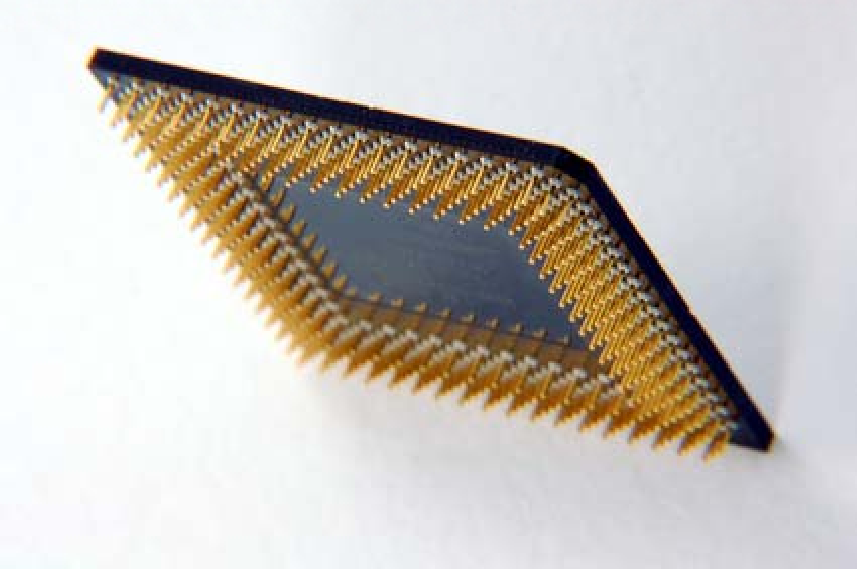
[A3]
Delphi / C++Builder XE2
Deep Dive

David Intersimone "David I"
Vice President of Developer Relations and Chief Evangelist
Embarcadero Technologies
davidi@embarcadero.com

- Delphi 64-bit
- VCL Styles
- RTL
- LiveBindings
- RAD Cloud
- DataSnap

Delphi 64-bit



Delphi 32 and 64-bit Type Sizes

Signed types	Delphi/32	Delphi/64
ShortInt	1 byte	←
SmallInt	2 bytes	←
LongInt	4 bytes	←
Integer	4 bytes	←
Int64	8 bytes	←

Unsigned types	Delphi/32	Delphi/64
Byte	1 byte	←
Word	2 bytes	←
LongWord	4 bytes	←
Cardinal	4 bytes	←
UInt64	8 bytes	←

Delphi 32 and 64-bit Type Sizes

Signed types	**Delphi/32**	**Delphi/64**
NativeInt	**4 bytes**	**8 bytes**

Unsigned types	**Delphi/32**	**Delphi/64**
NativeUInt	**4 bytes**	**8 bytes**

Delphi 32 and 64-bit Pointer Types

Pointer types	**Delphi/32**	**Delphi/64**
Pointer		
String		
Class instance		
Class reference		
Interface		
AnsiString	**4 bytes**	**8 bytes**
WideString		
UnicodeString		
Procedure pointer		
Dynamic array		
PAnsiChar		
PWideChar		
PChar		

Delphi 64-bit on Windows

- Same Windows API
  - – CreateWindowEx, PeekMessage, etc..
- Same Delphi RTL
  - – SysUtils, Classes, Generics.Collections,
- Same VCL
  - – Forms, Graphics, Controls, Menus, etc..

What Can Do in your Code?

- Find all Integer<->Pointer casts, including Integer<->instance casts.
  - Check for Pointer size assumptions
- Ensure external dependencies are also 64bit
  - Image/bitmap libraries
  - Hardware interfaces libraries
  - ActiveX controls
- Consider rewriting Assembler in pure-Pascal
  - Better future portability (think ARM CPUs…)
  - Rely more on algorithmic performance rather than raw assembly performance.
- Make sure you know what bit-size (32 vs 64) you’re shifting with SHL/SHR.

