

Linguagem de Programação Java - Introdução

Jose.wellington@uniceub.br

Calendário

◀ agosto de 2013 ▶

D	S	T	Q	Q	S	S
28	29	30	31	1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

◀ setembro de 2013 ▶

D	S	T	Q	Q	S	S
25	26	27	28	29	30	31
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	1	2	3	4	5

◀ outubro de 2013 ▶

D	S	T	Q	Q	S	S
29	30	1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31	1	2

◀ novembro de 2013 ▶

D	S	T	Q	Q	S	S
27	28	29	30	31	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

◀ dezembro de 2013 ▶

D	S	T	Q	Q	S	S
24	25	26	27	28	29	30
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	1	2	3	4

Agenda

- **Primeiro Programa**
- **Exercício**
- **Arquitetura Java**
- **Netbeans**
- **Variáveis**

Primeiro Programa

Meu Primeiro Programa

```
// Primeiro Programa

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

Meu Primeiro Programa

- Certifique-se de ter adicionado a sua lista de path's o path do compilador e interpretador Java, Javac e Java respectivamente.
- Crie o arquivo ao lado em um diretório qualquer e salve com o nome: `Olah.Java`
- Chame o compilador Java para este arquivo: `javac Olah.Java`
- Seu diretório deve ter recebido um novo arquivo após essa compilação: **`Olah.class`**
- Chame o interpretador Java para este arquivo (omite a extensão `.class` de arquivo): `java Olah`
- Observe o resultado na tela: `Olah mundo !!!`

//Comentário de uma linha

⇒ Comentários em Java seguem a mesma sintaxe de C++.

// Linha

/* Bloco - delimitadores

*/

class Olá

- **class** é a palavra reservada que marca o início da declaração de uma classe.

O trecho do programa

```
public static void main (String[] args)
{
    System.out.println("Olah mundo");
}
```

public:

public significa que o método é acessível de qualquer ponto do código.

static

```
public static void main (String[] args)
{
    System.out.println("Olah mundo");
}
```

- Significa que não é necessário criar um objeto para usar esse método, ele pertence à classe e não ao objeto.

void

```
public static void main (String[] args)
{
    System.out.println("Olah mundo");
}
```

- Semelhante ao `void` C++ ou C, é o valor de retorno da função, quando a função não retorna nenhum valor ela retorna `void`, uma espécie de valor vazio que tem que ser especificado.

main

```
public static void main (String[] args)
{
    System.out.println("Olah mundo");
}
```

- Este é um nome particular de método que indica para o compilador o início do programa;
- É dentro deste método e através das iterações entre os atributos, variáveis e argumentos visíveis nele que o programa se desenvolve.

(String args[])

```
public static void main (String[] args)
{
    System.out.println("Olah mundo");
}
```

- É o argumento do método `main` e por consequência do programa todo, ele é um vetor de `Strings` que é formado quando são passados ou não argumentos através da invocação do nome do programa na linha de comando do sistema operacional, exemplo:

⇒ Java Aloh argumento1 argumento2

{ ... }

```
public static void main (String[] args)
{
    System.out.println("Olah mundo");
}
```

- Os comandos são terminados com **ponto-e-vírgula**.
- Um **bloco** é delimitado por chaves - { e } e constitui um comando composto.

```
{
    int x;
    x = 23 * 54;
}
```

System.out.println - Line

Pois essa função imprime uma linha, e linha inclui uma quebra de linha (ou newline, ou \n, ou [enter], ou parágrafo).

```
System.out.print("Olah mundo\n");
```

System.out.print - normal

Não tem quebra de linha

System.out.printf - formatação

`System.out.println("Olah mundo!");`

```
public static void main (String[] args)
{
    System.out.println("Olah mundo");
}
```

- Chamada do método `println` para o atributo `out` da classe `System`, o argumento é uma constante do tipo `String` para imprimir a cadeia “Olah mundo!” e posicionar o cursor na linha abaixo.
- Por hora guardar esta linha de código como o comando para imprimir mensagens na tela, onde o argumento que vem entre aspas é a `String` a ser impressa.

Exercício

Exercício 01

Implemente um programa que desenhe um "pinheiro" na tela, similar ao abaixo.

