Virginia Beach IBM i User Conference 2016

Uniting .NET and the IBM i: Tools, Tips and Techniques.

October 7, 2016 – (11:00 AM-12:00 PM)

Presented by: Derek Maciak
CTO/Partner
Surround Technologies
surroundtech.com | @SurroundTech

SURROUND TECHNOLOGIES™

Start Looping Title Slide

Today's Speaker

Derek Maciak

CTO/Partner Surround Technologies

dmaciak@surroundtech.com | www.surroundtech.com

Socialize:

- linkedin.com/company/128638
- tweet me @SurroundTech
- facebook.com/surroundtech

Agenda

What we'll discuss:

The IBM .NET data provider can be utilized to enable .NET applications to access data or call programs that reside on the IBM i. This session will provide the best practices for allowing windows, web and mobile applications to leverage the power of the IBM i while following Service Oriented Architecture (SOA) design principles. We will discuss tips and techniques for connecting to the IBM i, accessing data, and calling Programs(COBOL, RPG, Java, C, IBM i commands, etc.) on the IBM i.

Agenda Objectives:

- Best Practices for connecting .NET to the IBM i using the .NET data provider and following a Service Oriented Architecture
- Best Practices for accessing data using ADO.NET
- How to call programs using stored procedures
- Ways to effectively reuse existing 5250 screens in .NET

Connecting .NET to the IBM i

SURROUND TECHNOLOGIES™

Best Practices:

1. Use an IBM i ADO.NET Data Provider
2. Use Exception Handling and Logging
3. Put Connection information in a configuration file
4. String Properties
   - Use Library List
   - Use Connection Pooling
5. Design for a Service Oriented Architecture (SOA)
6. Stream Data in Blocks using multiple calls

IBM Business Partner

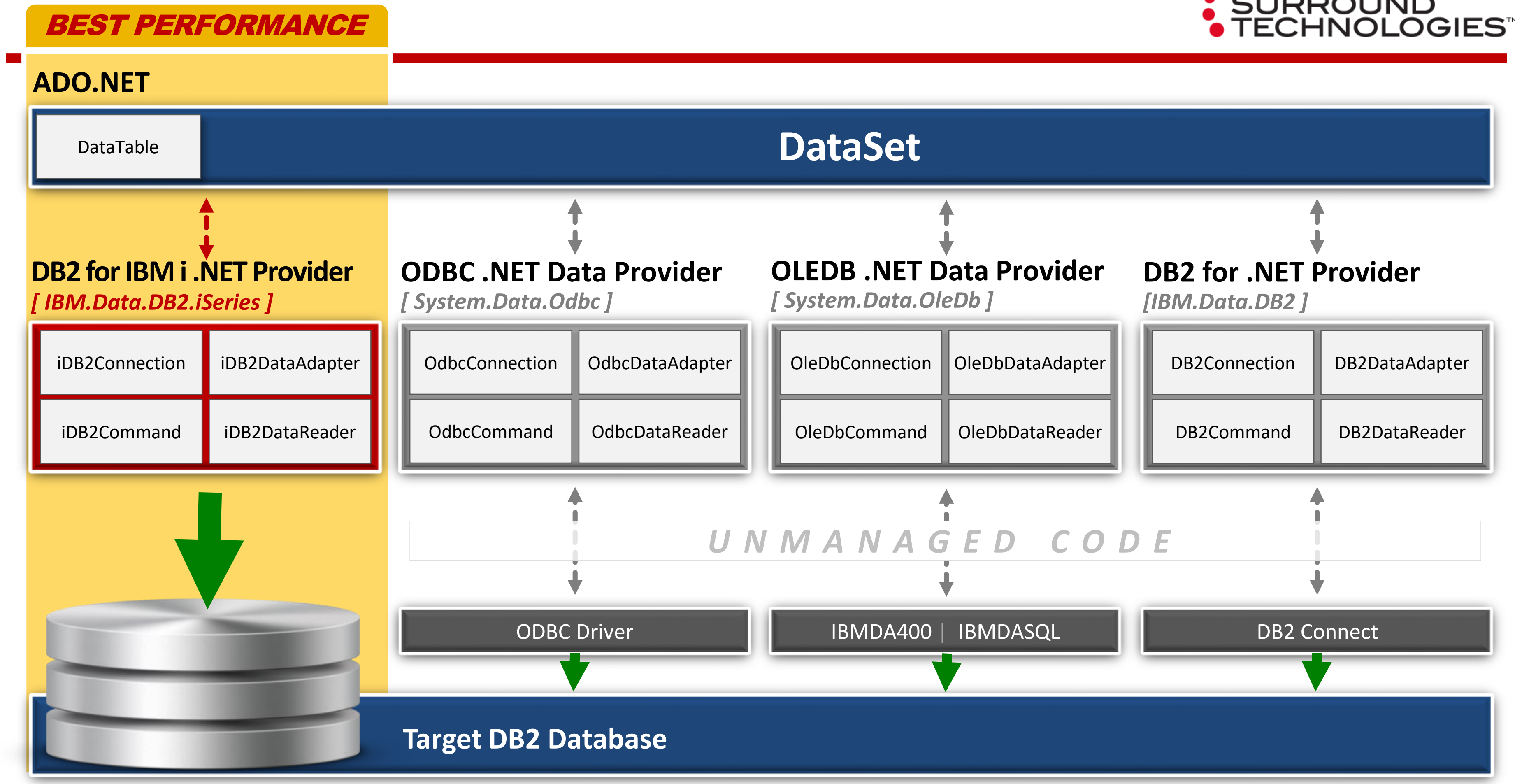
Microsoft CERTIFIED Partner

Agenda Objectives:

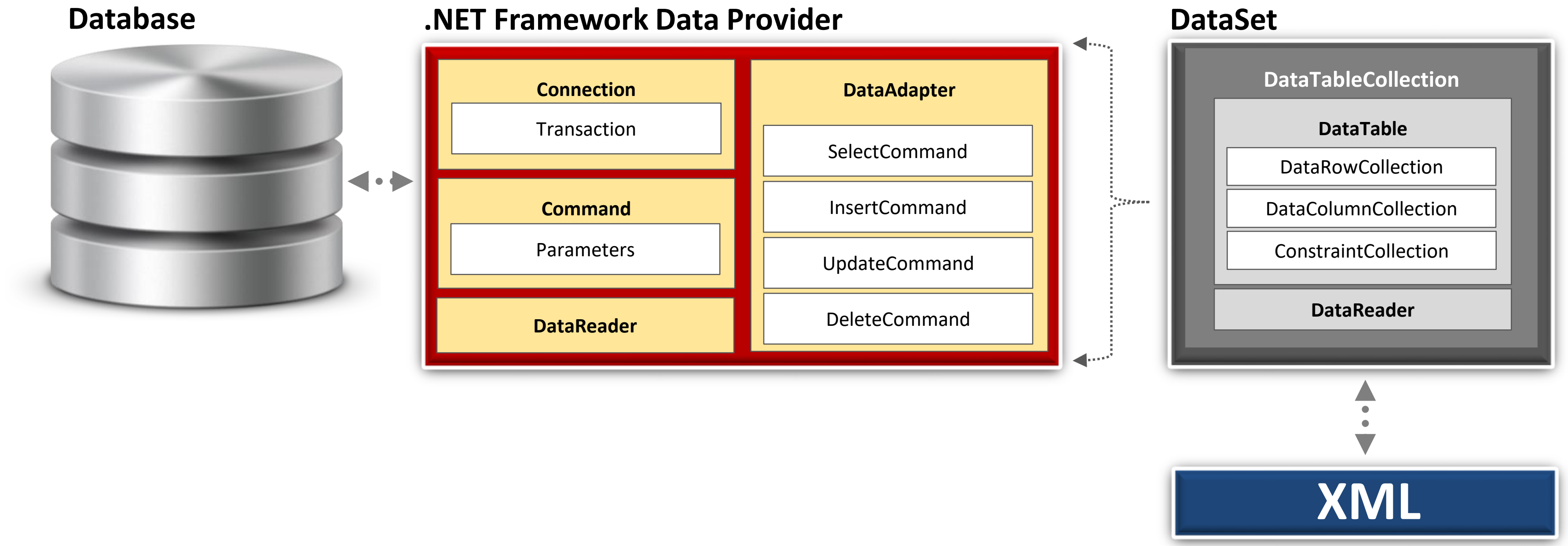
➡ ***Best Practices for connecting .NET to the IBM I using the .NET data provider and following a Service Oriented Architecture***

- Best Practices for accessing data using ADO.NET
- How to call programs using stored procedures
- Ways to effectively reuse existing 5250 screens in .NET

1. Use an IBM i ADO.NET Provider



1. ADO .NET Architecture



1. Benefits of ADO.NET

.NET Managed Code (Targets the CLR)

- *Better Performance*
- No COM Interop Module
- No Marshalling of Data
- Memory Management
- Thread Execution Support
- Security
- Reliability
- Remoting
- Enforces strict type safety

1. IBM DB2 for IBM i .NET Provider

- Available as part of IBM i Access for Windows starting with V5R3M0 and IBM i Access Client Solutions
- Not a default install option
  - Must use Selective install with IBM i Access for Windows
  - Optional Windows Application Package that is part of IBM i Access Client Solutions
- Can connect to down-level OS/400 versions to V5R1
- Need TCP/IP connection from PC to IBM i
- Uses the Optimized host database server job (QZDASOINIT) on the IBM i
- .NET class needs to reference IBM.Data.DB2.iSeries
- 5.4 version supports .NET Framework 1.0 and 1.1
- 6.1 and 7.1 versions support .NET Framework 2.0 or later
- IBM i Access for Windows 7.1 is the most current version available and will connect to any supported IBM i release.
- Always check and update to current Service Pack

2. Use Exception Handling and Logging

- Don’t assume that nothing will go wrong
- Use try/catch blocks
- You can trap specific Exceptions and code a recovery
- iDB2SQLErrorException and the iDB2CommErrorException will be the most common Exceptions thrown
- Suggest logging the exception

```
try
{
    // Opening connection, if this fails we will log the error and return a null
    connection.Open();
}
catch (Exception ex)
{
    logger.am_WriteLog("EasyBuyService", AB_LogLevel.Error, "SelectCustomers - Opening Connection", ex.Message);
    return null;
}
```

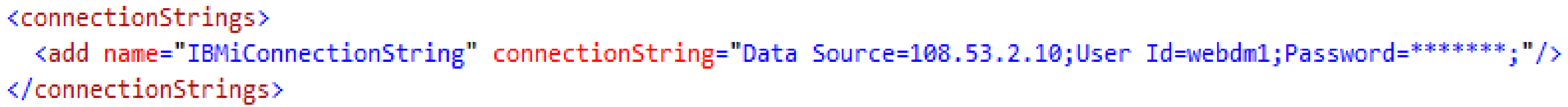
2. Provider Specific Exceptions

- iDB2CommErrorException
- iDB2ConnectionFailedException
- iDB2ConnectionTimeoutException
- iDB2ConversionException
- iDB2DCFunctionErrorException
- iDB2ExitProgramErrorException
- iDB2HostErrorException
- iDB2InvalidConnectionStringException
- iDB2MaximumPoolSizeExceededException
- iDB2SQLErrorException
- iDB2SQLParameterErrorException
- iDB2UnsupportedHostVersionException

3. Put Connection Information in a Config

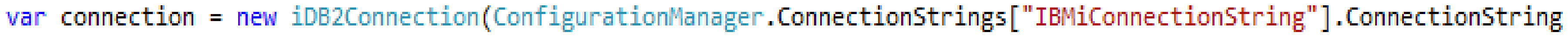
- Soft coding the connection information allows you to easily configure the connection properties without having to do a build

Configuration:



```
<connectionStrings>
  <add name="IBMiConnectionString" connectionString="Data Source=108.53.2.10;User Id=webdm1;Password=*******;"/>
</connectionStrings>
```

Loading Configuration at runtime:



```
var connection = new iDB2Connection(ConfigurationManager.ConnectionStrings["IBMiConnectionString"].ConnectionString
```

