

## 8.2 Caixa de Mensagem

As caixas de mensagem são componentes visuais que possibilitam apresentar mensagens diversas e chamar a atenção do usuário para eventuais ações executadas pelo programa.

Para usar o recurso de caixa de mensagem, considere um programa que faça a leitura de dois valores inteiros. O programa deve efetuar a adição, e caso o resultado obtido seja maior ou igual a 10, deve ser apresentada uma caixa de mensagem com o valor calculado acrescido de 5. Caso o valor calculado não seja maior ou igual a 10, deve ser apresentada uma caixa de mensagem com o valor subtraído de 7.

Execute o comando **File/New Project**, selecione na caixa de diálogo **New Project** a opção **Windows Form Application**, informe no campo **Name** o nome **Cap08\_Ex03** e acione o botão **OK**.

O formulário do programa terá seu desenvolvimento semelhante aos outros formulários já desenvolvidos. Assim sendo, insira no formulário dois controles **Label**, dois controles **TextBox** e um controle **Button**. Os controles **Label** devem possuir os textos "**Entre um valor para <A>:**" e "**Entre um valor para <B>:**". Para o controle **Button** altere seu texto para **Processar**. A Figura 8.18 apresenta o modelo do formulário.

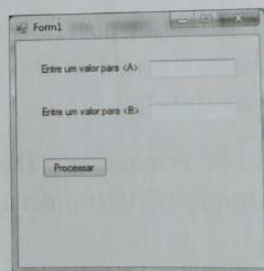


Figura 8.18 - Formulário para exibição de caixa de mensagem.

Na sequência acione o botão **Processar**, para que seja escrito na rotina o código em negrito apresentado em seguida:

```
int R, X, A, B;  
A = int.Parse(textBox1.Text);  
B = int.Parse(textBox2.Text);  
X = A + B;  
if (X >= 10)  
{
```

```

        R = X + 5;
        MessageBox.Show("R = " + R.ToString(), "Resultado V");
    }
    else
    {
        R = X - 7;
        MessageBox.Show("R = " + R.ToString(), "Resultado F");
    }
    textBox1.Focus();

```

Se forem fornecidos os valores 4 e 5, o resultado apresentado será 2. Caso sejam fornecidos os valores 5 e 7, o resultado será 17.

Execute o comando **File/Save All** para gravar o projeto do programa.

O programa anterior usa a instrução **MessageBox.Show()** para apresentar a resposta de saída do programa. A caixa de mensagem gerada pela instrução **MessageBox.Show()** apresenta uma mensagem dentro de uma caixa com um botão **OK**. Note que foi feita a concatenação do resultado da operação com a mensagem que será exibida pelo recurso, com o uso do método **ToString()**, que converte um valor numérico em um valor *string*. A funcionalidade **MessageBox.Show()** possui a seguinte sintaxe:

```

MessageBox.Show(MSG, TIT, BOT, ÍCO, PAD, ALI);

```

sendo:

- ☒ **MSG:** é a mensagem de saída que será exibida dentro da caixa de mensagem.
- ☒ **TIT:** a mensagem de identificação do título a ser exibida na barra de título da caixa de mensagem.
- ☒ **BOT:** é uma constante interna que identifica os tipos de botões a serem apresentados na caixa de mensagem. Veja a tabela mais adiante.
- ☒ **ICO:** é uma constante interna que identifica os tipos de ícones a serem apresentados na caixa de mensagem. Veja a tabela mais adiante.
- ☒ **PAD:** é uma constante interna que identifica qual dos botões apresentados estará selecionado com execução de ação padrão. Veja a tabela mais adiante.
- ☒ **ALI:** é uma constante interna que permite escrever uma mensagem ao lado direito da caixa. Esse parâmetro normalmente não é usado. Veja tabela mais adiante.



Os parâmetros podem ser omitidos desde que não haja omissões de parâmetros intermediários. Por exemplo, podem ocorrer as seguintes combinações: só MSG; só MSG, TIT; só MSG, TIT, BOT; só MSG, TIT, BOT, ICO; só MSG, TIT, BOT, ICO, PAD e por fim só MSG, TIT, BOT, ICO, PAD, ALI. Combinação do tipo MSG, BOT, PAD não é aceita.

A seguir apresenta-se a tabela contendo a descrição de cada constante a ser fornecida como argumento para formatação de uma caixa de mensagem.

