

Objetivos: Conhecer a estrutura básica de um programa em JAVA;  
Conhecer os tipos de dados primitivos de JAVA;  
Adicionar funcionalidades aos botões dos formulários;

Recursos: Aula em laboratório com uso do computador; Quadro e pincel; Roteiro de Aula;

## UM PROGRAMA SIMPLES EM JAVA

**Arquivo:** PrimeiroPrograma.java

```
1. public class PrimeiroPrograma{  
2.     public static void main (String args[]){  
3.         System.out.println("Olá Mundo!");  
4.     }  
5. }
```

Linha 1 – Esta linha define uma classe chamada Programa do tipo pública, ou seja, esta poderá ser acessada por qualquer pessoa ou programa, sem implementar segurança. Observe que o nome da classe possui o MESMO NOME DO ARQUIVO, incluindo a letra maiúscula no início cada palavra.

Linha 2 – Esta linha cria o que chamamos de método, que são as ações que o programa irá realizar. Todo método deve COMEÇAR COM LETRA MINÚSCULA e possuir parênteses. Ex.: acendeFarol() ou pisca(). Este método em específico se chama main() e será executado automaticamente quando rodarmos o programa. Ele estará presente em todos os programas que criarmos durante a disciplina. As outras palavras definem que ele é público (public), que não é necessário criar um objeto para usá-lo (static) e que ao final da execução ele não necessita retornar qualquer valor (void).

Linha 3 – Este comando é usado para exibir a mensagem Olá Mundo! na linha de comando.

Linha 4 e 5 – As chaves são usadas para delimitar blocos de comando, a da linha 4 fecha o bloco do método main() e a da linha 5 fecha a classe PrimeiroPrograma.

## CÓDIGO AUTOMÁTICO

```
package javaapplication4;  
  
public class NovoJFrame extends javax.swing.JFrame {  
  
    /** Creates new form NovoJFrame */  
    public NovoJFrame() {  
        initComponents();  
    }  
  
    private void jButton2ActionPerformed(java.awt.event.  
        // TODO adicione seu código de manipulação aqui:  
    }  
  
    public static void main(String args[]) {  
        java.awt.EventQueue.invokeLater(new Runnable() {  
            public void run() {  
                new NovoJFrame().setVisible(true);  
            }  
        })  
    }  
}
```

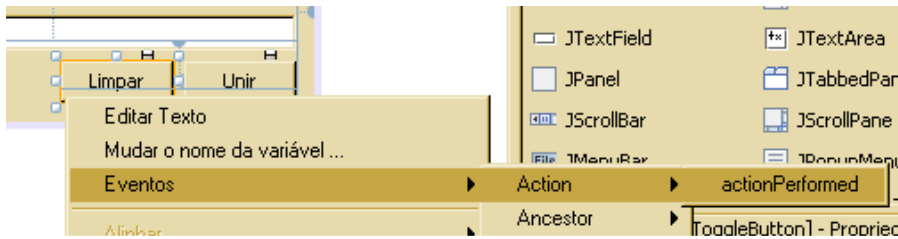
Boa parte do código de um programa será gerado automaticamente pelo NetBeans. As partes em AZUL não poderão ser modificadas diretamente pelo usuário. Os trechos em AMARELO permitem modificações, e é onde nós escreveremos nossos comandos.

É possível expandir ou recolher o código para facilitar a edição, basta clicar nos símbolos + e – ao lado do início de cada bloco de código.

Observe que o editor destaca as palavras com cores diferentes para facilitar a leitura do código.

## ADICIONANDO FUNCIONALIDADE A UM BOTÃO

Para fazer com que um botão no formulário tenha funcionalidade, clique sobre ele com o botão direito do mouse e selecione as opções **Eventos > Action > actionPerformed**. Logo após será exibida a tela de edição de código e o cursor já estará posicionado no local adequado.



## CRIANDO VARIÁVEIS EM JAVA

Para criar uma variável em Java, declare da seguinte forma:

```
int numero;
```

Onde **int** é o tipo do dado e **numero** é o nome da variável.

Uma variável Java pode ser definida em qualquer local do código.

Os nomes das variáveis devem sempre começar com letras e não pode conter espaços.

## TIPOS DE DADOS

String	Com S maiúsculo. Usado para guardar palavras ou números em formato de texto. Ex. Casa.
int	Inteiro, para números sem parte decimal. Ex: 150. Usado para idade ou quantidades.
double	Números reais, com parte fracionária. Ex. 1,3514. Usado para dinheiro, divisões, etc.
boolean	Aceita como valor true (verdadeiro) ou false (falso).
char	Armazena um único caractere. Ex. V.

Existem diversos outros tipos de dados, a princípio iremos trabalhar apenas com estes.

