Módulo 1

Introdução à Programação I

Lição 3

Primeiros passos no ambiente de programação

Versão 1.0 - Jan/2007

Autor
Florence Tiu Balagtas

Equipe
Joyce Avestro
Florence Balagtas
Rommel Feria
Reginald Hutcherson
Rebecca Ong
John Paul Petines
Sang Shin
Raghavan Srinivas
Matthew Thompson

Necessidades para os Exercícios

Sistemas Operacionais Suportados

NetBeans IDE 5.5 para os seguintes sistemas operacionais:

- Microsoft Windows XP Profissional SP2 ou superior
- Mac OS X 10.4.5 ou superior
- Red Hat Fedora Core 3
- Solaris™ 10 Operating System (SPARC® e x86/x64 Platform Edition)

NetBeans Enterprise Pack, poderá ser executado nas seguintes plataformas:

- Microsoft Windows 2000 Profissional SP4
- Solaris™ 8 OS (SPARC e x86/x64 Platform Edition) e Solaris 9 OS (SPARC e x86/x64 Platform Edition)
- Várias outras distribuições Linux

Configuração Mínima de Hardware

Nota: IDE NetBeans com resolução de tela em 1024x768 pixel

Sistema Operacional	Processador	Memória	HD Livre
Microsoft Windows	500 MHz Intel Pentium III workstation ou equivalente	512 MB	850 MB
Linux	500 MHz Intel Pentium III workstation ou equivalente	512 MB	450 MB
Solaris OS (SPARC)	UltraSPARC II 450 MHz	512 MB	450 MB
Solaris OS (x86/x64 Platform Edition)	AMD Opteron 100 Série 1.8 GHz	512 MB	450 MB
Mac OS X	PowerPC G4	512 MB	450 MB

Configuração Recomendada de Hardware

Sistema Operacional	Processador	Memória	HD Livre
Microsoft Windows	1.4 GHz Intel Pentium III workstation ou equivalente	1 GB	1 GB
Linux	1.4 GHz Intel Pentium III workstation ou equivalente	1 GB	850 MB
Solaris OS (SPARC)	UltraSPARC IIIi 1 GHz	1 GB	850 MB
Solaris OS (x86/x64 Platform Edition)	AMD Opteron 100 Series 1.8 GHz	1 GB	850 MB
Mac OS X	PowerPC G5	1 GB	850 MB

Requerimentos de Software

NetBeans Enterprise Pack 5.5 executando sobre Java 2 Platform Standard Edition Development Kit 5.0 ou superior (JDK 5.0, versão 1.5.0_01 ou superior), contemplando a Java Runtime Environment, ferramentas de desenvolvimento para compilar, depurar, e executar aplicações escritas em linguagem Java. Sun Java System Application Server Platform Edition 9.

- Para **Solaris**, **Windows**, e **Linux**, os arquivos da JDK podem ser obtidos para sua plataforma em http://java.sun.com/j2se/1.5.0/download.html
- Para **Mac OS X**, Java 2 Plataform Standard Edition (J2SE) 5.0 Release 4, pode ser obtida diretamente da Apple's Developer Connection, no endereço: http://developer.apple.com/java (é necessário registrar o download da JDK).

Para mais informações: http://www.netbeans.org/community/releases/55/relnotes.html

Colaboradores que auxiliaram no processo de tradução e revisão

Alexandre Mori
Alexis da Rocha Silva
Aline Sabbatini da Silva Alves
Allan Wojcik da Silva
André Luiz Moreira
Andro Márcio Correa Louredo
Antoniele de Assis Lima
Antonio Jose R. Alves Ramos
Aurélio Soares Neto
Bruno da Silva Bonfim
Bruno dos Santos Miranda
Bruno Ferreira Rodrigues
Carlos Alberto Vitorino de Almeida
Carlos Alexandre de Sene
Carlos André Noronha de Sousa
Carlos Eduardo Veras Neves
Cleber Ferreira de Sousa
Cleyton Artur Soares Urani
Cristiano Borges Ferreira
Cristiano de Siqueira Pires
Derlon Vandri Aliendres
Fabiano Eduardo de Oliveira
Fábio Bombonato
Fernando Antonio Mota Trinta
Flávio Alves Gomes
Francisco das Chagas
Francisco Marcio da Silva
Gilson Moreno Costa
Givailson de Souza Neves
Gustavo Henrique Castellano
Hebert Julio Gonçalves de Paula
Heraldo Conceição Domingues
Hugo Leonardo Malheiros Ferreira
Ivan Nascimento Fonseca
Jacqueline Susann Barbosa
Jader de Carvalho Belarmino
João Aurélio Telles da Rocha
João Paulo Cirino Silva de Novais
João Vianney Barrozo Costa
José Augusto Martins Nieviadonski
José Leonardo Borges de Melo
José Ricardo Carneiro
Kleberth Bezerra G. dos Santos
Lafaiete de Sá Guimarães
Leandro Silva de Morais
Leonardo Leopoldo do Nascimento
Leonardo Pereira dos Santos
Leonardo Rangel de Melo Filardi
Lucas Mauricio Castro e Martins
Luciana Rocha de Oliveira
Luís Carlos André
Luís Octávio Jorge V. Lima
Luiz Fernandes de Oliveira Junior
Luiz Victor de Andrade Lima
Manoel Cotts de Queiroz
Marcello Sandi Pinheiro
Marcelo Ortolan Pazzetto
Marco Aurélio Martins Bessa
Marcos Vinicius de Toledo
Maria Carolina Ferreira da Silva
Massimiliano Giroldi
Mauricio Azevedo Gamarra
Mauricio da Silva Marinho
Mauro Cardoso Mortoni
Mauro Regis de Sousa Lima
Namor de Sá e Silva
Néres Chaves Rebouças
Nolyanne Peixoto Brasil Vieira
Paulo Afonso Corrêa
Paulo José Lemos Costa
Paulo Oliveira Sampaio Reis
Pedro Antonio Pereira Miranda
Pedro Henrique Pereira de Andrade
Renato Alves Félix
Renato Barbosa da Silva
Reyderson Magela dos Reis
Ricardo Ferreira Rodrigues
Ricardo Ulrich Bomfim
Robson de Oliveira Cunha
Rodrigo Pereira Machado
Rodrigo Rosa Miranda Corrêa
Rodrigo Vaez
Ronie Dotzlaw
Rosely Moreira de Jesus
Seire Pareja
Sergio Pomerancblum
Silvio Sznifer
Suzana da Costa Oliveira
Tásio Vasconcelos da Silveira
Thiago Magela Rodrigues Dias
Tiago Gimenez Ribeiro
Vanderlei Carvalho Rodrigues Pinto
Vanessa dos Santos Almeida
Vastí Mendes da Silva Rocha
Wagner Eliezer Roncoletta

Auxiliadores especiais

Revisão Geral do texto para os seguintes Países:

- **Brasil** – Tiago Flach
- **Guiné Bissau** – Alfredo Cá, Bunene Sisse e Buon Olossato Quebi – ONG Asas de Socorro

Coordenação do DFJUG

- **Daniel deOliveira** – JUGLeader responsável pelos acordos de parcerias
- **Luci Campos -** Idealizadora do DFJUG responsável pelo apoio social
- **Fernando Anselmo -** Coordenador responsável pelo processo de tradução e revisão, disponibilização dos materiais e inserção de novos módulos
- **Regina Mariani -** Coordenadora responsável pela parte jurídica
- **Rodrigo Nunes -** Coordenador responsável pela parte multimídia
- **Sérgio Gomes Veloso -** Coordenador responsável pelo ambiente JEDI™ (Moodle)

Agradecimento Especial

John Paul Petines – Criador da Iniciativa JEDI™
Rommel Feria – Criador da Iniciativa JEDI™

1. Objetivos

Nesta lição discutiremos como escrever, compilar e rodar os programas em Java. Existem duas maneiras para se fazer isso: a primeira é por intermédio de uma console e um editor de texto e a segunda é utilizando a IDE NetBeans como ambiente integrado de desenvolvimento.