```
C:\well>java entrada
      X
     XXX
    XXXXX
   XXXXXX
  XXXXXXX
 XXXXXXXX
XXXXXXX
XXXXXXXX
XXXXXXXXX
XXXXXXXXX
XXXXXXXXX
      XX
      XX
     XXXX
```

Arquitetura Java

A arquitetura Java

- O coletor de lixo (“**Garbage Collection**”)
 - ⇒ É um processo executado em background a nível de sistema operacional que registra toda memória alocada e mantém um contagem do número de referências para cada ponto da memória.
 - ⇒ A JVM (“thread” em background) de tempos em tempos verifica se há na memória alguma referência com valor 0, procedendo então a desalocação da mesma.

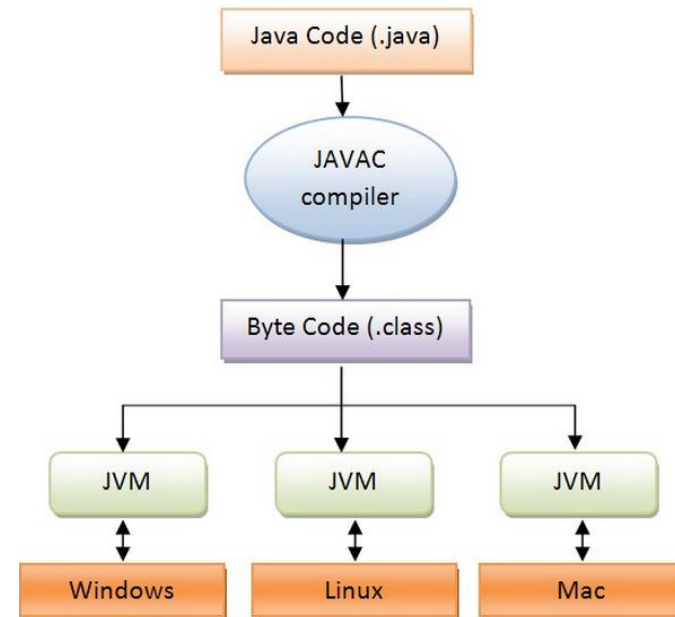


Performance tuning technique number 99:
avoiding long pauses by the garbage collector

A arquitetura Java

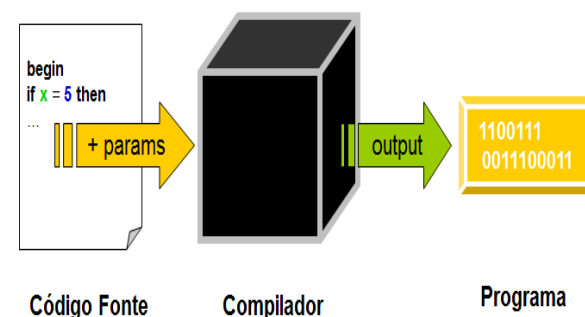
- **Segurança do código**

- ⇒ Os programas javas são convertidos de código fonte para um conjunto de código binário (**byte code**).
