

- [Canais](#)
- [Fóruns](#)
- [Multimídia](#)
- [Blogs](#)
- [Shop](#)
- [Eventos](#)
- [Pro](#)
- [Banco de Currículos](#)



Por uma internet mais criativa e dinâmica

[Faça Login](#) ou [Cadastre-se](#)


E-mail: Senha: ☐ Manter conectado

[Esqueci a senha](#)

- [Arquitetura e Design](#)
 - [3ds max](#)
 - [Acessibilidade](#)
 - [Arquitetura da Informação](#)
 - [Fireworks](#)
 - [Flash](#)
 - [Lightwave 3D](#)
 - [Photoshop](#)
 - [Suite Corel](#)
 - [Teoria/Design](#)
 - [Usabilidade](#)
- [Banco de Dados](#)
 - [Access](#)
 - [DB2](#)
 - [Interbase](#)
 - [MySQL](#)
 - [Oracle](#)
 - [PostgreSQL](#)
 - [SQL Server](#)
- [Carreira](#)
 - [Certificações](#)
 - [Mercado](#)
 - [Tendências](#)
- [CMS e Framework](#)
 - [CakePHP](#)
 - [Django](#)
 - [Drupal](#)
 - [Joomla](#)
 - [Ruby on Rails](#)
 - [Sistemas de E-commerce](#)
 - [Wordpress](#)
 - [Zend Framework](#)
- [Desenvolvimento](#)
 - [Agile](#)
 - [Ajax](#)
 - [Aplicativos móveis](#)
 - [Dreamweaver](#)
 - [Flash](#)
 - [Flex](#)

- [Gerência de Projetos](#)
 - [SEO](#)
 - [Software Livre](#)
 - [Visual FoxPro](#)
 - [Visual Studio](#)
 - [Web Standards](#)
 - [WebServices](#)
- [Gerência de TI](#)
 - [B. Intelligence](#)
 - [Computação Forense](#)
 - [Direito e Web](#)
 - [E-Gov](#)
 - [Governança de TI](#)
 - [Mercado](#)
- [Linguagens](#)
 - [.NET](#)
 - [ADO.NET](#)
 - [ASP.NET](#)
 - [C#.NET](#)
 - [VB.NET](#)
 - [ActionScript](#)
 - [ASP](#)
 - [ColdFusion](#)
 - [CSS](#)
 - [Delphi](#)
 - [Java](#)
 - [Java para Mobile](#)
 - [Java para web](#)
 - [Javascript](#)
 - [Ajax](#)
 - [JQuery](#)
 - [Prototype](#)
 - [Perl](#)
 - [PHP](#)
 - [Ruby](#)
 - [UML](#)
 - [Visual Basic](#)
 - [XHTML](#)
 - [XML](#)
- [Mídia e Marketing Digital](#)
 - [E-commerce](#)
 - [E-Learning](#)
 - [E-mail Marketing](#)
 - [Mídia Social](#)
 - [Mobile Marketing](#)
 - [Publicidade Online](#)
 - [SEO](#)
 - [Web Analytics](#)
 - [Web Marketing](#)
 - [Web Writing](#)
- [Redes e Servidores](#)
 - [Apache](#)
 - [Cisco](#)
 - [Linux](#)
 - [Segurança](#)
 - [Windows Server](#)
- [Tecnologia](#)
 - [Gadgets](#)
 - [TV Digital](#)
 - [VoIP](#)

[Desenvolvimento](#) + [.NET](#) + [VB.NET](#)

 [Feeds](#)  [Newsletter](#)

Quinta-feira, 03 de dezembro de 2009 às 11h15

NHibernate - usando o ActiveRecord - Parte 02



Conheça o curso ao vivo:

HTML5 na Prática, com Paulino Michelazzo.

Na [primeira parte deste artigo](#) eu apresentei o ActiveRecord. Agora vou mostrar como usar este recurso em um projeto Windows Forms usando a linguagem VB .NET.

Apenas recordando, o ActiveRecord é um recurso que facilita a utilização do NHibernate poupando o trabalho de criar os arquivos XML de mapeamento OR/M, para isso é criada uma classe onde usamos atributos para realizar esta tarefa.