Demonstration

VCL Styles

Themes in Delphi/C++Builder XE and earlier

- Only 1 "theme" supported (native Windows)
- 3rd party products required to change look of VCL apps
- Current “skinning" products for Windows common controls
  - Requiring hooking WndProc's
  - Intercepting control messages
- Separate controls needed for "custom" controls
  - TCustomControl / TGraphicControl descendants

VCL Styles

- Multiple styles provided/supported
- Custom style file creation
- Custom style classes
- Pluggable style engine
- Works with TWinControl and TGraphicControl descendants

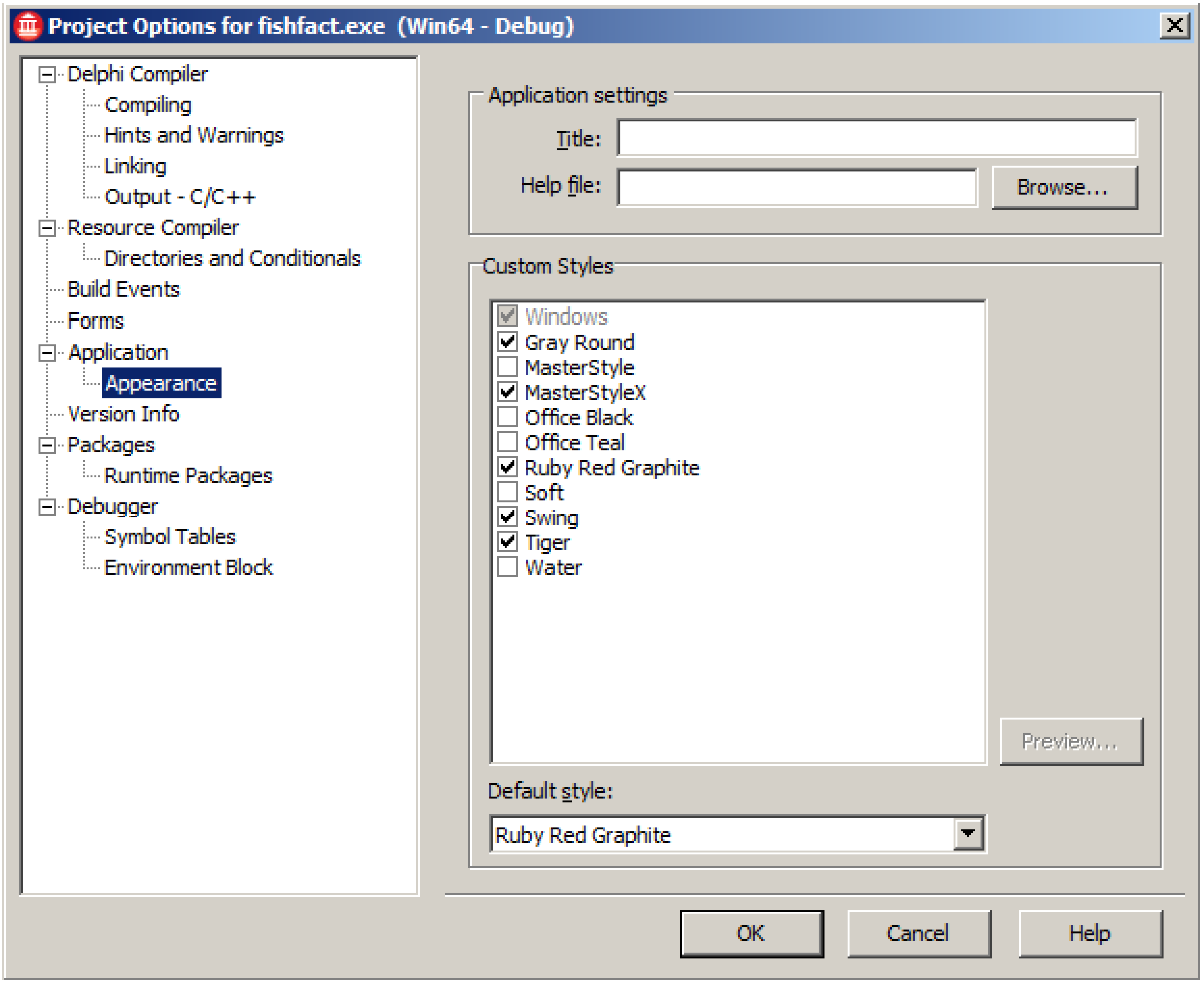
VCL Styles

- TThemeServices replaced by TCustomStyleServices
  - Extends TThemeServices with more parts, states and methods
- TStyleEngine
  - Processes control messages
  - Calls StyleServices to render parts
- TStyleManager
  - Provides services such as
    - LoadFromFile
    - LoadFromResource
    - RegisterStyle
    - RegisterStyleClass - Associates file extension with Style class
    - SetStyle
    - Style selection
    - Style engine selection
- Providing custom styles
  - Create style file (using provided style building tool)
  - Implement TCustomStyleServices descendant and render parts directly

What can’t be styled?