- ⇒ Em tempo de execução, os códigos binários são carregados, checados e executados pelo interpretador. O interpretador possui dois serviços:
 - ♦ Executar o código binário;
 - ♦ Fazer as chamadas apropriadas ao sistema operacional para o hardware em uso.



Segurança do código

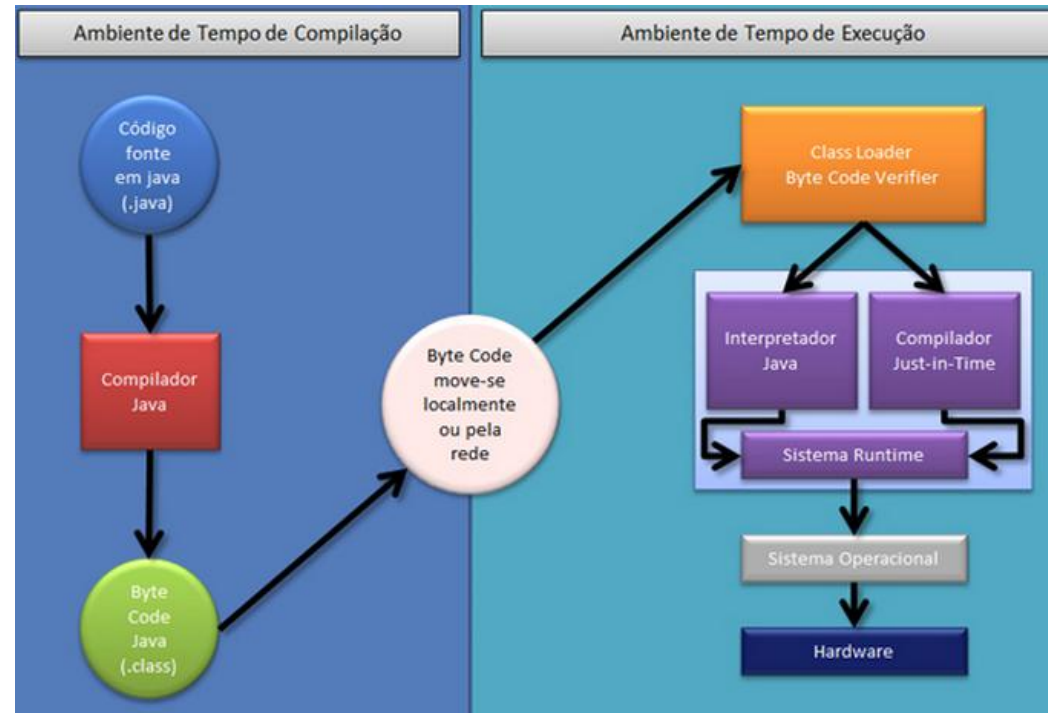
⇒ Em alguns ambientes de execução do java, uma porção do código binário verificado são compiladas para código nativo da máquina e é executado diretamente no hardware da plataforma alvo (JIT-**just-in-time**).



Nota: Todas os arquivos classe que são importados da internet são verificados pelo verificador de código binário.

Segurança do código

⇒ **JIT** é o acrônimo para **compilador just-in-time**, que é um tradutor que converte, em tempo de execução, instruções de um formato para outro, por exemplo, de bytecode para código de máquina.



