Aula 02 – Componentes Avançados do WinForms (Parte 1)

Nesta segunda aula, daremos continuidade ao aprendizado em C# com Windows Forms. Na primeira aula, conhecemos o ambiente do Visual Studio, exploramos o Designer de Formulários e trabalhamos com Label e Button para criar uma aplicação simples. Agora, avançaremos com novos componentes e construiremos aplicações práticas para consolidar o aprendizado.

1. Revisão da Aula Anterior

Na aula anterior, criamos nossa primeira aplicação WinForms utilizando os componentes Label e Button. Exploramos o conceito de eventos e vimos como manipular o evento Click de um botão.

2. Conhecendo Novos Componentes

O Windows Forms possui uma grande variedade de componentes que podem ser utilizados para criar interfaces gráficas mais ricas e interativas. Nesta aula, iremos explorar os seguintes:

- TextBox: Permite a entrada de texto pelo usuário.
- ComboBox: Permite a seleção de opções em uma lista suspensa.
- RichTextBox: Caixa de texto avançada que aceita formatação.
- Panel: Contêiner que ajuda a organizar outros componentes.
- Button: Botão de ação, já conhecido, mas agora usado em contextos mais elaborados.

TextBox

Propriedade	Descrição
Name	Identificação do componente no código
Text	Texto exibido ou digitado no campo
Multiline	Permite entrada de múltiplas linhas
PasswordChar	Caractere para ocultar senha
MaxLength	Número máximo de caracteres
ReadOnly	Define se é somente leitura
Enabled	Ativa ou desativa o campo
Visible	Define se o campo está visível

ComboBox

Propriedade	Descrição
Name	Identificação no código
Text	Texto ou item selecionado
Items	Lista de itens disponíveis
SelectedIndex	Índice do item selecionado
DropDownStyle	Estilo de exibição da lista
MaxDropDownItems	Máximo de itens exibidos na lista
Sorted	Define se os itens devem ser ordenados

RichTextBox

Propriedade	Descrição
Name	Identificação no código
Text	Texto simples exibido
Rtf	Texto com formatação RTF
Multiline	Permite múltiplas linhas
ReadOnly	Define se é somente leitura
Font	Fonte do texto
SelectionStart	Posição inicial da seleção
SelectionLength	Tamanho da seleção de texto
SelectionFont / SelectionColor	Fonte e cor do texto selecionado

Panel

Propriedade	Descrição
Name	Identificação no código
Controls	Coleção de controles dentro do Panel
BorderStyle	Estilo da borda
BackColor	Cor de fundo
Dock	Define acoplamento no formulário
Enabled	Ativa ou desativa todos os controles internos
Visible	Define se o Panel está visível

Button

Propriedade	Descrição
Name	Identificação no código
Text	Texto exibido no botão
Enabled	Ativa ou desativa o botão
Visible	Define se aparece na tela
BackColor	Cor de fundo
ForeColor	Cor do texto
Image	Imagem exibida no botão
DialogResult	Resultado usado em diálogos
FlatStyle	Estilo visual do botão

3. Exemplo Prático: Gerador de Recibo

Vamos criar uma aplicação simples para geração de recibos. Este exemplo nos permitirá manipular vários componentes ao mesmo tempo.

Passos:

1. Crie um novo projeto Windows Forms no Visual Studio.
2. Insira os seguintes componentes: Panel, Label, TextBox, ComboBox, RichTextBox e Button.
3. Configure as propriedades dos componentes da seguinte forma:

- TextBox: txtPagador, txtRecebedor, txtValor, txtExtenso
- ComboBox: cmbReferente (adicione itens: 'Serviços prestados em treinamento', 'Suporte técnico', 'Serviços prestados no desenvolvimento de sistemas')
- Button: btnGerar (Texto: 'Gerar')
- RichTextBox: ricTexto