- TMainMenu/TPopupMenu (main menu bar is styled, but not the pop-up parts)
- Ribbon controls
- OS dialogs

Project Options | Appearance



Project Options for fishfact.exe (Win64 - Debug)

Application settings

Title:

Help file:

Custom Styles

☑ Windows
☑ Gray Round
□ MasterStyle
☑ MasterStyleX
□ Office Black
□ Office Teal
☑ Ruby Red Graphite
□ Soft
☑ Swing
☑ Tiger
□ Water

Default style: Ruby Red Graphite

Typical Usage

- Uses Vcl.Styles, Vcl.Themes;
- TStyleManager.SetStyle(‘Ruby Red Graphite’);

Demonstration

RTL

What’s new in the RTL?

- 64-bit for Windows!!!
- MacOSX - No COM, No ActiveX. :)
  - Windows specifics has been mapped to MacOSX equivalents wherever possible
- TOSVersion
  - Architecture (32/64-bit), Platform (Win/Mac), Major, minor OS version, Service Pack info
- TFormatSettings
  - replaces global format setting variables (that don't exist on MacOSX)
- TZipFile - cross platform Zip support
- SymLinks in IOUtils and SysUtils file handling routines
  - class function TFile.CreateSymLink(const Link, Target: string): Boolean;
- TLoginCredentialService
  - Extensible framework agnostic login credential services with support for callbacks (success, failure, etc)
- TPoint, TRect, TSize - new methods and properties
- TPointF, TRectF, TSizeF - floating point versions

LiveBindings

What is LiveBindings?

“Provides a way for developers to create a read/write link between components/**objects** and their **data** in the the application.”

LiveBindings Overview

- Supported by FireMonkey and VCL
- Based on relational expressions, called binding expressions
  - Can be unidirectional or bidirectional
- Can be used programmatically or visually
- **Is all about control objects and source object**

LiveBinding Engine

- Allow developers to create live binding in run-time between components/objects at design-time.
- The LiveBinding visual components use the engine:
  - TBindScope for bindings to components
  - TBindScopeDB for bindings to databases
  - TBindingsList for holding of binding lists

- A number of conversion services in the box and allows to create new conversion service
- Automatic updates
  - Expressions are automatically re-evaluated when a variable (e.g.; property) referenced by an expression has changed value.
  - Requires external notification

LiveBinding Engine

- Allow developers to create live binding in run-time between components/objects at design-time.
- The LiveBinding visual components use the engine:
  - TBindScope for bindings to components
  - TBindScopeDB for bindings to databases
  - TBindingsList for holding of binding lists

- A number of conversion services in the box and allows to create new conversion service
- Automatic updates
  - Expressions are automatically re-evaluated when a variable (e.g.; property) referenced by an expression has changed value.
  - Requires external notification