Netbeans

IDE - Integrated Development Environment



- O NetBeans é uma apenas uma Interface para auxiliar o desenvolvimento.
- Nessa IDE conseguimos trabalhar com a linguagem Java.

IDE - Integrated Development Environment



- Editor de código fonte
- Colorização de sintaxe
- Teclas de acesso rápido a recursos
- Edição de telas
- Debug
- Gerenciamento de versões

IDE - Integrated Development Environment

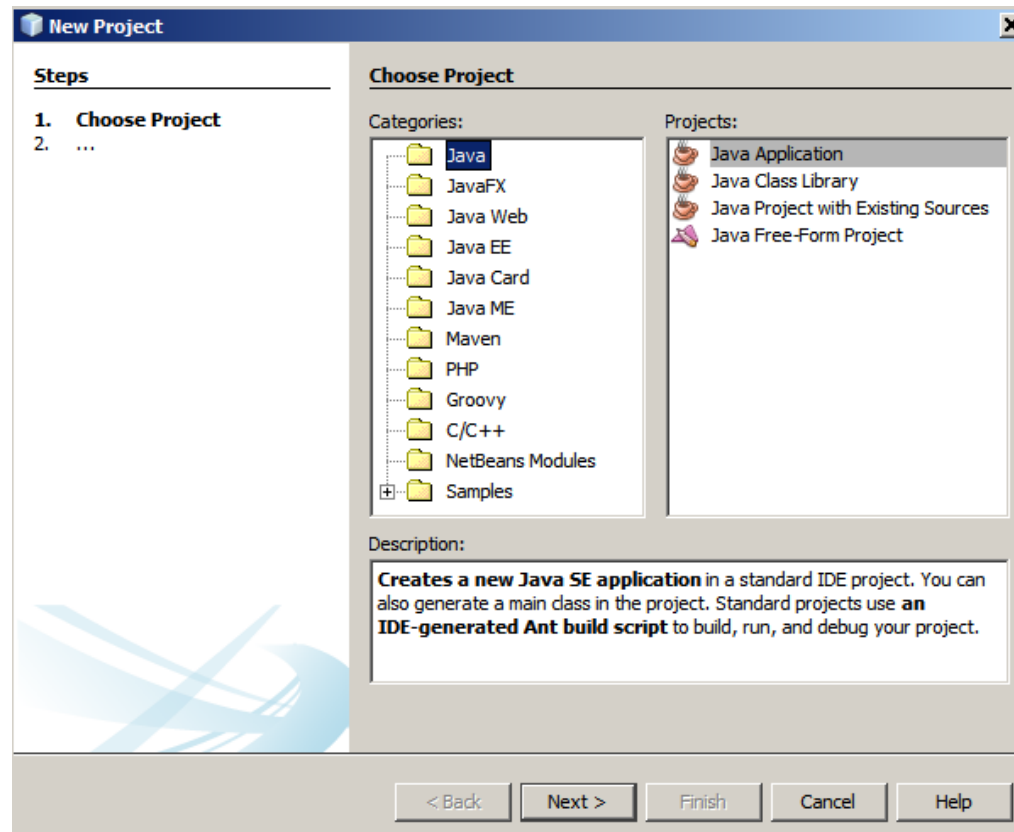
Criando um novo Projeto

Clique em Arquivo-Novo projeto... ou use o atalho Ctrl+Shift+N, ou ainda tem um botão na barra de ferramentas para criar o novo projeto



IDE - Integrated Development Environment

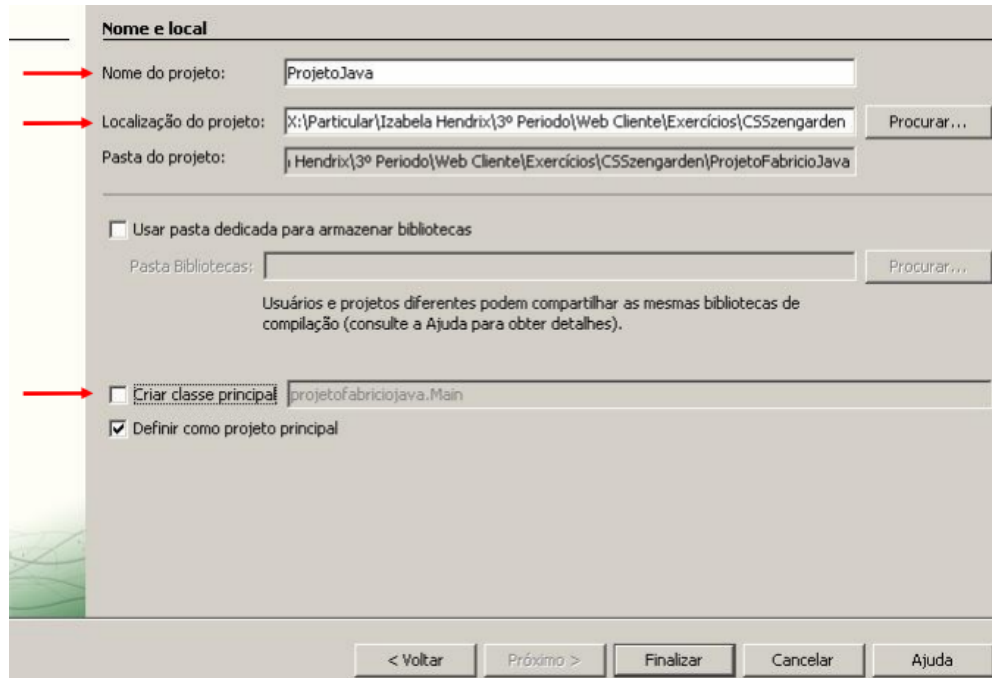
Vai aparecer uma janela como na figura abaixo, selecione no lado esquerdo a Categoria Java.



IDE - Integrated Development Environment