Ao final desta lição, o estudante será capaz de:

- Criar programas usando o editor de texto com uma console de desenvolvimento do Linux (sugerimos o **Ubuntu Dapper**) ou Windows
- Diferenciar entre erros de sintaxe e de tempo de execução (Run Time)
- Criar programas utilizando a IDE NetBeans

2. Introdução

Uma IDE é um ambiente de desenvolvimento integrado. É um software aplicativo que provê um construtor de interfaces GUI, um editor de códigos, um compilador e/ou interpretador e um depurador.

Nesta lição utilizaremos o **Ubuntu Dapper** como sistema operacional ou o **Windows**. Antes de realizar esta tenha certeza de que já tenha instalado no sistema operacional a Java JDK e o NetBeans. Instruções como instalar o Java JDK e o NetBeans podem ser vistas no **Apêndice A** e para os ambientes na versão Windows XP no **Apêndice B**.

Antes de entrar em detalhes, veremos o primeiro programa Java que poderemos escrever.

3. Primeiro Programa Java

Antes de explicar o que o programa significa, vamos escrevê-lo e executá-lo.

3.1 Utilizando a console e um editor de texto

Neste exemplo utilizaremos um simples editor de texto, que pode ser o **gedit** do Linux ou o **notepad** do Windows, para editar o código fonte. Em seguida será necessário abrir uma janela terminal para compilar e executar os programas.

Passo 1: executar um editor de texto

Para iniciar um editor de texto no **Linux** selecione Applications ⟹ Accessories ⟹ Text Editor.

Para iniciar um editor de texto no **Windows** selecione Start ⟹ Programs ⟹ Accessories ⟹ Notepad.

Passo 2: Abrir a janela de console

Para abrir o terminal no **Linux**, selecione Applications ⟹ Accessories ⟹ Terminal.

Para abrir o terminal no **Windows**, selecione Start ⟹ Run... e na janela que se apresenta, digite **cmd** e pressione o botão OK.

Passo 3: Escrever as instruções utilizando o Editor de Texto

Digite as seguintes instruções no editor de textos:

```
public class Hello
{
    /**
     * Meu primeiro programa Java
     */
    public static void main(String[] args) {
      // Mostra na tela o texto "Hello world"
         System.out.println("Hello world!");
    }
}
```

Passo 4: Salvar o programa Java

Chamaremos o programa de "Hello.java" e o colocaremos em uma pasta denominada "myJavaPrograms".

Caso esta pasta não tenha sido criada, retorne à janela de terminal aberta e insira as seguintes instruções:

Para o **Linux**:

```
$ md myJavaPrograms
```

Para o **Windows**:

```
C:\> md myJavaPrograms
```

Retorne ao Editor de textos e salve o programa. Para abrir a caixa de diálogo salvar selecione a

opção "File" localizada na barra de menus e depois clique na opção "Save".

Selecione a nova pasta criada como myJavaPrograms para entrar nela. A pasta deve estar vazia porque ainda não salvamos nada dentro dela.

Na caixa de texto "Name", digite o nome do programa (Hello.java), e depois clique no botão salvar.

ATENÇÃO: Para o **Notepad** no Windows, mude o Tipo para "All Files" (em Save as Type).

Após salvar o arquivo observe que o título da janela mudou de "Untitled" para "Hello.java", caso deseje alterar novamente o arquivo basta editá-lo e depois salvá-lo novamente clicando em File ⇒ Save.

Passo 5: Entrar na pasta que contém o programa

O próximo passo deve ser o de compilar o programa. Inicialmente, precisamos entrar na pasta que o contém. Retorne à janela do terminal.

Em **Linux**:

Normalmente, quando abrimos uma janela terminal, ela vai diretamente para sua pasta home (identificada por $). Para ver o que tem dentro do diretório digite **ls** (LS em minúscula, significando "List Sources") e pressione ENTER. Isso fará com que sejam listados os arquivos e pastas da pasta home.

Verifique a existência de uma pasta chamada "myJavaPrograms", criada a pouco, sendo esta o local em que foi salvo o programa "Hello.java". Mudaremos o contexto para esta pasta.

Para entrar nesta pasta devemos utilizar o comando: cd [nome da pasta]. O comando "cd" significa "Change Directory". Digitaremos:

```
$ cd myJavaPrograms
```

Agora que estamos dentro da pasta onde o arquivo do programa está, poderemos então compilá-lo. Certifique-se de que o arquivo está realmente dentro desta, executando o comando **ls** (LS em minúscula) novamente.

Em **Windows**:

Normalmente, quando abrimos uma janela terminal ela vai diretamente para sua pasta raiz (identificada por C:\). Para conhecer o conteúdo do diretório digite **dir** (significando "directory") e pressione ENTER. Isso fará com que sejam listados os arquivos e pastas da pasta principal.

Verifique a existência de uma pasta chamada "myJavaPrograms", criada a pouco, sendo esta o local em que foi salvo o programa "Hello.java". Mudaremos o contexto para esta pasta.

Para entrar nesta pasta devemos utilizar o comando: cd [nome da pasta]. O comando "cd" significa "Change Directory". Digitaremos:

```
C:\>cd myJavaPrograms
```

Agora que estamos dentro da pasta onde o arquivo do programa está, poderemos então compilá-lo. Certifique-se de que o arquivo está realmente dentro desta, executando o comando **dir** novamente.

Passo 6: Compilar o programa

Para compilar o programa, utilizamos o comando: **javac [Nome do Arquivo]**. Ou seja:

```
javac Hello.java
```

Durante a compilação, é criado o arquivo: **[Nome do Arquivo].class,** neste caso, **Hello.class**, que contém o código em linguagem de máquina (chamado de **bytecode**).

Passo 7: Executar o programa

Assumindo que não ocorreu problemas na compilação (caso tenha ocorrido qualquer problema refaça os passos realizados), estamos prontos para executar o programa.

Para executar o programa, utilizamos o comando: **java [nome do arquivo sem a extensão]**. No caso do exemplo, digite:

```
java Hello
```

Veremos na mesma tela, em que foi executado o comando, a seguinte mensagem:

```
Hello world!
```

3.2 Erros

Vimos um pequeno programa Java, geralmente não encontraremos qualquer problema para compilar e executar esses programas, entretanto nem sempre este é o caso, como mencionamos na primeira parte deste curso, ocasionalmente encontramos erros durante esse processo.

Como mencionamos antes, há dois tipos de erros: o primeiro pode ocorrer durante a compilação, chamado de erro de sintaxe, o segundo pode ocorrer durante a execução, chamado *runtime error*.

3.2.1 Erros de Sintaxe

Os erros de sintaxe normalmente são erros de digitação, ocasionados pelo programador que pode ter se equivocado e digitar uma instrução errada, ou por esquecimento de alguma parte da instrução, por exemplo, um ponto e vírgula. O Compilador tenta isolar o erro exibindo a linha de instrução e mostrando o primeiro caractere incorreto naquela linha, entretanto, um erro pode não estar exatamente neste ponto.

Outros erros comuns são a troca de letras, troca de letras maiúscula por minúscula (a linguagem Java é completamente **case-sensitive**, ou seja, o caractere "a" é completamente diferente do caractere "A", e o uso incorreto da pontuação.