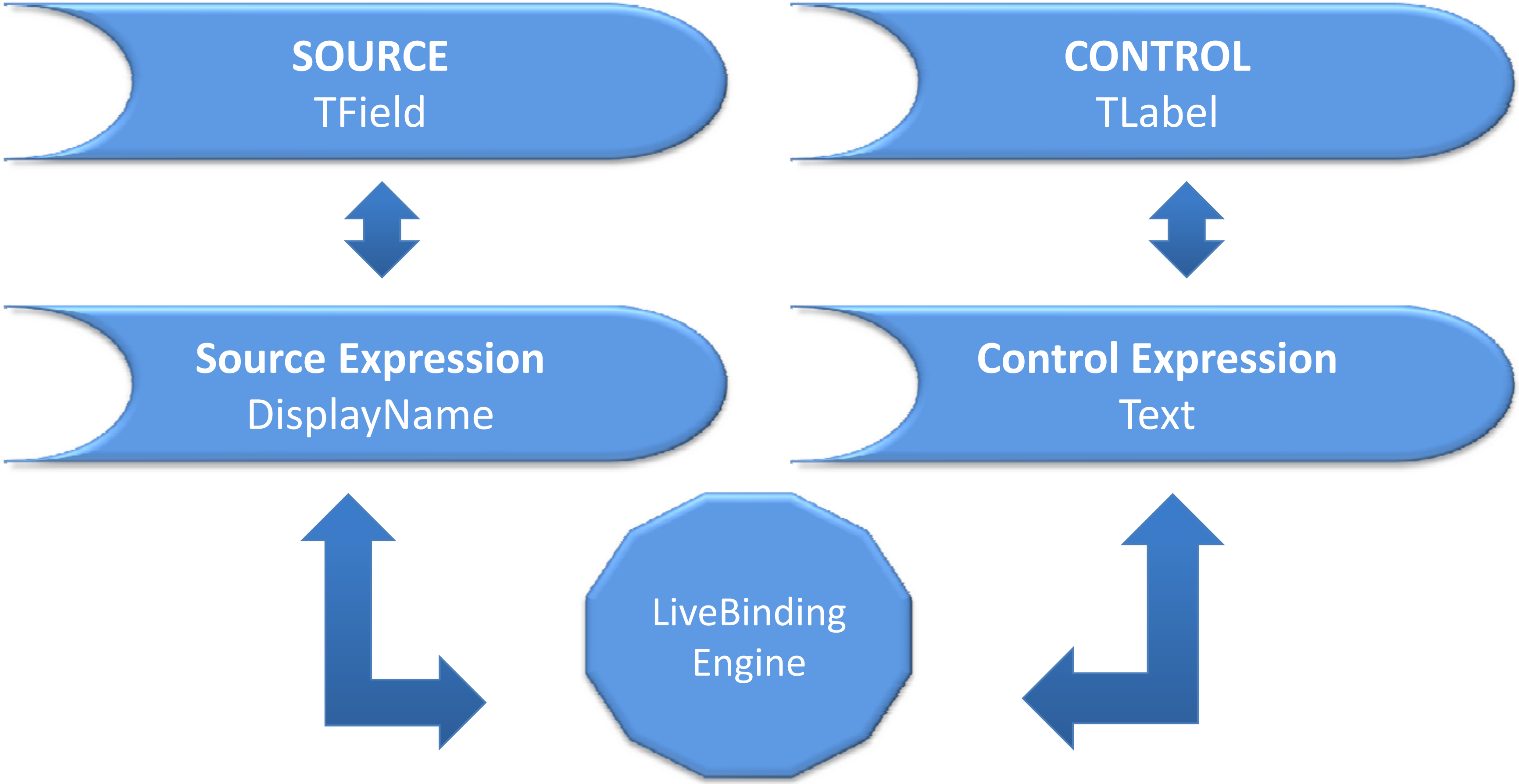
LiveBinding components

- TBindScopeDB – magic binding
  - Bind data from DataSet to components
  - “Link to DB Field” wizard create the expressions to bind the data from DataSet to components

- TBindScope – non-dataset binding
  - Bind data between components/objects
  - Support single object or collection of objects

- TBindingsList
  - Hold the binding lists
  - Manage the list of methods and output converters used by the expression

