6 - Trabalhando com Variáveis

Nos exercícios desenvolvidos até este momento não necessitamos de recursos para "guardar" a informação digitada pelo usuário do aplicativo, como por exemplo no último exercício, o nome e o sexo do usuário eram tratados diretamente no controle TextBox (Caixa de Texto), até porque os dados digitados eram textos e, em regra, tudo o que é digitado em um TextBox é tratado na linguagem como texto (também chamado tecnicamente de "string").

Os próximos projetos que serão desenvolvidos utilizarão novos "tipos de dados", como por exemplo: solicitar a digitação da idade da pessoa, o peso, o valor de uma mercadoria, entre outros. Estes dados serão numéricos e precisarão ser "convertidos" de "texto" (string) para "números" para podermos trabalhar com eles, fazer operações matemáticas...

Variáveis são "áreas" (espaços) da memória do computador, que utilizamos para armazenar dados (números, palavras, datas...). Esses dados ficarão armazenados "temporariamente" na memória do computador enquanto o programa estiver em execução e serão descartados após o término do programa. Oportunamente estudaremos como "guardar" (gravar) estes dados no HD do computador, para serem utilizados em outros momentos.

IMPORTANTE:

a) C# é "**case sensitive**". Significa que uma variável declarada como NOMEALUNO (maiúsculo) é diferente da variável NomeAluno.

b) C# é "**fortemente tipada**", isto é, exige que os valores armazenados em uma variável sejam compatíveis com o tipo de variável.

c) C# é "**estaticamente tipada**", isto é, exige que as variáveis sejam declaradas e inicializadas antes de sua utilização.

d) Por convenção, as variáveis são declaradas com o **primeiro caracter em minúsculo**, exemplo: cliente, nomeCliente...

Para a criação de variáveis recomenda-se:

a) Se houver mais de uma palavra, a primeira letra da segunda palavra deve ser maiúscula, exemplo: nomeCliente, codAluno.

b) Pode-se usar um prefixo para expressar, no início da variável, o tio de dado ao qual ela pertence: strNomecliente, floPesoMedio, intIdadeAluno...

c) Os nomes não poderão conter somente números, por exemplo: chamar uma variável de 123 e também não poderão ter o mesmo nome de uma instrução utilizada pelo programa (chamados de nomes reservados). Por exemplo, chamar uma variável de if.

6.1 - Tipos de variáveis:

byte - Número inteiro de 8-bits sem sinal (0 a 255)

sbyte - Número inteiro de 8 bits com sinal (-127 a 128)

ushort - Número inteiro de 16 bits sem sinal (0 a 65.535)

short - Número inteiro de 16 bits com sinal (-32.768 a 32.767)

uint - Número inteiro de 32 bits sem sinal (0 a 4.294.967.295)

int - número Inteiro de 32-bits com sinal (-2.147.483.648 a 2.147.483.647)

ulong - Número inteiro de 64 bits sem sinal (0 a 18.446.744.073.709.551.615)

long - Número inteiro de 64 bits com sinal (-9.223.372.036.854.775.808 a 9.223.372.036.854.775.807)

double - Número de ponto-flutuante de 64bits - 15 dígitos de precisão

float - Número de ponto flutuante de 32 bits - 7 dígitos de precisão

decimal - Número de ponto flutuante de 128 bits - 28 dígitos de precisão

bool - Valor Booleano, verdadeiro ou falso.

Char - Um único caracter.

string - Texto, número ou qualquer caracter.

6.2 - Onde declarar as vairáveis?

Dependendo do "local" onde as variáveis forem declaradas (criadas), elas terão um "escopo" (visibilidade) diferenciado.

Por exemplo:

- Se declarar uma variável dentro de um bloco de programação, por exemplo: "private void BtnExibir...", esta variável existirá somente para este bloco, será "visível" somente para este "pedaço de código". Para todos os demais "blocos" esta variável não existirá. Dependendo do projeto, esta opção poderá atender perfeitamente...

- Para os casos em que a variável for necessária em todos os blocos de programação (um alcance mais global), a criação da variável precisará ser feita "fora do bloco". No próximo exercício estudaremos em mais detalhe este procedimento.

