo e ele funcione corretamente. Para isso, execute o seguinte passo a passo:

1. Clique no ícone do Windows localizado na barra de tarefas e digite “variáveis” para pesquisar o termo entre os *softwares* do seu computador.
2. Selecione a opção **Editar as variáveis de ambiente do sistema**.
Tenha cuidado para não confundir com “Editar as variáveis de ambiente para sua conta”.
3. A janela **Propriedades do sistema** será aberta na guia **Avançado**.
Clique no botão **Variáveis de ambiente....**
4. Em **Variáveis do sistema**, clique em **Novo**.
5. No campo **Nome da variável**, digite o nome **JAVA_HOME**.
6. No campo **Valor da variável**, informe o caminho da instalação do JDK.
Se você não tiver alterado o caminho de instalação durante a instalação do Java JDK 17, então seu caminho provavelmente será **C:\Program Files\Java\jdk-17.0.2**. Caso você não saiba o caminho, use o botão **Procurar no diretório...** para localizar o caminho.

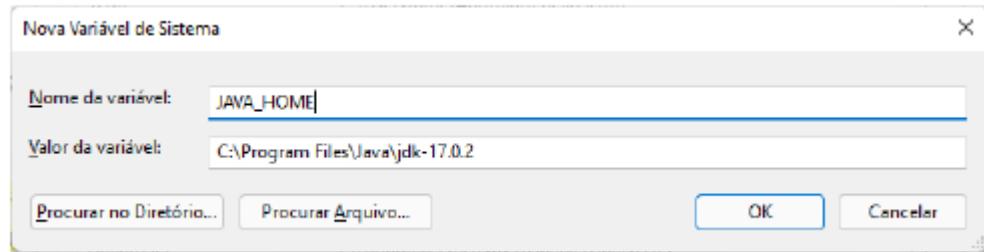


Figura 8 – Nova variável de sistema do Windows

Fonte: Senac EAD (2022)

7. Clique em **Ok** para adicionar a variável de ambiente.

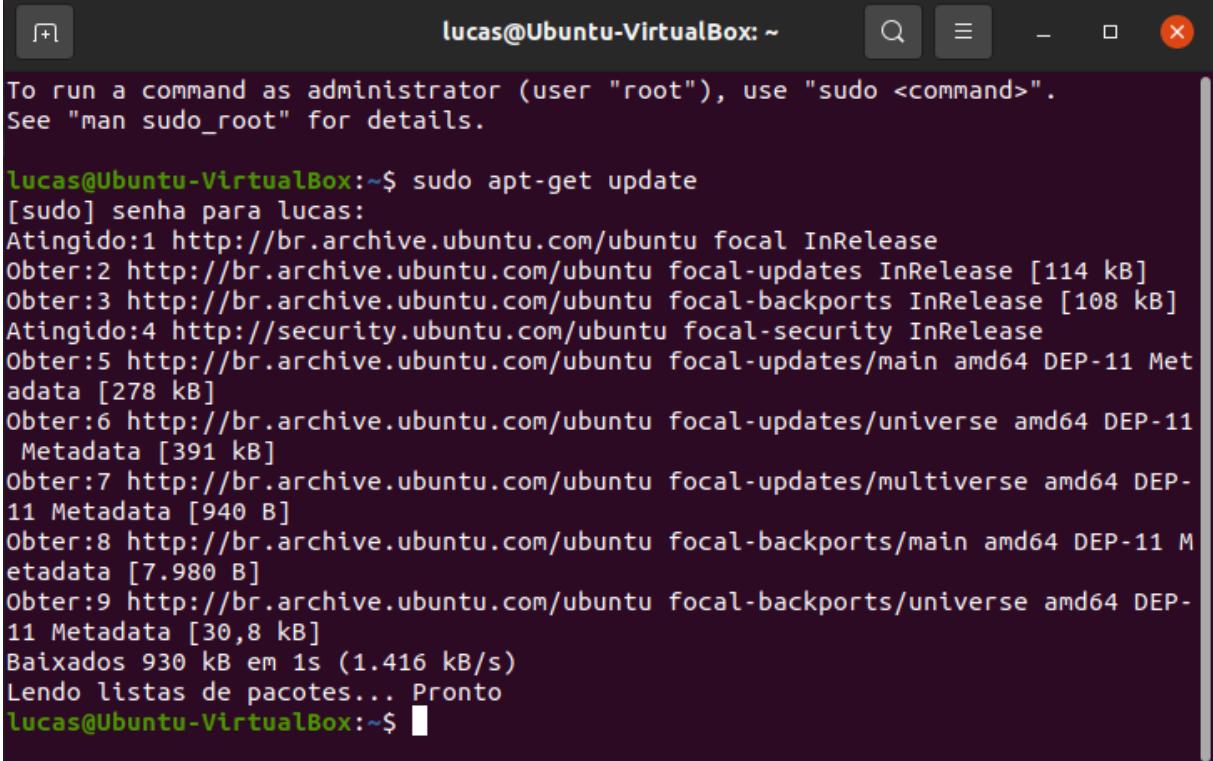
Para conferir se a variável foi configurada corretamente, abra o **Prompt de Comando** do Windows e execute o comando **echo %JAVA_HOME%**. Se estiver tudo certo, então o caminho de instalação do Java será exibido na tela. Caso não apareça, reinicie seu computador para que as alterações sejam processadas.

JDK 17 – Instalação e configuração (Linux)

Para esse passo a passo de instalação e configuração do Java no Linux, a distribuição **Ubuntu** será utilizada como base.

O primeiro passo para instalar o Java no Linux é atualizar o índice de pacotes do sistema. No terminal do Ubuntu, utilize este comando:

```
sudo apt-get update
```



```
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

lucas@Ubuntu-VirtualBox:~$ sudo apt-get update
[sudo] senha para lucas:
Atingido:1 http://br.archive.ubuntu.com/ubuntu focal InRelease
Obter:2 http://br.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Obter:3 http://br.archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
Atingido:4 http://security.ubuntu.com/ubuntu focal-security InRelease
Obter:5 http://br.archive.ubuntu.com/ubuntu focal-updates/main amd64 DEP-11 Met
adata [278 kB]
Obter:6 http://br.archive.ubuntu.com/ubuntu focal-updates/universe amd64 DEP-11
Metadata [391 kB]
Obter:7 http://br.archive.ubuntu.com/ubuntu focal-updates/multiverse amd64 DEP-
11 Metadata [940 B]
Obter:8 http://br.archive.ubuntu.com/ubuntu focal-backports/main amd64 DEP-11 M
etadata [7.980 B]
Obter:9 http://br.archive.ubuntu.com/ubuntu focal-backports/universe amd64 DEP-
11 Metadata [30,8 kB]
Baixados 930 kB em 1s (1.416 kB/s)
Lendo listas de pacotes... Pronto
lucas@Ubuntu-VirtualBox:~$
```

Figura 9 – Terminal do Ubuntu

Fonte: Ubuntu (2022)

Após isso, verifique se o Java já está instalado (ou não) com o comando:

```
java -version
```

Se o Java não estiver instalado, você terá a seguinte saída:

```
lucas@Ubuntu-VirtualBox: ~
Obter:6 http://br.archive.ubuntu.com/ubuntu focal-updates/universe amd64 DEP-11
Metadata [391 kB]
Obter:7 http://br.archive.ubuntu.com/ubuntu focal-updates/multiverse amd64 DEP-
11 Metadata [940 B]
Obter:8 http://br.archive.ubuntu.com/ubuntu focal-backports/main amd64 DEP-11 M
etadada [7.980 B]
Obter:9 http://br.archive.ubuntu.com/ubuntu focal-backports/universe amd64 DEP-
11 Metadata [30,8 kB]
Baixados 930 kB em 1s (1.416 kB/s)
Lendo listas de pacotes... Pronto
lucas@Ubuntu-VirtualBox:~$ java -version

Comando 'java' não encontrado, mas poder ser instalado com:

sudo apt install openjdk-11-jre-headless # version 11.0.14+9-0ubuntu2~20.04, o
r
sudo apt install default-jre           # version 2:1.11-72
sudo apt install openjdk-13-jre-headless # version 13.0.7+5-0ubuntu1~20.04
sudo apt install openjdk-16-jre-headless # version 16.0.1+9-1~20.04
sudo apt install openjdk-17-jre-headless # version 17.0.2+8-1~20.04
sudo apt install openjdk-8-jre-headless # version 8u312-b07-0ubuntu1~20.04

lucas@Ubuntu-VirtualBox:~$
```

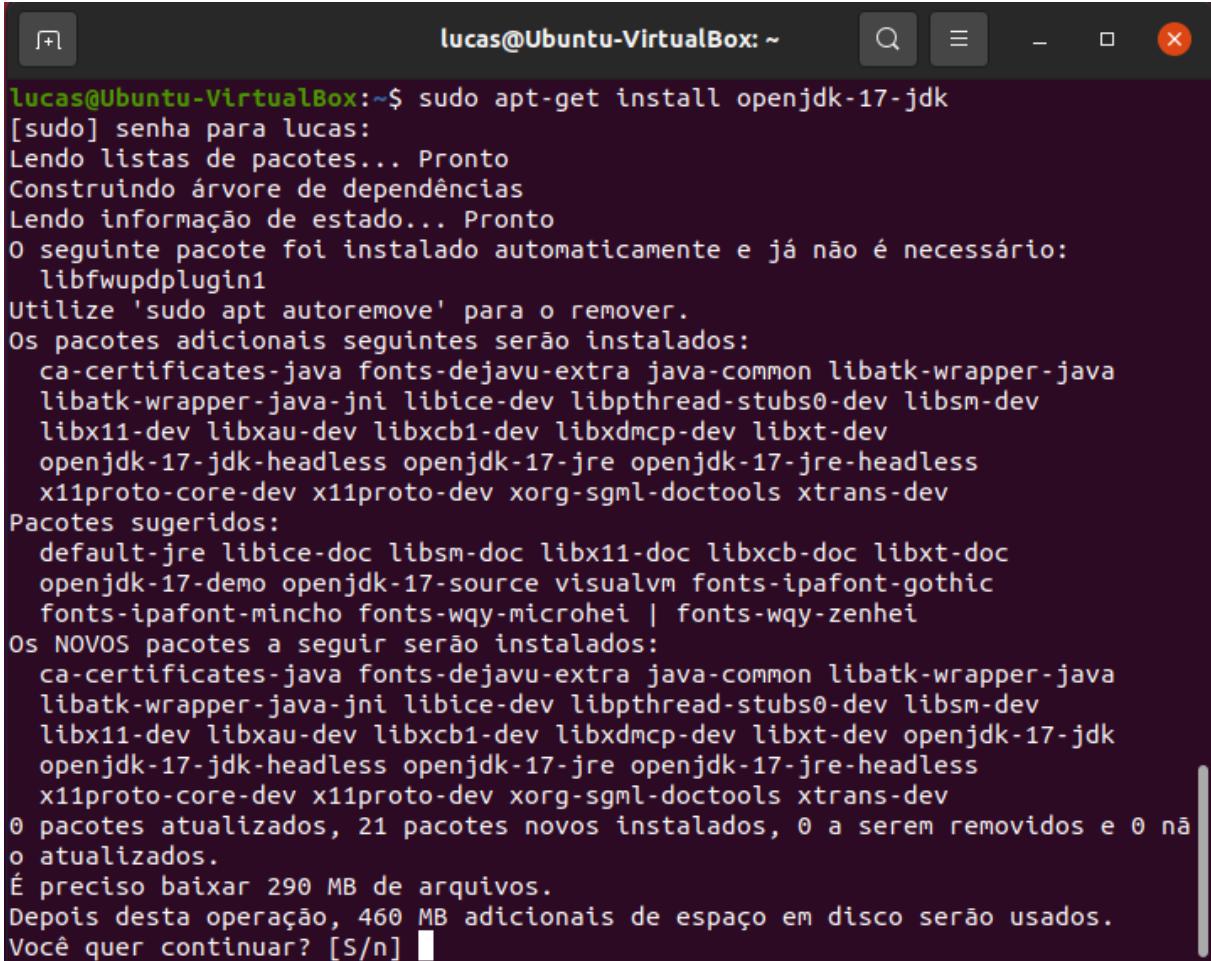
Figura 10 – Terminal do Ubuntu

Fonte: Ubuntu (2022)

