Visual Studio .NET - C#

Parte 2 – Windows Forms e Web Services

Cláudio Luís Vieira Oliveira
prof.claudioluis@fatec.sp.gov.br

1

Visual Studio .NET – C#

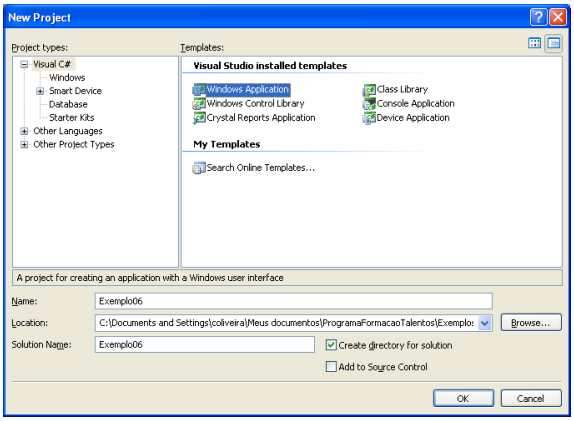
- **Conteúdo**
  - Aplicações para o Windows
  - Web Service

2

Visual Studio .NET – C#

Aplicações para o Windows

Outro tipo aplicação que pode ser criado com o Visual Studio .NET é a “Windows Application”.



Visual Studio .NET – C#

Formulário e controles básicos

Nas aplicações para Windows as principais diferenças em relação às aplicações para a Web são:

- maior controle sobre o lay-out;
- uso da propriedade Name que substitui a propriedade Id, utilizada nos projetos de web sites para identificar os controles.

Com o intuito de ilustrar os conceitos básicos de formulário e controles será construída uma aplicação que realizará a conversão entre temperaturas conforme ilustrado pela tabela a seguir:

Visual Studio .NET – C#

Fórmulas para conversão de temperatura

De	Para	Fórmula
Celsius	Fahrenheit	°F = °C × 1,8 + 32
Fahrenheit	Celsius	°C = (°F − 32) / 1,8
Celsius	Kelvin	K = °C + 273,15
Kelvin	Celsius	°C = K − 273,15
Fahrenheit	Kelvin	K = (°F + 459,67) / 1,8
Kelvin	Fahrenheit	°F = K × 1,8 - 459,67

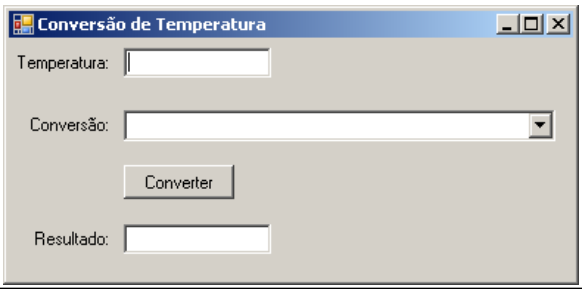
5

Visual Studio .NET – C#

Formulário e controles básicos

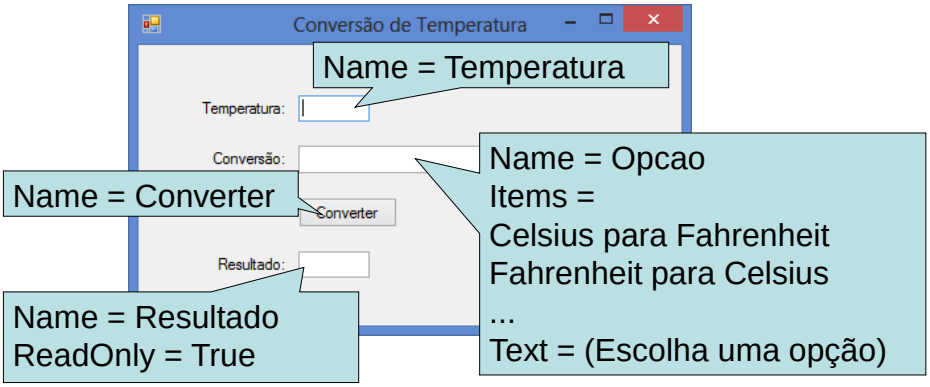
No formulário definir as propriedades:
- Name = Conversor
- Text = Conversão de Temperaturas
- StartPosition = CenterScreen

Em seguida adicionar os controles conforme mostrado na figura:



Visual Studio .NET – C#

Configuração dos controles



7

Visual Studio .NET – C#

Programação dos eventos

O funcionamento desta aplicação consiste no preenchimento do campo temperatura, escolha da conversão desejada e, ao clicar sobre o botão converter, o resultado da conversão deverá ser exibido no campo resultado. Desta forma, o evento a ser programado é o clique sobre o botão Converter:

```
private void Converter_Click(object s, EventArgs e)
{

}
```

8

Visual Studio .NET – C#

Obter os valores digitados

O primeiro passo consiste em realizar a leitura dos valores digitados pelo usuário:

```
private void Converter_Click(object s, EventArgs e)
{ double temperatura =
    Convert.ToDouble(Temperatura.Text);
  int opcao = Opcao.SelectedIndex;
  double resultado = 0;
}
```

9

Visual Studio .NET – C#

Escolha da fórmula que deverá ser usada

Após identificar a opção de conversão que foi escolhida pelo usuário (`Opcao.SelectedIndex`) torna-se necessário definir qual a fórmula que deverá ser usada, por exemplo:

```
...
switch (opcao)
{
  case 0:
    // Celsius para Fahrenheit
    resultado = temperatura * 1.8 + 32;
    break;
  case 1:
    // Fahrenheit para Celsius
...
```

10

Visual Studio .NET – C#

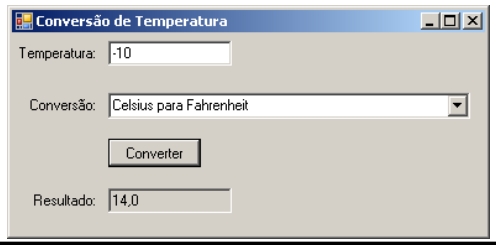
Exibição do resultado

Concluindo a aplicação será necessário exibir o resultado na respectiva caixa de texto:

```
Resultado.Text = resultado.ToString("f1");
```

Ou ainda:

```
Resultado.Text = resultado.ToString("##0.0");
```

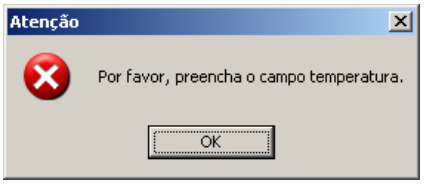


Visual Studio .NET – C#

Caixas de Mensagem

As caixas de mensagem são um importante recurso que pode ser usado no tratamento de erros e envio de informações para o usuário. No exemplo a seguir será mostrada uma caixa de mensagem se o usuário não preencher o campo temperatura:

```
if (Temperatura.Text == "")
  MessageBox.Show("Por favor, preencha o campo
    temperatura.",
    "Atenção",
    MessageBoxButtons.OK,
    MessageBoxIcon.Stop);
```



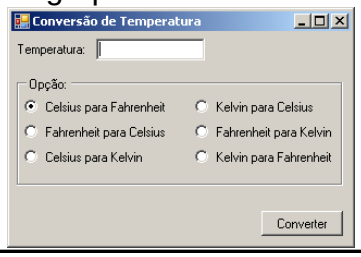
Visual Studio .NET – C#

Exercícios

1)Exibir uma caixa de mensagem se o usuário não realizar a escolha da conversão desejada.

2)Alterar o programa de conversão de temperatura para exibir o resultado em uma caixa de mensagem ao invés da caixa de texto.

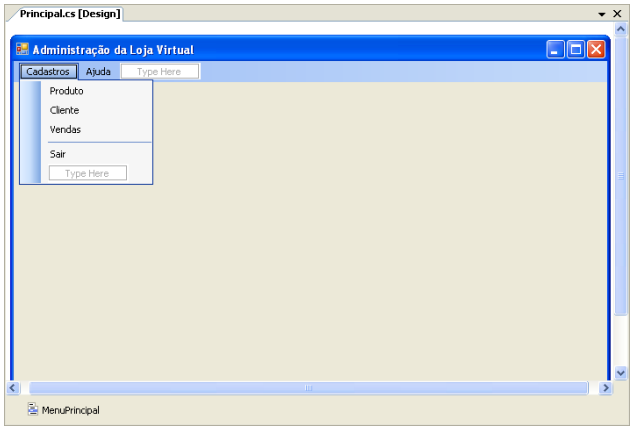
3)Substituir a caixa de seleção por botões de rádio. Dica: Utilizar o controle GroupBox para agrupar os botões de rádio utilizados.