Nome do projeto (lembrando que nome de projetos e classes são sempre escritos em letras minúsculas, mas a primeira letra é maiúscula.

Sempre sem espaços entre as palavras e a primeira letra da segunda palavra maiúscula. Ex.: ProjetoJava).



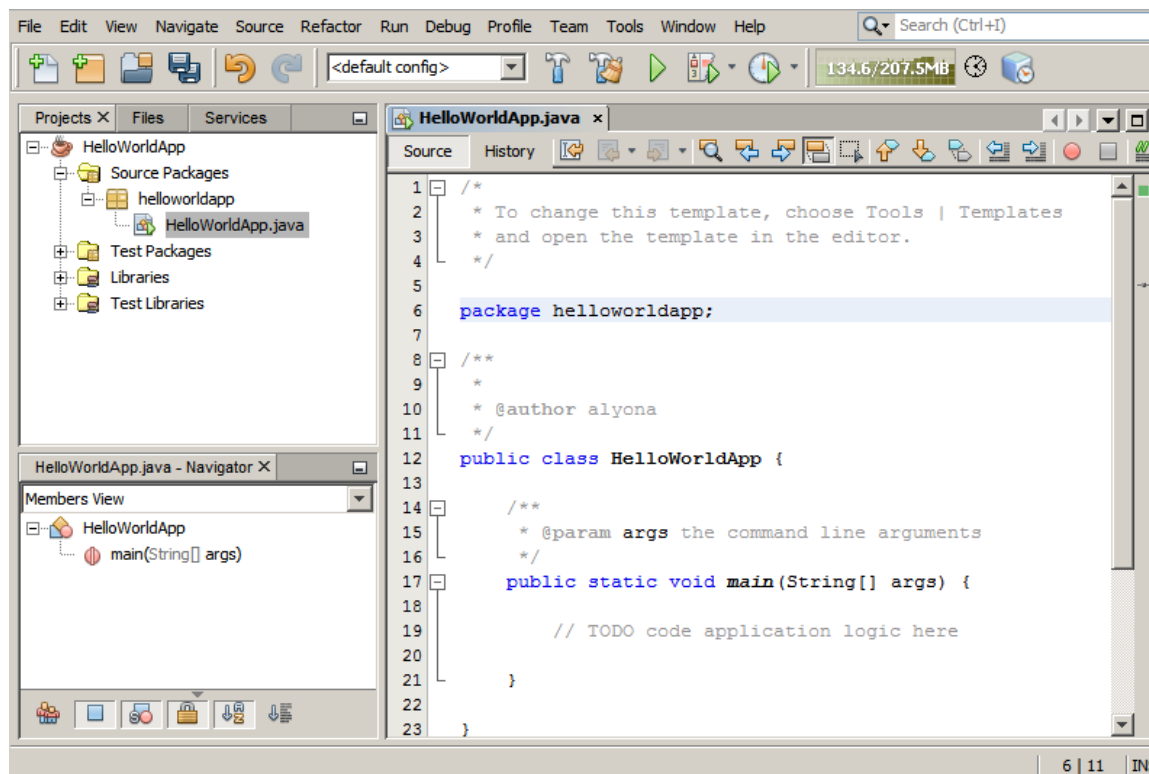
The screenshot shows the 'Nome e local' (Name and location) dialog box in an IDE. It contains the following fields and options:

- Nome do projeto:** ProjetoJava
- Localização do projeto:** X:\Particular\Izabela Hendrix\3º Período\Web Cliente\Exercícios\CSSzengarden (with a 'Procurar...' button)
- Pasta do projeto:** Hendrix\3º Período\Web Cliente\Exercícios\CSSzengarden\ProjetoFabricioJava
- ☐ Usar pasta dedicada para armazenar bibliotecas
- Pasta Bibliotecas:** (with a 'Procurar...' button)
- Usuários e projetos diferentes podem compartilhar as mesmas bibliotecas de compilação (consulte a Ajuda para obter detalhes).