Obs: O projeto **Castle ActiveRecord** oferece uma maneira rápida e fácil para implementar o padrão de registro ativo para aplicativos do Microsoft com base no .NET Framework.

Usando o ActiveRecord e o NHibernate em uma aplicação Windows Forms

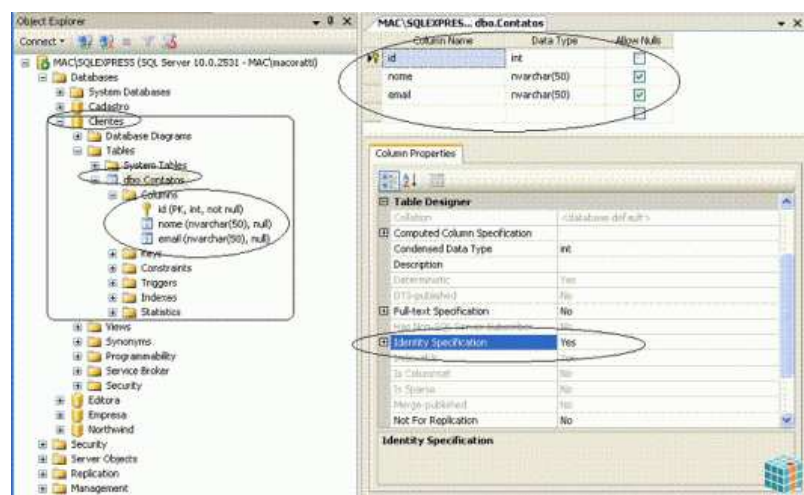
Para que você possa entender como implementar a utilização dos recursos do ActiveRecord em sua aplicação C# ou VB .NET, eu vou criar um exemplo usando a linguagem VB .NET no qual iremos realizar as seguintes tarefas:

- Criação do banco de dados e da tabela usando o SQL Server 2008 Express;
- Criação do projeto Windows Forms;
- Definição das referências para as DLLs do NHibernate e do ActiveRecord;
- Criação das classe para realizar o mapeamento;
- Definição do arquivo de configuração;
- Inicializando o Framework;
- Usando as classes na aplicação Windows Forms;

1. Criando o banco de dados e a tabela

A primeira coisa que eu vou fazer será criar o banco de dados Clientes e a tabela contatos no SQL Server 2008. Eu poderia criar as minhas classes de domínio e mandar gerar o banco de dados e as tabelas, mas por ser uma introdução vou seguir o caminho feliz.

Eu vou criar o banco de dados e a tabela usando o Microsoft SQL Server Management Studio. Abaixo vemos a estrutura da tabela definida com os campos : id, nome e email, sendo que id é um campo chave primária do tipo identity;



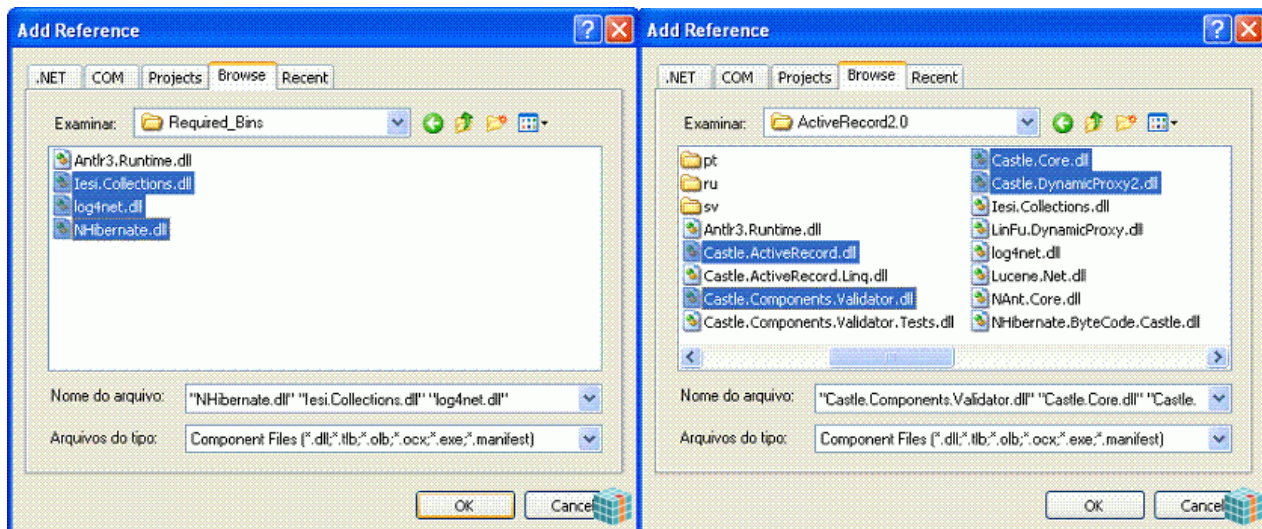
2. Criando o projeto Windows Forms e fazendo a referência ao NHibernate e ao ActiveRecord