6.3 - Como Declarar as variáveis dentro do programa

A declaração de variaveis segue o formato:

[TIPO] [NOME DA VARIAVEL] = [VALOR];

ou

[TIPO] [NOME DA VARIAVEL];

Vale lembrar que o C# é case-sensitive, então a varíavel "Nome" é diferente da variável "nome".

Exemplo: Declaração de uma variável chamada "idade", do tipo "inteira" → int idade;

6.4 - Como Atribuir conteúdo às variáveis:

Podemos atribuir conteúdo às variáveis de duas formas diferentes:

a) Diretamente, no momento da declaração:

int idade = 10;

float numeroComplexo = 4.5F;
(**Observe** a utilização do sufixo "F" - ou "f" – após o "ponto". A não utilização acarretará um erro)

double altura = Convert.ToDouble(TxtAltura.Text);
(**Observe** a necessidade da "conversão" do dado que está no TextBox para a variável "altura").

byte numChamada = Convert.ToByte(TxtnumChamada.Text);

b) Após a declaração:

int c;

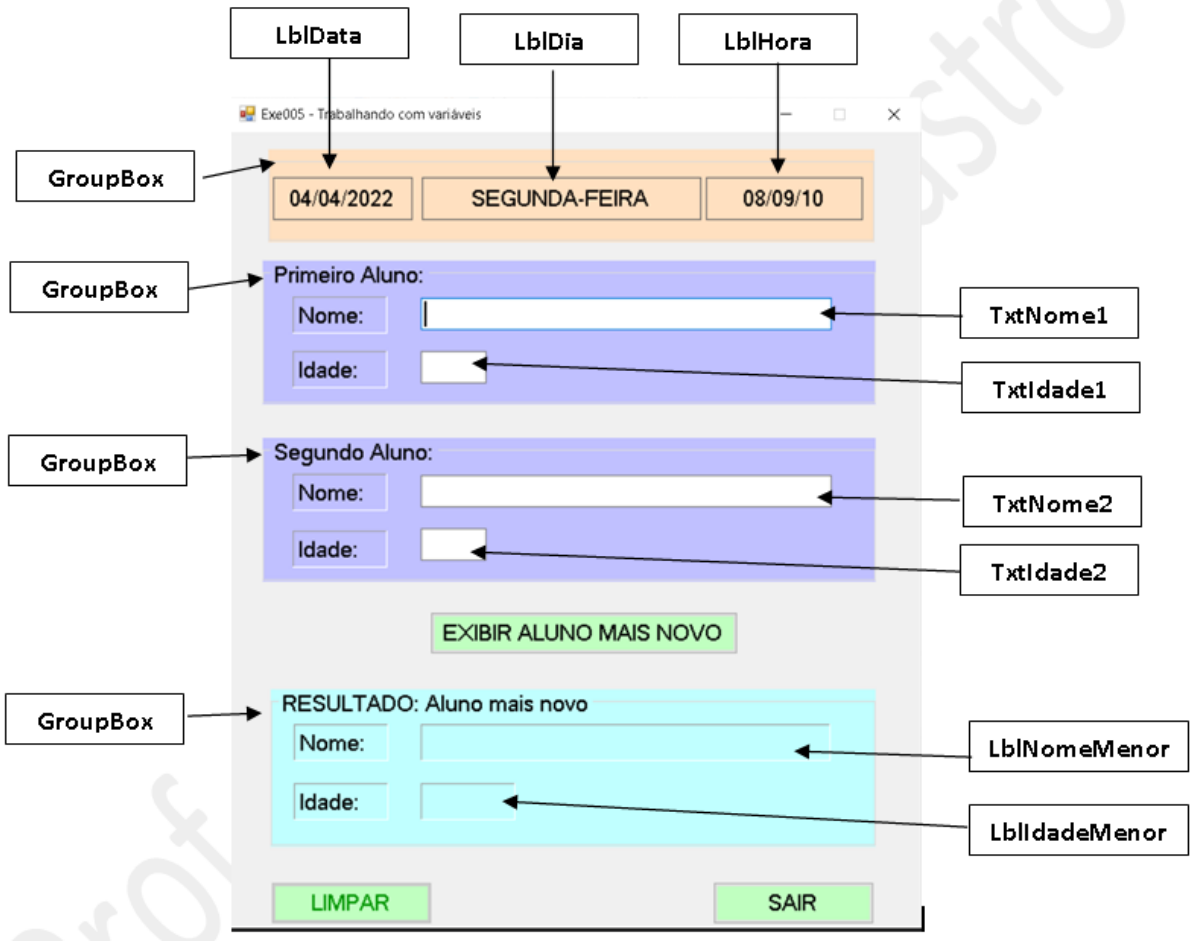
double altura;

c=10;

altura= Convert.ToDouble(TxtAltura.Text);

Agora, para colocar em prática este novo conhecimento, vamos para o próximo projeto, o **Exe005**.