4. 4. Dê um duplo clique no botão Gerar e adicione o seguinte código:

```
// Limpando o RichText
ricTexto.Clear();

string linha;
const string branco = "\n";

linha = "R E C I B O " + branco;
ricTexto.AppendText(linha);
```

```
linha = "Eu " + txtRecebedor.Text + ", recebi a importância de " +
txtValor.Text + " (" + txtExtenso.Text + "), referente a(o) " +
cmbReferente.Text + "." + branco;
ricTexto.AppendText(linha);

ricTexto.AppendText("-------------------" + branco);
ricTexto.AppendText(txtRecebedor.Text);

ricTexto.SelectionStart = 0;
ricTexto.SelectionLength = 11;
ricTexto.SelectionFont = new Font(ricTexto.SelectionFont, FontStyle.Bold);
```

4. Exemplo Prático: Diferença entre Datas

Outro componente útil do Windows Forms é o **DateTimePicker**, que permite ao usuário selecionar datas. Junto com as classes DateTime e TimeSpan, podemos realizar cálculos com datas.

DateTimePicker

Propriedade	Descrição
Name	Identificação no código
Value	Data/hora atualmente selecionada
Format	Formato de exibição (Short, Long, Time, Custom)
CustomFormat	Formato personalizado de data/hora
MinDate	Data mínima permitida
MaxDate	Data máxima permitida
ShowUpDown	Exibe setas para ajuste em vez do calendário
Checked	Indica se um valor foi selecionado
Enabled	Ativa ou desativa o controle
Visible	Define se aparece na tela

- Classe DateTime

Representa uma data e hora no .NET.
Permite manipular datas, horas e realizar cálculos.

Principais propriedades

- **Now** → retorna a data e hora atual do sistema.
- **Today** → retorna apenas a data atual (hora zerada).

- **UtcNow** → retorna a data/hora atual em UTC.
- **Day, Month, Year** → obtêm partes da data.
- **Hour, Minute, Second** → obtêm partes do horário.
- **DayOfWeek** → retorna o dia da semana (enum).

Principais métodos

- **AddDays(double)** → adiciona dias.
- **AddMonths(int)** → adiciona meses.
- **AddYears(int)** → adiciona anos.
- **AddHours/AddMinutes** → adiciona tempo.
- **ToShortDateString() / ToLongDateString()** → formata a data.
- **Compare(DateTime d1, DateTime d2)** → compara duas datas.

Exemplo:

```
DateTime hoje = DateTime.Now;

DateTime futuro = hoje.AddDays(7);

MessageBox.Show("Hoje: " + hoje.ToShortDateString() +

        "\nDaqui 7 dias: " + futuro.ToShortDateString());
```

Obs: O MessageBox.Show é um dos métodos mais usados em WinForms para exibir caixas de mensagem ao usuário. Ele possui sobrecargas (várias versões do mesmo método, aceitando diferentes parâmetros).

➔ Forma mais simples: somente texto

```
MessageBox.Show("Olá, mundo!");
```

➔ Texto + título:

```
MessageBox.Show("Olá mundo", "Hello World");
```

➔ Texto + título + botão:

```
MessageBox.Show("Deseja salvar as alterações?", "Salvar",
MessageBoxButtons.YesNoCancel);
```

➔ Texto + título + botão + ícone:

```
MessageBox.Show("Deseja salvar as alterações?", "Salvar",
MessageBoxButtons.YesNoCancel,
MessageBoxIcon.Question);
```

➔ Texto + Título + Botões + Ícones + Botão Padrão

```
MessageBox.Show("Deseja salvar as alterações?", "Salvar",
MessageBoxButtons.YesNo, MessageBoxIcon.Question,
MessageBoxDefaultButton.Button2);
```

O MessageBox.Show retorna um valor do tipo **DialogResult**, que indica qual botão o usuário clicou. Por exemplo:

```
DialogResult resposta = MessageBox.Show("Deseja sair da aplicação?",
"Confirmação",MessageBoxButtons.YesNo,MessageBoxIcon.Question);
if (resposta == DialogResult.Yes)
{
   Application.Exit();
}
```

- Classe TimeSpan

Representa um **intervalo de tempo** (diferença entre duas datas/horas).