A maneira mais simples de instalar o Java no Ubuntu é por meio dos repositórios do sistema. Por padrão, o Ubuntu empacota o OpenJDK, que é uma alternativa de código aberto para o JRE e o JDK da Oracle. Logo, siga com a instalação do OpenJDK 17 usando o seguinte comando:

```
sudo apt-get install openjdk-17-jdk
```

Após isso, basta confirmar a operação pressionando a tecla **Enter**.



```
lucas@Ubuntu-VirtualBox:~$ sudo apt-get install openjdk-17-jdk
[sudo] senha para lucas:
Lendo listas de pacotes... Pronto
Construindo árvore de dependências
Lendo informação de estado... Pronto
0 seguinte pacote foi instalado automaticamente e já não é necessário:
 libfwupdplugin1
Utilize 'sudo apt autoremove' para o remover.
Os pacotes adicionais seguintes serão instalados:
 ca-certificates-java fonts-dejavu-extra java-common libatk-wrapper-java
 libatk-wrapper-java-jni libice-dev libpthread-stubs0-dev libsm-dev
 libx11-dev libxau-dev libxcb1-dev libxdmcp-dev libxt-dev
 openjdk-17-jdk-headless openjdk-17-jre openjdk-17-jre-headless
 x11proto-core-dev x11proto-dev xorg-sgml-doctools xtrans-dev
Pacotes sugeridos:
 default-jre libice-doc libsm-doc libx11-doc libxcb-doc libxt-doc
 openjdk-17-demo openjdk-17-source visualvm fonts-ipafont-gothic
 fonts-ipafont-mincho fonts-wqy-microhei | fonts-wqy-zenhei
Os NOVOS pacotes a seguir serão instalados:
 ca-certificates-java fonts-dejavu-extra java-common libatk-wrapper-java
 libatk-wrapper-java-jni libice-dev libpthread-stubs0-dev libsm-dev
 libx11-dev libxau-dev libxcb1-dev libxdmcp-dev libxt-dev openjdk-17-jdk
 openjdk-17-jdk-headless openjdk-17-jre openjdk-17-jre-headless
 x11proto-core-dev x11proto-dev xorg-sgml-doctools xtrans-dev
0 pacotes atualizados, 21 pacotes novos instalados, 0 a serem removidos e 0 não
atualizados.
É preciso baixar 290 MB de arquivos.
Depois desta operação, 460 MB adicionais de espaço em disco serão usados.
Você quer continuar? [S/n]
```

Figura 11 – Terminal do Ubuntu

Fonte: Ubuntu (2022)

Observação

Outra forma de instalar o OpenJDK seria por meio deste comando:

```
sudo apt-get install default-jdk
```

Esse comando instalará a última versão disponível do JDK e, por isso, é importante que ele seja usado com cautela, pois é possível que uma versão diferente do JDK seja instalada.

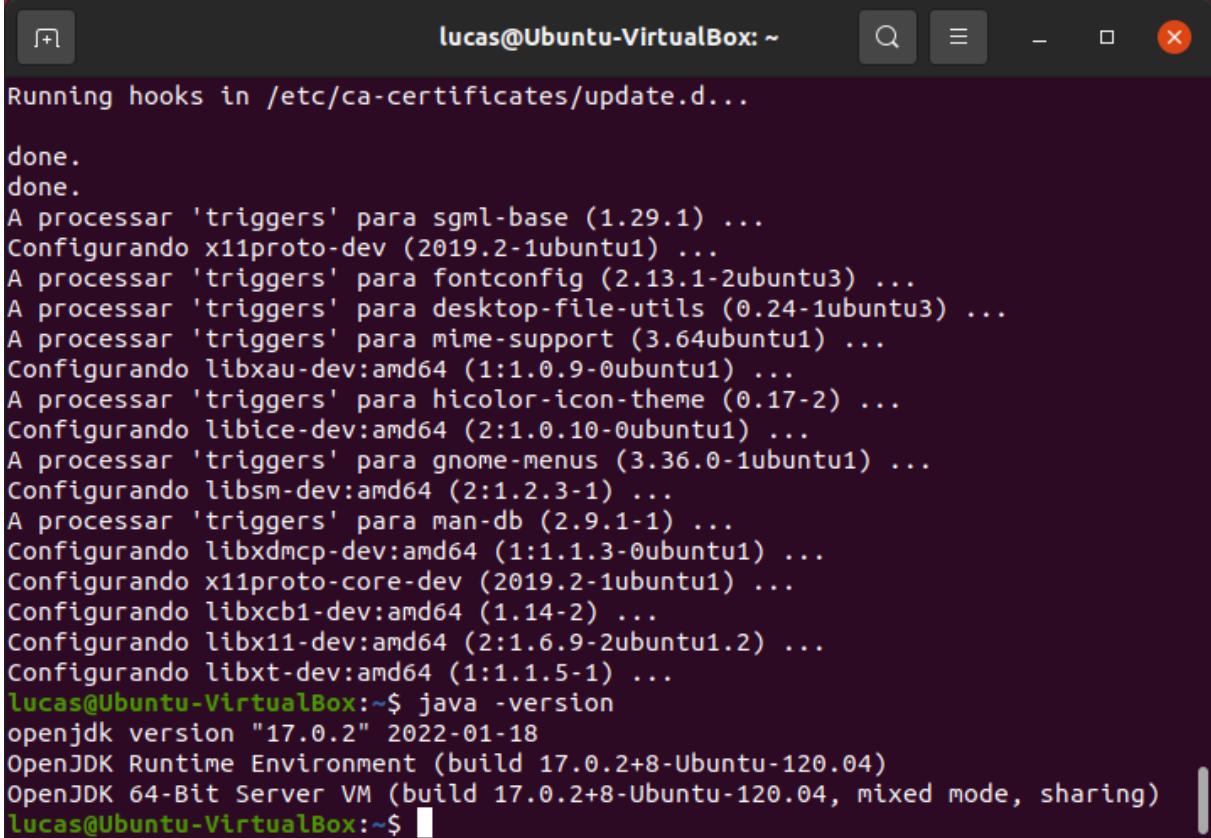
Caso você queira fazer o *download* do JDK oficial da Oracle, acesse o *site* oficial do Oracle JDK 17 e prossiga com a instalação do pacote **Linux x64 Debian Package** oferecido na lista de *downloads*. Após isso, abra o terminal, acesse a pasta de *downloads* e execute o seguinte comando:

```
sudo dpkg -i jdk-17.0.3_linux-x64_bin.deb
```

Caso escolha essa abordagem, lembre-se de que o nome do arquivo pode variar, de acordo com a versão disponibilizada para *download*.

Assim que o processo de instalação terminar, verifique se está tudo correto executando este comando:

```
java -version
```



A screenshot of a terminal window titled "lucas@Ubuntu-VirtualBox: ~". The terminal displays the output of a command that runs hooks in /etc/ca-certificates/update.d/. It shows various packages being processed, such as sgml-base, x11proto-dev, fontconfig, desktop-file-utils, mime-support, libxau-dev:amd64, hicolor-icon-theme, libice-dev:amd64, gnome-menus, libsm-dev:amd64, man-db, libxdmcp-dev:amd64, x11proto-core-dev, libxcb1-dev:amd64, libx11-dev:amd64, and libxt-dev:amd64. The output ends with the command "java -version" which returns the OpenJDK version 17.0.2.

```
Running hooks in /etc/ca-certificates/update.d...
done.
done.
A processar 'triggers' para sgml-base (1.29.1) ...
Configurando x11proto-dev (2019.2-1ubuntu1) ...
A processar 'triggers' para fontconfig (2.13.1-2ubuntu3) ...
A processar 'triggers' para desktop-file-utils (0.24-1ubuntu3) ...
A processar 'triggers' para mime-support (3.64ubuntu1) ...
Configurando libxau-dev:amd64 (1:1.0.9-0ubuntu1) ...
A processar 'triggers' para hicolor-icon-theme (0.17-2) ...
Configurando libice-dev:amd64 (2:1.0.10-0ubuntu1) ...
A processar 'triggers' para gnome-menus (3.36.0-1ubuntu1) ...
Configurando libsm-dev:amd64 (2:1.2.3-1) ...
A processar 'triggers' para man-db (2.9.1-1) ...
Configurando libxdmcp-dev:amd64 (1:1.1.3-0ubuntu1) ...
Configurando x11proto-core-dev (2019.2-1ubuntu1) ...
Configurando libxcb1-dev:amd64 (1.14-2) ...
Configurando libx11-dev:amd64 (2:1.6.9-2ubuntu1.2) ...
Configurando libxt-dev:amd64 (1:1.1.5-1) ...
lucas@Ubuntu-VirtualBox:~$ java -version
openjdk version "17.0.2" 2022-01-18
OpenJDK Runtime Environment (build 17.0.2+8-Ubuntu-120.04)
OpenJDK 64-Bit Server VM (build 17.0.2+8-Ubuntu-120.04, mixed mode, sharing)
lucas@Ubuntu-VirtualBox:~$
```

Figura 12 – Terminal do Ubuntu

Fonte: Ubuntu (2022)

Agora, o terminal deve retornar a versão instalada do Java. Se a versão 17 for retornada no seu terminal, isso significa que o JDK 17 foi devidamente instalado no seu sistema Ubuntu.

IDE

Um IDE é um *software* usado para criar aplicações que combina ferramentas comuns de desenvolvimento em uma única interface gráfica do usuário para facilitar o desenvolvimento de sistemas. Geralmente, um IDE ajuda você a organizar seus projetos de *software*, escrever código, testar, depurar e compilar o código. Como criar *softwares* é uma tarefa que demanda bastante tempo, é muito importante saber escolher um IDE que realmente torne o desenvolvimento de *software* mais produtivo e agradável.

A escolha de um IDE de um desenvolvedor pode ser uma coisa muito pessoal e o Java contém diversas opções de IDEs disponíveis. Cada IDE tem seus pontos positivos e negativos, por isso é importante fazer uma boa escolha. Como você está recém iniciando no mundo da programação Java, a melhor opção de IDE para começar é o **Apache NetBeans IDE**.

Apache NetBeans IDE



Figura 13 – Apache NetBeans IDE (logo)

Fonte: Ferreira (2020)

O NetBeans começou como um projeto estudantil (originalmente chamado Xelfi) na República Tcheca, em 1996. Xelfi foi o primeiro IDE Java escrito em Java, com seus primeiros pré-lançamentos em 1997.

Em junho de 2000, o NetBeans tornou-se *open source* (“código aberto”) pela Sun Microsystems, empresa que permaneceu como patrocinadora do projeto até janeiro de 2010, quando se tornou uma subsidiária da Oracle.

Ao longo de sua história na Sun Microsystems e na Oracle, o NetBeans tem sido gratuito e de código aberto e foi aproveitado por seu patrocinador como um mecanismo para impulsionar o ecossistema Java. Em 2016, a Oracle doou o código-fonte do NetBeans para a Apache Software Foundation, resultando assim no atual **Apache Netbeans IDE**.

O **Apache NetBeans IDE** permite que você desenvolva rapidamente e com facilidade aplicativos de *desktop* Java, Java EE e da *web*, bem como aplicativos HTML5 com HTML, JavaScript e CSS. O IDE também fornece um grande conjunto de ferramentas para desenvolvedores PHP e C/C++. É gratuito e de código aberto e contém uma grande comunidade de usuários e desenvolvedores em todo o mundo.

O Apache NetBeans IDE está disponível para as plataformas Windows, Linux, macOS e Solaris.

Por ser um IDE bastante leve, multiplataforma e que está sempre sendo atualizado para acompanhar as versões do Java, o Apache NetBeans IDE é uma boa porta de entrada para desenvolvedores ao mundo Java. Atualmente, a versão mais recente é a versão 13, lançada em 4 de março de 2022.

A versão mais recente do **Apache Netbeans IDE** pode variar, dependendo da data em que você estiver lendo este conteúdo.

Nos próximos tópicos, você estudará como instalar, configurar e criar projetos no **Apache Netbeans IDE**. Porém, é essencial que você já tenha concluído a instalação do Java Development Kit primeiro, pois, durante a instalação do Apache

Netbeans IDE, precisará informar onde o JDK está instalado para seguir com a instalação.

Apache Netbeans IDE 13 – Instalação e configuração (Windows)

O primeiro passo é baixar o instalador do Apache NetBeans IDE no *site* oficial. No seu navegador, busque por “Download Apache NetBeans 13” para acessar a página de *downloads*. Como você está fazendo a instalação no sistema operacional Windows, selecione a primeira opção de instalador: **Apache-NetBeans-13-bin-windows-x64.exe**.