Visual Studio .NET – C#

Formulário principal e menu de acesso

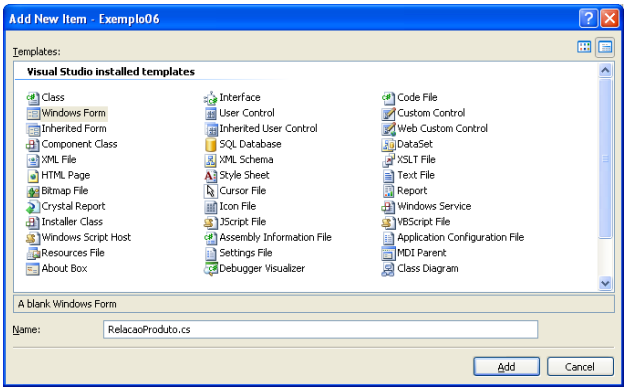
Criar um novo projeto e adicionar ao formulário o componente MenuStrip e inserir os itens que irão compor o menu:



Visual Studio .NET – C#

Formulário com a relação de produtos

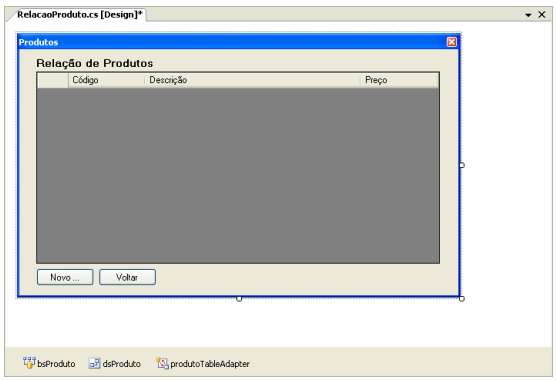
A seguir será criado um novo formulário que terá como objetivo exibir a relação dos produtos que estão cadastrados no banco de dados.



Visual Studio .NET – C#

Formulário com a relação de produtos

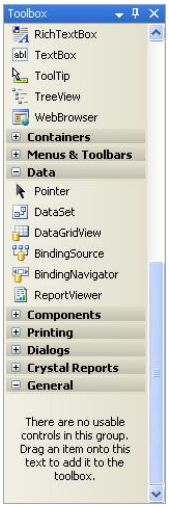
A seguir será criado um novo formulário que terá como objetivo exibir a relação dos produtos que estão cadastrados no banco de dados.



Visual Studio .NET – C#

Formulário com a relação de produtos

Os componentes principais deste formulário são aqueles que permitem a integração da aplicação com o banco de dados, ou seja:



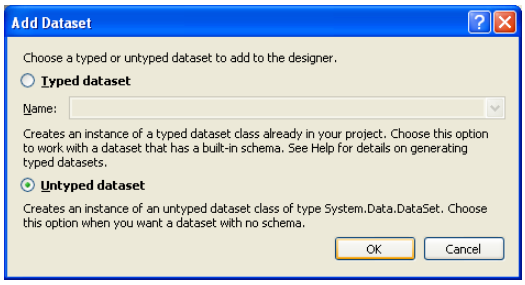
DataSet – Realiza a conexão com o banco de dados;
BindingSource – Atribuído ao DataSet obtém os dados de um objeto do banco de dados, normalmente uma tabela ou uma visão;
DataGridView – Conecta-se ao BindingSource criado e faz a exibição dos dados obtidos pelo mesmo.

17

Visual Studio .NET – C#

Criação do DataSet

Após o componente ser inserido ao formulário o mesmo deverá ser configurado para utilizar a opção Untyped dataset.
Em seguida deve-se definir os atributos Name e DataSetName como dsProduto.

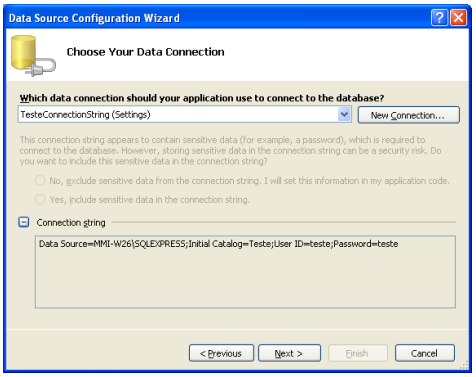


Visual Studio .NET – C#

Criação do BindingSource

Inserir o componente BindingSource, ajustar o atributo name para bsProduto, para o atributo DataSource escolher a opção Add Project Data Source, selecionar Database e pressionar o botão Next.