Tipo de botão	Constante
Apresenta o botão OK por padrão quando este valor é omitido.	MessageBoxButtons.OK
Mostra os botões OK e Cancelar.	MessageBoxButtons.OKCancel
Apresenta os botões Anular, Repetir e Ignorar.	MessageBoxButtons.AbortRetryIgnore
Exibe os botões Sim, Não e Cancelar.	MessageBoxButtons.YesNoCancel
Apresenta os botões Sim e Não.	MessageBoxButtons.YesNo
Indica os botões Repetir e Cancelar.	MessageBoxButtons.RetryCancel
Apresenta o ícone de advertência. O desenho de um X branco dentro de um círculo vermelho.	MessageBoxIcon.Error ou MessageBoxIcon.Hand ou MessageBoxIcon.Stop
Mostra o ícone de questionamento. O desenho de um balão azul com uma interrogação em branco.	MessageBoxIcon.Question
Apresenta o ícone de atenção. O desenho de um triângulo amarelo com um símbolo de exclamação preto.	MessageBoxIcon.Warning ou MessageBoxIcon.Exclamation
Exibe um ícone de informação. O desenho de um balão azul com a letra I em branco.	MessageBoxIcon.Information ou MessageBoxIcon.Asterisk
Não exibe nenhum ícone, sendo esta a forma padrão quando é omitido.	MessageBoxIcon.None
Determina qual dos botões será marcado como botão padrão. Deverá ser substituído o N pelo número do botão desejado.	MessageBoxDefaultButton.ButtonN
Determina o alinhamento à direita para a mensagem e título da caixa de mensagem.	MessageBoxOptions.RightAlign

Veja as possibilidades de combinações de uso obtidas pelo método **Show()** da estrutura **MessageBox** de acordo com alguns dos seus parâmetros. A Figura 8.19 mostra da esquerda para a direita de cima para baixo os exemplos conseguidos a partir dos seguintes trechos de códigos. Os códigos apresentados podem ser escritos da forma apresentada ou no sentido horizontal. Para executá-los, basta colocar em um formulário um controle **Button** e acrescentar o trecho de código à instrução **MessageBox.Show()**;

```
private void button1_Click(object sender, EventArgs e)
{
    MessageBox.Show
    (
        "Mensagem",
        "Título"
    );
}
```

```
private void button1_Click(object sender, EventArgs e)
{
    MessageBox.Show
    (
        "Mensagem",
        "Título",
        MessageBoxButtons.OKCancel
    );
}
```

```
private void button1_Click(object sender, EventArgs e)
{
    MessageBox.Show
    (
        "Mensagem",
        "Título",
        MessageBoxButtons.AbortRetryIgnore
    );
}
```

```
private void button1_Click(object sender, EventArgs e)
{
    MessageBox.Show
    (
        "Mensagem",
        "Título",
        MessageBoxButtons.YesNoCancel
    );
}
```

```
private void button1_Click(object sender, EventArgs e)
```

```

{
    MessageBox.Show
    (
        "Mensagem",
        "Título",
        MessageBoxButtons.YesNo
    );
}

```

```

private void button1_Click(object sender, EventArgs e)
{
    MessageBox.Show
    (
        "Mensagem",
        "Título",
        MessageBoxButtons.RetryCancel
    );
}

```

```

private void button1_Click(object sender, EventArgs e)
{
    MessageBox.Show
    (
        "Mensagem",
        "Título",
        MessageBoxButtons.OK,
        MessageBoxIcon.Error
    );
}

```

```

private void button1_Click(object sender, EventArgs e)
{
    MessageBox.Show
    (
        "Mensagem",
        "Título",
        MessageBoxButtons.OK,
        MessageBoxIcon.Question
    );
}

```

```

private void button1_Click(object sender, EventArgs e)
{
    MessageBox.Show
    (
        "Mensagem",
        "Título",
        MessageBoxButtons.OK,
        MessageBoxIcon.Warning
    );
}

```

```

private void button1_Click(object sender, EventArgs e)
{
    MessageBox.Show
    (
        "Mensagem",
        "Título",
        MessageBoxButtons.OK,
        MessageBoxIcon.Information
    );
}