Interface gráfica



Inserimos neste projeto um novo controle (componente visual): o GroupBox, com o objetivo de melhor visualizar o resultado.

O primeiro "GroupBox" para "agrupar" a data/dia e hora atual; o segundo "GroupBox" para agrupar os dados do "primeiro aluno"; o terceiro "GroupBox" para agrupar os dados do "segundo aluno" e o quarto GroupBox para agrupar e exibir o resultado.

Principais funcionalidades do projeto

- Ao carregar o formulário (Form_Load) o projeto exibirá a Data, o Dia e a Hora corrente.

- A caixa de texto "TxtIdade1" e a "TxtIdade2" estarão "bloqueadas" para a digitação de dado "não numérico".

- Botão "Exibir Aluno mais Novo":

1. Verificar se todas as informações estão preenchidas (nomes e idades). Exibir mensagem de alerta caso alguma informação esteja ausente;
2. Atribuir os dados do TextBox para as variáveis;
3. Identificar, através da "estrutura condicional if", o aluno com a "menor idade" e exibir os dados deste aluno (nome e idade) na área específica.

   OBS: No caso de alunos com idades iguais (poderá ocorrer), exibir a mensagem informando sobre a igualdade.
4. Desabilitar o botão exibir;
5. Habilitar o botão limpar.

- Botão "Limpar":

1. Limpar as caixas de texto (nomes e idades);
2. Limpar os labels com os resultados;
3. Habilitar o botão Exibir;
4. Desabilitar o botão limpar;
5. Colocar o foco na primeira caixa de texto.

Solução do exercício:

```
using System;
using System.Windows.Forms;

namespace Exe005
{
   public partial class FrmExe005 : Form
   {
      //Declaração das variáveis
      string nome1 = "";
      string nome2 = "";
      byte idade1 = 0;
      byte idade2 = 0;


      public FrmExe005()
      {
```

```
    InitializeComponent();
}

private void BtnSair_Click(object sender, EventArgs e)
{
    MessageBox.Show("Volte Sempre!!!", "Obrigado");
    Application.Exit();
}

private void FrmExe005_Load(object sender, EventArgs e)
{
    timer1.Enabled = true;
    timer1.Interval = 1000; //evento ocorre a cada 1 segundo
    LblData.Text = DateTime.Now.ToString("dd/MM/yyyy");
    LblDia.Text = DateTime.Now.ToString("dddd").ToUpper();
    LblHora.Text = DateTime.Now.ToString("HH/mm/ss");
}

private void timer1_Tick(object sender, EventArgs e)
{
    LblHora.Text = DateTime.Now.ToString("HH/mm/ss");
}

private void BtnExibir_Click(object sender, EventArgs e)
{
    if ((TxtNome1.Text == "") ||
        (TxtIdade1.Text == "") ||
        (TxtNome2.Text == "") ||
        (TxtIdade2.Text == ""))
    {
        MessageBox.Show("Preencha TODOS os campos!!", "ATENÇÃO");
        TxtNome1.SelectAll();
        TxtNome1.Focus();
    }
    else
    {
        nome1 = TxtNome1.Text;
        idade1 = Convert.ToByte(TxtIdade1.Text);

        nome2 = TxtNome2.Text;
        idade2 = Convert.ToByte(TxtIdade2.Text);

        if (idade1 == idade2)
        {
            MessageBox.Show("Alunos com a mesma idade!!!", "ATENÇÃO");
            BtnExibir.Enabled = false;
            BtnLimpar.Enabled = true;
        }
        else if (idade1 < idade2)
        {
            LblNomeMenor.Text = nome1;
```

```
            LblIdadeMenor.Text = idade1.ToString();
            BtnExibir.Enabled = false;
            BtnLimpar.Enabled = true;
          }
          else
          {
            LblNomeMenor.Text = nome2;
            LblIdadeMenor.Text = idade2.ToString();
            BtnExibir.Enabled = false;
            BtnLimpar.Enabled = true;
          }
        }
      }

      private void BtnLimpar_Click(object sender, EventArgs e)
      {
        TxtNome1.Text = "";
        TxtIdade1.Text = "";
        TxtNome2.Text = "";
        TxtIdade2.Text = "";
        LblNomeMenor.Text = "";
        LblIdadeMenor.Text = "";
        nome1 = "";
        nome2 = "";
        idade1 = 0;
        idade2 = 0;
        BtnExibir.Enabled = true;
        BtnLimpar.Enabled = false;
        TxtNome1.Focus();
      }

      private void TxtIdade1_KeyPress(object sender, KeyPressEventArgs e)
      {

        //faixa dos números: 48 a 57
        if (e.KeyChar != 08 && (e.KeyChar < 48 || e.KeyChar > 57))
        {
          e.KeyChar = (char)0; //anula a tecla digitada --> 0 = nulo
        }
      }
    }
}
```