LiveBinding Scenario



SOURCE
TField

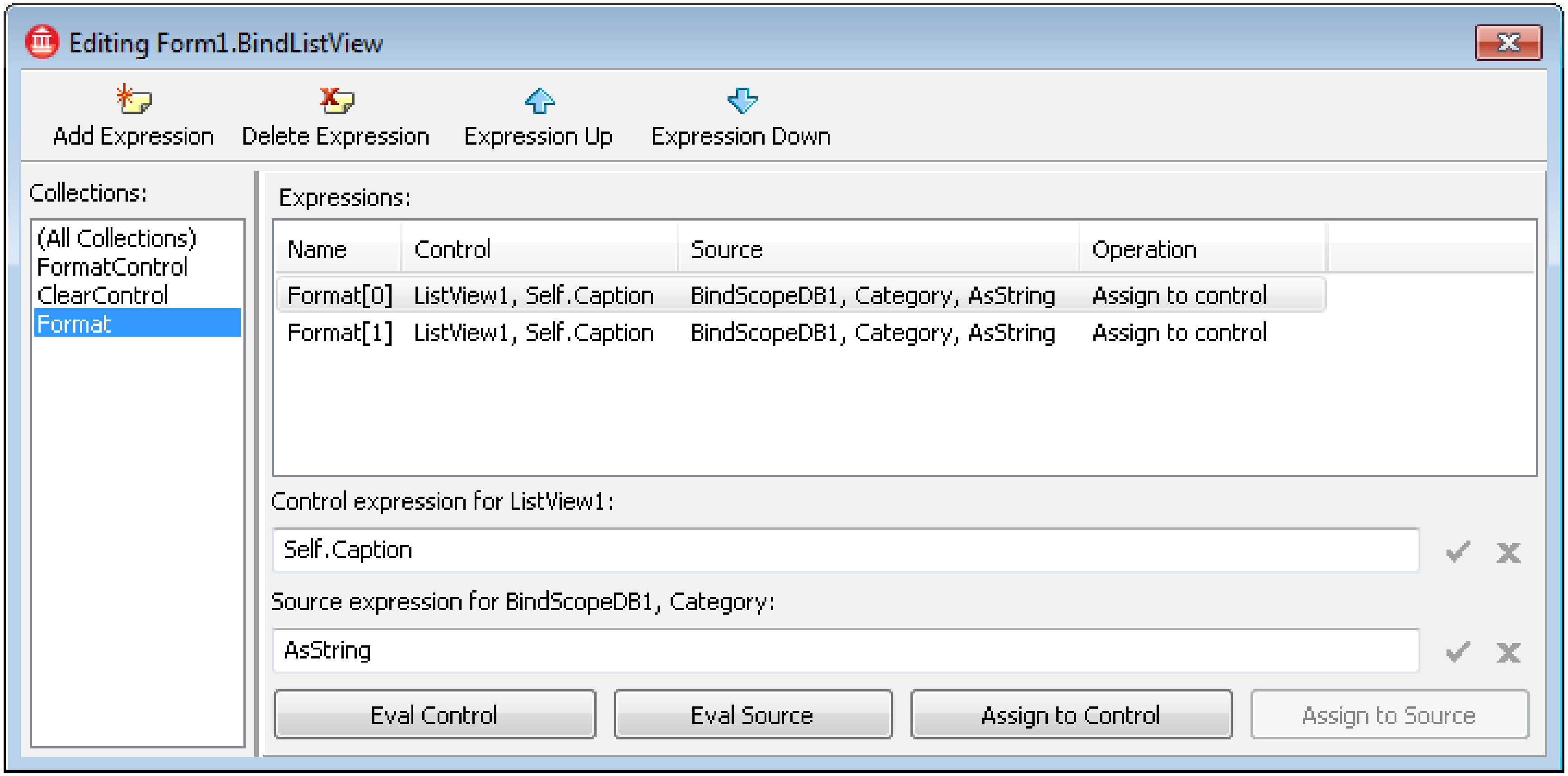
CONTROL
TLabel

Source Expression
DisplayName

Control Expression
Text

LiveBinding
Engine

Binding Expression Editor



Name	Control	Source	Operation
Format[0]	ListView1, Self.Caption	BindScopeDB1, Category, AsString	Assign to control
Format[1]	ListView1, Self.Caption	BindScopeDB1, Category, AsString	Assign to control

Control expression for ListView1:

Self.Caption

Source expression for BindScopeDB1, Category:

AsString

Code Sample – Binding an object

```
var
  BindingExpression2 : TBindingExpression;
begin
   BindingExpression2 := TBindings.CreateManagedBinding(
   { inputs }
   [TBindings.CreateAssociationScope([
     Associate(MyObject1, 'o1'),
     Associate(MyObject2, 'o2’) ])], 'o1.StringValue + o2.StringValue',
   {output}
   [TBindings.CreateAssociationScope([
     Associate(MyResultObject, 'res’) ])], 'res.StringValue’,          nil);
```