4. Provider Specific Connection String Properties

- ConnectionTimeout
- CheckConnectionOnOpen
- Database
- DataCompression
- DataSource
- DefaultCollection
- DefaultIsolationLevel
- HexParserOption
- LibraryList
- MaximumDecimalPrecision
- MaximumDecimalScale
- MaximumInlineLobSize
- MaximumPoolSize
- MaximumUseCount
- MinimumDivideScale
- MinimumPoolSize
- Naming
- Password
- Pooling
- QueryOptionsFileLibrary
- SortLanguageId
- SortSequence
- SortTable
- SSL
- Trace
- UserID

SURROUND TECHNOLOGIES™

4. Provider Specific Connection String Properties

- **Naming (Default is “SQL”)**
  - When set to “SQL” , a period(.) is used to separate schema. ex: schema.object
  - When set to “System”, a forward slash(/) is used to separate schema. ex: schema/object

- **Pooling (Default is “True”)**
  - Allows reuse of connections by an application that frequently open/close connections
  - Reused Connections open more quickly
  - Consider calling iDB2ProviderSettings.CleanupPooledConnections() when application is about to end to make sure all pooled connections are terminated

4. Provider Specific Connection String Properties

- **LibraryList**
  - List a schema names separated by commas
  - *USRLIBL can be used to add the library list for the User Profile of the current host server job
  - Placing schemas before or after the *USRLIBL can control the referenced order
  - DefaultCollection should be added to the LibraryList
  - LibraryList is great for testing against different Libraries and for allowing different User Profiles to access different Libraries

- **DefaultCollection**
  - If specified the default Schema is used for all unqualified object names
  - If Not specified
    - If Naming is “SQL”, Default Collection is set to the user ID that opened the connection
    - If Naming is “System”, Default Collection is not set, unqualified object names use the host server job’s library list

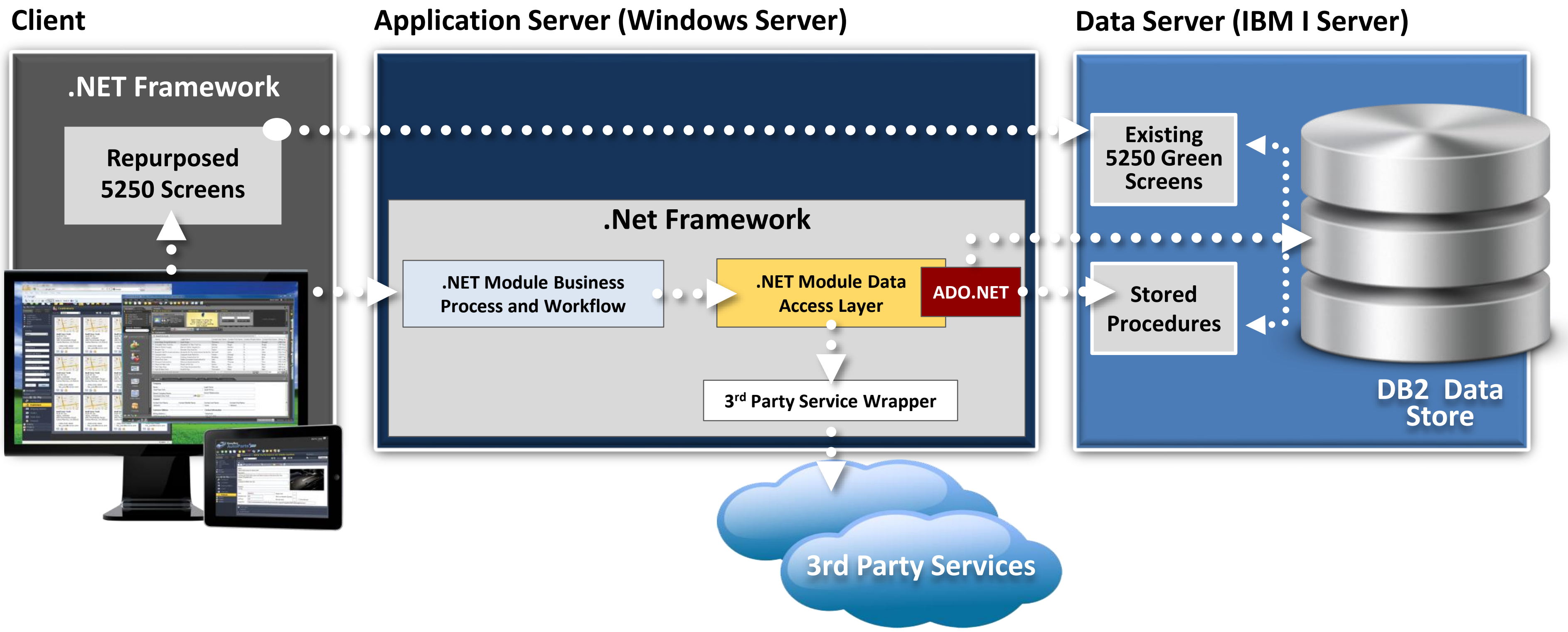
SURROUND TECHNOLOGIES™

5. Service Oriented Architecture

You may need to support all of this!