The screenshot shows the Apache NetBeans official website's download page for version 13. At the top, there's a navigation bar with links for Community, Participate, Blog, Get Help, Plugins, and Download. The main title is "Downloading Apache NetBeans 13". Below the title, it says "Apache NetBeans 13 was released on March 4, 2022. See Apache NetBeans 13 Features for a full list of features." It then lists download options: "Binaries: netbeans-13-bin.zip (SHA-512, PGP ASC)" and "Installers: Apache-NetBeans-13-bin-windows-x64.exe (SHA-512, PGP ASC), Apache-NetBeans-13-bin-linux-x64.sh (SHA-512, PGP ASC), Apache-NetBeans-13-bin-macosx.dmg (SHA-512, PGP ASC)". There's also a link for "Source: netbeans-13-source.zip (SHA-512, PGP ASC)". To the right, there's a sidebar with links for "Community Installers", "Deployment Platforms", "Building from Source", "Community Approval", and "Earlier Releases". At the bottom, it says "Officially, it is important that you verify the integrity of the downloaded files using the PGP signatures (.asc file) or a hash (.sha512 files). The PGP keys used to sign this release are available here." and "Apache NetBeans can also be installed as a self-contained snap package on Linux."

Figura 14 – Site oficial do Apache NetBeans IDE 13

Fonte: Apache NetBeans (2022)

Após fazer o *download* do Apache Netbeans 13 no *site* oficial, abra o instalador para iniciar o processo de instalação.

Na primeira tela do instalador, você terá uma breve mensagem de boas-vindas e apresentação do *software*. Apesar de ser possível especificar quais componentes serão instalados pelo botão **Customize...**, opte pela instalação completa, pois você precisará desses outros recursos no futuro. Então, pressione o botão **Next** para continuar.

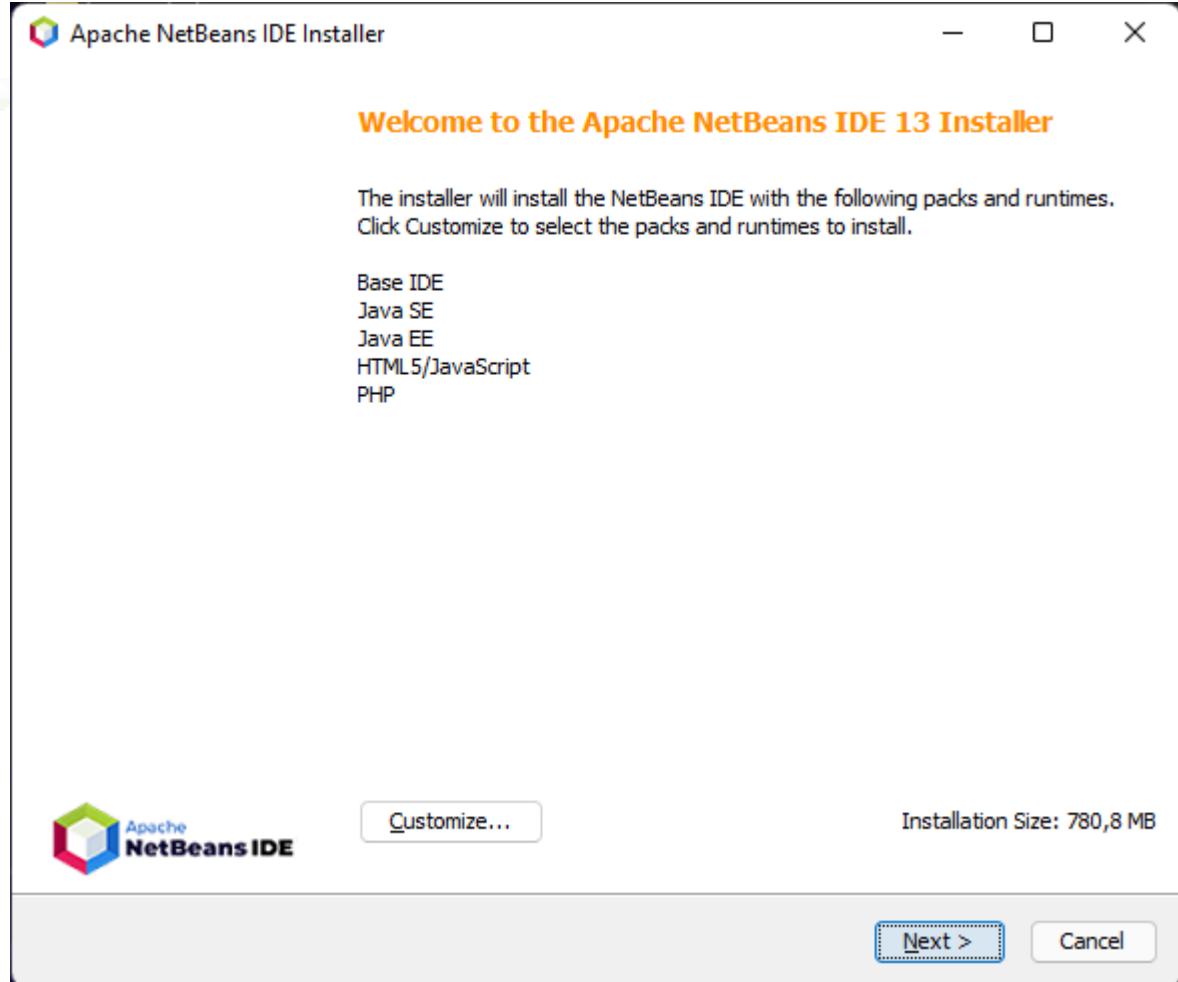


Figura 15 – Tela de boas-vindas do instalador do Apache NetBeans IDE

Fonte: Apache NetBeans IDE (2022)

Na tela seguinte, será apresentado o contrato de licença do Apache NetBeans IDE. Assim que finalizar a leitura, você pode marcar a opção **I accept the terms in the license agreement** e clicar no botão **Next** para continuar a instalação.

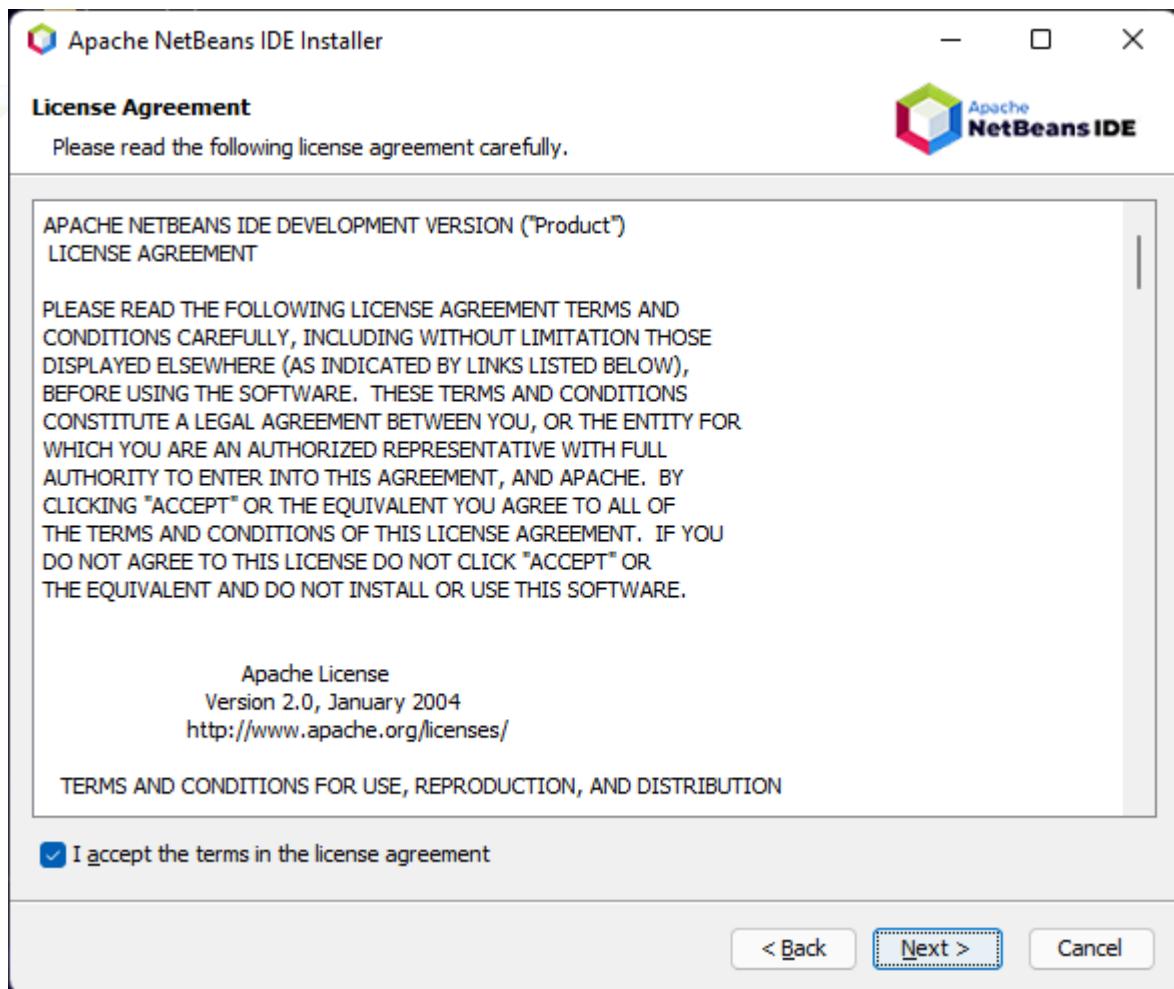


Figura 16 – Tela de contrato do instalador do Apache NetBeans IDE

Fonte: Apache NetBeans IDE (2022)

Na tela seguinte, o instalador solicitará que você indique dois locais:

Install the Apache NetBeans IDE in

Essa opção refere-se ao local em que o IDE será instalado. Por padrão, o local de instalação será na pasta **Program Files** do Windows.

JDK for the Apache NetBeans IDE

Nessa opção, é preciso informar onde o JDK está instalado. Por padrão, o instalador apontará para a versão mais recente do JDK que existir na pasta Java. Caso não haja, você precisará indicar manualmente o local em que o JDK foi

instalado. Se você seguiu o conteúdo passo a passo até aqui, então esse campo já estará preenchido, pois já terá feito a instalação do JDK.

Se estiver tudo correto, siga com a instalação clicando em **Next**.

