

LINGUAGEM DE PROGRAMAÇÃO

Orientada a objeto

Roteiro da aula

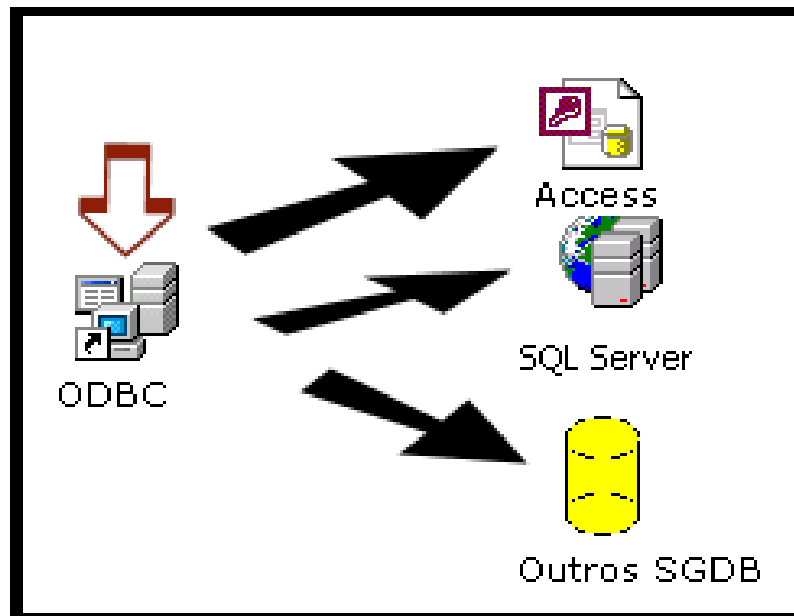
- Introdução ao ADO .NET
- Exemplo com Banco de Dados inserido dentro do projeto.
- Conexão com Banco de Dados

Histórico do acesso aos Dados.

- No início, para se ter acesso às informações do banco de dados era necessário ter vasto conhecimento das API de comunicação,
- O programador acabava implementando seu código exclusivamente para cada versão de driver como a DBLIB (SQL Server)
- Então foi criado em 1990 com apoio da Microsoft e um consórcio de empresas, o padrão **ODBC (Open DataBase Connectivity)**.

Histórico do acesso aos Dados.

- **ODBC (Open DataBase Connectivity)** → é uma camada intermediária encarregada de cuidar da comunicação com a API deixando para o programador uma interface única e padrão para todo acesso ao SGDB.



Histórico do acesso aos Dados.

- A partir do ODCB foram surgindo outras propostas:
- **DAO (Data Access Objects)** focado no MS ACCESS, mas era realmente lento em conjunto com o ODBC.
- Substituído depois pelo **RDO (Remote Data Objects)**
- **OLEDB** semelhante a arquitetura do ODCB porém trouxe a implementação de interfaces COM e a estratégia da Microsoft UDA (Universal Data Access) com objetivo de armazenamento distribuído. Foi aderido até por banco de dados de padrão aberto.
- Para facilitar sua utilização foi criado o **ADO (Activex Data Objects)** com objetivo de consumir os recursos oferecidos pelos OLEDB.

Histórico do acesso aos Dados.

- Concluímos quer foram criadas várias camadas de acesso ao banco de dados, com objetivo de **simplificar** e **padronizar** sua utilização.
- Isso foi fundamental para o desenvolvimento e evolução das aplicações deixando o padrão OLEDB em conjunto com o ADO na liderança no acesso a dados devido a um melhor desempenho no acesso a dados.
- Vale ressaltar que mesmo usando OLEDB ou ODBC a aplicação vai ter uma **perda de desempenho** pois vai ter um camada intermediária entre sua aplicação e as APIs de acesso a dados.

Introdução ADO.NET

- Maioria das aplicações hoje utilizam Banco de dados
- Com a globalização o acesso tornou-se complexo porque as aplicações passam a ser distribuídas em vários locais e acessadas em mais do que um banco de dados.
- O acesso a dados pode ser através de arquivos XML a banco de dados distintos como SQL Serve ou Oracle.
- A plataforma ADO .Net se destaca pela sua produtividade e facilidade no desenvolvimento através do Visual Studio.
- Com a plataforma ADO.Net a Microsoft procurou facilitar a comunicação com outros programas e Banco de dados

Introdução ADO.NET

- O padrão mais conhecido é o XML
- O ADO .Net possui classes e recursos que permitem a sua manipulação de forma simplificada.
- Com o ADO .Net podemos trabalhar de forma desconectada do Banco de Dados
- As classes estão agrupadas no namespace **System.Data**, quando for utilizar você precisará importar este namespace.
- Deverá importar a classe do Banco de dados utilizado:
 - System.Data.SqlClient
 - System.Data.OracleClient
 - System.Data.OleDb
 - System.Data.MySqlClient

Introdução ADO.NET

- Essas classes seguem um padrão
- Isso quer dizer, um objeto criado por uma classe para acesso a dados utilizando ...SQLCliente, e criado e manipulado da mesma forma que um objeto com o namespace ...MySqlClient.
 - System.Data.MyClient – usado para acessar banco de dados MySql (necessita instalar classe / dll)
 - System.Data.SqlClient –usado para acessar banco de dados Microsoft SqlServer a partir da versão 7.0
 - System.Data.OleDb –usado para acessar banco de dados Microsoft Access e Microsoft SqlServer versão anterior a 7.0

Exemplo:

- Exerc2-Ado.Net – parte 1
- Criar um formulário completo com inserção, atualização e exclusão de registros.
- Criar um Banco de Dados SQL de forma simplificada através de diagrama

Pratique...