Vamos retornar ao exemplo, o programa **Hello.java**. Intencionalmente, escreveremos a palavra-chave "static" de forma errada e omitiremos o ponto-e-vírgula em uma instrução e a deixaremos errada.

```
public class Hello
{
    /**
     * Meu primeiro programa Java
     */
    public statict void main(String[] args) {
      // A linha abaixo foi retirado o ;
         System.out.println("Hello world!")
    }
```

```
}
```

Salve o programa e execute os passos necessários para compilá-lo. Observe a mensagem de erro gerada ao se tentar compilar novamente o programa:

```
Hello.java:6: <identifier> expected
        public statict void main(String[] args) {
                       ^
Hello.java:10: ';' expected
        }
        ^
1 error
```

A primeira mensagem de erro sugere que existe um erro na linha 6 do programa apontado para a palavra **void**, entretanto esta palavra está correta. O erro é na palavra anterior **statict** que deve ser digitada como **static**.

A segunda mensagem de erro sugere que faltou um ponto-e-vírgula na linha 10, entretanto, esta contém simplesmente o comando de fechar o bloco do método main. O erro está exatamente na linha anterior.

Como regra, ao encontrar muitas mensagens de erros devemos corrigir o primeiro erro da lista e tente novamente compilar o programa. Deste modo reduziremos o número total de mensagens de erro dramaticamente, pois podem existir o que chamamos de erros derivados, ou seja, um erro que tem por causa a instrução anterior.

3.2.2 Erros em tempo de execução (Erros de run-time)

Os erros em tempo de execução são erros que não aparecerão até que tentemos executar o programa. Os programas são compilados com sucesso, mas apresentarão respostas erradas, que podem ter como causa se o programador não obedeceu uma lógica coerente ou no caso em erro de estruturas do programa.

4. Usando NetBeans

Construímos o programa sem utilizar nenhum recurso sofisticado, iremos aprender como fazer todo o processo da seção anterior utilizando uma IDE.

Nesta parte da lição utilizaremos o **NetBeans** que é um Ambiente de Desenvolvimento Integrado (IDE - Integrated Development Environment).

Um ambiente de desenvolvimento integrado é um software aplicativo que possui uma interface construtora, um editor de texto, um editor de código, um compilador e/ou interpretador e um depurador.

Passo 1 : executar o NetBeans

Existem duas formas de executar o NetBeans: a primeira é utilizando a linha de comandos de uma janela terminal e segunda é selecionar o ícone de atalho encontrado na janela da área de trabalho.

Para executar o NetBeans por intermédio da linha de comando, abra uma janela terminal (Os passos para abrir a janela terminal foram discutidos anteriormente) e digite:

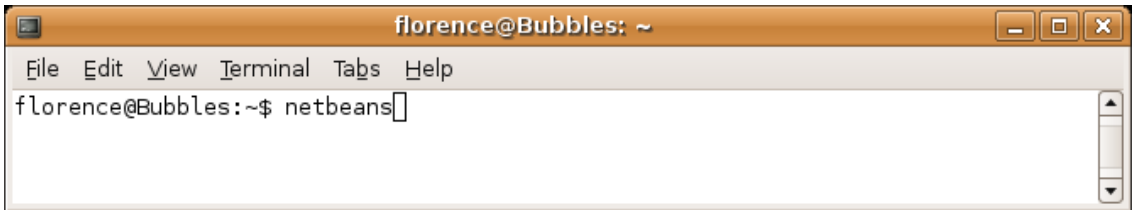


Figura 1: Executando o NetBeans pela linha de comandos

Para o Windows, este comando deve ser executado na pasta em que o NetBeans foi instalado, por exemplo:

```
C:\Program Files\netbeans-5.5\bin>netbeans
```

A segunda maneira de executar o NetBeans é clicando no ícone de atalho encontrado na área de trabalho do computador.

Figura 2: Ícone do NetBeans 5.5 no Desktop

Depois de abrir a IDE NetBeans será mostrada a interface gráfica GUI, conforme à Figura 3:

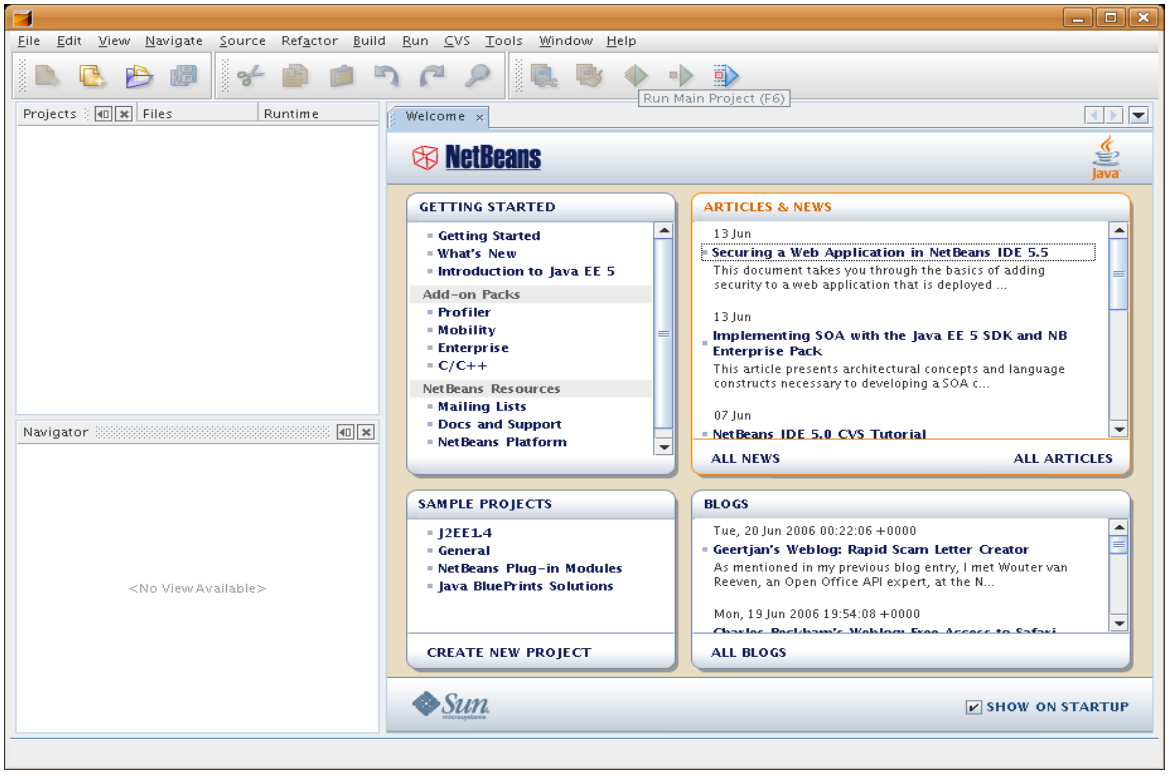


Figura 3: Janela de Welcome do NetBeans

Passo 2: construir o projeto

Clique em File ⇒ New Project, depois de fazer isso, uma janela de diálogo aparecerá. Neste momento deve-se clicar em "Java Application" e em seguida clicar no botão "Next >".