## ATRIBUINDO VALOR ÀS VARIÁVEIS

Usamos o operador = (igual) para adicionar o valor a uma variável. Ex.:

```
String nome = "JOÃO";           (use aspas duplas)
char c = 'L';                   (use aspas simples)
int x = 10;                      (sem aspas)
```

Se uma variável foi criada anteriormente, não é necessário repetir o nome do tipo. Ex.:

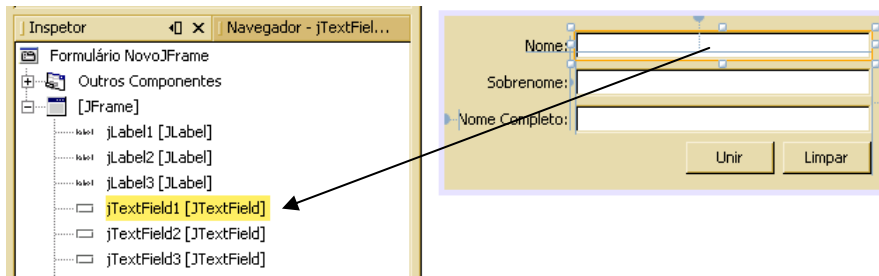
```
x = 20;
```

## CAPTURANDO E EXIBINDO TEXTO DO FORMULÁRIO

O componente `textField` possui diversos métodos para manipulação de dados, entre eles dois que serão bastante utilizados durante nossos primeiros projetos. São eles:

`getText()` - Usado para ler o conteúdo digitado no campo e salvá-lo numa variável `String`.  
`setText()` - Usado para atribuir um conteúdo `String` ao campo.

O primeiro passo é saber o nome que o NetBeans deu ao campo. Clique sobre o campo e observe o nome do item no Inspetor. Este nome poderá ser alterado caso o usuário deseje (selecione e pressione F2).



Adicione funcionalidade a um botão, e no editor de código digite:

```
String nome = jTextField1.getText();
```

Este código irá capturar o valor que foi digitado em **jTextField1** e salva-o na variável **nome** que foi criada.

Agora adicione a funcionalidade ao segundo botão, e no editor de código digite:

```
jTextField3.setText(nome);
```

Este código irá atribuir o conteúdo da salva na variável **nome** ao campo **jTextField3**.

OBS.: O componente **jTextArea** também utiliza esses mesmos métodos.

## CONCATENANDO (UNINDO) STRINGS

Uma das possibilidades de se trabalhar com Strings é você poder montar em uma variável um texto composto de diversas partes de texto. Vejamos o exemplo:

```
String dia1 = "segunda";  
String fim = "-feira";
```

```
String seg = dia1 + fim;
```

Se você exibir o conteúdo da variável **seg** verá "segunda-feira" escrito na tela. O operador **+** quando usado em Strings, coloca o conteúdo de uma variável *colado* do lado da outra. Veja outro exemplo.

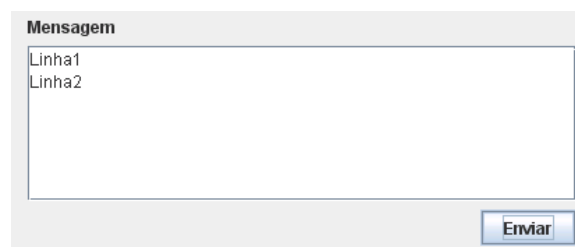
```
String t1 = "Texto 1"  
String tudo = t1 + " Texto 2";
```

Neste segundo exemplo concatenamos o valor contido em **t1** com um texto que não está numa variável. Para que haja um espaço entre as duas palavras, devemos colocá-lo antes da primeira letra de **Texto 2**.

Para inserir quebras de linhas, devemos usar o modificador **\n** no final de cada linha. Ex.:

```
String t1 = "Linha 1 \n";  
String t2 = "Linha 2 \n";  
String tudo = t1 + t2;  
jTextArea1.setText(tudo);
```

Ao visualizar o componente **jTextArea1** você verá que as linhas foram escritas uma abaixo da outra.



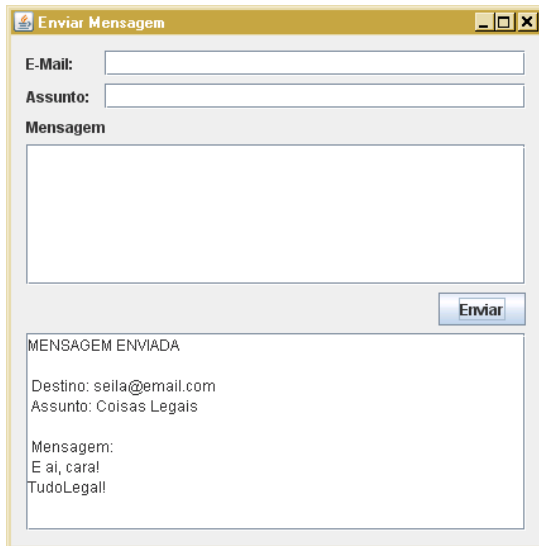
Existem outros modificadores como **\** que usamos quando queremos exibir aspas duplas.

## EXERCÍCIOS

Modifique os programas da aula anterior a fim de atribuir as seguintes funcionalidades.

1. No primeiro programa, faça com que o botão UNIR colete as informações dos campos Nome e Sobrenome e exiba no campo Nome Completo a concatenação destes. E o botão LIMPAR deve apagar as informações de todos os campos. **Dica:** Para limpar o conteúdo do campo, basta colocar uma string vazia.

2. Adicione um segundo JTextArea e faça com que o botão Enviar colete as informações dos três primeiros campos e exiba na nova área de texto o conteúdo destes, conforme o modelo abaixo. Esse mesmo botão deve limpar o conteúdo dos campos superiores ao exibir os dados na nova área.



The image shows a Java Swing window titled "Enviar Mensagem". It contains the following elements:

- Two text input fields: "E-Mail:" and "Assunto:".
- A large text area labeled "Mensagem".
- An "Enviar" button located at the bottom right of the message area.
- A confirmation area below the message area with the text: "MENSAGEM ENVIADA", "Destino: seila@email.com", "Assunto: Coisas Legais", and "Mensagem: E ai, cara! TudoLegal!".