Abra agora o Visual Basic 2008 Express Edition e crie um novo projeto do tipo Windows Forms Application com o nome NHibernateActiveRecordDemo;

A seguir clique com o botão direito do mouse sobre o nome do projeto e selecione a opção Add Reference:

Na guia Browse primeiro localize onde você instalou os arquivos do NHibernate e inclua as referências conforme a figura 1.0;

Em seguida repita a operação selecionando as dlls na pasta onde você instalou as dlls do ActiveRecord conforme a figura 2.0;



3. Criando a classe para efetuar o mapeamento OR/M

No menu Project selecione Add Class e informe o nome Contato.vb. Esta classe deverá herdar da classe ActiveRecordBase e usar os atributos do ActiveRecord para realizar o mapeamento OR/M para a tabela Contatos.

```
Imports Castle.ActiveRecord

<ActiveRecord("contatos")> _
Public Class Contato
    Inherits ActiveRecord(Of Contato)

    Private m_id As Integer
    Private m_nome As String
    Private m_email As String

    <PrimaryKey("Id")> _
    Public Property Id() As Integer
        Get
            Return (m_id)
        End Get
        Set(ByVal value As Integer)
            m_id = value
        End Set
    End Property

    <[Property]("Nome")> _
    Public Property Nome() As String
        Get
            Return (m_nome)
        End Get
        Set(ByVal value As String)
            m_nome = value
        End Set
    End Property

    <[Property]("Email")> _
    Public Property Email() As String
        Get
            Return (m_email)
        End Get
        Set(ByVal value As String)
            m_email = value
        End Set
    End Property
End Class
```

Neste arquivo usamos o *namespace* Castle.ActiveRecord para ter acesso às classes e aos atributos;

A classe Contato herda da classe ActiveRecordBase() e dessa forma poderá usar os métodos para mapear e persistir informações para o banco de dados;

Definimos 3 propriedades e usamos os seguintes atributos:

```
- <PrimaryKey("Id")> _
- <[Property]("Nome")> _
- <[Property]("Email")> _
```

Para mapear as propriedades para os campos da tabela Contatos.

Não definimos explicitamente o nome da tabela nem seu esquema, mas o ActiveRecord é capaz de inferir estas informações.

Observe que a classe Contato estende a classe ActiveRecord e usa atributos para sinalizar quais propriedades mapeiam para colunas na tabela de banco de dados e a coluna que serve como a chave primária.

A classe de ActiveRecord define métodos de instância, como criar e salvar e métodos estáticos incluindo Find, FindAll e DeleteAll.

Junto com a criação da classe Contatos, você também vai precisar definir algumas configurações de configuração, como a sequência de conexão de banco de dados e algumas poucas opções específicas de NHibernate.

Mas não há necessidade de escrever nenhum código de acesso a dados. que é realizado para você automaticamente pelo projeto de ActiveRecord e NHibernate.

Na verdade, você nem precisa ter criado as tabelas do banco de dados neste ponto, como o projeto ActiveRecord pode automaticamente gerar as tabelas com base nas classes que você criou. (Não vou usar este recursos neste exemplo.)

Obs: Nosso exemplo possui somente uma tabela, mas podemos trabalhar com relacionamentos entre tabelas. A sintaxe do atributo do projeto ActiveRecord também é usada para estabelecer relacionamentos entre classes. Por exemplo, os atributos HasMany e BelongsTo podem ser usados em uma classe pai e filho, respectivamente, para indicar uma relação um-para-muitos.