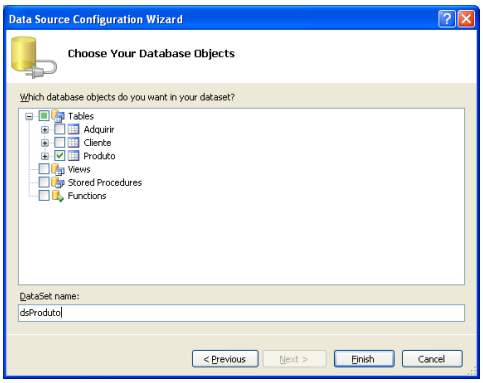
Clicar no botão New Connection ou escolher uma string de conexão já existente, depois pressionar o botão Next.



Visual Studio .NET – C#

Criação do BindingSource

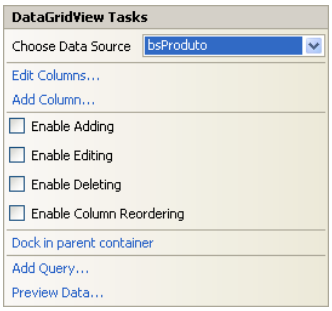
Na pasta Tables escolher Produto e definir o DataSet name como dsProduto. Depois na janela de Properties colocar o valor Produto para o atributo DataMember e Codigo para o atributo Sort.



Visual Studio .NET – C#

Configuração do DataGridView

Colocar no formulário o componente DataGridView e escolher o Data Source bsProduto. Na janela Properties atribuir o valor GridProduto para o atributo Name.



21

Visual Studio .NET – C#

Acesso ao formulário através do menu

No formulário Principal deve-se programar o evento do menu que irá abrir o formulário RelacaoProduto:

```
private void ProdutoMenuItem_Click(object sender, EventArgs
e)
{ RelacaoProduto rp = new RelacaoProduto();
  rp.ShowDialog(this);
}
```

Já no formulário RelacaoProduto o evento clique no botão Voltar deverá fechar o formulário:

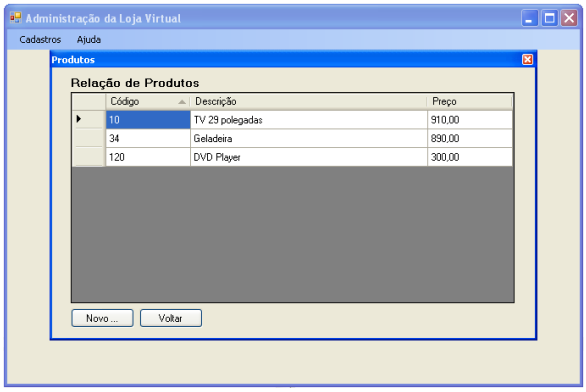
```
private void Voltar_Click(object sender, EventArgs e)
{ this.Close();
}
```

22

Visual Studio .NET – C#

Testando a aplicação

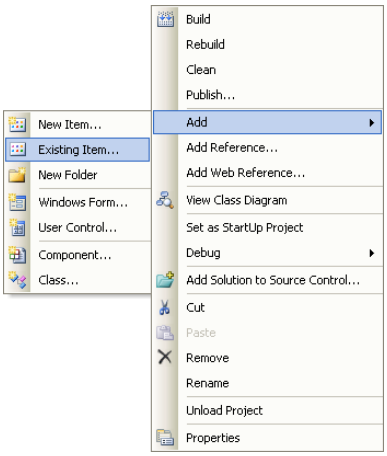
Neste momento já é possível realizar um teste da aplicação e os registros existentes na tabela Produto deverão ser mostrados no DataGridView:



Visual Studio .NET – C#

Reutilização de código

As classes Produto e AcessoDadosProduto criadas no projeto da aplicação Web deverão ser reutilizadas neste projeto com o intuito de prover os métodos necessários para realizar a manutenção na tabela de Produtos. Isso pode ser feito na Solution Explorer clicando com o botão direito do mouse sobre o nome do projeto e escolhendo as opções Add e Existing Item ...