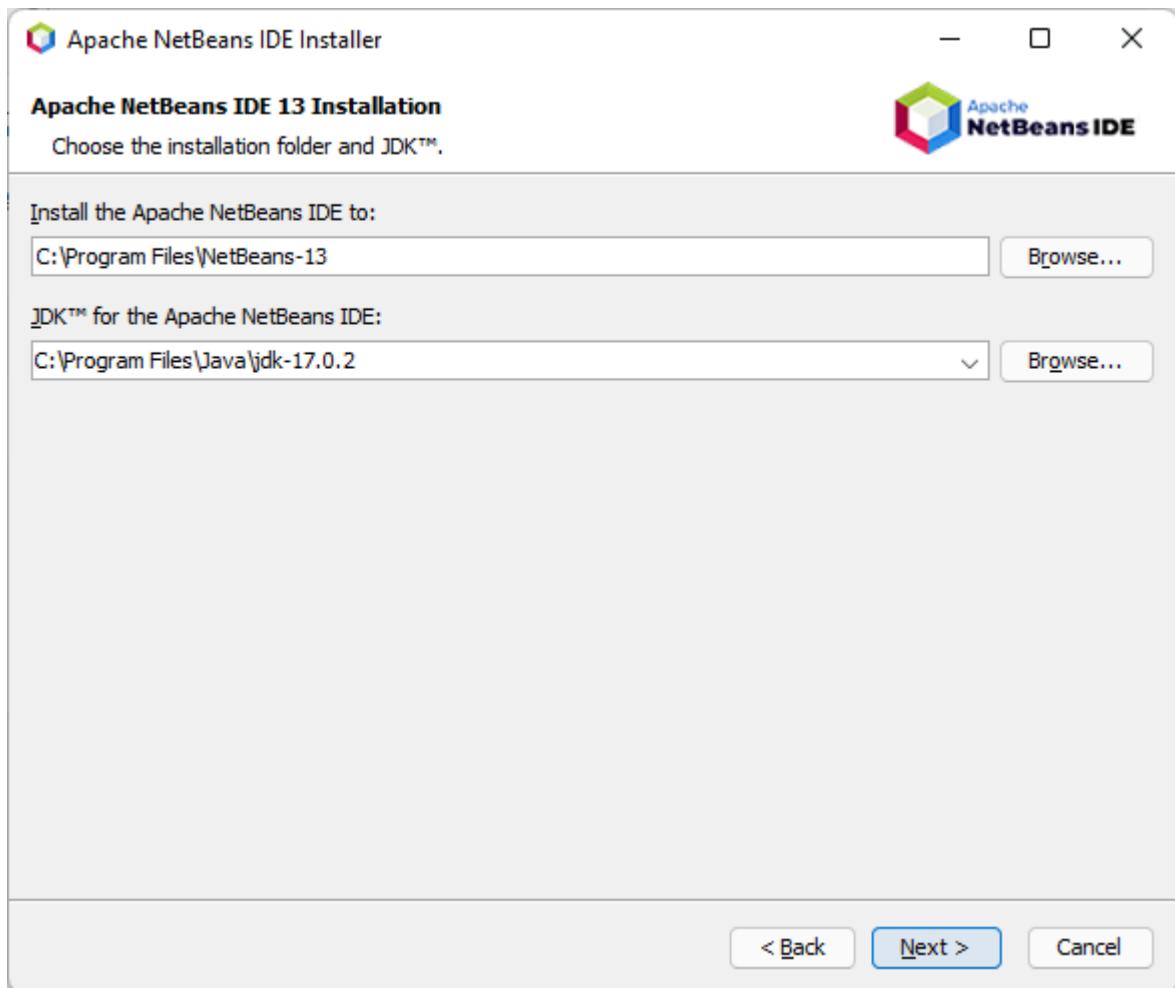


Figura 17 – Tela de local de instalação do instalador do Apache NetBeans IDE

Fonte: Apache NetBeans IDE (2022)

Na tela seguinte, o instalador perguntará se você quer receber as atualizações do Apache NetBeans IDE. Para que seu ambiente esteja sempre atualizado, mantenha a opção **Check for Updates** marcada. Por fim, clique no botão **Install** para iniciar a instalação do IDE.

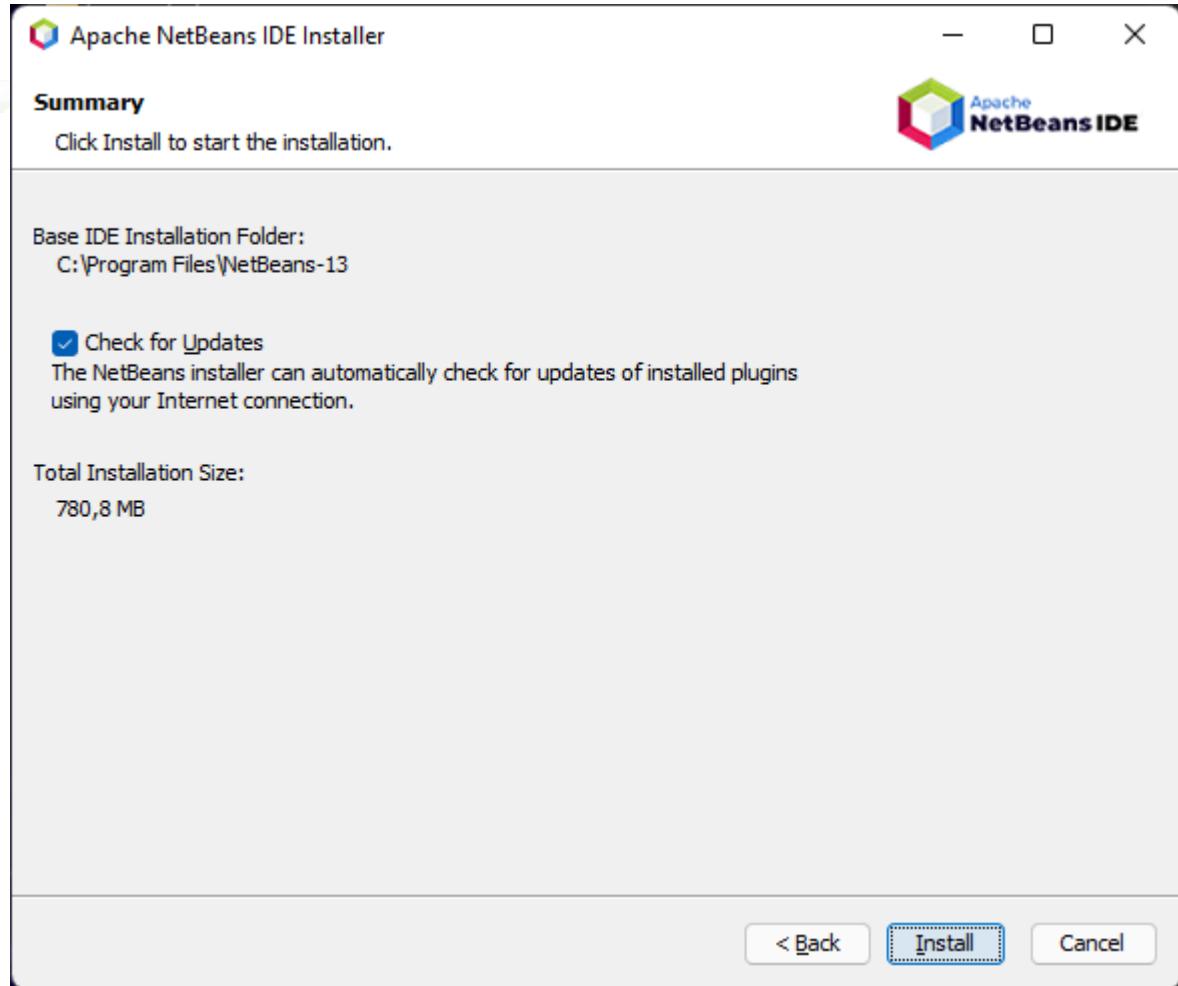


Figura 18 – Tela de atualização do instalador do Apache NetBeans IDE

Fonte: Apache NetBeans IDE (2022)

Assim que a instalação for finalizada, uma mensagem de “instalação concluída com sucesso” será apresentada e o Apache NetBeans IDE estará pronto para ser utilizado. Finalize a instalação clicando no botão **Finish**.

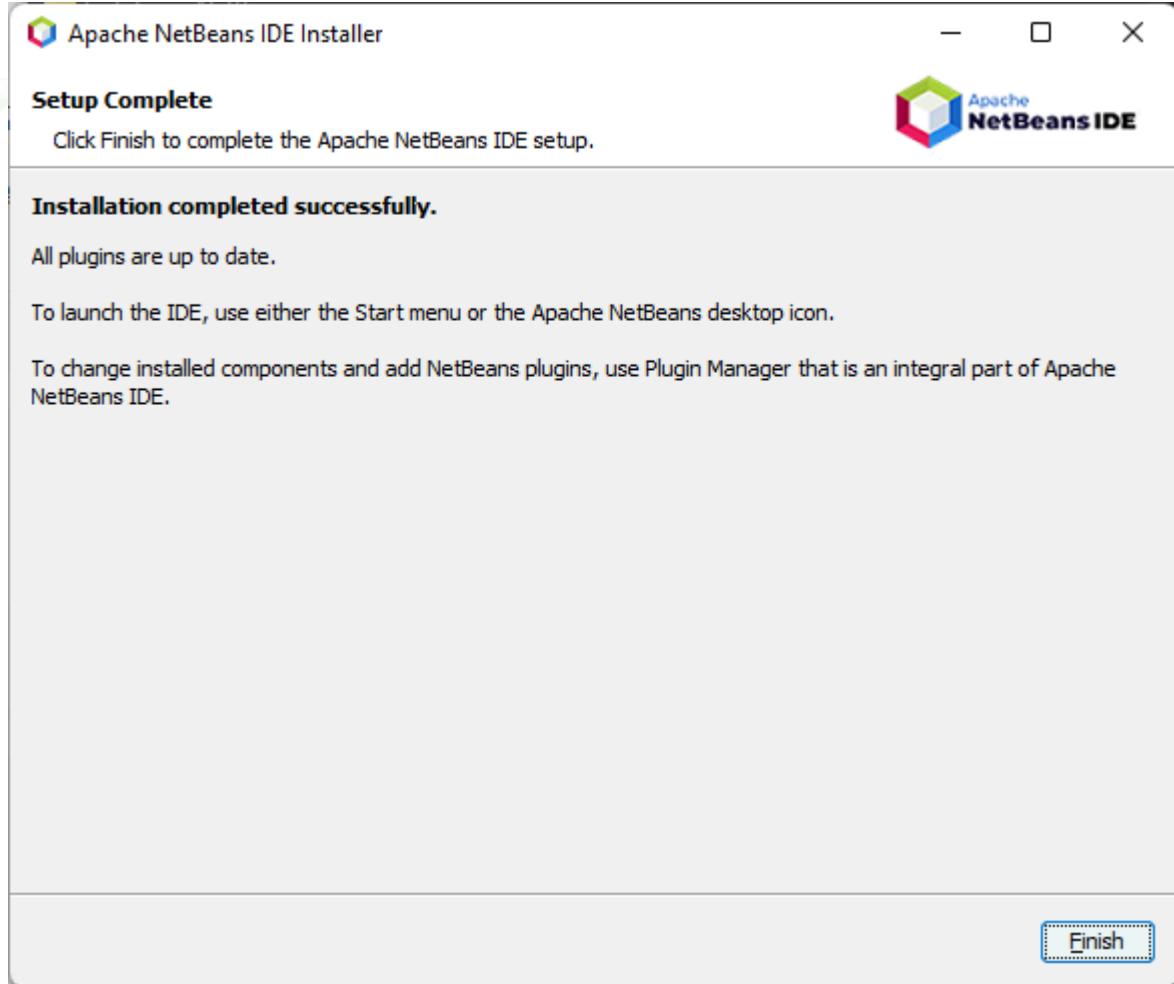


Figura 19 – Tela de conclusão do instalador do Apache NetBeans IDE

Fonte: Apache NetBeans IDE (2022)

Apache Netbeans IDE 13 – Instalação e configuração (Linux)

Para instalar o Apache Netbeans 13 no Ubuntu, existem três formas:

- ◆ Usar o gerenciador de pacotes (comando **apt-get**).
- ◆ Baixar o instalador do site oficial (arquivo **.deb**).
- ◆ Fazer a instalação a partir da loja Ubuntu Software Center.

Dessas, a terceira opção é a mais prática e garantirá a versão mais recente do software. Então, acesse o aplicativo Ubuntu Software Center.

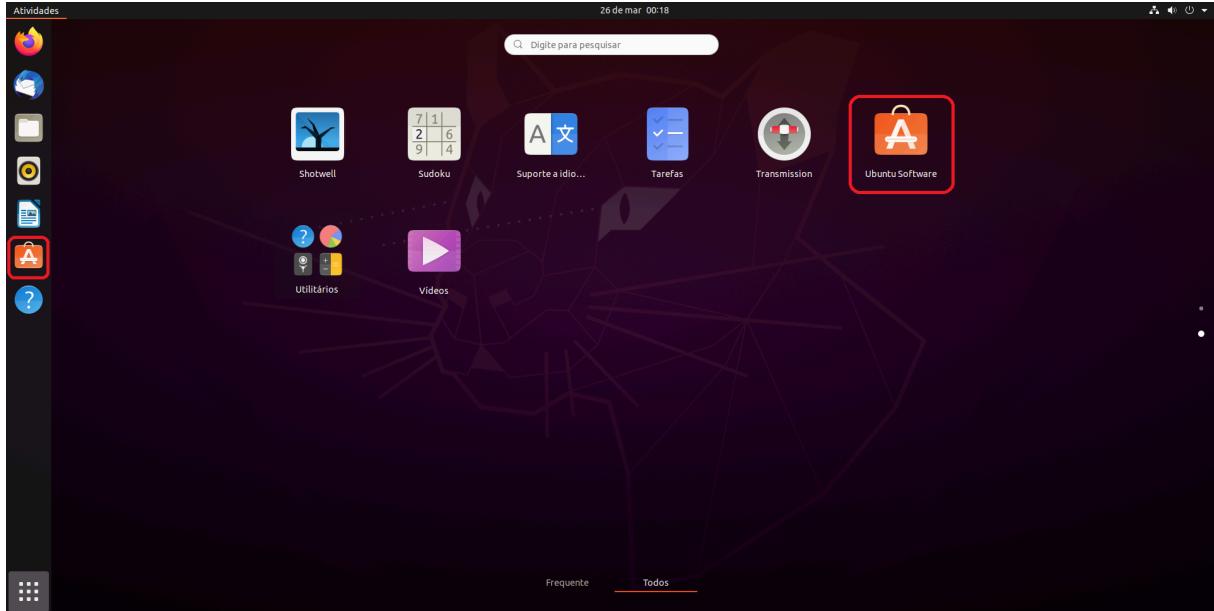


Figura 20 – Exemplo de menu de aplicativos do Ubuntu

Fonte: Senac EAD (2022)

Assim que a loja abrir, clique no ícone de lupa localizado no canto esquerdo da tela e preencha a barra de pesquisa que aparecerá com NetBeans. Você perceberá que há dois resultados: o **NetBeans** e o **Apache NetBeans**.