Considerações sobre a solução proposta:

1) A estrutura condicional if, apresenta uma modalidade em que incluímos, em um único "if" várias condições (chamamos de condição composta):

```
if ((TxtNome1.Text == "") || (TxtIdade1.Text == "") || (TxtNome2.Text == "") || (TxtIdade2.Text == ""))
        {
          MessageBox.Show("Preencha TODOS os campos!!", "ATENÇÃO");

        }
```

O símbolo "||" (duas barras verticais) indica o operador OU.

Perceba que incluímos no mesmo "if" todas as condições de teste possíveis para as caixas de texto.

A instrução "MessageBox.Show..." será executa se UMA das condições forem verdadeiras, isto é, se o TxtNome1 estiver em branco OU se o TxtIdade1 estiver em branco OU se o TxtNome2 estiver em branco OU se o TxtIdade2 estiver em branco, a mensagem de alerta será exibida.

A execução a partir da linha 42, somente será executada SE TODOS OS CAMPOS ESTIVEREM PREENCHIDOS!!!

Outra consideração também importante, são as linhas em que atribuímos para as variáveis "idade" o conteúdo dos TextBox.

Considerando que um TextBox SEMPRE conterá uma informação do tipo "texto", as instruções de "conversão de tipo" SERÃO OBRIGATÓRIAS. No caso em questão, precisaremos converter o texto em uma informação numérica do tipo byte. A instrução que realiza esta operação é a Convert.ToByte. (conheceremos outras ao longo do curso).

Situação de erro!!!

Iniciamos, apartir deste projeto, o trabalho com informações numéricas e isto poderá causar em nossos aplicativos alguns erros que precisarão ser corrigidos, por exemplo.

Execute o projeto e digite na primeira idade alguma letra. Preencha os demais campos e clique em EXIBIR.

O projeto será "abortado" (encerrado) e uma mensagem de erro será exibida, algo do tipo:

System.FormatException: 'A cadeia de caracteres de entrada não estava em um formato correto.'

O erro ocorreu exatamente onde temos a instrução:

```
idade1 = Convert.ToByte(TxtIdade1.Text);
```

Não podemos deixar que isto ocorra nos aplicativos, os possíveis erros deverão ser tratados e JAMAIS um aplicativo deverá ser finalizado com uma operação de erro!!

Temos um capítulo específico sobre "Tratamentos de erro", através da estrutura "try...catch" e será estudado mais adiante.