```

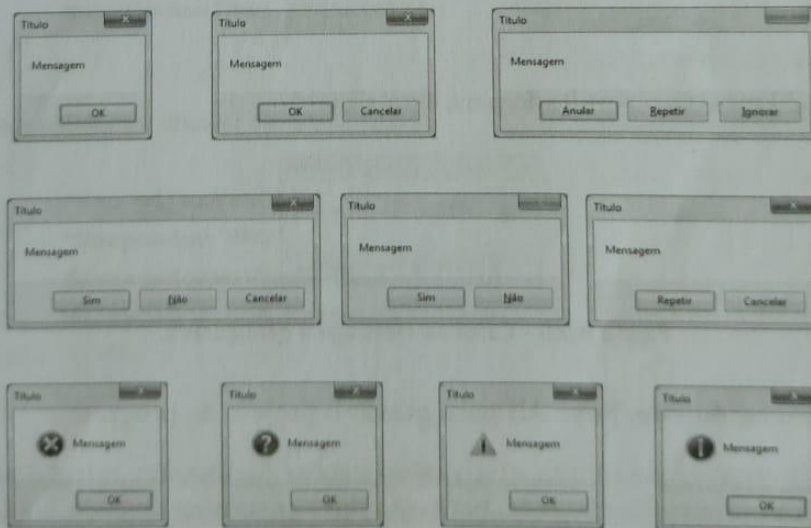


Figura 8.19 - Exemplos de uso MessageBox.Show().



O tipo enumerado **DialogResult** disponibiliza como constantes de captura de ações de botões, além do valor **Yes**, os valores **Abort**, **Cancel**, **Ignore**, **No**, **None**, **OK** e **Retry**.

As Figuras 8.21, 8.22 e 8.23 mostram as telas que aparecem quando da execução do programa.

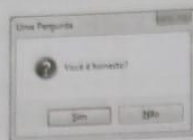


Figura 8.21 - Pergunta a ser executada.



Figura 8.22 - Ação para condição verdadeira.

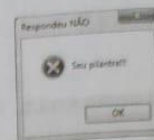


Figura 8.23 - Ação para condição verdadeira.

Execute o comando **File/Save All** para gravar o projeto do programa.

### 8.3 Botão de Rádio

O botão de rádio é um conjunto de componentes que permite selecionar uma opção da lista. Para demonstrar sua ação, considere um programa de calculadora simples, que faz a leitura de dois valores reais e apresenta o resultado da operação matemática escolhida. Observe a Figura 8.24 com o formulário que será usado pelo programa.

Execute o comando **File/New Project**, selecione na caixa de diálogo **New Project** a opção **Windows Form Application**, informe no campo **Name** o nome **Cap08\_Ex06** e acione o botão **OK**.

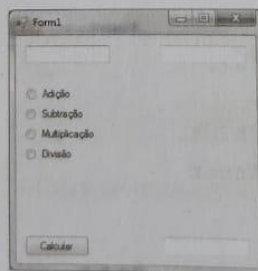


Figura 8.24 - Formulário do programa calculadora.

Para montar o formulário que deve ter a sua propriedade **Text** com o texto **Calc-Sim - Calculadora Simples**, tome o cuidado de inserir os dois controles **TextBox** superiores um ao lado do outro. Observe também que o tamanho do formulário do programa foi um pouco diminuído. Acrescente um controle **TextBox** no canto inferior direito.

No formulário é usado um controle novo, denominado **RadioButton**. Insira quatro controles desse tipo de forma que o controle **radioButton1** tenha seu **Text** alterado para o texto **Adição**, o controle **radioButton2** tenha seu **Text** alterado para o texto **Subtração**, o controle **radioButton3** tenha seu **Text** alterado para o texto **Multiplicação** e o controle **radioButton4** tenha seu **Text** alterado para o texto **Divisão**.

O outro controle é um **Button** com sua propriedade **Text** alterada para **Calcular** colocado no canto inferior esquerdo.

Em relação ao terceiro controle **TextBox** (**textBox3**), serão feitas algumas alterações em suas propriedades. Assim sendo, selecione-o, localize na janela **Properties** a propriedade **Locked**, alterando-a para **True**. Desta forma, esse controle somente terá a capacidade de apresentar um valor, não permitindo a sua alteração. Como será utilizado para apresentar o resultado da operação, essa alteração se torna conveniente. Outra alteração é com a propriedade **TabStop** alterada para **False**. Desta forma, se você utilizar a tecla **<Tab>** para movimentar o foco nos controles do formulário, ele estará inibido.

O controle **button1**, que deve estar com seu **Text** alterado para **Calcular**, deve ter o código em negrito, indicado em seguida:

```
private void button1_Click(object sender, EventArgs e)
{
    float R = 0, A, B;
    A = float.Parse(textBox1.Text);
    B = float.Parse(textBox2.Text);
    if (radioButton1.Checked)
        R = A + B;
    if (radioButton2.Checked)
        R = A - B;
    if (radioButton3.Checked)
        R = A * B;
    if (radioButton4.Checked)
        if (B == 0)
            MessageBox.Show("ERRO : Divisão zero!");
        else
            R = A / B;
    textBox3.Text = R.ToString();
    textBox1.Focus();
}
```