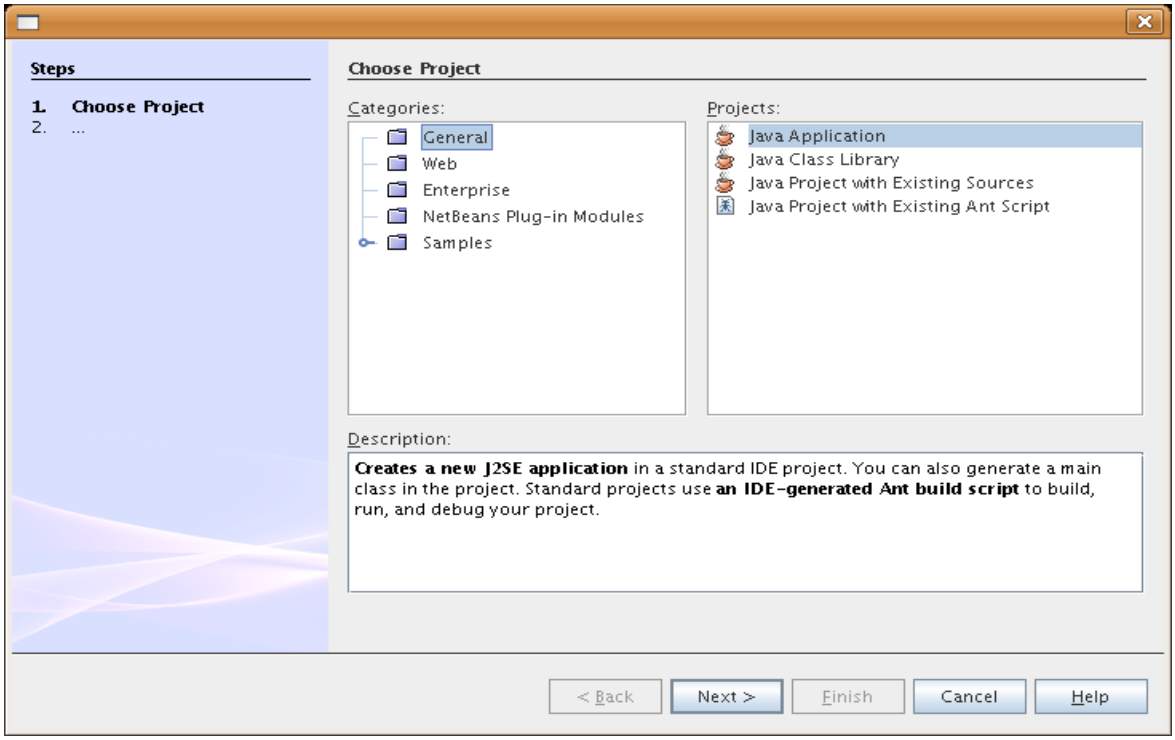


Figura 4: Escolhendo o tipo do projeto

Será mostrada uma nova janela de diálogo, conforme a figura 5.

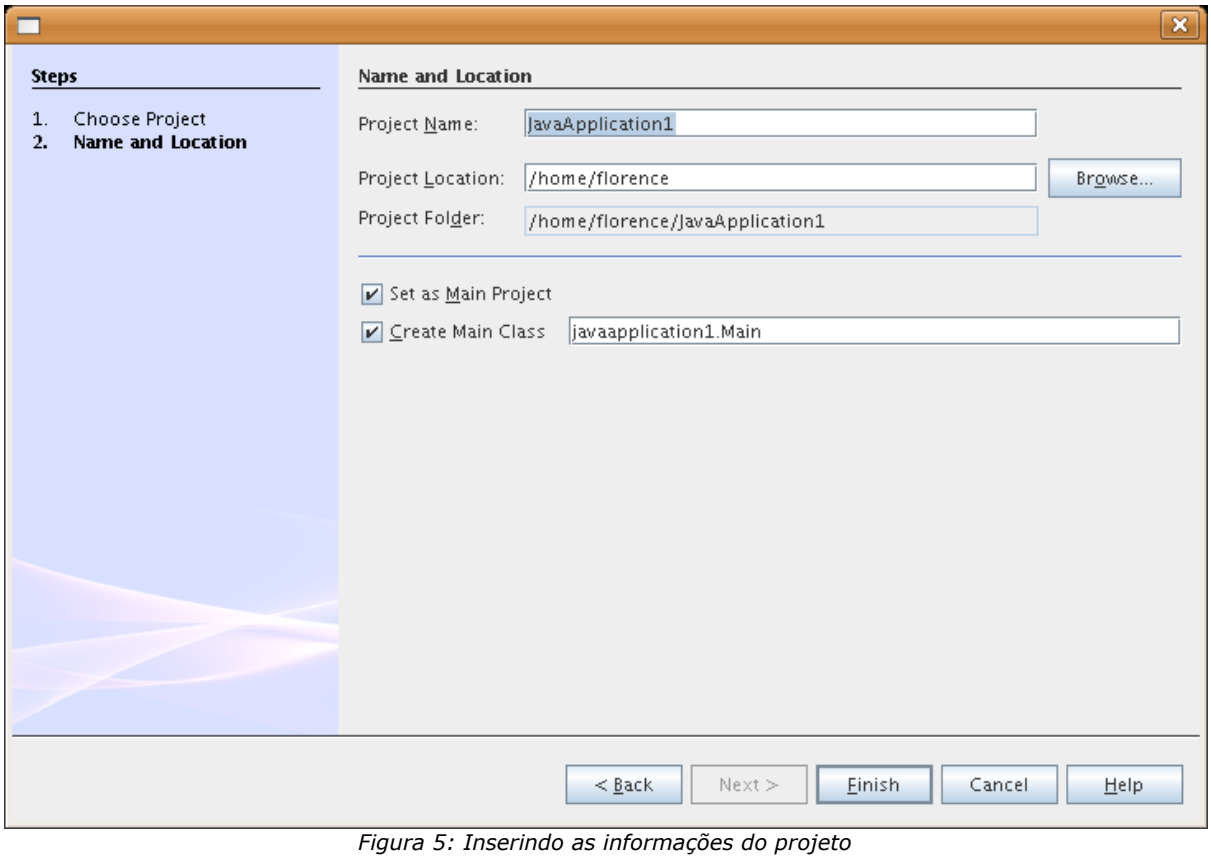


Figura 5: Inserindo as informações do projeto

Troque o local da aplicação clicando no botão "Browse...". Aparecerá uma janela de diálogo para localização do diretório. Dê um clique duplo no seu diretório home.