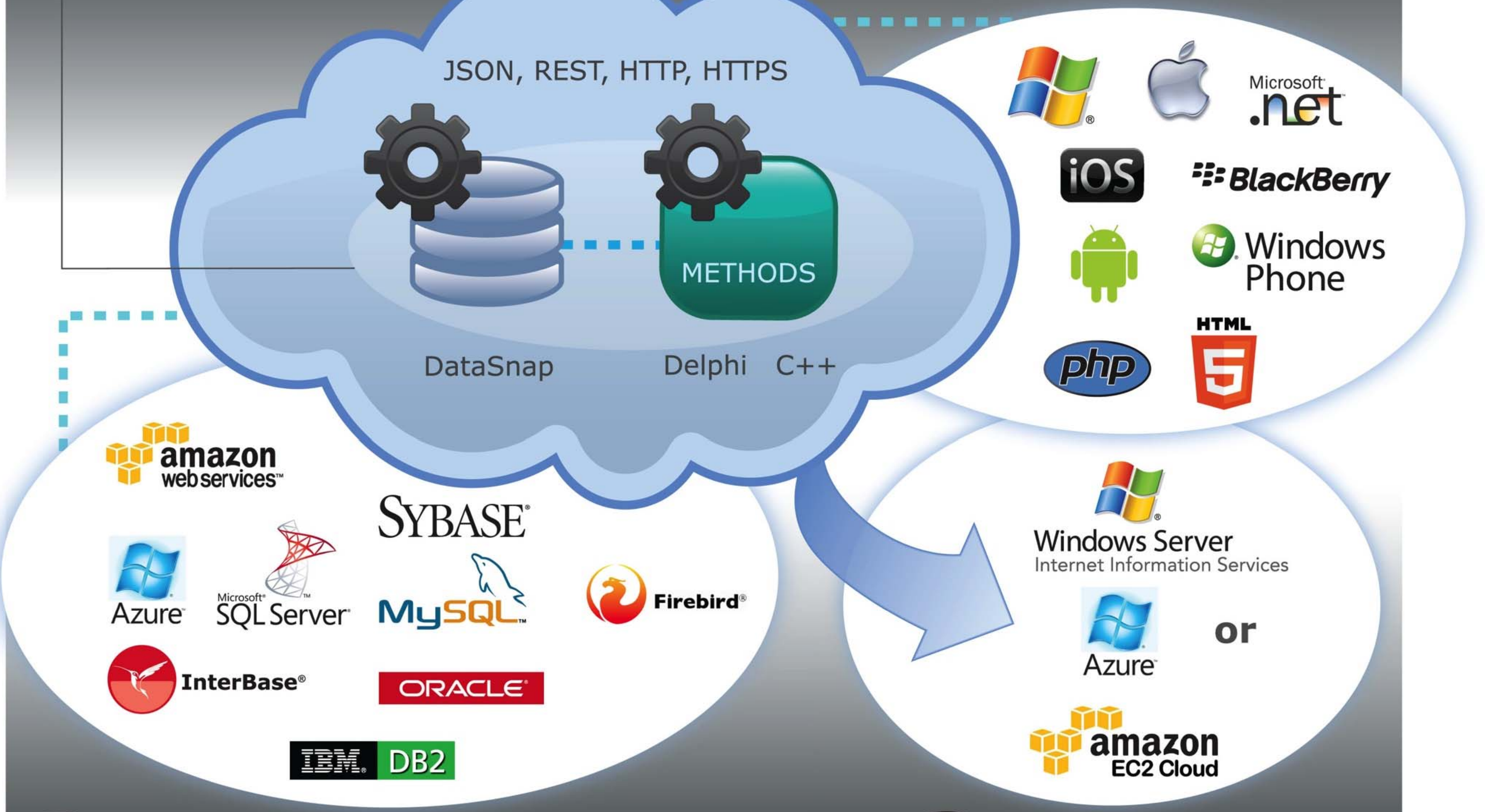
Demonstration

RAD Cloud

RAD Cloud Services

1 Build RAD Cloud services in Delphi® or C++Builder®

4 Connect to your data and services from any platform



2 Connect to back-end services and databases

3 Deploy on Windows, IIS, Azure or Amazon EC2

Cloud API service support

	Windows Azure	amazon web services
Storage	Azure Blobs Service	Amazon S3
Table	Azure Table Service	SimpleDB
Queue	Azure Queue Service	Amazon SQS