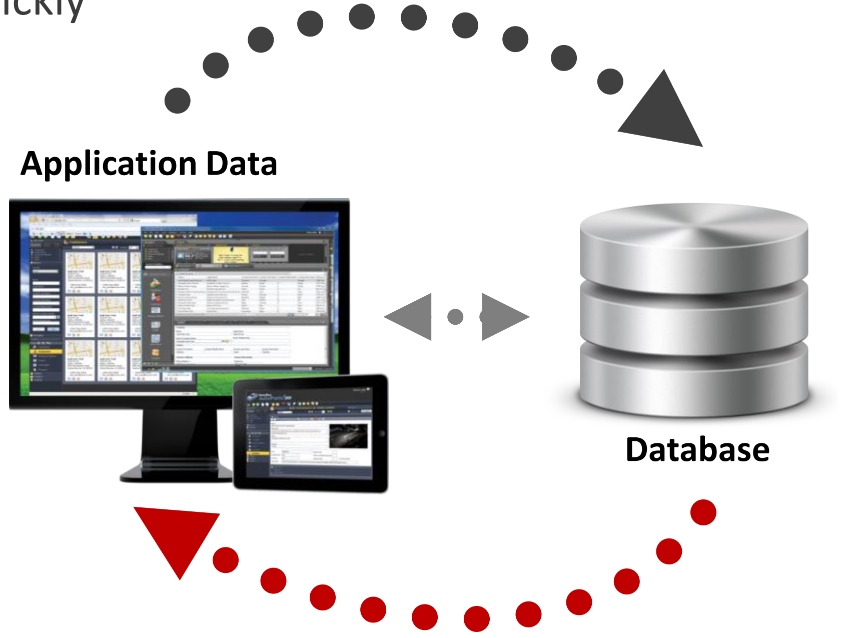
SURROUND TECHNOLOGIES™

5. Service Oriented Architecture


Client
Application Server (Windows Server)
Data Server (IBM I Server)
.NET Framework
Repurposed 5250 Screens
.Net Framework
.NET Module Business Process and Workflow
.NET Module Data Access Layer
ADO.NET
3rd Party Service Wrapper
Existing 5250 Green Screens
Stored Procedures
DB2 Data Store
3rd Party Services

6. Streaming Data in Blocks

- Useful when needing to output a lot of data
- A requestor can receive a block of data very quickly in which the user can begin to work with
- Data should be accessed Asynchronously
- Useful to get around webserver message size quotas



6. Streaming Data in Blocks

Demo

Accessing Data for ADO.NET

SURROUND TECHNOLOGIES™

Best Practices:

1. Using Connection Object
2. Using Command Object
3. DataReader vs. DataSet
4. Use Parameterized Queries
5. Choosing the correct Execution Method
6. Always Close and Dispose
7. Use the "Using" Statement in C#
8. Performing Transactions
9. Benefits of Stored Procedures

Agenda Objectives:

- Best Practices for connecting .NET to the IBM I using the .NET data provider and following a Service Oriented Architecture
- ***Best Practices for accessing data using ADO.NET***
- How to call programs using stored procedures
- Ways to effectively reuse existing 5250 screens in .NET

1. Using Connection Object

- Connection object for DB2 for IBM i .NET Provider is **iDB2Connection**
- In ADO.net, the Connection Object is used to Connect to a database
- Public Properties
  - **ConnectionString** - Use Provider specific connection string properties
  - **ConnectionTimeout** – Gets the time to wait while establishing a connection before terminating and generating an error
- Public Methods
  - **Open()** – Opens a connection to the data source using the settings specified in the ConnectionString
  - **Close()** – Closes a connection to the data source
  - **CreateCommand()** – Creates a new iDB2Command object for use with this connection
  - **BeginTransaction()** – Begins a database transaction for this connection using isolation level
  - **Dispose()** - Releases the resources used by this connection.

2. Using Command Object

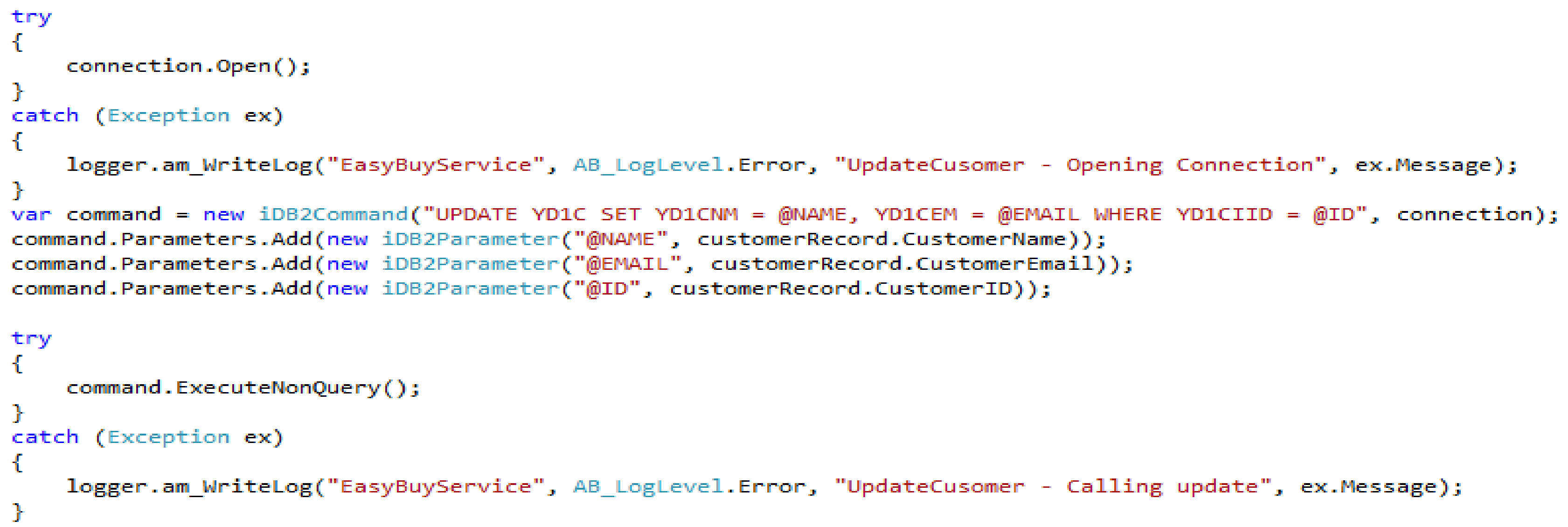
- Command object for DB2 for IBM i .NET Provider is **iDB2Command**
- In ADO.net, the Command Object is used to execute SQL statements or Stored Procedures against a data source
- Public Properties
    - **CommandText** – Contains the SQL statement or stored procedure to run against a data source
    - **CommandType** – Values of *StoredProcedure*, *TableDirect* or *Text* specify how the CommandText Property is interpreted
    - **ConnectionTimeout** – maximum number of seconds to wait for the command to execute before terminating and generating an error. 0 = wait indefinitely.
    - **Parameters** – Collection of iDB2Parameters. Used with Parameterized Queries.
- Public Methods
    - **Cancel()** – Attempts to cancel execution of a command
    - **Dispose()** – Releases the resources used by this command
    - **CreateParameter()** – Creates a new instance of an iDB2Paramter Object. The return value parameter could be input-only, output-only, bidirectional or a stored procedure. The default is Input.
    - **ExecuteNonQuery()** – Executes the command and ignores any result set
    - **ExecuteReader()** – Executes the command and returns an iDB2DataReader object
    - **ExecuteScalar()** – Executes the command and returns the first column of the first row in the result set