- ☐ Criar classe principal: projetofabriciojava.Main
- ☒ Definir como projeto principal

At the bottom, there are buttons: < Voltar, Próximo >, Finalizar, Cancelar, and Ajuda. Red arrows point to the 'Nome do projeto', 'Localização do projeto', and 'Criar classe principal' fields.

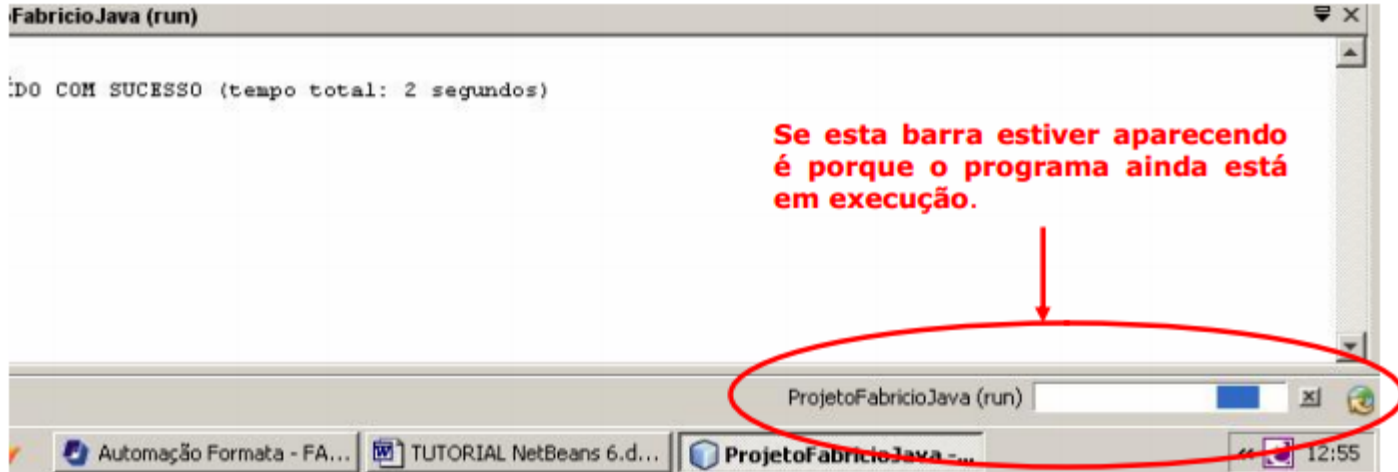
IDE - Integrated Development Environment

A janela Projeto, que contém uma view em árvore dos componentes do projeto, incluindo arquivos de código-fonte, bibliotecas de que seu código depende, e assim por diante.



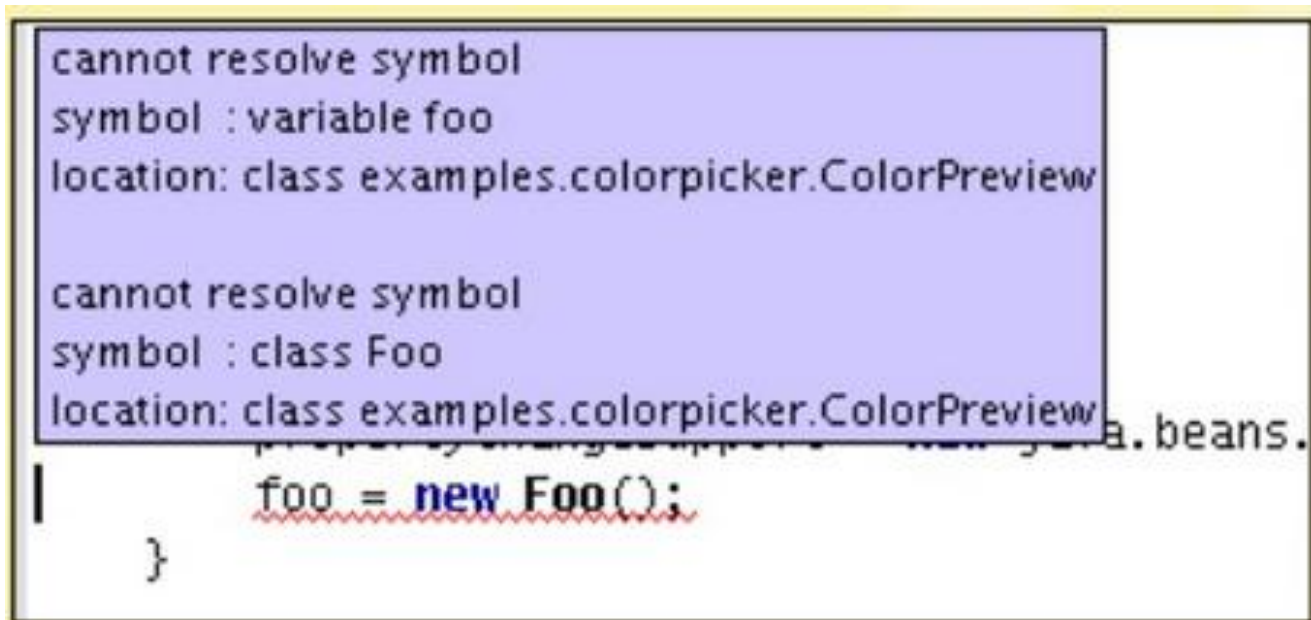
IDE - Integrated Development Environment

Para executar o projeto você pode utilizar o atalho Shift+F6 ou ir no clicar com o botão direito do mouse encima da classe que você quer executar e clique em Executar arquivo.



IDE - Integrated Development Environment

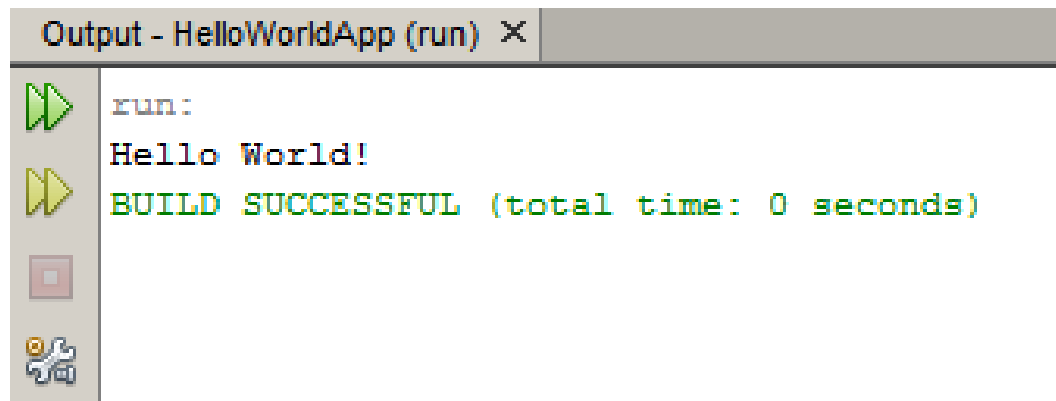
Correção em tempo real