O aplicativo NetBeans trata-se da versão 10 do IDE, quando a Oracle era responsável pelas atualizações. Logo, essa é a versão desatualizada. A versão atualizada é a que está sendo mantida pela Apache e se chama **Apache NetBeans**. Selecione essa opção para continuar.

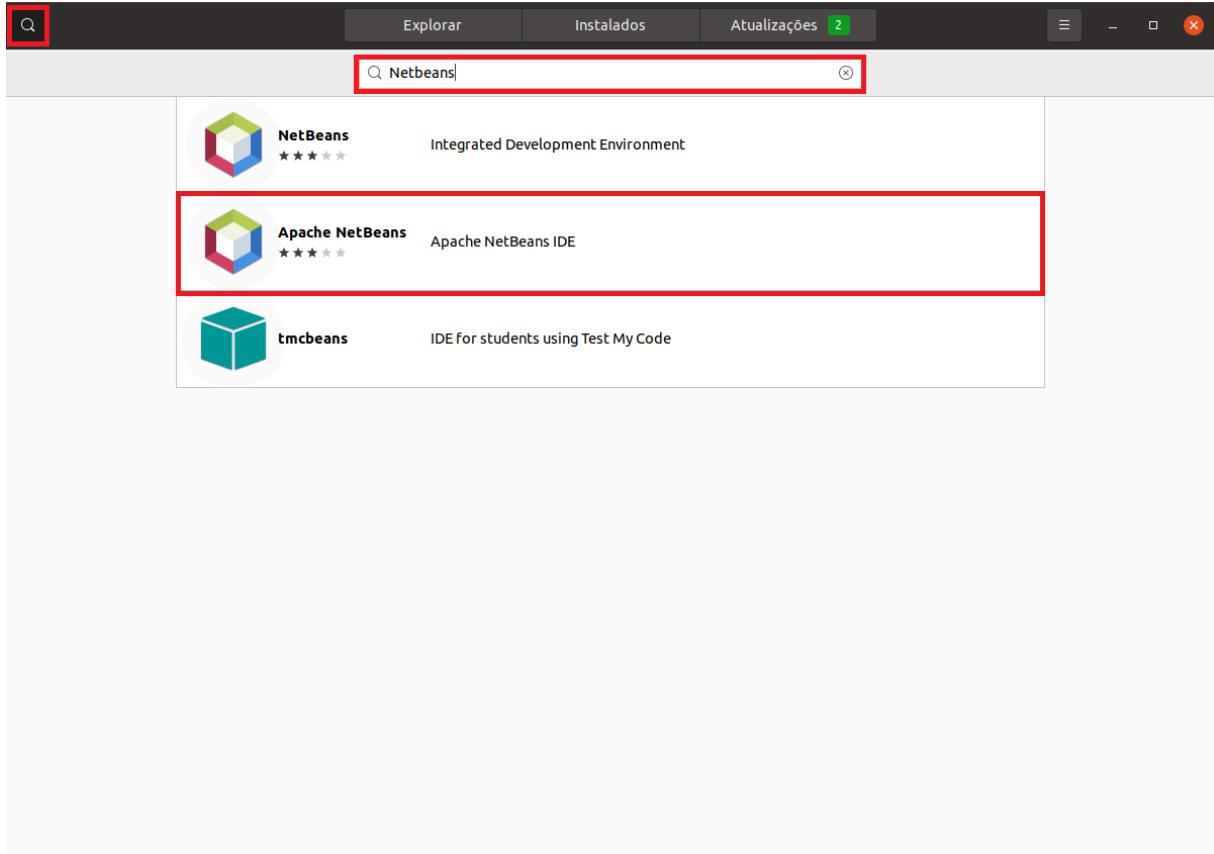


Figura 21 – Ubuntu Software Center

Fonte: Ubuntu Software Center (2022)

Na página do Apache NetBeans, clique no botão **Instalar**.

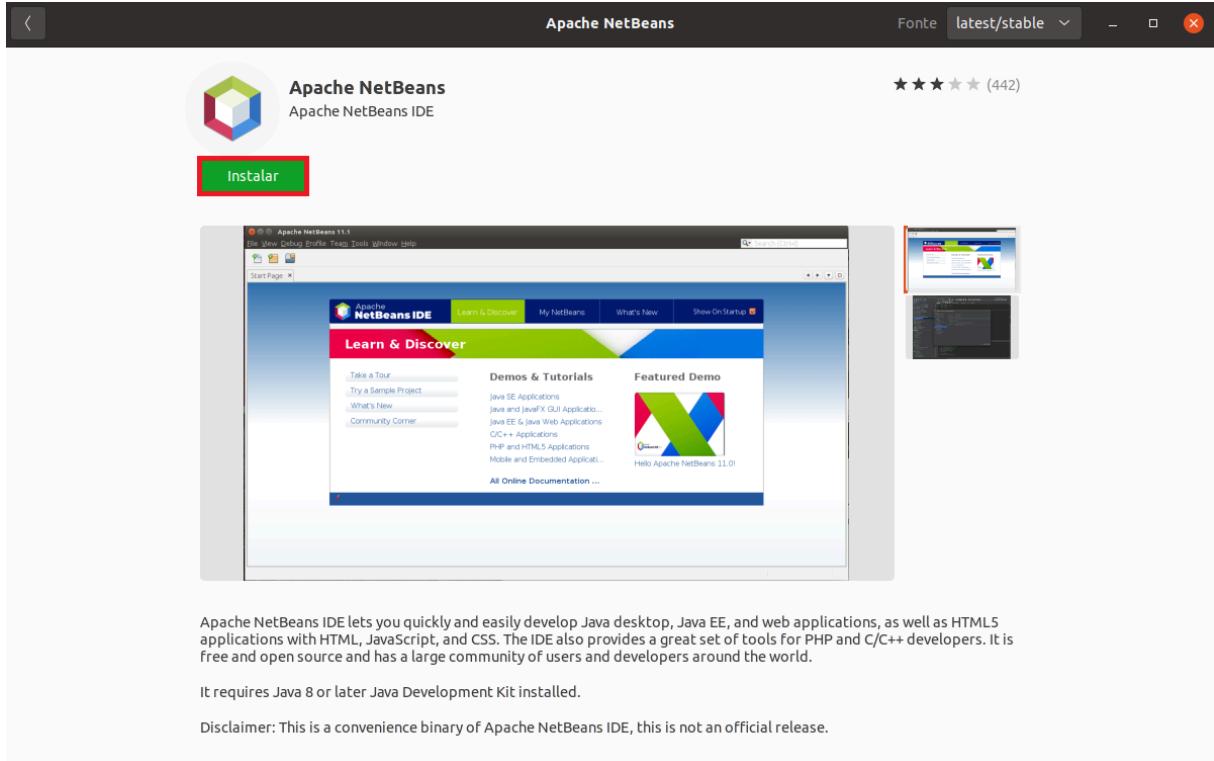


Figura 22 – Ubuntu Software Center

Fonte: Ubuntu Software Center (2022)

Assim que a instalação for finalizada, você encontrará o Apache NetBeans no seu menu de aplicativos pronto para ser utilizado.

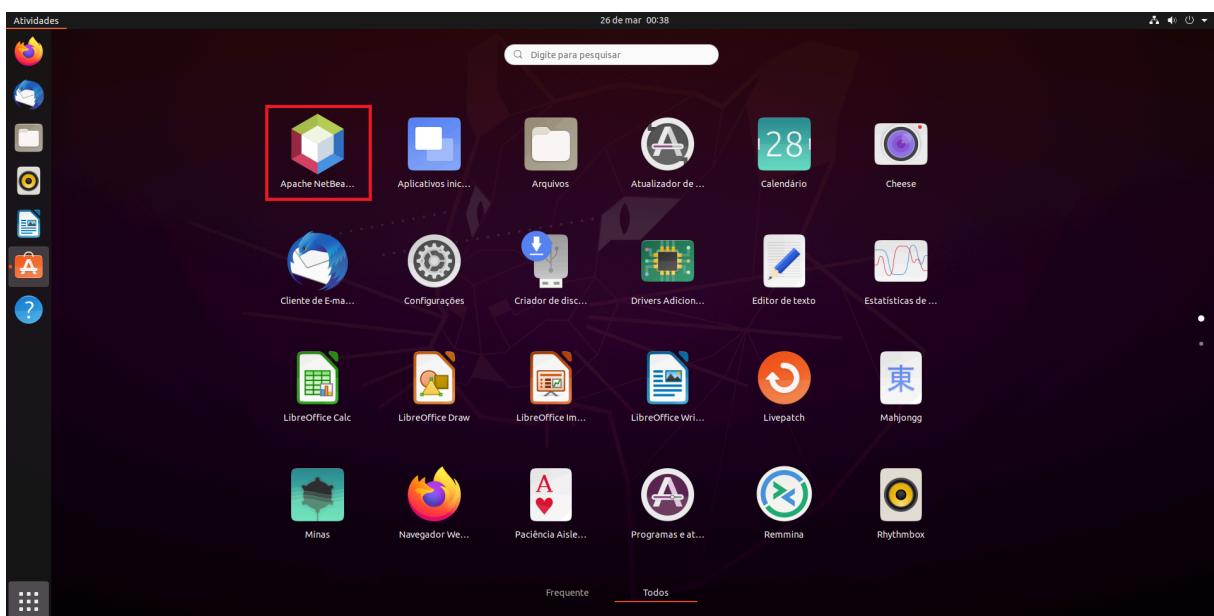


Figura 23 – Exemplo de menu de aplicativos do Ubuntu Software Center

Fonte: Senac EAD (2022)

Criação de projetos

Agora que você já está com seu ambiente de desenvolvimento pronto, chegou a hora de criar seu primeiro projeto Java. Para isso, comece criando um projeto bem simples chamado “Hello, World!”.

Por enquanto, não serão abordadas questões técnicas da linguagem, como sintaxes e chamada de funções. Esse conteúdo será estudado em outros tópicos.

Programa “Hello, World!”

O “Hello, World!” (ou “Olá, mundo!”, em português) é um programa simples, com poucas linhas de código, que retorna a mensagem “Hello, World!” na tela. Desenvolver um programa “Hello, World!” é uma prática comum entre as pessoas que estão aprendendo programação ou estão conhecendo uma nova linguagem de programação.

Ele também serve para verificar se um ambiente de programação está devidamente configurado, já que algumas linguagens de programação podem requerer algumas instalações e configurações adicionais. Assim, o “Hello, World!” também é uma forma de verificar se o ambiente está configurado adequadamente. Portanto, se o programa funcionar, significa que está tudo correto com o seu ambiente de desenvolvimento.

Com o Apache NetBeans IDE aberto, clique no primeiro item do menu superior **File** para revelar novas opções no menu e selecione a primeira opção, **New Project...**, para começar a criação de um novo projeto. Você também pode usar o atalho **CTRL + SHIFT + N** para acessar essa opção.