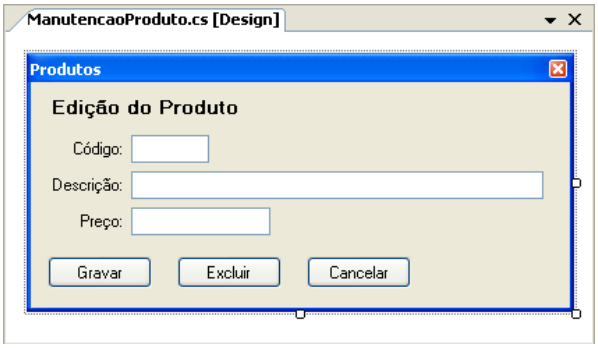


24

Visual Studio .NET – C#

Formulário para manutenção dos produtos

Agora deverá ser criado o formulário ManutencaoProduto que será responsável pelas tarefas de inclusão, alteração e exclusão dos registros:



Visual Studio .NET – C#

Abertura do formulário de manutenção

A chamada do formulário criado será realizada através do clique em uma linha das linhas da DataGridView ou através do botão Novo, ambos existentes no formulário RelacaoProduto. O método AbrirManutençãoProduto terá a função de realizar a abertura do formulário:

```
private void AbrirManutencaoProduto(int codigo)
{
  ManutencaoProduto mp = new ManutencaoProduto(codigo);
  mp.ShowDialog(this);
}
```

Visual Studio .NET – C#

Abertura do formulário de manutenção

Em seguida serão programados os eventos no botão Novo:

```
private void Novo_Click(object sender, EventArgs e)
{
  AbrirManutencaoProduto(-1);
}
```


Visual Studio .NET – C#

Seleção do produto escolhido

... e na DataGridView:

```
private void GridProduto_Click(object sender,
DataGridViewCellEventArgs e)
{ int linhasSelecionadas = GridProduto.Rows.
    GetRowCount(DataGridViewElementStates.Selected);
  if (linhasSelecionadas > 0)
  { int codigo = -1;
    for (int i = 0; i < linhasSelecionadas; i++)
    { codigo = Convert.ToInt32(GridProduto.SelectedRows[i].
        Cells[0].Value.ToString());
    }
    AbrirManutencaoProduto(codigo);
  }
}
```

Visual Studio .NET – C#

Manutenção de produtos

No formulário de manutenção do produto serão criados dois atributos que terão a função de armazenar o código do produto e a operação, além de ajustes no método construtor:

```
public partial class ManutencaoProduto : Form
{ private int codigo;
  private string operacao;

  public ManutencaoProduto(int _codigo)
  { InitializeComponent();
    codigo = _codigo;
    if (codigo == -1)
      operacao = "i";
    else
      operacao = "a";
  }
  ...
```

29

Visual Studio .NET – C#

Manutenção de produtos

Quando o formulário é aberto para alteração ou exclusão (codigo diferente de -1), o valores dos campos deverão ser exibidos nas respectivas caixas de texto:

```
private void ManutencaoProduto_Load(object sender, EventArgs
e)
{
  if (codigo != -1)
  {
    Codigo.Text = codigo.ToString();
    Produto produto = AcessoDadosProduto.
      obterProduto(codigo);
    Descricao.Text = produto.Descricao;
    Preco.Text = produto.Preco.ToString("f2");
  }
}
```

30

Visual Studio .NET – C#

Inserção e alteração de produtos

O botão Gravar será responsável por obter os dados do formulário e enviá-los para o banco de dados utilizando, para isso, a classe AcessoDadosProduto:

```
private void Gravar_Click(object sender, EventArgs e)
{
  Produto produto = new Produto();
  produto.Codigo = Convert.ToInt32(Codigo.Text);
  produto.Descricao = Descricao.Text;
  produto.Preco = (float)Convert.ToDouble(Preco.Text);
  if (operacao == "i")
    AcessoDadosProduto.incluirProduto(produto);
  else
    AcessoDadosProduto.alterarProduto(produto);
  this.Close();
}
```

31

Visual Studio .NET – C#

Exclusão de um produto

Neste caso será utilizado o evento Click do botão Excluir, sendo que será solicitada através da MessageBox a confirmação do usuário:

```
private void Excluir_Click(object sender, EventArgs e)
{
  if (MessageBox.Show("Você tem certeza que deseja excluir
    este produto?", "Atenção", MessageBoxButtons.YesNo,
    MessageBoxIcon.Question) == DialogResult.Yes)
  {
    AcessoDadosProduto.excluirProduto(Convert.
      ToInt32(Codigo.Text));
    this.Close();
  }
}
```