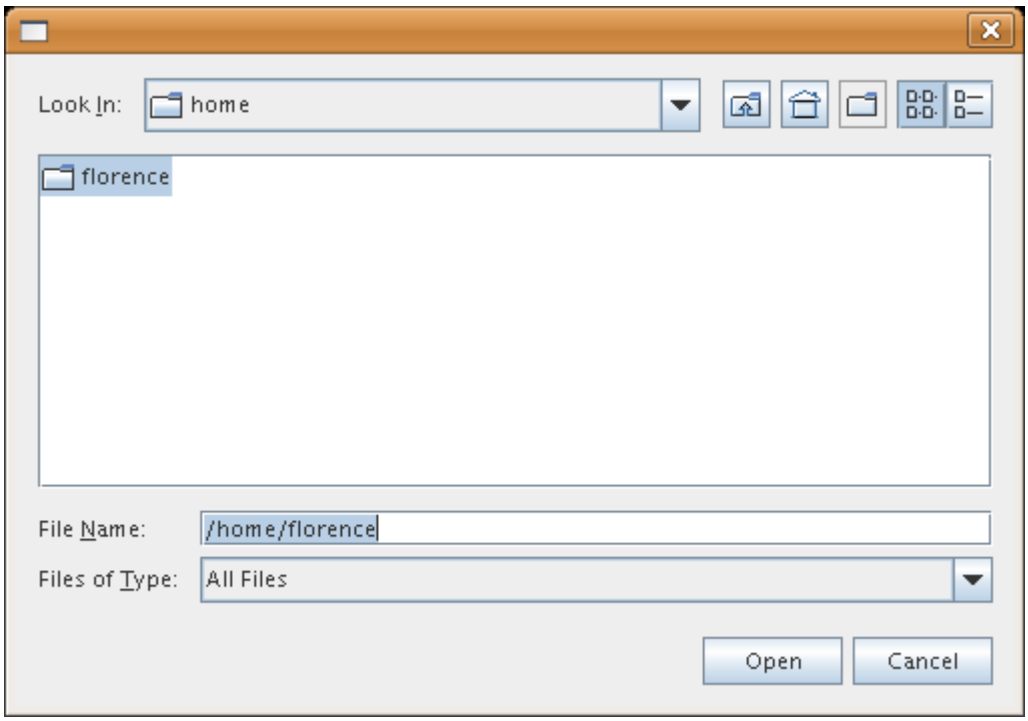
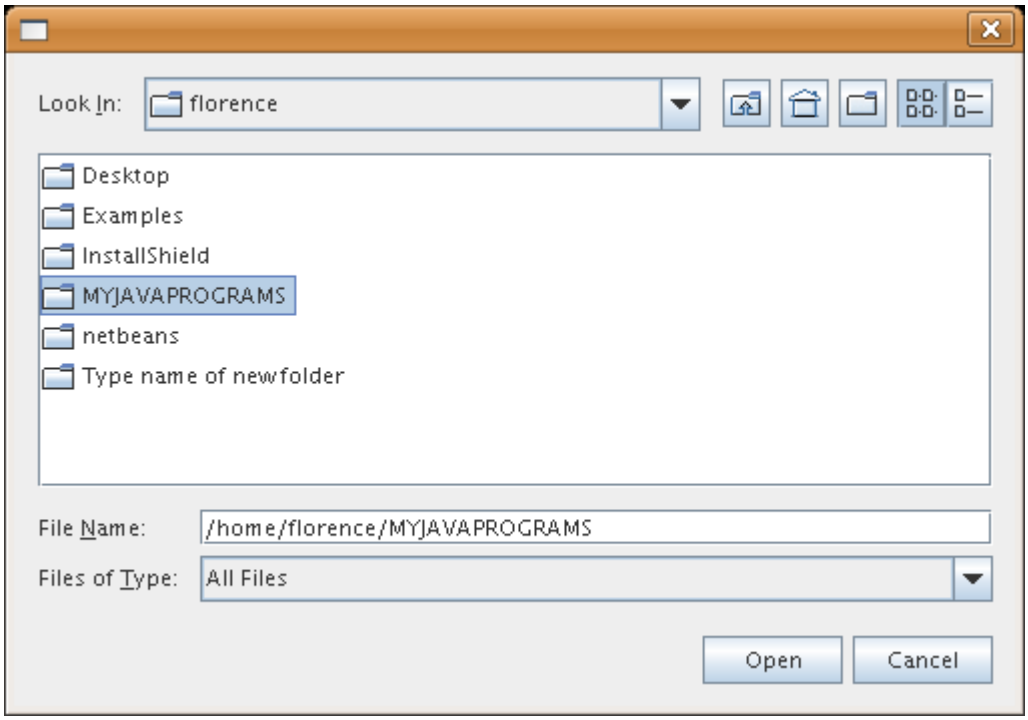


Figura 6: Acertando a Localização do Projeto

O conteúdo da raiz do diretório será apresentado. Dê um clique duplo no diretório MYJAVAPROGRAMS e depois dê um clique no botão "Open".



Veja que a localização do projeto mudou para /home/florence/MYJAVAPROGRAMS.
Finalmente, no campo "Create Main Class", digite "Hello", que será o nome da classe principal, e em seguida clique no botão "Finish".

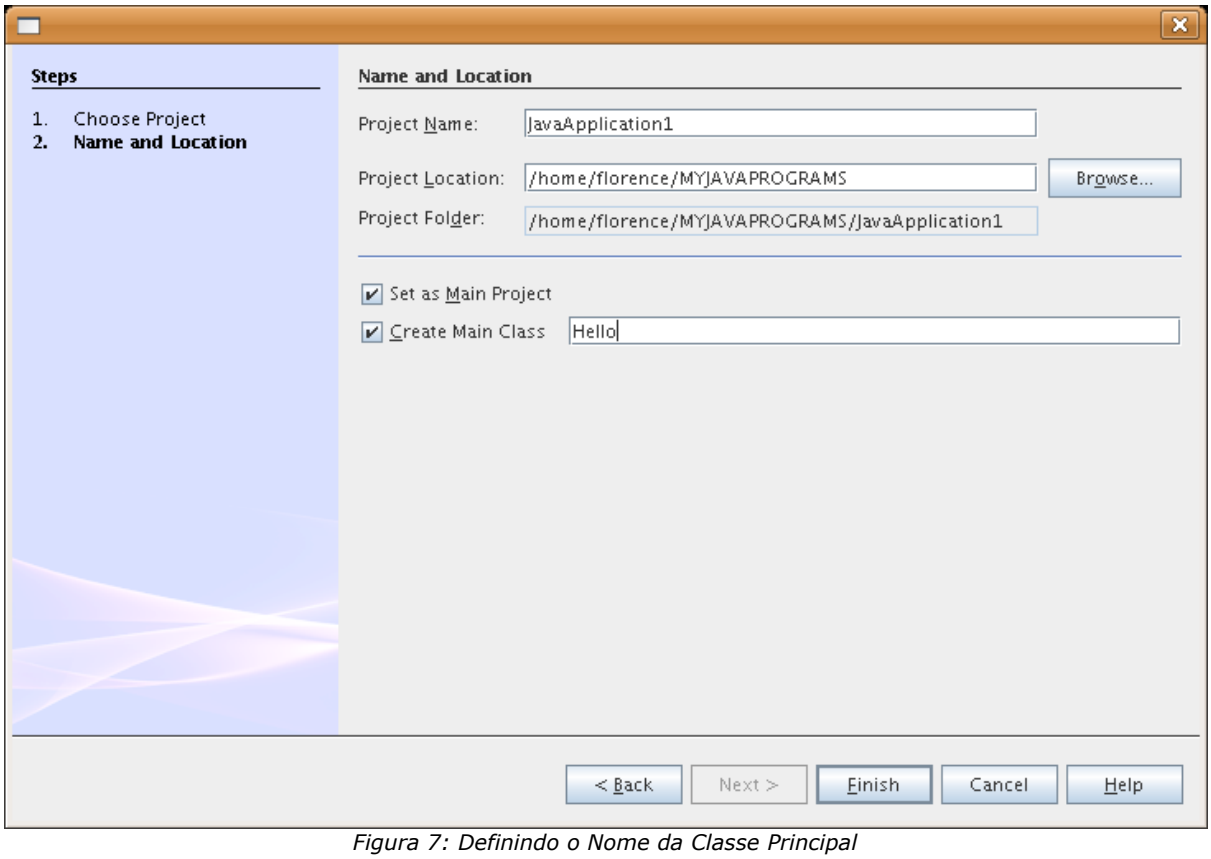


Figura 7: Definindo o Nome da Classe Principal

Passo 3: escrever os detalhes do programa

Antes de escrever o programa descreveremos a janela principal.

Como mostrado na **figura 8**, automaticamente, o NetBeans cria um código básico para o programa Java. Poderemos adicionar as declarações neste código gerado. No lado esquerdo da janela visualizamos uma lista de pastas e arquivos que o NetBeans gerou antes de criar o projeto. Tudo se encontra dentro da sua pasta MYJAVAPROGRAMS, onde foi configurado o local do projeto. No lado direito, visualizamos o código gerado.