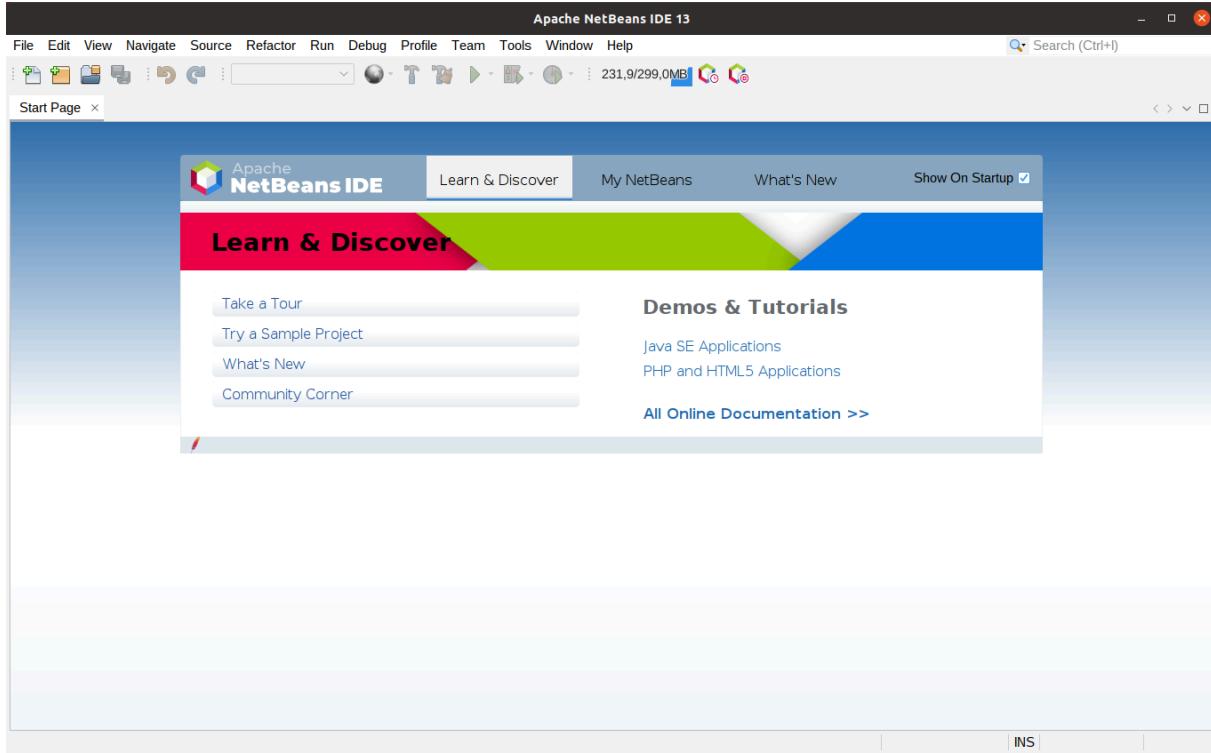


Figura 24 – Apache NetBeans IDE

Fonte: Apache NetBeans IDE (2022)

Uma nova janela será aberta e você será questionado sobre que tipo de projeto deseja criar. O Apache NetBeans IDE oferece três opções para projetos em Java:

- ◆ Java with Maven
- ◆ Java with Gradle
- ◆ Java with Ant

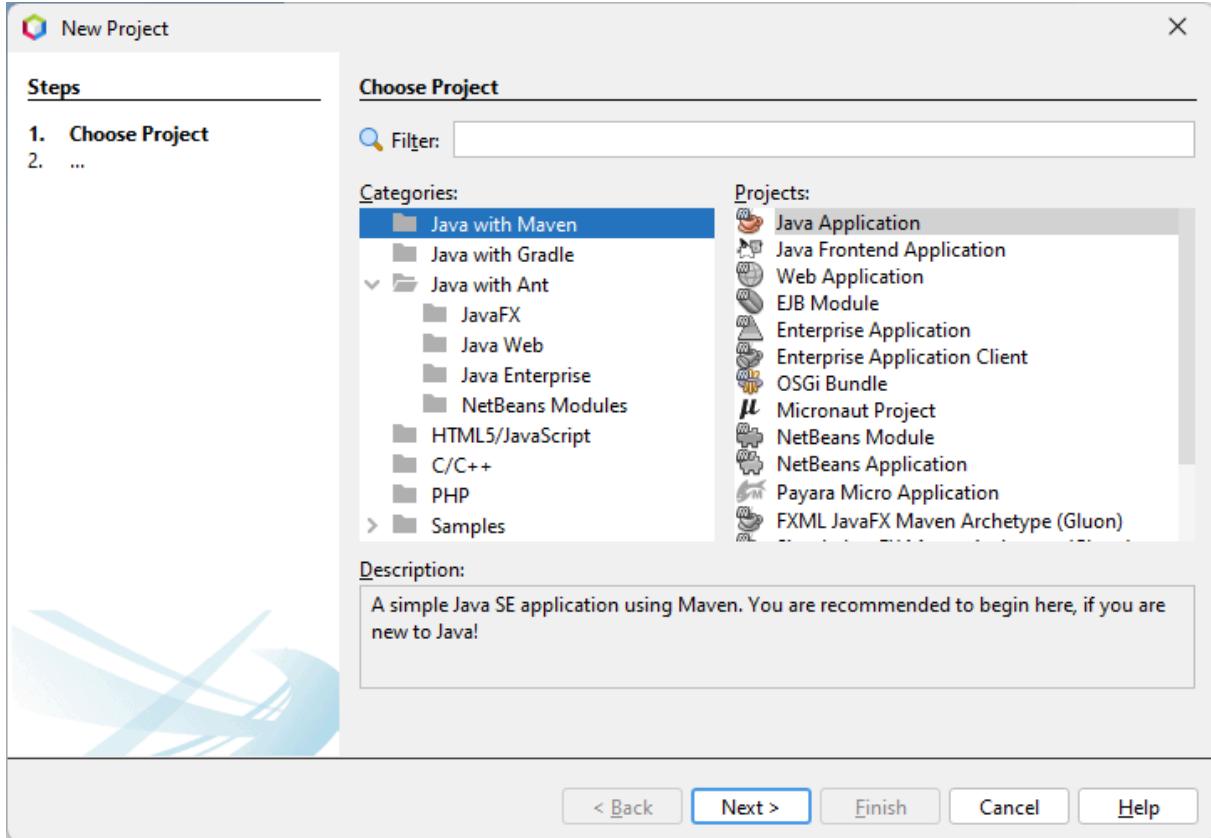


Figura 25 – Apache NetBeans IDE

Fonte: Apache NetBeans IDE (2022)

Assim que você finalizar a codificação de um *software*, a etapa seguinte será compilar e rodar o projeto. Quando é necessário realizar a compilação (*build*) de um projeto, normalmente esse processo é realizado manualmente. Conforme o projeto se torna maior e mais complexo, principalmente quando são inseridas novas bibliotecas nele, esse processo tende a se tornar maior e mais complexo também. Para economizar tempo e trabalho, é comum desenvolvedores recorrerem a sistemas de compilação automático e, ao criar um projeto Java no NetBeans, o IDE oferece três opções, usando um sistema de compilação automático integrado. As opções oferecidas são o Maven, o Gradle e o Ant, respectivamente. Para compreender como cada uma dessas bibliotecas funciona na prática, crie três projetos de “Hello, World!” usando cada uma delas.

Lembre-se de que a compilação é o processo de transformar o código escrito pelo programador em um programa executável. Ao contrário de outras linguagens, como C++, que geram arquivos de programa (extensão .exe), no caso do Java, o resultado serão arquivos chamados bytecodes (extensão .class), que, por sua vez, são interpretados pela JVM e executados.

Para se manter uma abordagem didática neste momento, a ordem sugerida pelo NetBeans será invertida. Ao invés de criar respectivamente um projeto com o Maven, seguido do Gradle e por fim com o Ant, você começará pelo Ant, em seguida partirá para o Gradle e só depois utilizará o Maven.

Vale destacar ainda que é possível a compilação simples e direta de código Java a partir de terminal, usando o comando javac. Em projetos maiores, o mais interessante é recorrer a ferramentas de automatização, como as utilizadas pelo NetBeans.

Ant

O Apache Ant é uma biblioteca Java usada para automatizar processos de compilação e construção de aplicativos Java, lançado em 2000. Antes do seu surgimento, a única ferramenta de automação de *build* disponível era o Make, e este foi muito usado para construir aplicativos nos primeiros anos do Java. Em muitos aspectos, o Ant é muito semelhante ao seu antecessor e é a melhor opção para criação de projetos simples e para programadores que estão iniciando na área de programação.

Para criar um projeto com o Ant, selecione a opção **Java with Ant** em **Categories** e **Java Application** em **Projects**. Após isso, clique em **Next** para continuar.

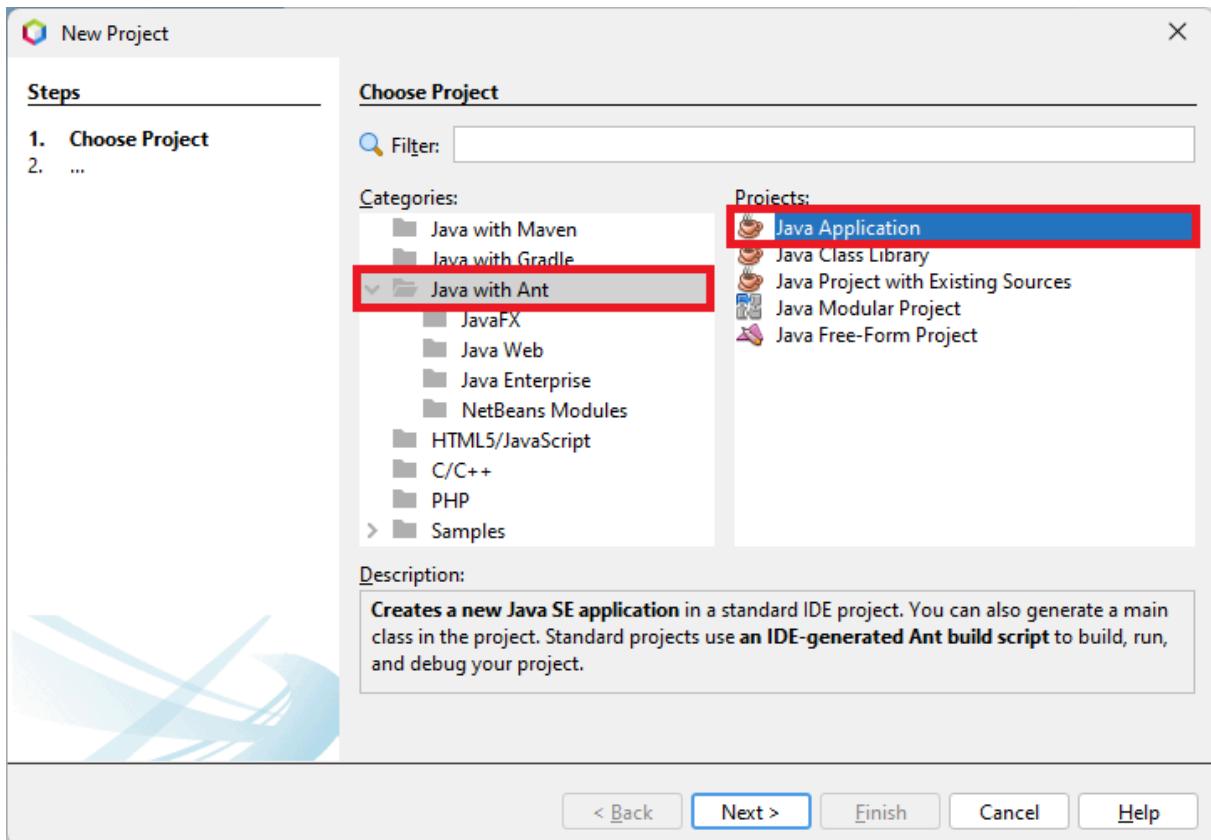


Figura 26 – Apache NetBeans IDE

Fonte: Apache NetBeans IDE (2022)

Na tela seguinte, você precisará informar qual será o nome do seu projeto. Dessa forma, será criada uma pasta com o mesmo nome no local definido em **Project Location**. Por conta disso, é importante que o nome do projeto não contenha espaços nem caracteres especiais. Nesse exemplo, você criará um projeto “HelloWorld”. Após isso, tenha certeza de manter a opção **Create Main Class** selecionada para o IDE já criar o arquivo que será a classe principal e em seguida clique em **Finish**.