3. DataReader vs. DataSet

- **Use DataSet when you:**
  - are working with Tabular Data
  - Need minimal business logic
  - want to do minimal coding
  - Want to cache data locally in your application so that you can manipulate it
- **Use DataReader when you:**
  - need to quickly access data once, in a forward-only and read-only manner
  - are processing a large result set of data
  - want to use custom business entities
  - want to use a Service Oriented Architecture

4. Use Parameterized Queries

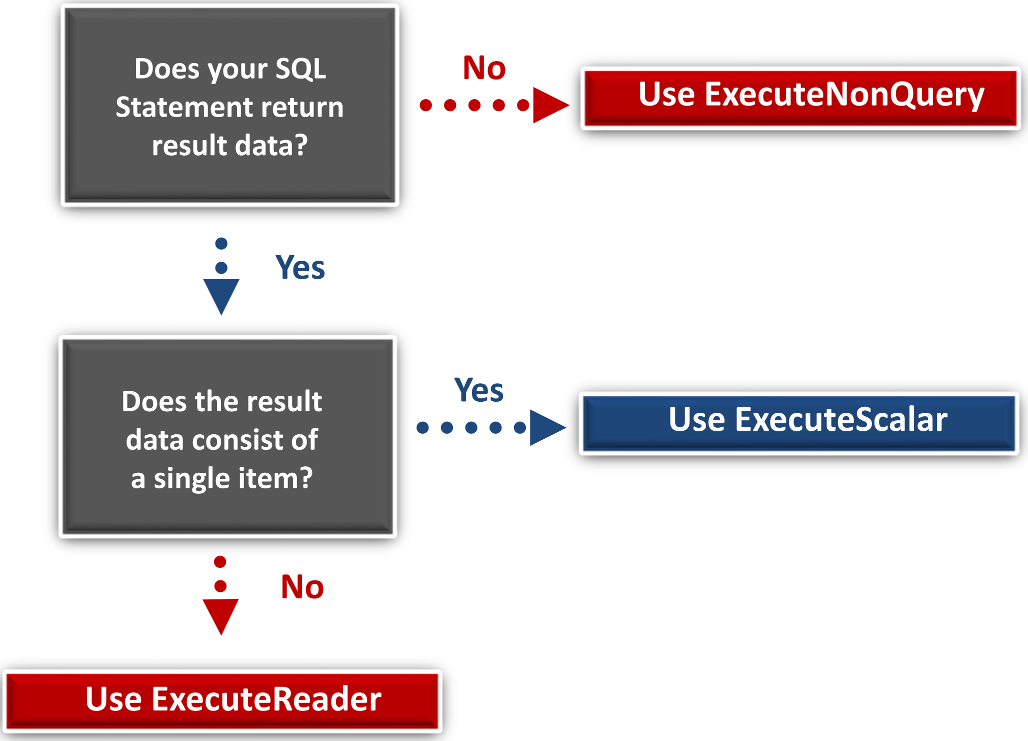
- Can improve performance
- Cleaner and more flexible code
- Prevents SQL Injection



```
try
{
    connection.Open();
}
catch (Exception ex)
{
    logger.am_WriteLog("EasyBuyService", AB_LogLevel.Error, "UpdateCusomer - Opening Connection", ex.Message);
}
var command = new iDB2Command("UPDATE YD1C SET YD1CNM = @NAME, YD1CEM = @EMAIL WHERE YD1CIID = @ID", connection);
command.Parameters.Add(new iDB2Parameter("@NAME", customerRecord.CustomerName));
command.Parameters.Add(new iDB2Parameter("@EMAIL", customerRecord.CustomerEmail));
command.Parameters.Add(new iDB2Parameter("@ID", customerRecord.CustomerID));

try
{
    command.ExecuteNonQuery();
}
catch (Exception ex)
{
    logger.am_WriteLog("EasyBuyService", AB_LogLevel.Error, "UpdateCusomer - Calling update", ex.Message);
}
```

5. Choosing Your Execution Method


Does your SQL Statement return result data?
No
Use ExecuteNonQuery
Yes
Does the result data consist of a single item?
Yes
Use ExecuteScalar
No
Use ExecuteReader

6. Close Connection and Dispose Objects

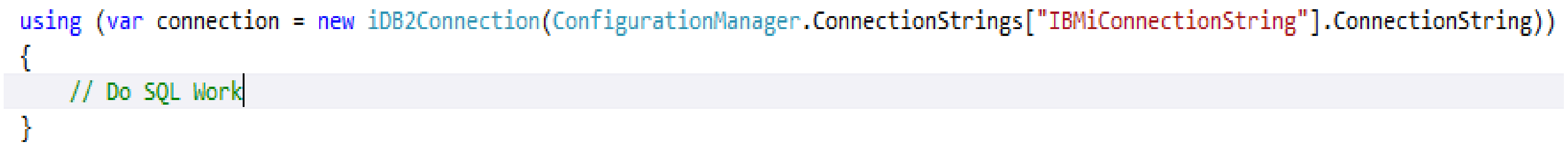
SURROUND TECHNOLOGIES™

- **Always Close() connection to the data source**
  - Keeping connection open wastes system resources and may even prevent others from connecting due to connection limits
  - Connection pooling will minimize the performance impact of opening and closing connections

- **Always Dispose() of Objects**
  - Cleans up resources associated with an object
  - When Disposing iDB2Command, the IBM i host resources associated with the object will be cleaned up