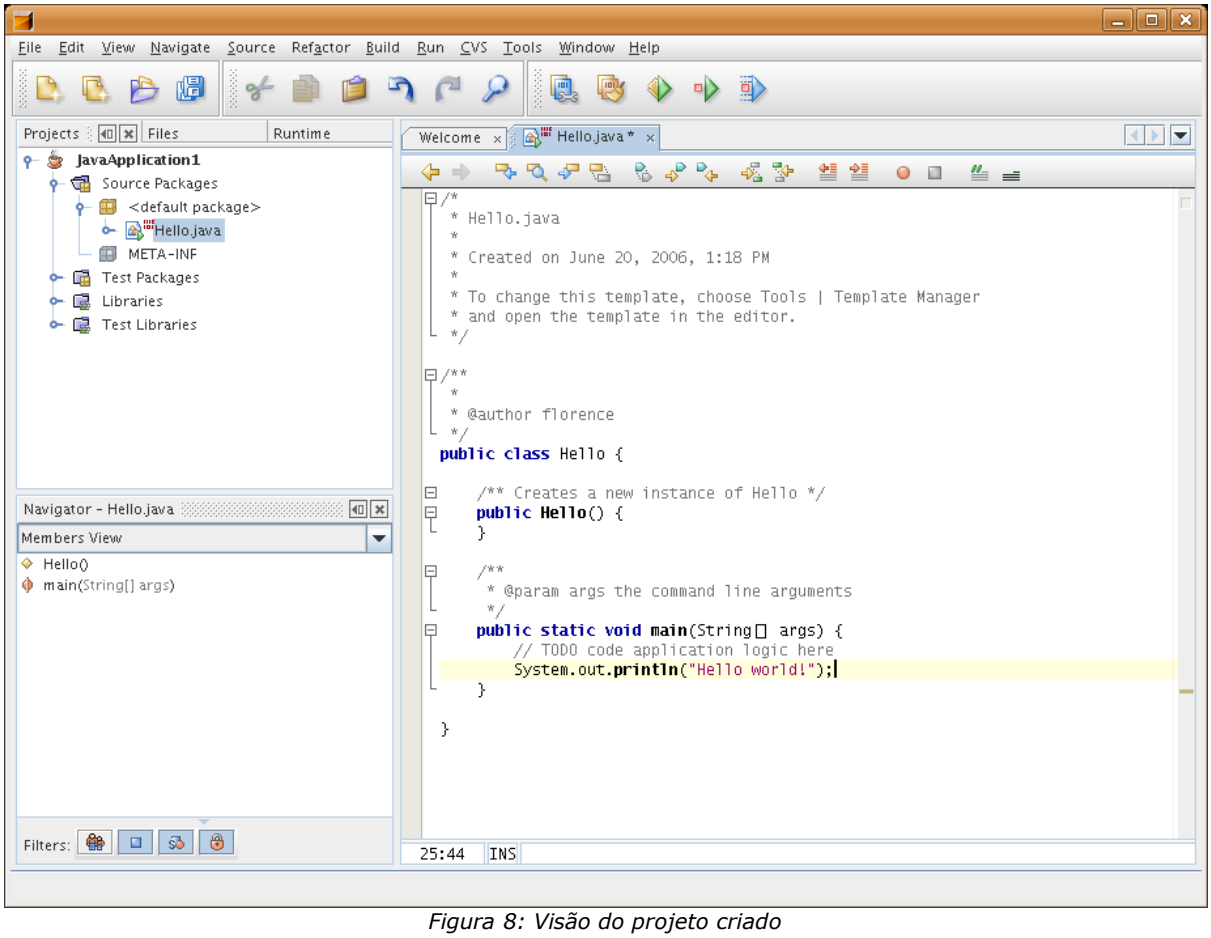


Figura 8: Visão do projeto criado

Modifique o código gerado pelo NetBeans, por hora ignoraremos as outras partes das instruções discutindo os detalhes destas posteriormente. Insira a seguinte instrução:

```
System.out.println("Hello world!");
```

Isto significa que você deseja que seja mostrada a mensagem "Hello world!" na saída padrão do computador, em seguida seja feito um salto de linha. Poderíamos substituir esta instrução por duas equivalentes:

```
System.out.print("Hello");
System.out.println(" world!");
```

O método print() faz com que não seja provocado o salto de linha, utilizaremos para este exemplo a primeira instrução. Insira esta instrução após a linha de comentário (que será desprezada pelo compilador):

```
//TODO code application logic here.
```

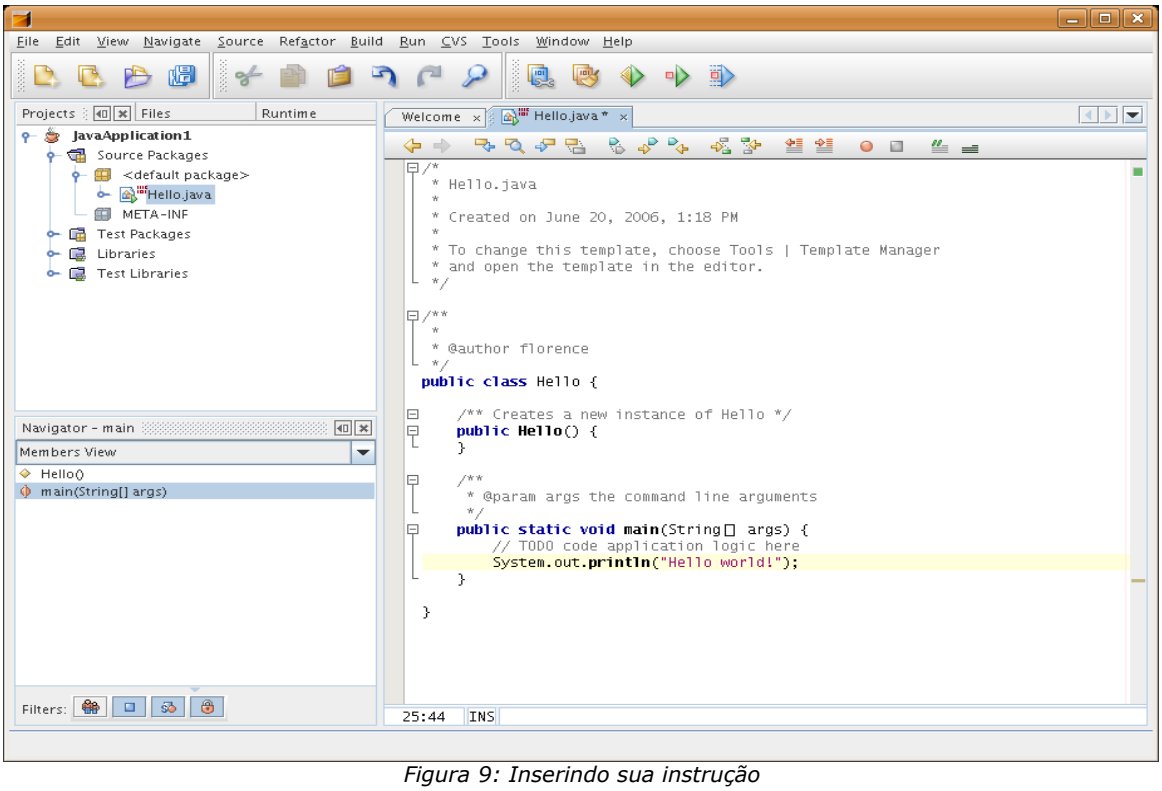


Figura 9: Inserindo sua instrução

Passo 4 : compilar o projeto

Para compilar o programa, a partir do Menu Principal selecione Build ⟹ Build Main Project, ou utilize a tecla de atalho F11, ou utilize o botão de atalho para compilar o código.