4. Criando o arquivo XML para definir as configurações usadas pelo NHibernate

Vamos agora criar o arquivo de configuração para o NHibernate com o nome AppConfig.xml conforme abaixo:

```
<?xml
version="1.0"
encoding="utf-8"
?>

<activerecord>

<config>

  <add

    key="connection.driver_class"

    value="NHibernate.Driver.SqlClientDriver" />

  <add

    key="dialect"

    value="NHibernate.Dialect.MsSql2005Dialect" />

  <add

    key="connection.provider"

    value="NHibernate.Connection.DriverConnectionProvider" />

  <add

    key="connection.connection_string"

    value="Data Source=.\SQLEXPRESS;Initial Catalog=Clientes;Integrated Security=SSPI" />

  <add

    key="proxyfactory.factory_class" />

    value="NHibernate.ByteCode.Castle.ProxyFactoryFactory, NHibernate.ByteCode.Castle"

  </add>

</config>

</activerecord>
```

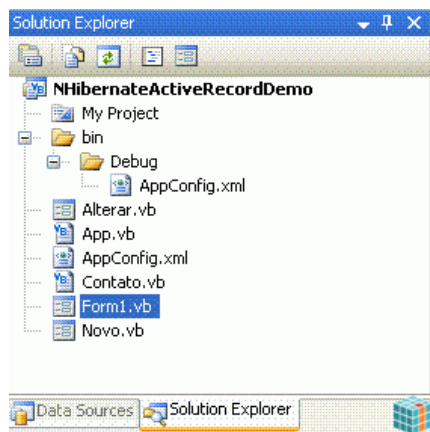
Neste arquivo estamos definindo as seguintes informações:

- O driver do banco de dados usados no projeto: key="connection.driver_class"
- O dialeto do banco de dados : key="dialect"
- O provedor da conexão : key="connection.provider"
- A string de conexão usada: key="connection.connection_string"
- O proxy : key="proxyfactory.factory_class" />

Estas informações referem-se ao banco de dados que estamos usando, no caso o SQL Server 2008 Express.

Este arquivo XML deve estar localizado na pasta bin/debug quando o projeto estiver sendo executado no modo de debug.

Neste momento o seu projeto deverá ter a seguinte estrutura:



Vamos partir para a criação da interface e utilização na prática do ActiveRecord.

5. Criando o arquivo para inicializar o framework

No menu Project selecione Add Class e informe o nome App.vb e inclua nele o seguinte código:

```
Imports Castle.ActiveRecord
Imports Castle.ActiveRecord.Framework
Imports Castle.ActiveRecord.Framework.Config

Public Class App

    Public Shared Sub Main()
        Dim source As XmlConfigurationSource = New XmlConfigurationSource("AppConfig.xml")
        ActiveRecordStarter.Initialize(source, GetType(Contato))
    End Sub

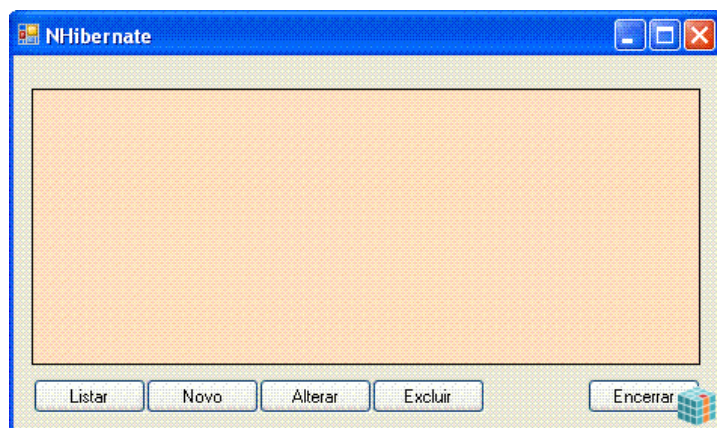
End Class
```

Este arquivo irá ler as informações do arquivo AppConfig.xml criado e iniciar o framework usando o método Initialize da classe ActiveRecordStarter.

Temos que executar este código antes de poder usar os recursos do ActiveRecord, e, sem fazer isso o projeto não vai funcionar.