7. Using Statement in C# vs. Dispose() Method

- Declare and instantiate an IDisposable object in a using statement
  - Provides a convenient syntax that ensures the correct use of IDisposable objects
  - Causes the object itself to go out of scope as soon as Dispose() is called
  - Ensures that Dispose is called even if an exception occurs while you are calling methods in the object



```
using (var connection = new iDB2Connection(ConfigurationManager.ConnectionStrings["IBMiConnectionString"].ConnectionString))
{
    // Do SQL Work
}
```

8. Performing Transactions

- Ensures that a set of database operations are performed with integrity
- A set of database operations can be committed permanently are rolled back in case a failure occurs during a database operation
- All iDB2Commands will run under the same transaction within a connection that starts the transaction
- The table(s) that are used within a transaction must be journaled
- To cancel changes made during the transaction, use the transaction's Rollback() method
- To permanently commit changes made during the transaction to the database, use the transaction's Commit() method
- If Close() is issued during a transaction, it will be rolled back
- Supports only Local transactions (Single Connection) and not Distributed transactions (Multiple Connections)

9. Benefits of Stored Procedures

SURROUND TECHNOLOGIES™

- Reduced client/server traffic
- Performance
- Efficient reuse of code and programming abstraction
- Enhanced security controls
- Reduced development cost and increased reliability
- Centralized security, administration and maintenance
- Can call a program on the IBM i written in a supported programming language (COBOL, RPG, Java, C, IBM i commands, etc.)

How to Call programs on the IBM i

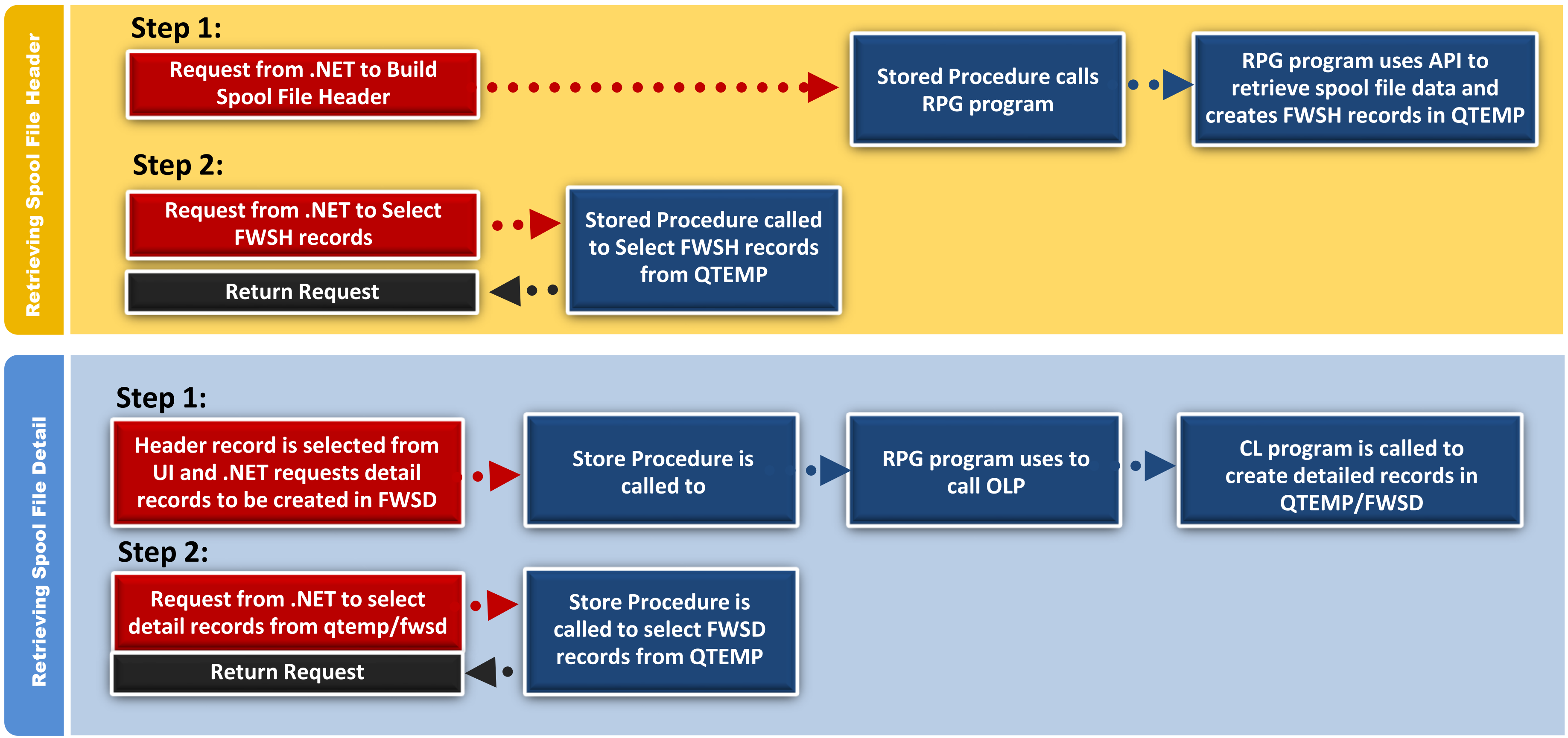
Calling Stored Procedures:

- 2 Types of Stored Procedures
  - External
  - SQL
- 2 methods
  - Use CommandType.Text
    - Set the CommandText to include the Call statement, the stored procedure name, and the parameter markers
  - Use CommandType.StoredProcedures
    - Set the CommandText to the stored procedure name only
- Use Execute Reader to read the results
- Use NextResult() Method to to read more than 1 result

Agenda Objectives:

- Best Practices for connecting .NET to the IBM I using the .NET data provider and following a Service Oriented Architecture
- Best Practices for accessing data using ADO.NET
- ➡ ***How to call programs using stored procedures***
- Ways to effectively reuse existing 5250 screens in .NET