Principais propriedades

- **Days, Hours, Minutes, Seconds** → partes do intervalo.
- **TotalDays, TotalHours, TotalMinutes** → valor total em unidades.

Principais métodos

- **FromDays(double), FromHours(double), etc.** → cria um intervalo a partir de um número.
- **Add(TimeSpan)** → soma intervalos.
- **Subtract(TimeSpan)** → subtrai intervalos.

Exemplo1:

```
DateTime inicio = new DateTime(2025, 1, 1);
```

```
DateTime fim = DateTime.Now;

TimeSpan diferenca = fim - inicio;

MessageBox.Show("Dias desde 01/01/2025: " + diferenca.Days);
```

Exemplo2:

```
// Definindo horários de início e fim

DateTime inicio = new DateTime(2025, 8, 28, 18, 30, 0);  // 18:30

DateTime fim    = new DateTime(2025, 8, 28, 22, 45, 0); // 22:45

// Calculando a diferença

TimeSpan duracao = fim - inicio;

// Mostrando resultado

MessageBox.Show("Diferença em horas: " + duracao.TotalHours.ToString("F2")
+

        "\nHoras inteiras: " + duracao.Hours +

        "\nMinutos restantes: " + duracao.Minutes);
```

Obs: O que significa "F2"?

- **F** → significa *Fixed-point* (ponto fixo, número decimal).
- **2** → indica a **quantidade de casas decimais**.

Ou seja, "F2" formata o número para ter **2 casas decimais**, arredondando se necessário.

Exemplo prático:

Passos:

5. 1. Crie um novo projeto Windows Forms.
6. 2. Adicione dois DateTimePickers (dtIni e dtFim), um Label e um Button (Texto: 'Diferença').

7. 3. No evento Click do botão, insira o seguinte código:

```
private void btnDiferenca_Click(object sender, EventArgs e)
{
    DateTime dtini = dtIni.Value;
    DateTime dtfim = dtFim.Value;

    TimeSpan ts = dtfim - dtini;
    label1.Text = ts.Days.ToString() + " dias de diferença";
}
```

5. Atividades Práticas

Atividade 1: Desenvolva uma aplicação que calcule o número de dias entre sua data de nascimento e a data atual.

Atividade Extra: Modifique a aplicação do recibo para salvar o texto gerado em um arquivo .txt.

Gabarito atividade extra:

```
using System;
using System.IO; //Adicione essa biblioteca para escrever ou ler arquivos
using System.Text;
using System.Windows.Forms;

//Acrescente esses método Click a um botão btnSalvar no app do recibo:
private void btnSalvar_Click(object sender, EventArgs e)
{
    if (string.IsNullOrWhiteSpace(ricTexto.Text))
    {
        MessageBox.Show("Gere o recibo antes de salvar.", "Atenção",
            MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return;
    }

    try
    {
        // Obtém caminho da pasta Documentos do usuário
        string pastaDocs =
Environment.GetFolderPath(Environment.SpecialFolder.MyDocuments);

        // Gera nome único para o arquivo (com data/hora)
        string nomeArquivo = $"recibo_{DateTime.Now:yyyyMMdd_HHmmss}.txt";

        // Caminho completo
        string caminhoCompleto = Path.Combine(pastaDocs, nomeArquivo);

        // Grava o conteúdo do RichTextBox como texto simples
        File.WriteAllText(caminhoCompleto, ricTexto.Text, Encoding.UTF8);

        MessageBox.Show($"Recibo salvo em:\n{caminhoCompleto}", "Sucesso",
            MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
    catch (Exception ex)
    {
        MessageBox.Show("Erro ao salvar o arquivo:\n" + ex.Message, "Erro",
            MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

Obs: **O que acontece aqui:**

- Environment.SpecialFolder.MyDocuments → pega a pasta **Documentos** do usuário logado.

- DateTime.Now:yyyyMMdd_HHmmss → gera nome único (ex.: recibo_20250828_104512.txt).
- File.WriteAllText → grava direto o conteúdo do RichTextBox.