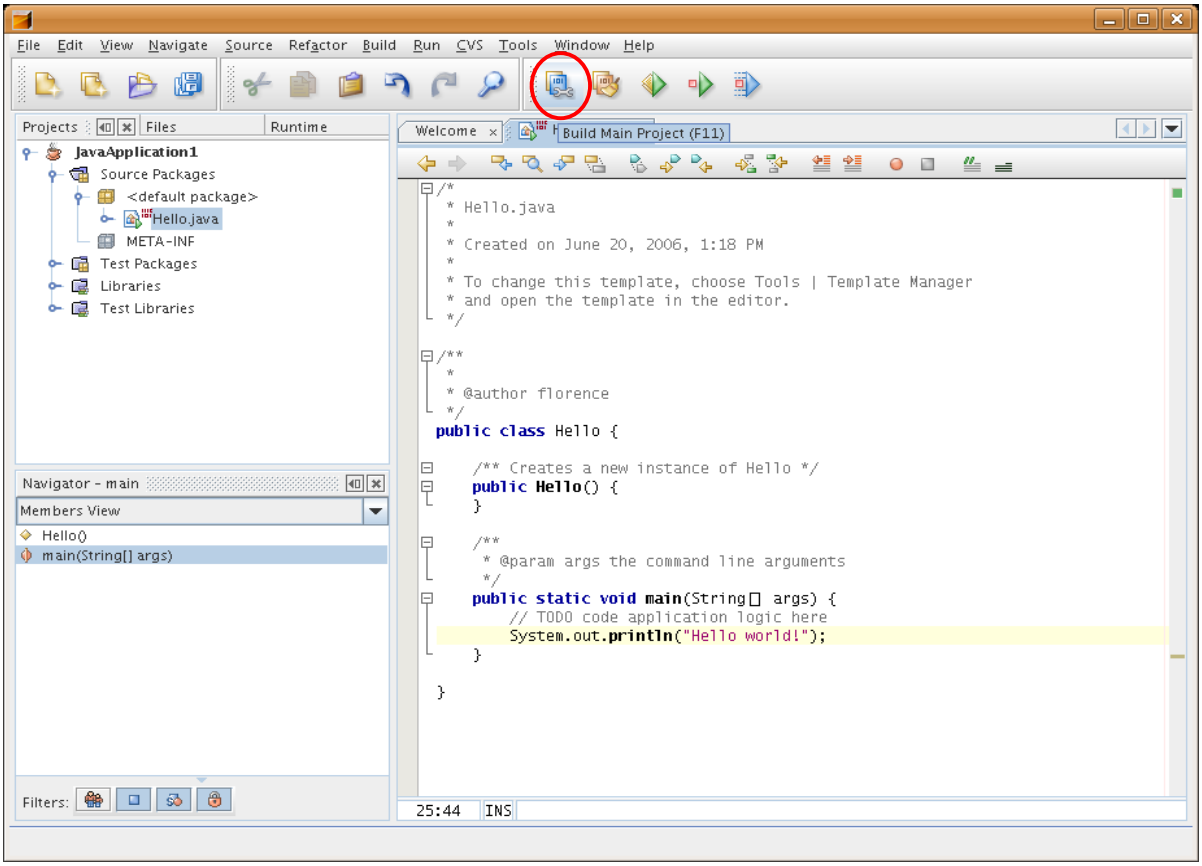


Figura 10: Botão de Atalho para executar o Projeto

Se não existir erros no programa, veremos a mensagem de sucesso na janela de saída.

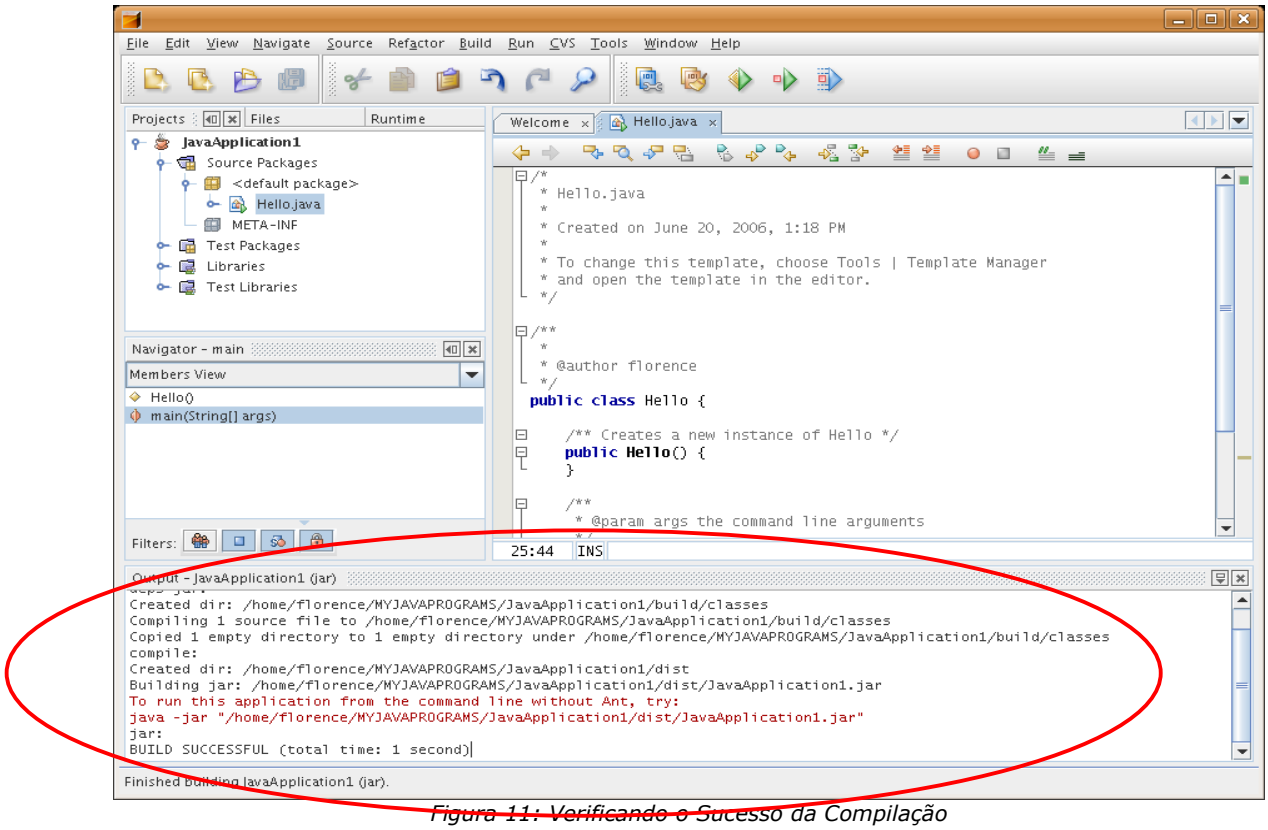


Figura 11: Verificando o Sucesso da Compilação

Passo 5: Executar o projeto

Para executar o programa, clique em Run ⟹ Run Main Project, ou utilize a tecla de atalho F6, ou utilize o botão de atalho para executar o programa.