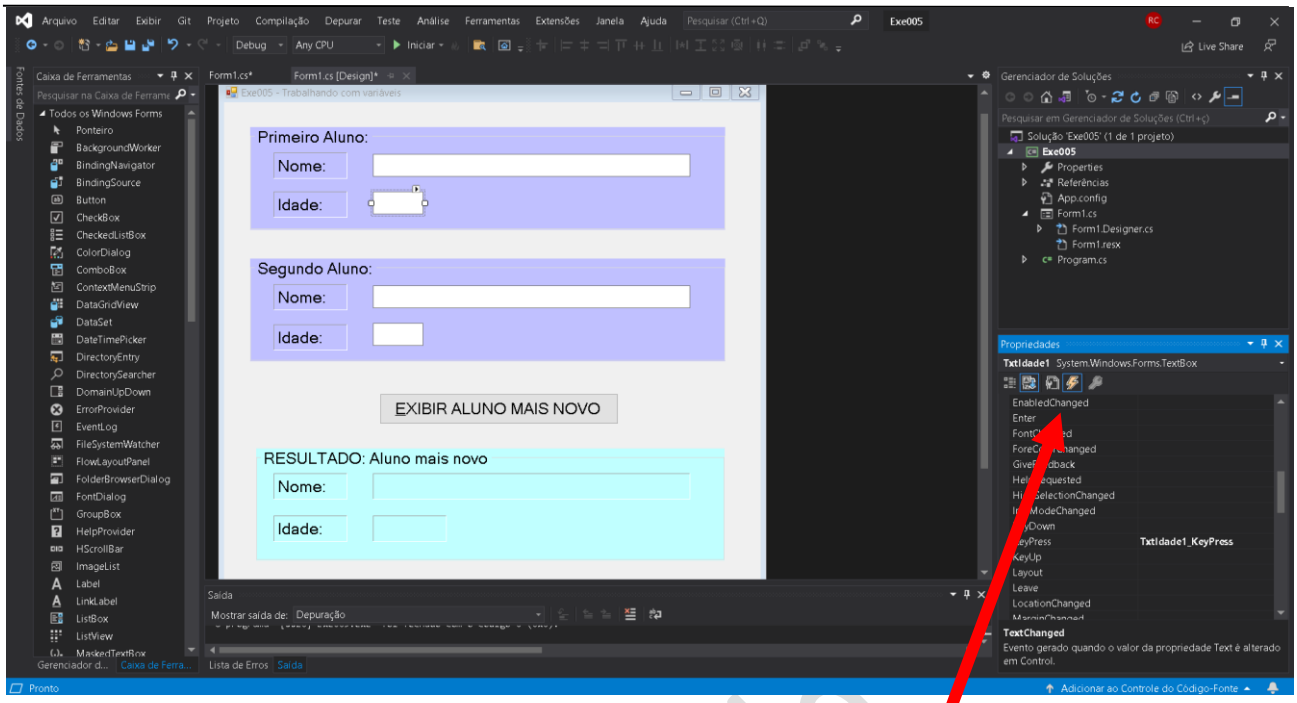
Mas podemos inserir uma programação específica, para impedir a digitação de textos em um TextBox que deva receber somente números, vamos conhecer??

O objetivo é impedir a digitação de qualquer tecla que não seja numérica.

A programação deverá ser digitada em um evento novo...

Até este momento estudamos apenas o evento Click de um botão, agora estudaremos o evento "KeyPress" (pressionamento de tecla). Este evento ocorre quando digitamos algo em um texto e estará relacionado com as caixas de texto "TxtIdade".

Em modo de "design" de projeto, de um único clique na caixa de texto idade (deixa-la apenas selecionada):



Na janela de "propridedades" (que está no lado direito) clique no símbolo de raio (que está em amarelo). A janela de propriedades é alterada para "janela de eventos". Localize o evento "keypress" e dê DOIS CLIQUES sobre ele. O Visual Studio entrará no modo de CODE (código), especificamente no bloco de código do KeyPress. Digite a programação abaixo:

```
private void TxtIdade1_KeyPress(object sender, KeyPressEventArgs e)
    {
      if (!char.IsDigit(e.KeyChar) && e.KeyChar != 08)
      {
        //Atribui True no Handled para cancelar o evento
        e.Handled = true;
      }


    }
```

Ou, se preferir, substitua pelo código abaixo, que terá o mesmo resultado:

```
            //faixa dos números: 48 a 57
        if (e.KeyChar != 08 && (e.KeyChar < 48 || e.KeyChar > 57))
        {
          e.KeyChar = (char)0; //anula a tecla digitada --> 0 = nulo

        }
```

Em uma tradução "livre", podemos interpretar a estrutura "if" da seguinte maneira:
se o caracter pressionando não for digito e também não for a tecla "backspace" (retorno)
{
Ignore a tecla digitada;
}

Execute o projeto e faça a tentativa de digitar qualquer coisa que não seja número nos TextBox da idade...

Códigos das teclas:

8 → BackSpace

13 → Tecla <ENTER>

32 → Espaço

48 a 57 → Digitos de 0 a 9