



DelphiLive!

The Delphi Developer Conference

Jim Tierney | Embarcadero Technologies

DataSnap Server Method Basics

Data Types and Client Connectivity

Content

- Server Method Example
- Parameter Types
- Server programming
- Client programming
 - Delphi
 - Delphi Prism
- Feature previews
- Questions

Server Method Example

- Server
 - DataSnap Components
 - TDSServer
 - TDSTCPSTransport
 - Port: 211
 - TDSServerClass
 - Server Method Class
 - DataModule
 - TDSServerModule
 - Simple class with no designer
 - TPersistent with {\$METHODINFO ON}
 - Server Methods
 - Public
 - Parameters must be supported by DataSnap

Server Method Example

- Data Explorer
 - Add DataSnap Connections
 - View Server Method Parameters
 - Execute Server Methods

Server Method Example

- Client
 - TSQLConnection
 - Driver: DataSnap
 - Driver.Port: 211
 - Driver.HostName: localhost
 - Proxy Generator

Parameter Types

- DataSnap supports specific parameter and function result types
- Parameter direction is supported on some types
 - var
 - out
- Methods declared with unsupported types/directions are not available
 - Method not listed in DataSnap server meta data
 - DataSnap server will raise “method not found” exception when called by client

Parameter Types

Basic
AnsiString
Boolean
Currency
TDateTime
DBXDate
DBXTime
Double
Int64
Integer
LongInt
Integer
LongInt
OleVariant
Single
SmallInt
String
WideString

Basic DBXValue
TDBXAnsiStringValue
TDBXAnsiCharsValue
TDBXBcdValue
TDBXBooleanValue
TDBXDateValue
TDBXDoubleValue
TDBXInt16Value
TDBXInt32Value
TDBXInt64Value
TDBXSingleValue
TDBXStringValue
TDBXTimeStampValue
TDBXTimeValue
TDBXWideCharsValue
TDBXWideStringValue

Streams
TStream
TDBXStreamValue

Tables
TParams
TDataSet
TDBXReader

Connection
TDBXConnection*

* TDBXConnection not covered in this presentation

Parameter Types - Basic

- String and Scalar
- Not nullable
- Default direction is "in"
- *var* and *out* supported
- Can be a function result type

Type	(in)	var	out	Result
AnsiString	✓	✓	✓	✓
Boolean	✓	✓	✓	✓
Currency	✓	✓	✓	✓
TDateTime	✓	✓	✓	✓
DBXDate	✓	✓	✓	✓
DBXTime	✓	✓	✓	✓
Double	✓	✓	✓	✓
Int64	✓	✓	✓	✓
Integer	✓	✓	✓	✓
LongInt	✓	✓	✓	✓
Integer	✓	✓	✓	✓
LongInt	✓	✓	✓	✓
OleVariant	✓	✓	✓	✓
Single	✓	✓	✓	✓
SmallInt	✓	✓	✓	✓
String	✓	✓	✓	✓
WideString	✓			✓

Parameter Types - Basic DBXValue

- String and Scalar
- Nullable
- Default direction is in/out
- *Var* and *Out* not supported
- Cannot be a function result type

Type	(in/out)	var	out	Result
TDBXAnsiStringValue	✓			
TDBXAnsiCharsValue	✓			
TDBXBcdValue	✓			
TDBXBooleanValue	✓			
TDBXDateValue	✓			
TDBXDoubleValue	✓			
TDBXInt16Value	✓			
TDBXInt32Value	✓			
TDBXInt64Value	✓			
TDBXSingleValue	✓			
TDBXStringValue	✓			

Parameter Types - Streams

- TStream
 - Not nullable
- TDBXStreamValue
 - Nullable
- Streams can be passed in both directions
 - Client reads stream from server
 - Server reads stream from client
- Large Streams (>32K) Marshaled in 32K chunks
 - Round trip for every chunk
 - Receive unidirectional stream
 - Seek not supported
 - Stream.Size = -1

Type	(in)	var	out	Result
TStream	✓	✓	✓	✓

Type	(in/out)	var	out	Result
TDBXStreamValue	✓			

Parameter Types - Tables

- Table Types
 - TDataSet
 - TParams
 - TDBXReader
- Tables can be passed in both directions
 - From server to client
 - From client to server
 - Limited column type support