Name conventions

- Amazon and Microsoft use different names for the same things

Windows Azure		amazon web services
Container	=	Bucket
Blobs	=	Object
Page	=	Multipart
ACL	=	ACL (Access Control List)

Microsoft Azure

- Blob Storage
  - Unstructured storage
  - Defined Container
  - For storing large pieces of data such as: Images, Video, Documents, etc.
- Azure Tables
  - Scalable structured storage
  - NOSQL type tables
  - Entities stored in a table
  - Different set of properties made up of different types
- Queues
  - Persistent asynchronous messaging
  - Each message is up to 8kb long

Amazon WS

- Simple Storage Service (S3)
  - Blob Storage
  - Unstructured storage
  - Defined Container
  - For storing large pieces of data such as: Images, Video, Documents, etc.
- SimpleDB - Tables
  - Scalable structured storage
  - NOSQL type tables
  - Entities stored in a table
  - Different set of properties made up of different types
- Simple Queue Service (SQS)
  - Persistent asynchronous messaging
  - Each message is up to 64kb long

Deploy to the Cloud

- Deployment Manager support:
  - Amazon EC2
  - Microsoft Azure
- Requires Platform Assistant

What's New in DataSnap

What’s new in DataSnap

- Enhancements
  - Server
  - REST Interfaces
  - Callback
  - Authentication Manager
- New DataSnap Monitoring and Control

Server

- Proxy info support for DataSnap HTTP connections in TSQLConnection and TDSRESTConnection
- New TDBXChannelInfo.DBXClientInfo (TDBXClientInfo record) provides:
  - IP Address
  - Protocol
  - Application Name (when possible) – only for HTTP

HTTPS Support

- For Stand-alone DataSnap servers
- For Stand-alone WebBroker servers
- DataSnap Wizards updated with HTTPS support

Session Manager

- Session support for TCP/IP protocol
- Support for store and retrieve TObject in Sessions
- Sessions created and destroyed with the TCP/IP protocol now notifies the AddSessionEvent and RemoveSessionEvent
- Support for multiple Callback Tunnels on DataSnap REST Interface

Callbacks

- Heavyweight callbacks now support broadcasting to specific callbacks
- GetAllChannelNames added to the DSServer to get the name of all callback channels

Callback client side events

- Notifications whenever the manager's tunnel is opened or closed, or when callbacks are added to it or removed from it. Available for:
  - DBX Client Events
  - REST Client Events
  - JavaScript REST Client Events

DataSnap Monitoring and Control

"DataSnap servers in XE2 are able to easily monitor connections, and close any TCP connection they wish."

What you can do

- Take control over the client connections
- Monitor client requests
- Limit the number of requests per connection
- Improve Security
- More…

How it works?

- TDSTCPServerTransport components has two new events:
  - onConnect
  - onDisconnect
- Connections are linked with a Session ID

Detecting Graceless Disconnects

- By default, the OnDisconnect event will not be notified if the client abruptly loses his internet connection
- This behavior depends on OS configuration
- Use TDSTCPChannel methods to control this behavior for each connection:
  - EnableKeepAlive
  - DisableKeepAlive

KeepAlive Enablement

- Available through TDSTCPServerTransport component
- Three new published properties
  - KeepAliveEnablement - (Set: Default, Enabled, Disabled)
  - KeepAliveTime - Integer Milliseconds value
  - KeepAliveInterval - Integer Milliseconds value

Closing Client Connections

- Connection can be closed any time
- Channel provide the connection instance

```
//Get the associated Channel for the given connection, and if successful close it
if FConnections.TryGetValue(Connection, Channel) then
   Channel.Close;
```

Demo

- Tracking all the TCP/IP connections and their corresponding sessions
- How to manually close a TCP/IP connection from the server
- How to detect when a client has lost internet connectivity while connected.

Q&A

Thank You ☺