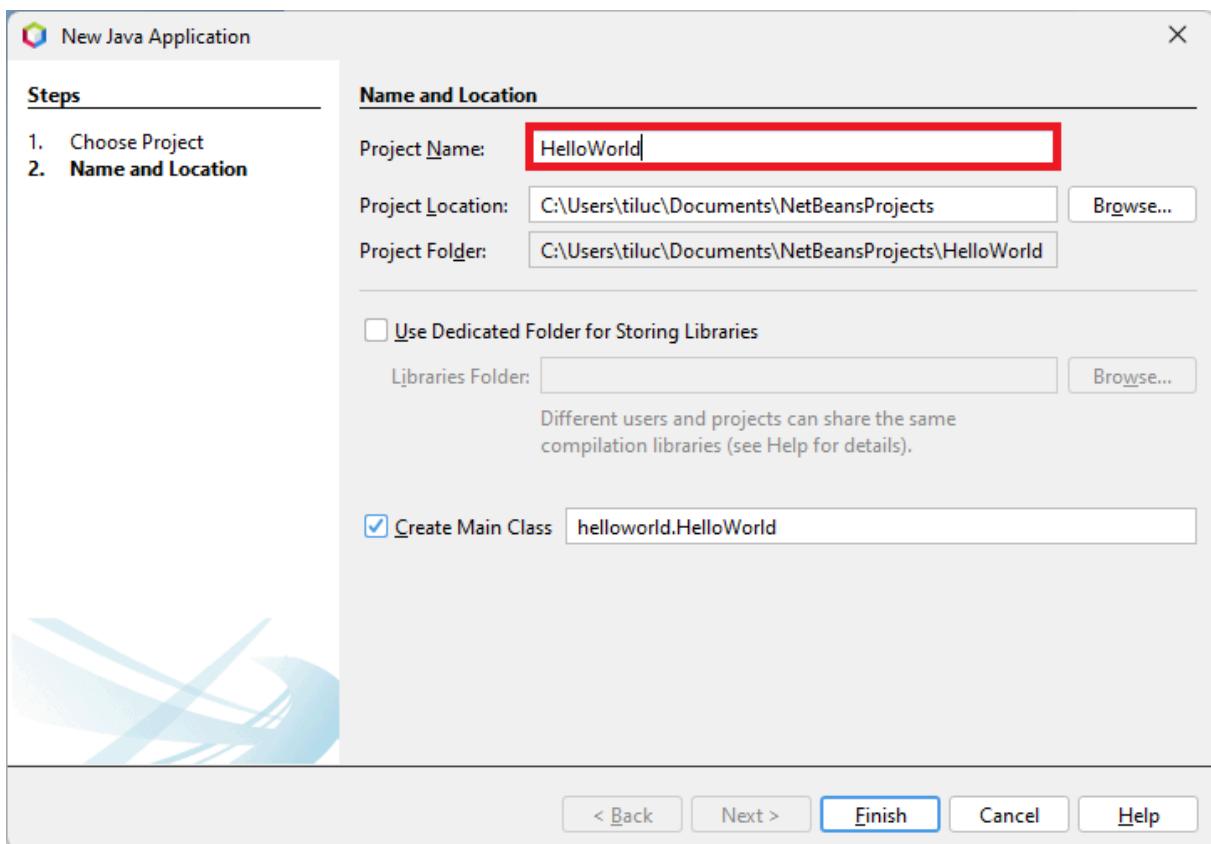


Figura 27 – Apache NetBeans IDE

Fonte: Apache NetBeans IDE (2022)

Assim que o projeto for criado, você verá a seguinte tela:

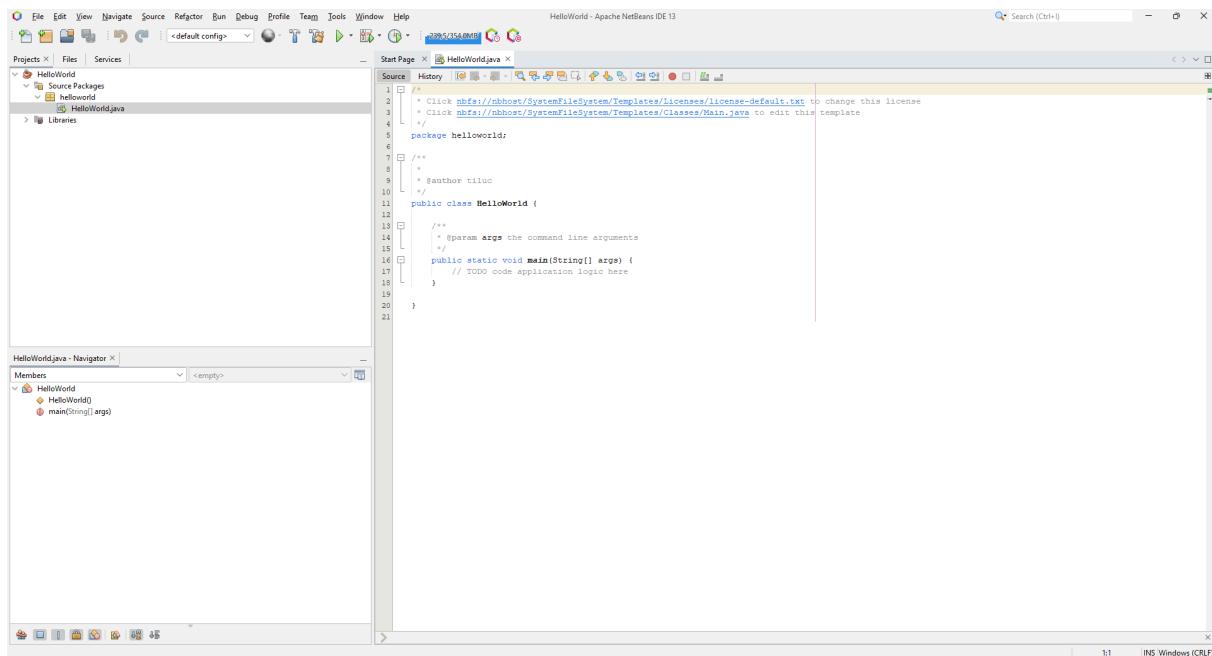


Figura 28 – Apache NetBeans IDE

Fonte: Apache NetBeans IDE (2022)

Antes de escrever seu código, conheça melhor o Apache NetBeans IDE.

No lado esquerdo da tela está a lista de projetos. Todos os seus projetos criados com o NetBeans aparecerão aqui. Dentro de cada projeto, há **pacotes** que funcionam como subpastas para organizar os arquivos do projeto e, dentro dos pacotes, estarão os seus arquivos, os quais serão, na maioria, os códigos Java que você criará e escreverá. Ainda dentro do projeto, haverá as dependências que são, em sua essência, as **bibliotecas** necessárias para que algum recurso específico do projeto funcione. Se você expandir o item **Libraries** (“bibliotecas”, em português), encontrará todas as dependências Java sendo usadas pelo projeto. Repare que mesmo um projeto base, como o que está sendo criado agora, necessita de várias bibliotecas para funcionar. Conforme seu projeto for ficando maior e mais complexo, a tendência é o número de bibliotecas aumentar.

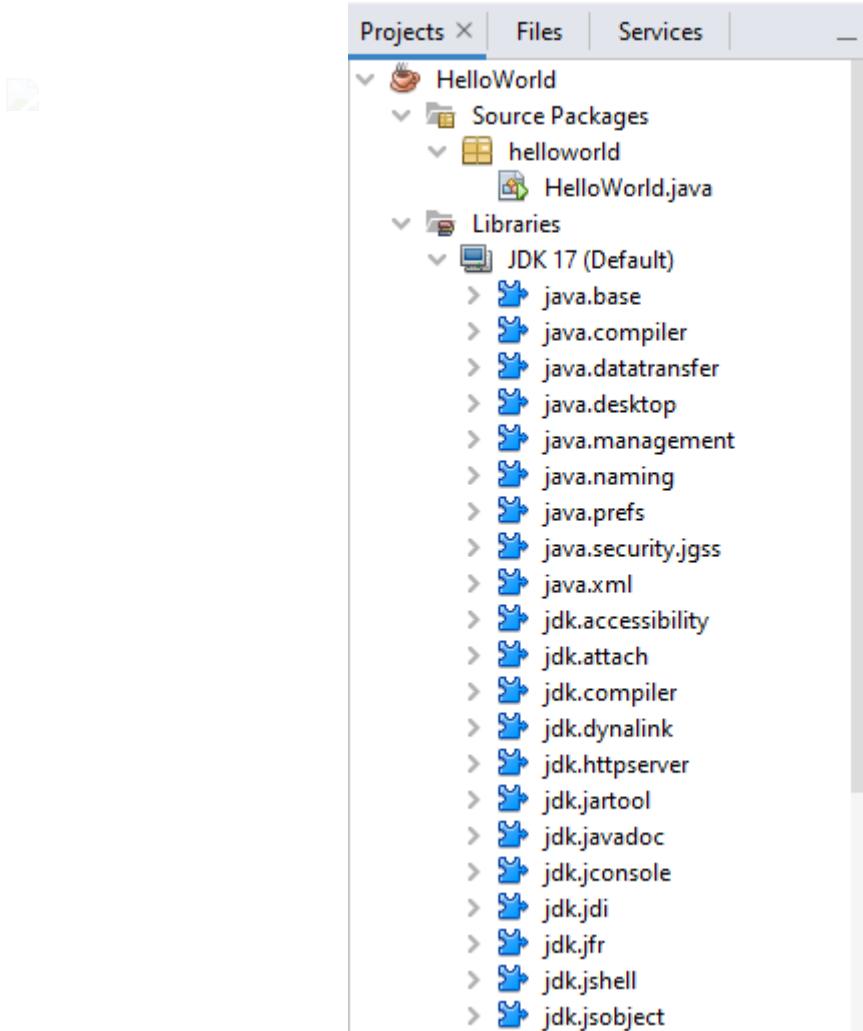


Figura 29 – Apache NetBeans IDE

Fonte: Apache NetBeans IDE (2022)

Logo abaixo, há uma área que mostrará um resumo das variáveis e funções declaradas nos arquivos do seu projeto. Isso pode ser muito útil para localizar áreas específicas do seu código quando você tiver centenas de linhas em apenas um arquivo.

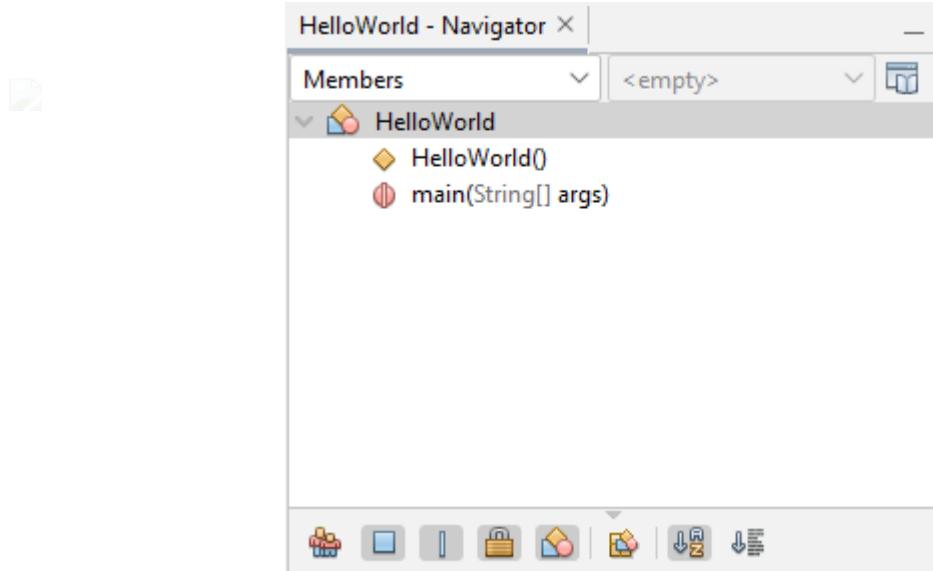


Figura 30 – Apache NetBeans IDE

Fonte: Apache NetBeans IDE (2022)

Por fim, é apresentada a área de edição de texto, que é onde você passará a maior parte do tempo escrevendo códigos. Quando se cria um arquivo Java no projeto, o IDE encarrega-se de criar toda a estrutura-base para se começar a escrever o código. Além disso, alguns comentários são adicionados, incluindo o nome do usuário do sistema operacional (linha 9), para representar uma assinatura de “quem criou esse código”. Esses comentários não são necessários para seu código funcionar, então você pode “limpá-los” para deixar seu código mais claro. Além disso, também adicione o seguinte trecho de código na sua função principal para exibir a mensagem “Hello, World!”:

```
System.out.println("Hello, World!");
```

O código completo, sem os comentários, ficará da seguinte maneira:

```
package helloworld;

public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Olá, mundo!");
    }
}
```

Perceba que, após editar o arquivo **HelloWorld.java**, o nome dele ficará em negrito. Isso significa que o arquivo teve alterações e elas ainda não foram salvas. Se você editar o código de um arquivo e não salvar as alterações, elas não serão aplicadas ao seu programa. Portanto, lembre-se de salvar seu projeto sempre para que erros não ocorram.