Type	(in)	var	out	Result
TDataSet	✓	✓	✓	✓
TParams	✓	✓	✓	✓
TDBXReader	✓	✓	✓	✓

Server Programming

- DBXValue types
- Streams
- Tables
- Exceptions

Server Programming

- DBXValue types

- IsNull property

```
If ADBXInt32Value.IsNull then
```

```
...
```

- SetNull method

```
ADBXInt32Value.SetNull;
```

- Getter methods

```
I := ADBXInt32Value.GetInt32;
```

- Setter methods

```
ADBXInt32Value.SetInt32(I);
```

Server Programming - DBXValue

Getter	Setter	Type	Use with
GetAnsiString	SetAnsiString	AnsiString	TDBXAnsiCharsValue, TDBXAnsiStringValue
GetBcd	SetBcd	TBcd	TDBXBcdValue
GetBoolean	SetBoolean	Boolean	TDBXBooleanValue
GetDate	SetDate	TDBXDate	TDBXDateValue
GetDouble	SetDouble	Double	TDBXDoubleValue
GetInt16	SetInt16	Int16	TDBXInt16Value
GetInt32	SetInt32	Int32	TDBXInt32Value
GetInt64	SetInt64	Int64	TDBXInt64Value
GetSingle	SetSingle	Single	TDBXSingleValue
GetString	SetString	String	TDBXStringValue, TDBXWideCharsValue, TDBXWideStringValue
GetTime	SetTime	TDBXTime	TDBXTimeValue
GetTimeStamp	SetTimeStamp	TSQLTimeStamp	TDBXTimeStampValue
GetWideString	SetWideString	UnicodeString (same as string)	TDBXStringValue, TDBXWideCharsValue, TDBXWideStringValue

Server Programming - Streams

- TDBXStreamValue

- IsNull method

- If ADBXStreamValue.IsNull then ...

- SetNull method

- ADBXStreamValue.SetNull;

- GetStream method

- S := ADBXStreamValue.GetStream(True (* InstanceOwner *));

- InstanceOwner parameter

- True – Caller owns stream and must free it
 - False – DataSnap will free the stream later

- SetStream method

- ADBXStreamValue.SetStream(S, True (* InstanceOwner *));

- InstanceOwner parameter

- True – DataSnap owns the stream and will free it when no longer in use
 - False – DataSnap will not free the stream

- Var, out or Result TStream

- Once server methods exits, DataSnap owns the stream and will free it when no longer in use

- Reading a stream from the client

- Don't rely on size property (has value of -1 for streams > 32K)
 - Do not use Seek method

Server Programming - Tables

- Reading a TDBXReader from the client

- Next method

- ```
While AReader.Next do ..
```

- Getter Properties

- ```
I := AReader.Value[ColumnIndex].AsInt32
```

- Reader/DataSet Conversions

- TDBXReader to TClientDataSet

- ```
DS := TDBXDataSetReader.ToClientDataSet(nil,
 AReader, True (* InstanceOwner *))
```

- TDataSet to TDBXReader

- ```
R := TDBXTableReader.Create(ADataset, True (*  
    InstanceOwner *))
```

Server Programming - Exceptions

- Raising exceptions to the client
 - Unhandled exception in a server method are marshaled to the client and raised as TDBXError exceptions

Client Programming - Delphi

- Calling a server method
 - Prepare a command
 - Set Parameter Values
 - Execute command
 - Get Parameter Values
- Working with Streams
- Working with Tables

Client Programming - Delphi

- Prepare a command
 - CommandType property
 - TDBXCommandTypes.DSServerMethod
 - value is 'DataSnap.ServerMethod'
 - Text property
 - Class name + '.' + method name
 - Prepare method
 - Populates command parameters list using metadata from the server

```
if FCommand = nil then
begin
    FCommand := FDBXConnection.CreateCommand;
    FCommand.CommandType := TDBXCommandTypes.DSServerMethod;
    FCommand.Text := ' TServerMethods.TestInt32 ';
    FCommand.Prepare;
end;
```

Client Programming - Delphi

- Set all input and input/output parameters
 - Setter Methods

```
FCommand.Parameters[0].Value.SetInt32(I);
```
 - SetNull method

```
FCommand.Parameters[0].Value.SetNull;
```