6. Criando a interface com o usuário

Agora vamos criar uma interface usando o formulário form1.vb conforme a figura abaixo:



Iremos implementar as funcionalidades de listar, incluir, alterar e excluir informações do banco de dados usando o NHibernate e o mapeamento gerado pelo ActiveRecord.

a. Listando os contatos no DataGridView

Vamos primeiro definir o código para listar as informações da tabela Contatos no evento Click do botão Listar:

```
Private Sub btnListar_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnListar.Click
    Try
        gdvContato.DataSource = Contato.FindAll()
        Catch ex As Exception
            MsgBox("ERRO : " & ex.Message)
        End Try
    End Sub
```

Olhe bem o código que usamos; ele é responsável por obter os dados da tabela contatos e exibir no datagridview.

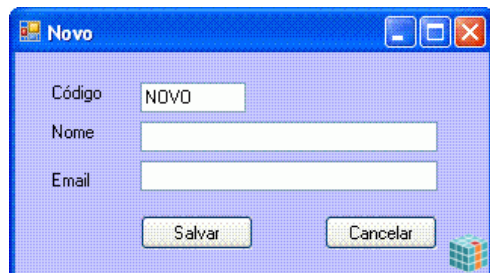
Cadê o comando SQL, a conexão, os objetos Command, Connection, dataset, dataadapter ????

Esta é a vantagem em usar o NHibernate. Não precisamos de nada nisso.

Estamos usando apenas o método FindAll() que retorna todas as instâncias do tipo Contato. Lembre-se de que a nossa classe Contato herda da classe ActiveRecordBase(Of Contato).

b. Criando um novo contato

Para criar um novo contato eu vou incluir um novo formulário a partir do menu Project opção Add Windows Forms com o nome Novo.vb conforme o leiaute abaixo:



Este formulário possui 3 Labels, 3 TextBox :

- txtNome
- txtEmail

Dois controles Buttons:

- btnSalvar
- btnCancelar

Para incluir um novo Contato não é preciso informar o código do mesmo, visto que este campo foi definido como do tipo identity na tabela e é gerenciado pelo próprio banco de dados.

O código do evento Click do botão Salvar do formulário Novo.vb é dado a seguir:

```
Private Sub btnSalvar_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnSalvar.Click
    If txtNome.Text <> String.Empty And txtEmail.Text <> String.Empty Then
        Dim contato As New Contato
        contato.Nome = txtNome.Text
        contato.Email = txtEmail.Text
        Try
            contato.Create()
            txtNome.Text = ""
            txtEmail.Text = ""
            MsgBox("Contato criado com sucesso.")
        Catch ex As Exception
            MsgBox("Erro " & ex.Message)
        End Try
    End If
End Sub
```

Neste código verificamos se foram informados valores para nome e email nas caixas de texto e em seguida criamos uma instância da classe Contato.

A seguir atribuímos os valores para o objeto criado e usamos o método Create() que cria (Salva) uma nova instância do objeto para o banco de dados.

Limpamos os controles e exibimos uma mensagem ao usuário. Ocorrendo erro emitimos um aviso.

Antes que eu esqueça o código que abre este formulário está no evento Click do botão Novo do formulário Form1.vb :

```
Private
    Sub btnNovo_Click(ByVal
        sender As
            System.Object, ByVal
                e As System.EventArgs)
        Handles
            btnNovo.Click

        My.Forms.Novo.ShowDialog()
    End Sub
```

c. Alterando um contato existente

O código do evento Click do botão Alterar do formulário Form1.vb apenas abre um novo formulário que eu crie e chamei de Alterar.vb. O código é:


```

Private
    Sub btnAlterar_Click(ByVal
        sender As
        System.Object, ByVal
        e As System.EventArgs)
        Handles
        btnAlterar.Click
        My.Forms.Alterar.ShowDialog()
End
    Sub

```

Para alterar um contato eu vou incluir um novo formulário a partir do menu Project opção Add Windows Forms com o nome Alterar.vb conforme o leiaute abaixo:

Este formulário possui 3 Labels, 3 TextBox :

- txtCodigo
- txtNome
- txtEmail

Dois controles Buttons:

- btnSalvar
- btnCancelar

Dica: Como os formulário Novo.vb e Alterar.vb são iguais poderíamos ter usado a herança visual para criá-los.

Neste formulário a propriedade Enabled do controle txtCodigo foi definida como False, pois não podemos alterar o código do contato.