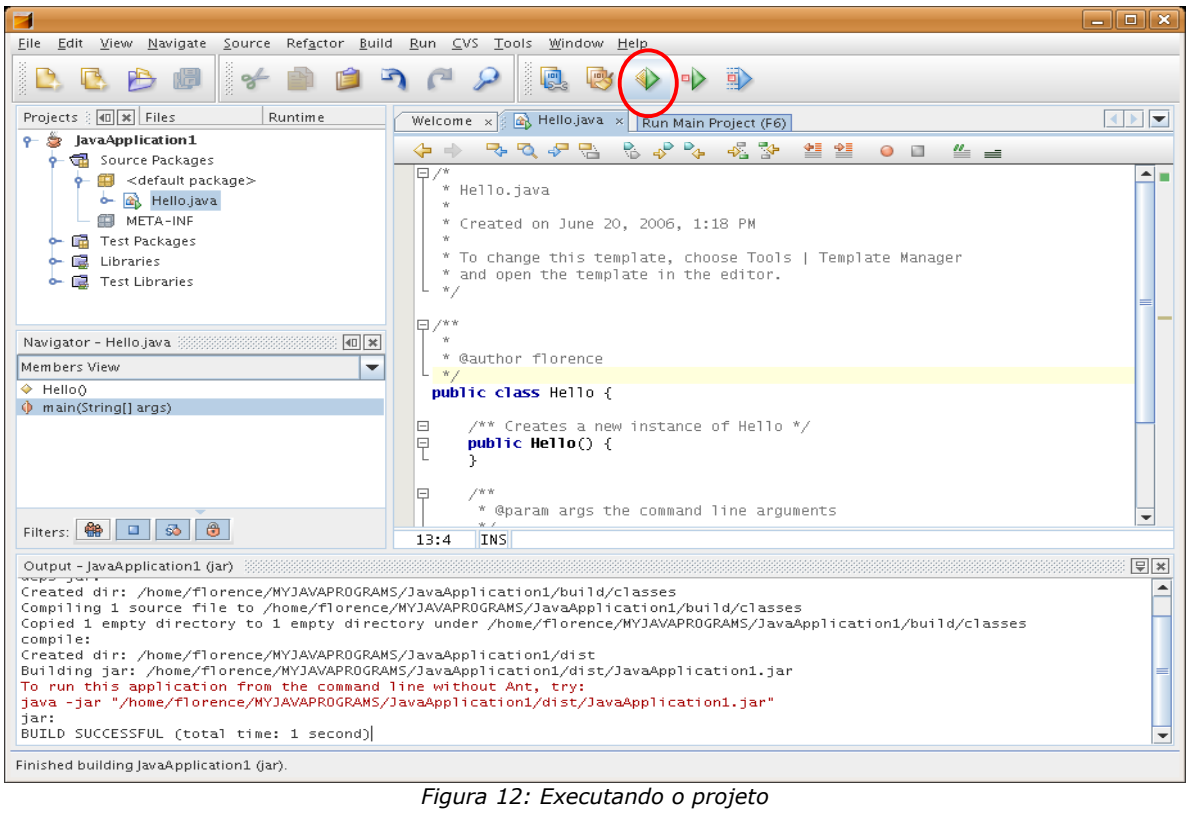


Figura 12: Executando o projeto

O resultado final do programa, será mostrado na janela de saída.

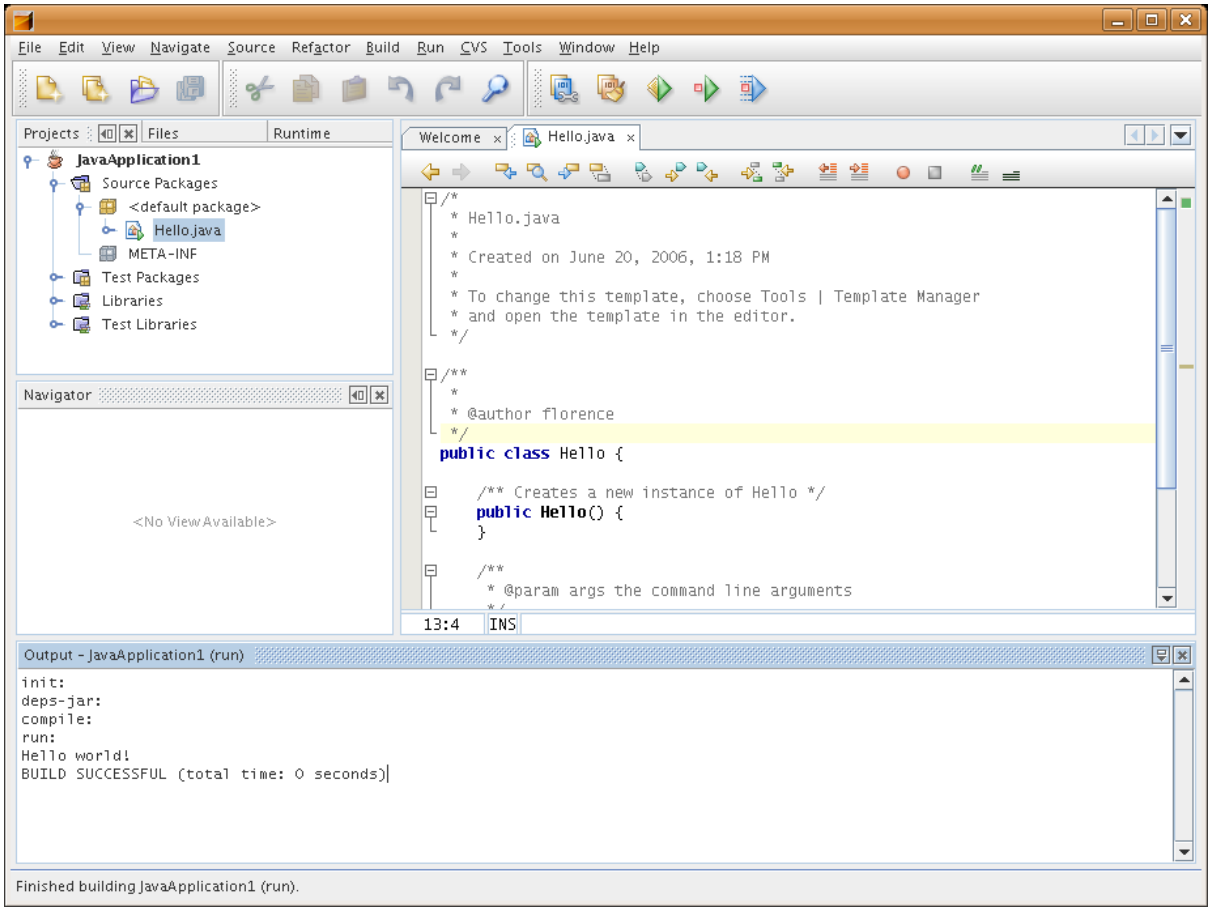


Figura 13: Resultado final da execução do projeto

5. Exercícios

5.1 Melhorando o Hello World!

Utilizando o NetBeans crie uma classe chamada [SeuNome], o programa deverá mostrar como resultado a mensagem:

```
Welcome to Java Programming [SeuNome]!!!
```

5.2 A árvore

Utilizando o NetBeans, crie uma classe chamada **TheTree**. O programa deverá mostrar as seguintes linhas na saída:

```
I think that I shall never see,
[Eu acho que nunca verei,]
a poem as lovely as a tree.
[um poema tão adorável quanto uma árvore.]
A tree whose hungry mouth is pressed
[Uma árvore cuja boca faminta é pressionada]
Against the Earth's sweet flowing breast.
[Contra a Terra fluindo em seu seio docemente.]
```

Parceiros que tornaram JEDI™ possível

Instituto CTS
Patrocinador do DFJUG.

Sun Microsystems
Fornecimento de servidor de dados para o armazenamento dos vídeo-aulas.

Java Research and Development Center da Universidade das Filipinas
Criador da Iniciativa JEDI™.

DFJUG
Detentor dos direitos do JEDI™ nos países de língua portuguesa.

Banco do Brasil
Disponibilização de seus *telecentros* para abrigar e difundir a Iniciativa JEDI™.

Politec
Suporte e apoio financeiro e logístico a todo o processo.

Borland
Apoio internacional para que possamos alcançar os outros países de língua portuguesa.

Instituto Gaudium/CNBB
Fornecimento da sua infra-estrutura de hardware de seus servidores para que os milhares de alunos possam acessar o material do curso simultaneamente.