 - For use with server methods using TDBXValue types (e.g.; TDBXInt32Value)
- Execute Command
 - FCommand.ExecuteUpdate
 - Some Possible Exceptions
 - Client side
 - Parameter value not set
 - DataSnap server method invocation errors
 - Unknown server method
 - Parameter errors (Value can't be Null)
 - Exceptions raised by a Server Method
- Get input/output, output and result parameter values
 - IsNull property

```
If FCommand.Parameters[1].Value.IsNull then ...
```

 - For use with server methods using TDBXValue type parameters
 - Getter Methods

```
I := FCommand.Parameters[1].Value.GetInt32;
```

```

// Server Method
function TServerMethods.TestInt32(
    AIn: Int32; var AVar: Int32; out AOut: Int32;
    AValue: TDBXInt32Value): Int32;

// Client Code
Procedure TServerMethodsClient.TestInt32(
    AIn: Int32; var AVar: Int32; out AOut: Int32;
    var AValue: Int32; var ANullValue: Boolean): Int32
begin
    if FCommand = nil then
        begin
            FCommand := FDBXConnection.CreateCommand;
            FCommand.CommandType := TDBXCommandTypes.DSServerMethod;
            FCommand.Text := ' TServerMethods.TestInt32 ';
            FCommand.Prepare;
        end;

        FCommand.Parameters[0].Value.SetInt32(AIn);
        FCommand.Parameters[1].Value.SetInt32(AVar);
        If ANullValue then
            FCommand.Parameters[3].Value.SetNull
        Else
            FCommand.Parameters[3].SetInt32(AValue);

        FCommand.ExecuteUpdate;

        AVar := FCommand.Parameters[1].Value.GetInt32;
        AOut := FCommand.Parameters[2].Value.GetInt32;
        ANullValue := FCommand.Parameters[3].Value.IsNull;
        If not ANullValue then
            AValue := FCommand.Parameters[3].Value.GetInt32;
        Result := FCommand.Parameters[4].Value.GetInt32; // Return value
    end;

```

Client Programming - Streams

- **SetStream method**

```
FCommand.Parameters[0].Value.SetStream(S, AInstanceOwner);
```

- **InstanceOwner parameter**

- True – Caller owns the stream and must free it
 - False – DataSnap will free the stream later

- **GetStream method**

```
S := FCommand.Parameters[0].Value.GetStream(AInstanceOwner);
```

- **InstanceOwner parameter**

- True – DataSnap owns the stream and will free it when no longer in use
 - False – DataSnap will not free the stream

- **Null streams**

- Handle null when the server method parameter type is TDBXStreamValue
 - **IsNull method**

```
If FCommand.Parameters[0].Value.IsNull then ...
```

- **SetNull method**

```
FCommand.Parameters[0].Value.SetNull;
```

- **Reading a stream from the server**

- Do not rely on Size property (size has a value of -1 for streams > 32K)
 - Do not use Seek method

Client Programming - Tables

- Client parameters support TDBXReader values
 - Use when server method parameter types are TDataSet, TParams, or TDBXReader
- SetDBXReader method

```
FCommand.Parameters[0].Value.SetDBXReader(R, AInstanceOwner);
```
- GetDBXReader method

```
R := FCommand.Parameters[0].Value.GetDBXReader(AInstanceOwner);
```
- Reading a TDBXReader from the server
 - Next method

```
While AReader.Next do ..
```
 - Getter Properties

```
I := AReader.Value[ColumnIndex].AsInt32
```
- TDataSet and TParams conversions
 - TDBXReader to TClientDataSet

```
DS := TDBXDataSetReader.ToClientDataSet(nil, AReader, AInstanceOwner)
```
 - TDBXReader to TParams

```
Params := TDBXParamsReader.ToParams(nil, AReader, AInstanceOwner);
```
 - TDataSet to TDBXReader

```
R := TDBXTableReader.Create(ADataSet, AInstanceOwner)
```
 - TParams to TDBXReader

```
R := TDBXParamsReader.Create(AParams, AInstanceOwner)
```

Client Programming - Delphi Prism

- Calling a server method
 - Prepare a command
 - Set Parameter Values
 - Execute command
 - Get Parameter Values
- Working with Streams
- Working with Tables