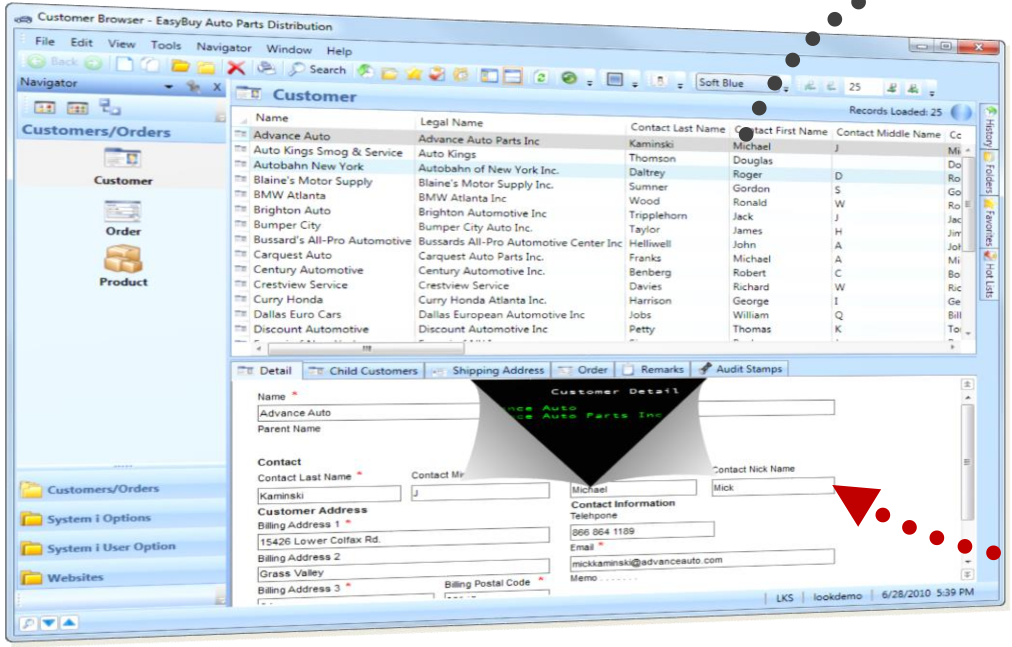
Spool File Retrieval


Retrieving Spool File Header
Step 1:
Request from .NET to Build Spool File Header
Stored Procedure calls RPG program
RPG program uses API to retrieve spool file data and creates FWSH records in QTEMP
Step 2:
Request from .NET to Select FWSH records
Stored Procedure called to Select FWSH records from QTEMP
Return Request
Retrieving Spool File Detail
Step 1:
Header record is selected from UI and .NET requests detail records to be created in FWSD
Store Procedure is called to
RPG program uses to call OLP
CL program is called to create detailed records in QTEMP/FWSD
Step 2:
Request from .NET to select detail records from qtemp/fwsd
Store Procedure is called to select FWSD records from QTEMP
Return Request

How to Call programs in the IBM i

Demo

Reuse 5250 Screens in .NET



Customer

Records Loaded: 25

Name	Legal Name	Contact Last Name	Contact First Name	Contact Middle Name	Cc
Advance Auto	Advance Auto Parts Inc	Kaminski	Michael	J	Mi
Auto Kings Smog & Service	Auto Kings	Thomson	Douglas		Do
Autobahn New York	Autobahn of New York Inc.	Daltrey	Roger	D	Ro
Blaine's Motor Supply	Blaine's Motor Supply Inc.	Sumner	Gordon	S	Go
BMW Atlanta	BMW Atlanta Inc	Wood	Ronald	W	Ro
Brighton Auto	Brighton Automotive Inc	Tripplehorn	Jack	J	Jac
Bumper City	Bumper City Auto Inc.	Taylor	James	H	Jim
Bussard's All-Pro Automotive	Bussards All-Pro Automotive Center Inc	Helliwell	John	A	Joh
Carquest Auto	Carquest Auto Parts Inc.	Franks	Michael	A	Mi
Century Automotive	Century Automotive Inc.	Benberg	Robert	C	Bo
Crestview Service	Crestview Service	Davies	Richard	W	Ric
Curry Honda	Curry Honda Atlanta Inc.	Harrison	George	I	Ge
Dallas Euro Cars	Dallas European Automotive Inc	Jobs	William	Q	Bill
Discount Automotive	Discount Automotive Inc	Petty	Thomas	K	To

Name *
Advance Auto
Parent Name

Contact
Contact Last Name *
Kaminski
Contact Mi
J
Michael
Contact Nick Name
Mick

Customer Address
Billing Address 1 *
15426 Lower Colfax Rd.
Billing Address 2
Grass Valley
Billing Address 3 *
Billing Postal Code *

Contact Information
Telehpone
866 864 1189
Email *
mickkaminski@advanceauto.com
Memo

LKS | lookdemo | 6/28/2010 5:39 PM

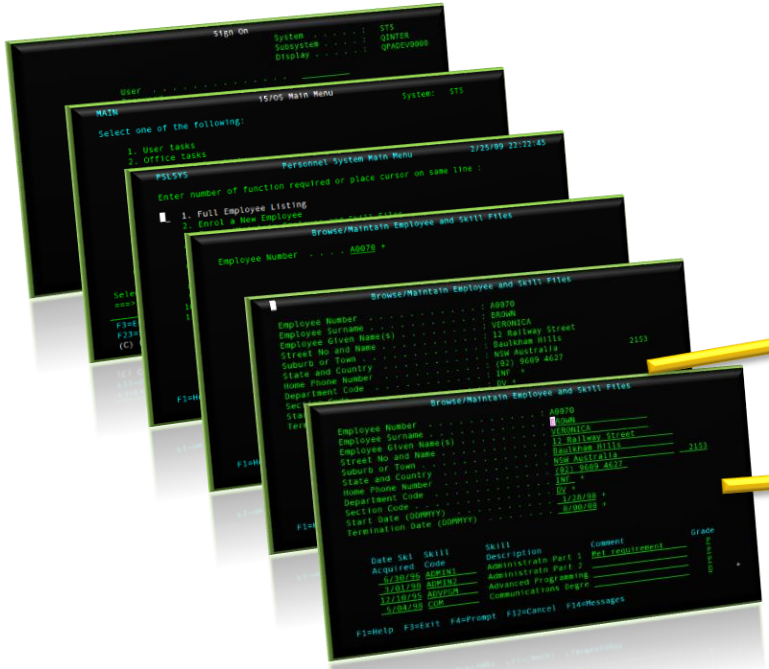
Agenda Objectives:

- Best Practices for connecting .NET to the IBM I using the .NET data provider and following a Service Oriented Architecture
- Best Practices for accessing data using ADO.NET
- How to call programs using stored
- ***Ways to effectively reuse existing 5250 screens in .NET***

Reuse 5250 Screens in .NET

SURROUND TECHNOLOGIES™

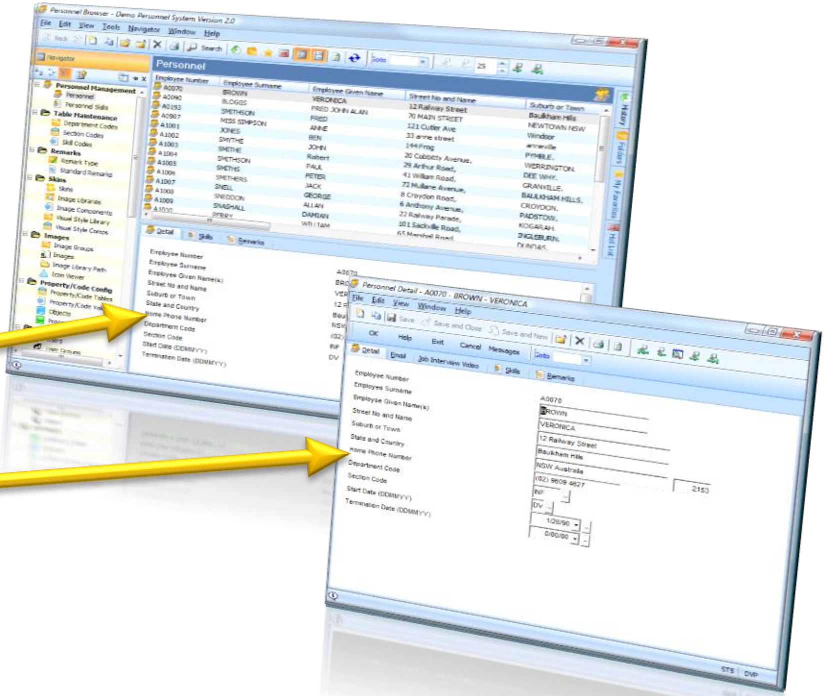
Go from these



```
                Browse/Maintain Employee and Skill Files
                                                     A0070
Employee Number . . . . . . . . . . . . . . . :     BROWN
Employee Surname . . . . . . . . . . . . . . :     VERONICA
Employee Given Name(s) . . . . . . . . . . . :     12 Railway Street
Street No and Name . . . . . . . . . . . . . :     Baulkham Hills              2153
Suburb or Town . . . . . . . . . . . . . . . :     NSW Australia
State and Country . . . . . . . . . . . . . :     (02) 9689 4627
Home Phone Number . . . . . . . . . . . . . :     INF  +
Department Code . . . . . . . . . . . . . . :     DV +
Section Code . . . . . . . . . . . . . . . . :      1/28/98 +
Start Date (DDMMYY) . . . . . . . . . . . . :      0/00/00 +
Termination Date (DDMMYY)
                                                                    Grade
                                Skill              Comment
   Date Skl   Skill             Description        Met requirement
   Acquired   Code              Administratn Part 1
    6/30/96   ADMIN1            Administratn Part 2
    3/01/99   ADMIN2            Advanced Programming
   12/10/95   ADVPGM            Communications Degre
    5/04/98   COM

 F1=Help  F3=Exit  F4=Prompt  F12=Cancel  F14=Messages
```

To this



Personnel Detail - A0070 - BROWN - VERONICA

Employee Number	A0070
Employee Surname	BROWN
Employee Given Name(s)	VERONICA
Street No and Name	12 Railway Street
Suburb or Town	Baulkham Hills
State and Country	NSW Australia
Home Phone Number	(02) 9609 4627
Department Code	INF
Section Code	DV
Start Date (DDMMYY)	1/28/90
Termination Date (DDMMYY)	0/00/00

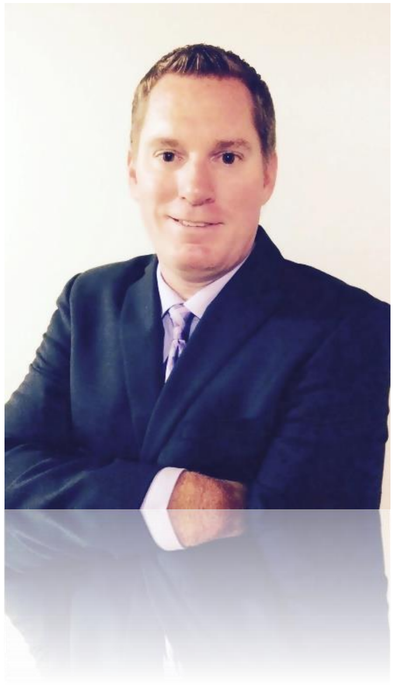
Personnel Browser - Demo Personnel System Version 2.0
File Edit View Tools Navigator Window Help
Personnel
Detail Email Job Interview Video Skills Remarks
OK Help Exit Cancel Messages
Save Save and Close Save and New
2153
STS DVP

Reuse 5250 Screens in .NET

Demo

Q&A

Thanks for watching!



Today's Speaker:

Derek Maciak

[CTO / Partner]

dmaciak@surroundtech.com | www.surroundtech.com

Socialize:

linkedin.com/company/128638

tweet me @SurroundTech

facebook.com/surroundtech

SURROUND TECHNOLOGIES™

Important Links

- IBM i Access
  - http://www-03.ibm.com/systems/power/software/i/access/index.html
- Redbook – Integrating DB2 Universal database for iSeries with Microsoft ADO.NET
  - https://www.redbooks.ibm.com/abstracts/sg246440.html