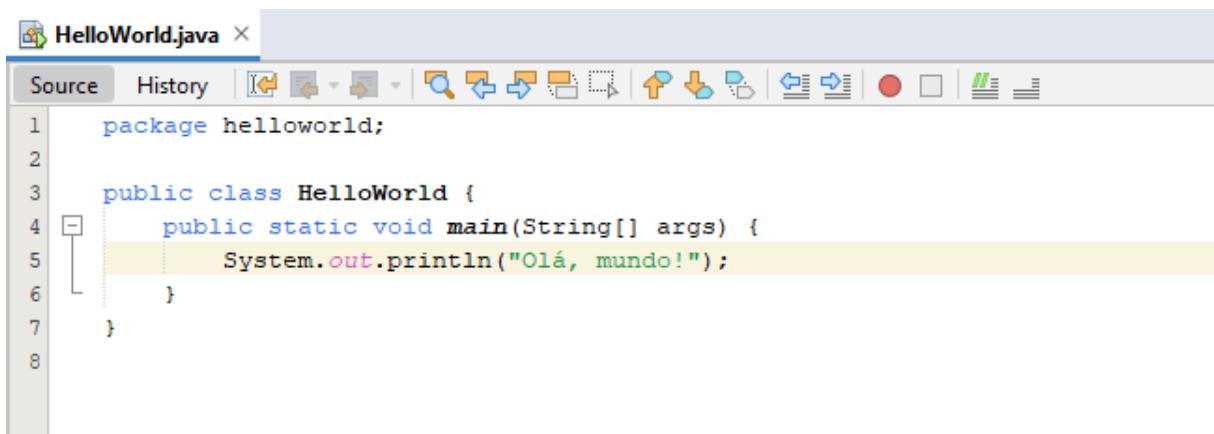


Figura 31 – Apache NetBeans IDE

Fonte: Apache NetBeans IDE (2022)

Para salvar as alterações, acesse o menu **File** e selecione a opção **Save**. Porém, com o passar do tempo, esse processo pode se tornar muito cansativo para você. Como alternativa, é possível utilizar o atalho **CTRL + S** para salvar as alterações no arquivo mais rapidamente. Assim que as alterações forem salvas, o nome do arquivo não estará mais em negrito.

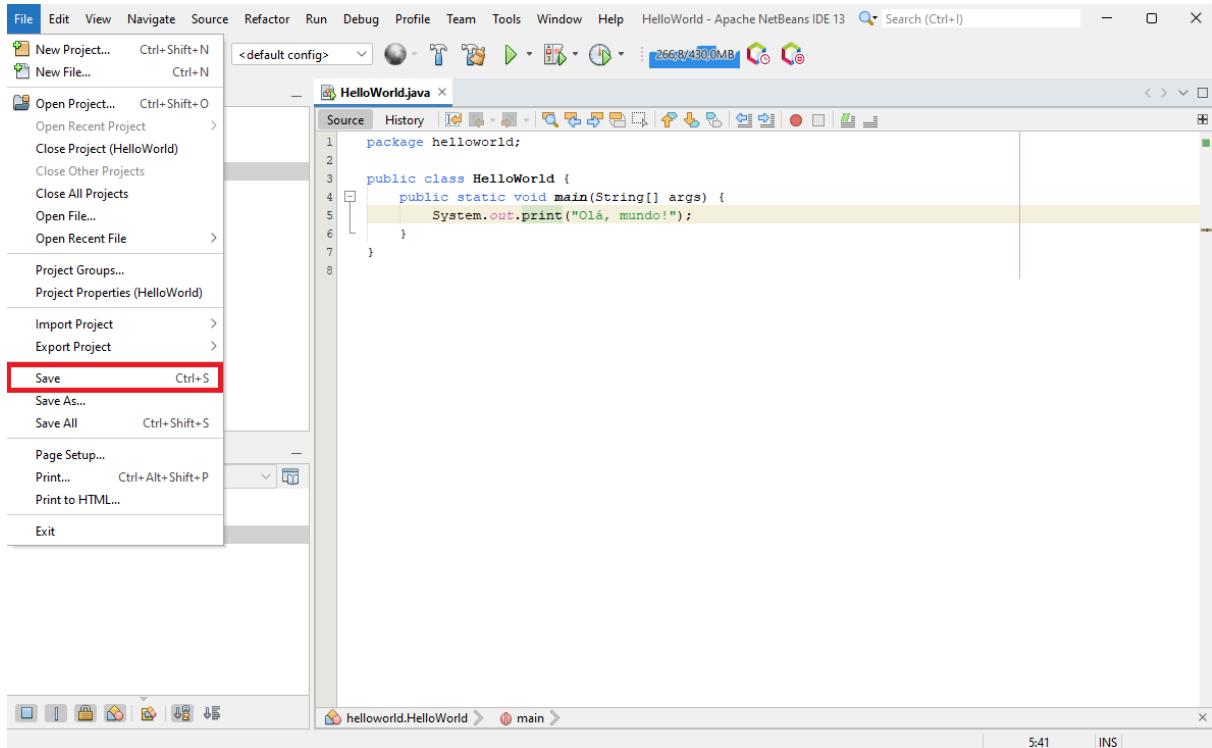


Figura 32 – Apache NetBeans IDE

Fonte: Apache NetBeans IDE (2022)

Agora que você já escreveu o seu código e salvou as alterações, já pode rodar o projeto! Para isso, clique no botão de **Run Project**.

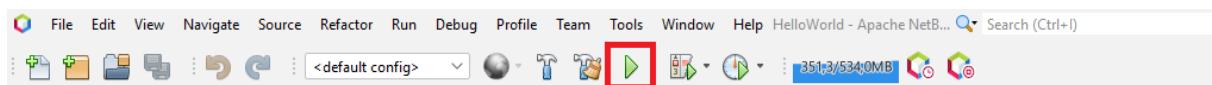
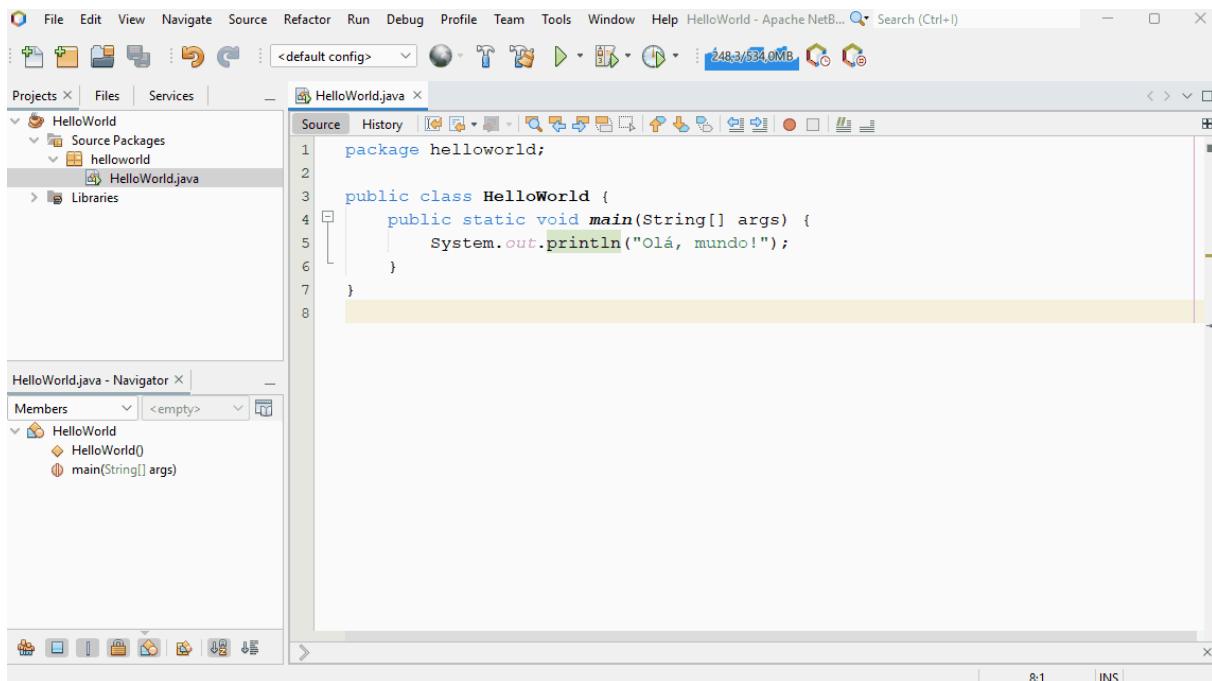


Figura 33 – Apache NetBeans IDE

Fonte: Apache NetBeans IDE (2022)

Quando você tentar rodar o projeto, o Ant entrará em ação para primeiro compilar o projeto e só depois executá-lo compilado.

No final, você terá o resultado mostrado no *gif*.



Maven

O Apache Maven é uma poderosa ferramenta de gerenciamento de projetos de software usada no ambiente de desenvolvimento Java para gerenciar e construir projetos, bem como para gerenciar dependências em projetos Java. Essas dependências são nada mais, nada menos do que as bibliotecas externas que a aplicação utilizará. Você pode definir as dependências e outras configurações do projeto por meio de um arquivo XML chamado **pom.xml**.

Agora que você já conhece um pouco do **Maven**, que tal praticar?

Crie um novo projeto no Apache NetBeans IDE com o Maven e utilize o mesmo código de “Hello, World!” usado na criação do projeto com o Ant. Por fim, execute o projeto e compare o tempo necessário para o projeto executar com o Ant e o Maven.

Gradle

Gradle é uma ferramenta de gerenciamento de dependência e automação de *build*, que foi construída sobre os conceitos do Ant e do Maven.

Uma das primeiras coisas que se pode notar sobre o Gradle é que ele não faz uso de arquivos XML, ao contrário do Maven. Isso levou a arquivos de configuração mais compactados e com menos confusão para os desenvolvedores. O arquivo de configuração do Gradle é por convenção chamado de **build.gradle**.

Crie um novo projeto no Apache NetBeans IDE com o **Gradle** e utilize o mesmo código de “Hello, World!” usado na criação do projeto com o Ant. Por fim, execute o projeto e compare o tempo necessário para o projeto executar com o Ant e o Maven.

Comparação entre Ant, Maven e Gradle

Se você realizou os desafios propostos sobre a criação do projeto “Hello, World!” com o Maven e o Gradle, você deve ter percebido que a construção do código-fonte permaneceu exatamente a mesma, em comparação com o Ant. Porém, existem algumas características diferentes, tais como:

- ◆ As informações solicitadas pelo IDE para criar o projeto contêm alguns campos diferentes.
- ◆ O tempo necessário para criar e executar o projeto é maior devido à necessidade de baixar pacotes e bibliotecas essenciais no projeto-base.
- ◆ A estrutura de arquivos apresentada pelo IDE é diferente por conta de novos arquivos existentes no projeto (pacotes e bibliotecas externas do JDK).
- ◆ O código-fonte do projeto inicial também é diferente, pois sugere uma outra abordagem para o desenvolvedor começar com a ferramenta designada.

Apesar de o IDE recomendar o uso do Maven na criação de projetos, essa ferramenta pode tornar o desenvolvimento de *software* mais complexo para aqueles que estão começando nessa área (o mesmo vale para o Gradle). Como, inicialmente, não será necessário fazer uso de pacotes e bibliotecas externas do JDK para desenvolver as suas aplicações, a melhor opção é utilizar o Ant, já que ele não tem um gerenciador de pacotes que tornará o projeto mais complexo.

Encerramento

Assim se concluem a introdução à linguagem de programação Java, a configuração do ambiente de desenvolvimento e a criação dos seus primeiros projetos usando o Apache NetBeans IDE. Esses são os primeiros passos para começar a aprender qualquer linguagem de programação e são a base para iniciar o desenvolvimento de sistemas com qualquer tecnologia. Agora que você já tem todas as ferramentas instaladas e em perfeito funcionamento, está pronto para aprofundar seus estudos na linguagem de programação Java.