No evento Load do formulário Alterar.vb devemos incluir o código a seguir:

```

Private Sub Alterar_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    txtCodigo.Text = Form1.gdvContato.CurrentRow.Cells(0).Value.ToString()
    txtNome.Text = Form1.gdvContato.CurrentRow.Cells(1).Value.ToString()
    txtEmail.Text = Form1.gdvContato.CurrentRow.Cells(2).Value.ToString()
End Sub

```

Este código apenas obtém os valores selecionados no controle DataGridView e os exibe nas caixas de texto do formulário Alterar.vb para que possamos realizar as mudanças.

No evento Click do botão Salvar temos o seguinte código :

```

Private Sub btnSalvar_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnSalvar.Click
    Dim codigo As Integer = Convert.ToInt32(txtCodigo.Text)
    Dim contato As Contato = contato.Find(Convert.ToInt32(codigo))

    contato.Nome = txtNome.Text
    contato.Email = txtEmail.Text
    Try
        contato.Update()
        MsgBox("Contato atualizado com sucesso.")
    Catch ex As Exception
        MsgBox("Erro " & ex.Message)
    End Try
End Sub

```

O código obtém o código da caixa de texto e usando o método Find() procura pelo objeto na coleção de objetos.

Em seguida atribui os valores definidos nas caixas de texto ao objeto encontrado e usando o método Update() atualiza as informações no banco de dados.

d. Excluindo um contato existente

Para excluir um contato existente incluímos o código abaixo no evento Click do botão Excluir:

```

Private Sub btnExcluir_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnExcluir.Click

```



```

If (MsgBox("Deseja excluir o contato : " &
gdvContato.CurrentRow.Cells(1).Value.ToString(), MsgBoxStyle.YesNo) =
MsgBoxResult.Yes) Then
    Dim codigo As Integer = Convert.ToInt32(gdvContato.CurrentRow.Cells(0).Value.ToString())
    Dim contato As Contato = contato.Find(Convert.ToInt32(codigo))

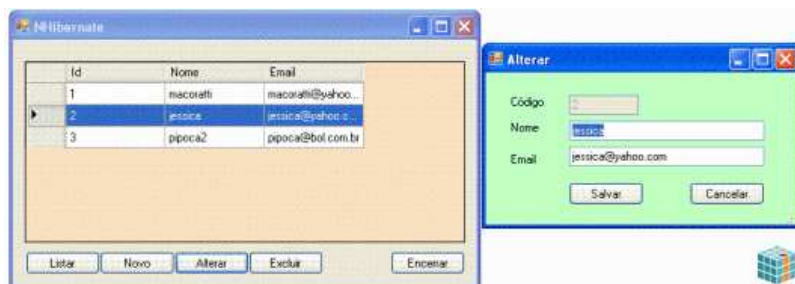
    Try
        contato.Delete()
        MsgBox("Contato excluído com sucesso.")
    Catch ex As Exception
        MsgBox("Erro " & ex.Message)
    End Try
End If
End Sub

```

No código, primeiro obtemos o valor do código do contato a partir da célula do DataGridView selecionada e em seguida, usando o método Find(), procuramos pelo objeto na coleção de objetos.

Para excluir o objeto usamos o método Delete() que deleta uma instância do banco de dados.

Abaixo vemos a aplicação em execução mostrando a alteração de um contato:



Como podemos notar, o NHibernate realiza as operações de persistência usando os métodos apropriados de forma que no nosso código não precisamos usar instruções SQL nem os objetos ADO .NET para realizar a conexão, criar comandos para efetivar tais operações.

De uma forma resumida podemos dizer que por trás dos panos, quando usamos o ActiveRecord, ocorrem as seguintes tarefas:

- os comandos SQL são gerados;
- os comandos SQL são passados para o Gerenciador de banco de dados em uso (RDBMS);
- o RDBMS compila e executa o SQL;
- retorna um enumerable;
- cria um modelo ActiveRecord para cada linha;

Pegue o projeto completo aqui: [NHibernateActiveRecordDemoSQLServer.zip](#) (sem as referências)

O código ficou mais limpo e fácil de manter e ainda temos uma vantagem adicional: se precisarmos mudar o banco de dados, a única alteração que devemos fazer será no arquivo de configuração AppConfig.xml. É isso que irei mostrar na continuação deste artigo.