```
cannot resolve symbol  
symbol : variable foo  
location: class examples.colorpicker.ColorPreview  
  
cannot resolve symbol  
symbol : class Foo  
location: class examples.colorpicker.ColorPreview  
... beans.  
|   foo = new Foo();  
    }
```


IDE - Integrated Development Environment

Parabéns! Seu programa funciona!



Output - HelloWorldApp (run) X

```
run:  
Hello World!  
BUILD SUCCESSFUL (total time: 0 seconds)
```

Variáveis

Tipos Primitivos

Tipo	Tamanho	Valor Mín	Valor Máx
boolean	1-bit	—	—
char	16-bit	Unicode 0	Unicode $2^{16} - 1$
byte	8-bit	-128	+127
short	16-bit	-2^{15}	$+2^{15} - 1$
int	32-bit	-2^{31}	$+2^{31} - 1$
long	64-bit	-2^{63}	$+2^{63} - 1$
float	32-bit	IEEE754	IEEE754
double	64-bit	IEEE754	IEEE754

Tipos Primitivos

O Java é uma linguagem fortemente tipada, ou seja, para usarmos **os tipos de informações, temos que declará-los.**

Vamos declarar um inteiro:
`int idade;`

Tipos Primitivos

Inicializando uma variável

Poderíamos atribuir o valor a uma variável de duas maneiras, uma na declaração:

```
int idade=21;
```

Outro meio é depois da declaração:

```
int idade;  
idade=21;
```

Tipos Primitivos

O tipo 'int', por exemplo, armazena 32 bits, ou qualquer inteiro entre -2.147.483.648 e 2.147.483.647

O tipo 'float', que armazena números decimais (quebrados, ou com vírgula) também armazenam 32 bits.

Já os 'long' armazenam 64 bits, assim como 'double' (que é um 'float' maior), ou seja, qualquer número inteiro de -9.223.372.036.854.775.808L até 9.223.372.036.854.775.807L.

Vamos declarar um tipo 'long':
`long idade_do_universo;`

Tipos Primitivos

O tipo char: armazenando e representando caracteres

Declaração

```
char nome_do_char = 'a';
```

Onde poderíamos substituir 'a' por qualquer caractere alfanumérico, caso tenhamos declarado assim.

Tipos Primitivos

O tipo char: armazenando e representando caracteres

Declaração

```
char nome_do_char = 'a';
```

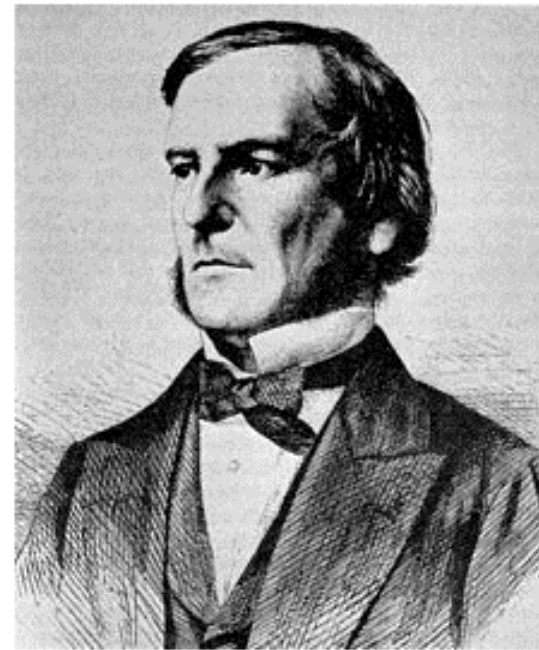
Onde poderíamos substituir 'a' por qualquer caractere alfanumérico, caso tenhamos declarado assim.

Tipos Primitivos

O tipo boolean: a base da lógica Booleano é um tipo de dado que permite apenas dois valores, true (verdadeiro) ou false (false).

Declaração

```
boolean nome_bool = true;  
boolean nome_bool2 = false;
```



George Boole

Agenda

- **Primeiro Programa**
- **Exercício**
- **Arquitetura Java**
- **Netbeans**
- **Variáveis**