Client Programming - Prism

- Prepare a command
 - CommandType property
 - System.Data.CommandType.StoredProcedure
 - CommandText property
 - Class name + '.' + method name
 - Prepare method
 - Populates command parameters list using metadata from the server

```
if FCommand = nil then
begin
    FCommand := FDBXConnection.CreateCommand as TAdoDbxCommand;
    FCommand.CommandType :=
System.Data.CommandType.StoredProcedure;
    FCommand.Text := ' TServerMethods.TestInt32 ';
    FCommand.Prepare;
end;
```

Client Programming - Prism

- Set all input and input/output parameters

```
FCommand.Parameters[0].Value := I;
```

Or

```
FCommand.Parameters[0].Value := DBNull.Value
```

- For use with server methods using TDBXValue types

- Execute a Command

```
FCommand.ExecuteNonQuery
```

- Some Possible Exceptions

- Client side
 - Parameter value not set
- DataSnap server method invocation errors
 - Unknown server method
 - Parameter errors (Value can't be Null)
- Exceptions raised by a Server Method

- Get input/output and result parameters values

```
I := FCommand.Parameters[1].Value as Int32;
```

Or

```
If FCommand.Parameters[1].Value = DBNull.Value then ...
```

- For use with server methods using TDBXValue types

```

// Server Method
function TServerMethods.TestInt32(
    AIn: Int32; var AVar: Int32; out AOut: Int32;
    AValue: TDBXInt32Value): Int32;

// Client Code
Procedure TServerMethodsClient.TestInt32(
    AIn: Int32; var AVar: Int32; out AOut: Int32;
    var AValue: nullable Int32): Int32
begin
    if FCommand = nil then
        begin
            FCommand := FDBXConnection.CreateCommand as TAdoDbxCommand;
            FCommand.CommandType := System.Data.CommandType.StoredProcedure;
            FCommand.Text := ' TServerMethods.TestInt32 ';
            FCommand.Prepare;
        end;

        FCommand.Parameters[0].Value := AIn;
        FCommand.Parameters[1].Value := AVar;
        If AValue = nil then
            FCommand.Parameters[3].Value := DbNull.Value
        Else
            FCommand.Parameters[3].Value := AValue;

        FCommand.ExecuteNonQuery;

        AVar := FCommand.Parameters[1].Value as Int32;
        AOut := FCommand.Parameters[2].Value as Int32;
        If FCommand.Parameters[3].Value = DBNull.Value then
            AValue := nil
        else
            AValue := FCommand.Parameters[3].Value as Int32;
        Result := FCommand.Parameters[4].Value as Int32; // Return value
    end;

```

Client Programming - Prism Streams

- Use `System.IO.Stream` when server method parameter type is `TStream` or `TDBXStreamValue`
 - Set
`FCommand.Parameters[0].Value := AStream;`
 - Get
`AStream := FCommand.Parameters[0].Value as System.IO.Stream;`
- Null streams
 - When the server method parameter type is `TDBXStreamValue`
 - Set
`FCommand.Parameters[0].Value := DBNull.Value;`
 - Get
`If FCommand.Parameters[0].Value = DBNull.Value then ...`
- Reading a `System.IO.Stream` from the server
 - Do not rely on `Length` property (`Length` has a value of `-1` for streams $> 32K$)
 - Do not use `Seek` method

Client Programming - Prism Tables

- Use `System.Data.IDataReader` when server method parameter type is `TDataSet`, `TParams` or `TDBXReader`

- Set

```
FCommand.Parameters[0].Value := AReader;
```

- Get

```
AReader := FCommand.Parameters[0].Value as  
System.Data.IDataReader;
```

- Passing a `System.Data.DataTable` to server methods

```
AReader := ADataTable.CreateDataReader();  
FCommand.Parameters[0].Value := AReader;
```

Feature Previews

- Delphi Prism proxy generator (just released)
- New transports and protocols
- New data types

Closing

- Questions
- DataSnap Blog
 - <http://blogs.embarcadero.com/jimtierney>
 - Includes links to sample clients and servers
- Other presentations from DataSnap R&D
 - Leonel Tognioli
 - Beyond the Basics of DataSnap 2009 Server Methods
 - 2:30 today
 - Adrian Andrei
 - DataSnap 2009 Client/Server Communication Protocol and Beyond
 - 6:15 tomorrow
- Thank you for coming!