32

Visual Studio .NET – C#

Atualização dos dados na DataGridView

Sempre que ocorrer qualquer manutenção na tabela através do formulário ManutencaoProduto, os dados exibidos na DataGridView deverão ser atualizados de forma a refletir as mudanças. Isso pode ser realizado através da utilização do evento Activated do formulário RelacaoProduto:

```
private void RelacaoProduto_Activated(object sender,
EventArgs e)
{
  this.produtoTableAdapter.Fill(this.dsProduto.Produto);
}
```


Visual Studio .NET – C#

Implementação dos outros cadastros

Conforme o modelo que foi criado para produtos, implementar as opções para realizar o cadastro de clientes e aquisições.

Visual Studio .NET – C#

Web Service

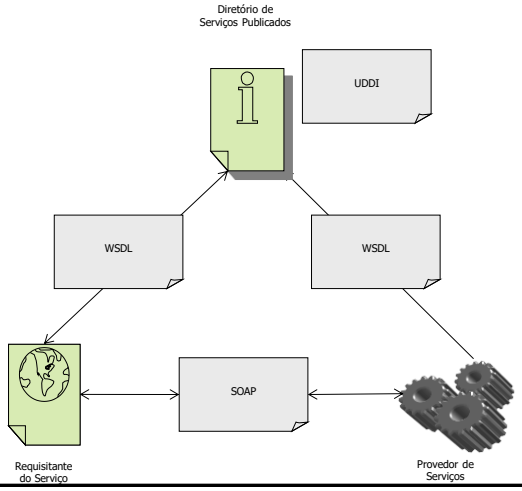
O termo **Arquitetura Orientada a Serviços** ou **Service-Oriented Architecture (SOA)** expressa um conceito onde, classes e métodos são disponibilizados como serviços dentro de uma rede de computadores, de forma independente e se comunicando através de protocolos fundamentados sobre padrões abertos. A maior parte das implementações de SOA utilizam o conceito de **Web Service.**

Web Service consiste em uma solução utilizada na integração de sistemas e na comunicação entre diferentes aplicações, permitindo a reutilização de código e a integração entre diferentes plataformas.

35

Visual Studio .NET – C#

Componentes de um Web Service



Visual Studio .NET – C#

Componentes de um Web Service

Para a representação e estruturação dos dados nas mensagens recebidas e enviadas é utilizada a linguagem XML. As chamadas às operações, incluindo os parâmetros de entrada e saída, são codificadas através da adoção do protocolo SOAP (Simple Object Access Protocol).

Os serviços disponibilizados, bem como as operações, mensagens e parâmetros entre outros aspectos, são descritos usando a linguagem WSDL (Web Services Definition Language). O processo de publicação, pesquisa e descoberta de Web Services dentro da rede utiliza o protocolo UDDI (Universal Description, Discovery and Integration).

37

Visual Studio .NET – C#

Criação de um Web Service

Para criar um Web Service no Visual Studio escolher a opção File, New e Web Site. Depois selecionar a opção ASP.NET Web Service.

```
[WebService(Namespace = "http://tempuri.org/")]
[WebServiceBinding(ConformsTo =
WsiProfiles.BasicProfile1_1)]
public class Service : System.Web.Services.WebService
{
  [WebMethod]
  public string Ola(string Nome)
  {
    return ("Olá, " + Nome);
  }
}
```

38

Visual Studio .NET – C#

Criação de um Web Service

Em uma visão simplificada um Web Service pode ser entendido como uma classe que tem os seus métodos publicados na Web.

```
[WebService(Namespace = "http://tempuri.org/")]
[WebServiceBinding(ConformsTo =
WsiProfiles.BasicProfile1_1)]
public class Service : S                    ebService
{
  [WebMethod]
  public string Ol(stri
  {
    return ("Olá, " + Nome);
  }
}
```

Determina que o método será acessível pela Web

Visual Studio .NET – C#

Consumindo um Web Service

Após a criação do Web Service o mesmo deverá ser publicado, sendo que a partir deste momento outras aplicações poderão utilizar os métodos do mesmo.