- Exerc2-Ado.Net – parte 2 – pratica exercitando...

Conexão Banco de Dados

- Conexão com Banco de dados através do objetoConnection.
- Utilizar objeto Command para recuperar dados de um Banco de dados

Conexão Banco de Dados

- O processo de conexão com o banco de dados e leitura de registros é executado em 5 passos:
 1. Criar um objeto de conexão com o banco de dados (**Connection**)
 2. Criar um objeto que vai executar um comando SQL (**Command**)
 3. Abrir o Banco de Dados
 4. Executar o comando SQL e processar o resultado
 5. Fechar a conexão com o banco de Dados

Command

- Objeto usado para executar um comando SQL é conhecido como Command
- Para criar este objeto você precisa informar:
 uma instrução SQL
 e qual objeto Connection ele deve usar .

Connection

- O objeto utilizado para conexão é conhecido como Connection

```
SqlConnection conn;
```

```
conn = new SqlConnection();
```

```
OleDbConnection conn;
```

```
conn = new OleDbConnection();
```

```
SqlConnection conn = new SqlConnection();
```

Connection

- Para criá-lo você vai precisar da string de conexão que informa ao objeto sobre como e em qual banco deve conectar.
- Você passa a string de conexão para o objeto Connection através da propriedade ConnectionString:

```
conn.ConnectionString = strconn;
```

- **strcon** é uma variável string que armazena a string de conexão.
- Pode ser feito em uma única linha

```
SqlConnection conn = new SqlConnection(strconn);
```


String de conexão

- É através dela que vamos informar ao objeto Connection onde está o Banco, o nome e como conectar(login, senha)
- A string pode receber parâmetros separados por virgula (DataSource, Initial Catalog, Integrated Security...)
- Exemplo:

```
"Data Source=.\SQLEXPRESS; AttachDbFilename=C:\Projetos.mdf;  
Integrated Security=True;User Instance=True;"
```

Pratica

- Continuando o aplicativo ControleProjetos
- Antes de continuar certifique que seu programa está funcionando corretamente
- Exercicio3 - ConexãoBD


Pratica

- Crie uma nova tabela Contato (PK-ContatoID e FK - EmpresaID)

Contato *			
	Column Name	Data Type	Allow Nulls
<input type="checkbox"/>	ContatoID	int	<input checked="" type="checkbox"/>
<input type="checkbox"/>	EmpresaID	int	<input checked="" type="checkbox"/>
<input type="checkbox"/>	Nome	varchar(50)	<input checked="" type="checkbox"/>
<input type="checkbox"/>	Cargo	varchar(50)	<input checked="" type="checkbox"/>
<input type="checkbox"/>	Telefone	varchar(15)	<input checked="" type="checkbox"/>
<input type="checkbox"/>	Celular	varchar(15)	<input checked="" type="checkbox"/>
<input type="checkbox"/>	Endereco	varchar(50)	<input checked="" type="checkbox"/>
<input type="checkbox"/>	Cidade	varchar(50)	<input checked="" type="checkbox"/>
<input type="checkbox"/>	UF	varchar(2)	<input checked="" type="checkbox"/>
<input type="checkbox"/>	CEP	varchar(9)	<input checked="" type="checkbox"/>
<input type="checkbox"/>	Email	varchar(50)	<input checked="" type="checkbox"/>
<input type="checkbox"/>	DataNascimento	datetime	<input checked="" type="checkbox"/>
<input type="checkbox"/>	Observacao	text	<input checked="" type="checkbox"/>
<input type="checkbox"/>			<input type="checkbox"/>


Pratica

- Crie mais uma tabela Projeto (PK-Projetoid e FK - Empresaid)

Projeto ^{PK}			
	Column Name	Data Type	Allow Nulls
	ProjetoID	int	<input type="checkbox"/>
	Empresaid	int	<input type="checkbox"/>
	Nome	varchar(50)	<input type="checkbox"/>
	DataInicio	datetime	<input type="checkbox"/>
	DataFim	datetime	<input type="checkbox"/>
	Obs	text	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

Pratica

- Crie mais uma tabela Tarefa(PK-TarefaId e FK - ProjetoId)

Tarefa *			
	Column Name	Data Type	Allow Nulls
	TarefaID	int	<input type="checkbox"/>
	ProjetoID	int	<input type="checkbox"/>
	Descricao	text	<input type="checkbox"/>
	Inicio	datetime	<input type="checkbox"/>
	Fim	datetime	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

Objeto Command

- **ExecuteScalar** → quando queremos retornar apenas 1 valor.
- **ExecuteReader** → quando queremos retornar um conjunto de registros através do comando SQL Select por exemplo. Associado a outro objeto – DataReader – para leitura de registros.
- **ExecuteNonQuery** → quando não espera retorno algum, como os comandos: Insert, Update ou Delete.

Pratica

- Continuando o aplicativo ControleProjetos
- Exercicio3 - ConexãoBD -- parte2

Leia – Anote – Entenda

- Classe **MySQLCommand** → através dessa classe executamos uma instrução SQL
- Na propriedades **Connection** → informa a conexão aberta
- Na propriedade **CommandText** → informa instruções SQL
- Metodo **ExecuteReader** → é o que retorna as linhas baseadas na execução da instrução SQL
- Objeto **MySqlDataReader** → é o retorno do ExecuteReader, para com o metodo **Read** , ler os dados e apresentá-los na tela

Links

- .NET para iniciantes:
- <http://aprendendodotnet.wordpress.com/2012/05/04/ado-net-classes-para-conexao-e-comandos-sql-server-conceitos/>
- Conceitos de ADO .NET
- <http://www.linhadecodigo.com.br/artigo/296/net-adonet-para-iniciantes.aspx>