Eu sei, é apenas NHibernate, mas eu gosto...

Interação



- [Relatar Link Quebrado](#) [Imprimir](#)

Desenvolvimento iPhone

Desenvolvemos aplicativos para smartphones
www.livetouch.com.br

Promocão PMI 4ª Edição

PMI Preparatório Ganhe PMP Fundam. e
Simulados R\$ 1490,00 Todo Brasil

Anúncios Google

5 comentários



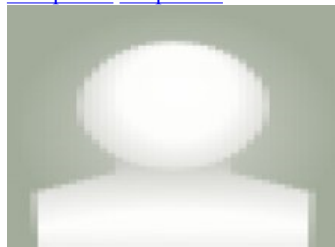
Clayton Lourenço

03/12/2009 15h42

Chave primária composta.

E se a tabela tiver chave primária composta?

[2 respostas](#) [Responder](#)



Leandro Ribeiro

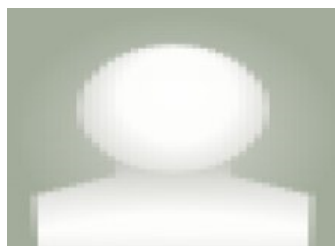
02/01/2010 14h26

Clayton,

Para aprender a trabalhar com chaves compostas sugiro você dar uma leitura na documentação do Active Record, lá inclusive tem um exemplo bem claro sobre chaves compostas.

Link

<http://www.castleproject.org/activerecord/documentation/trunk/usersguide/pks.html>



Leandro Ribeiro

02/01/2010 14h33

<http://www.castleproject.org/activerecord/documentation/trunk/usersguide/pks.html>



Leandro Ribeiro

02/01/2010 14h29

Bom Artigo

Macoratti,

Parabéns pelo artigo, ficou muito claro e simples.

Só gostaria de te sugerir que ao invés de passar o link para baixar o NHibernate, que fosse sugerido usar as DLL's que vem com o próprio Active Record, para evitar assim problemas de dependências de versão caso a DLL do NHibernate baixada seja mais recente do que a que o Active Record esteja usando.

Abraço e parabéns pelo trabalho.

[Responder](#)



Jefferson Shigueaki Sato

02/02/2010 09h20

Atualizar base de dados

Olá.. como pode ser feito nesse caso se for colocar mais um campo nesta tabela, por exemplo: telefone, para atualizar a base de dados sem perder os registros que ja estão ali...

Será que é possível...

[Responder](#)

[Cancelar resposta](#)

Qual a sua opinião?

Se você já possui conta iMasters, o login será feito abaixo.

Nome:

E-mail:

Comentário:

Atenção: comentários considerados spams e/ou ofensivos serão moderados.

Patrocínio: 

Sobre o Autor



José Carlos Macoratti é referência em Visual Basic no Brasil e autor dos livros "Aprenda Rápido: ASP" e "ASP, ADO e Banco de Dados na Internet". Mantenedor do site macoratti.net.

macoratti@yahoo.com

Outros artigos do mesmo autor:

- [ASP .NET - Gerenciamento de serviços usando o...](#)
- [VB .NET - Criando gráficos no VB 2010 Express...](#)
- [VB .NET - Criando gráficos no VB 2010 Express...](#)
- [Crystal Reports para o Visual Studio 2010...](#)

[Ver mais artigos de José Carlos Macoratti](#)

[Indique para um amigo](#)





Microsoft Visual Studio LightSwitch

[Desenvolva suas aplicações de uma maneira muito mais ágil e simples.](#)



RowFeeder

[Monitore e analise diversas palavras-chaves no Twitter e Facebook.](#)



Office 2010

[Versão de avaliação do pacote de escritório da Microsoft.](#)



VirtualBox

[Emule um sistema operacioanl dentro de outro.](#)



Google App Inventor

[Programa de criação de aplicativos para Android aberto para todos.](#)

Parceiros





- [**iMasters**](#)

- [Sobre o iMasters](#)
- [Política de privacidade](#)
- [Anuncie](#)
- [Feeds iMasters](#)
- [Fóruns iMasters](#)
- [Fale conosco](#)

2001 - iMasters FFPA Informática Ltda - Todos os direitos reservados.