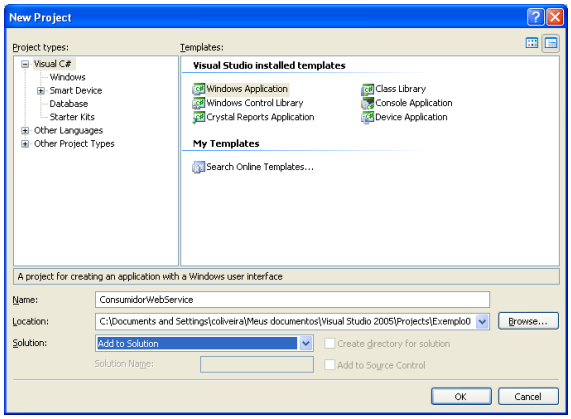
Neste exemplo, será criado um novo projeto para a Solution que consistirá em uma aplicação do tipo Windows Application que irá consumir o Web Service criado anteriormente.

Para criar este novo projeto, manter a Solution aberta e escolher File, New e depois Project ...

Visual Studio .NET – C#

Consumindo um Web Service

... selecionar Windows Application, definir o nome como ConsumidorWebService e, em seguida, escolher Add to Solution:



Visual Studio .NET – C#

Consumindo um Web Service

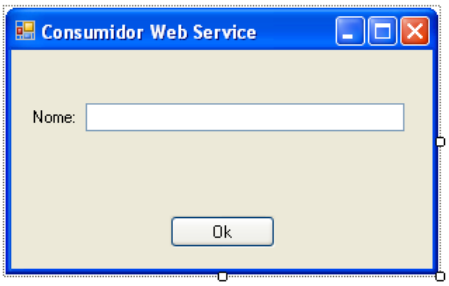
Criar um formulário semelhante ao mostrado abaixo, utilizando para isso:

Form1
- StartPosition = CenterScreen

Etiqueta
- Text = Nome

Caixa de texto
- Name = Nome

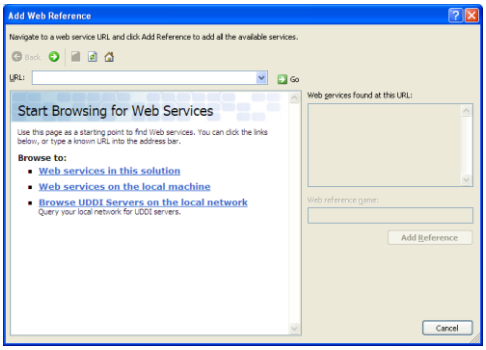
Botão:
- Name = Ok
- Text = Ok



Visual Studio .NET – C#

Consumindo um Web Service

Após a criação do formulário clicar com o botão direito do mouse sobre o nome do projeto e escolher o item Add Web Reference, na nova janela que foi aberta escolher a opção Web services in this solution ...

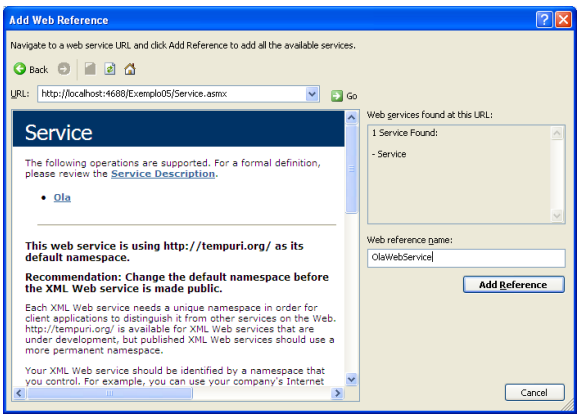


43

Visual Studio .NET – C#

Consumindo um Web Service

... escolher o nome da classe, ou seja Service, definir o nome da referência e clicar no botão Add Reference.

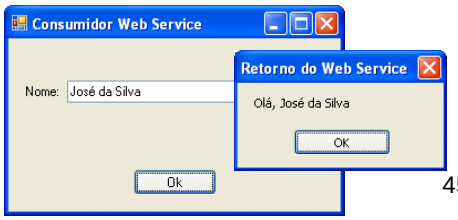


Visual Studio .NET – C#

Consumindo um Web Service

Utilizar o evento click do botão Ok para instanciar a classe do Web Service e realizar a chamada ao método Ola:

```
private void Ok_Click(object sender, EventArgs e)
{
  string resposta = "";
  OlaWebService.Service ows = new OlaWebService.Service();
  resposta = ows.Ola(Nome.Text);
  MessageBox.Show(resposta, "Retorno do Web Service");
}
```



45