SYNERGY 2015

SEATTLE, WA, USA

SYNERGY2015

SEATTLE, WA, USA

DataFlex 2014 / 18.1 Overview

Presented by: John Tuohy

DataFlex 18.1

- DataFlex 18.1 alpha 2 is now available for download
- It's most exciting addition is the support for web mobile/touch applications
  - Mobile/touch will be talked about extensively during the conference
- There's a lot more to 18.1 we want you to know about

- And that's the purpose of this talk...

New Compile Time Function

- RefTable()
  - Used to reference Table's file number

    Move (RefTable(Customer)) to iTable

  - Used to reference Table's column numbers

    Move (RefTable(Customer.Name)) to iColumn

    - Supported by code-sense
    - Adds to existing compile time functions – RefClass(), RefProc(), RefFunc(), RefProcSet()
    - Can be used in place of Get_FileNumber, Get_FieldNumber

RefTable() Example Usage

Example 1

```
Get_Attribute DF_Field_Length of Customer.File_Number (RefTable(Customer.Name)) to iLength
<or>
Get_Attribute DF_Field_Length of (RefTable(Customer)) (RefTable(Customer.Name)) to iLength
```

Example 2

```
Function Field_CommitNoEnterOnIndex Integer iField Returns Boolean
    Function_Return (RefTable(OrderHea.Order_Date)=iField)
End_Function
```

Example 3

```
Procedure ClearNamePrompt
    Set Prompt_Object of oCustomerDD  (RefTable(Customer.Name)) to hoMyPrompt
End_Function
```

Better Text and Binary Data Management

- Reading and writing text and binary data can be a challenge
  - Data can be very large
  - Data can contain embedded zeros

- Possible Solutions
  - Strings
  - Memory
  - UChar Arrays

UChar Array Changes

- The UChar Array

  UChar[] MyArrayOfBytes

  - A UChar array is an array of bytes:
    - Can represent character strings
    - Can represent binary data
    - Can be very big
  - Before 18.1, UChar arrays did not interoperate easily with the parts of DataFlex that read and write data
  - Now they do

UChar Array Changes

- New UChar Array command support for
  - Get_Field_Value
  - Set_Field_Value
  - Read_Block
  - Write

- New UChar Array Function support
  - AppendArray()
  - StringToUCharArray()
  - UCharArrayToString()

- This really opens things up!

UChar Array Changes

- For example, let's read a PDF file and write it to a database

```
Procedure ReadFromPDFFile String sOutFile
    UChar[] PDFManual
    // read a sequential file
    Direct_Input Channel 5 ("binary:" + sOutFile)
    If (SeqEOF) Begin
        Procedure_Return
    End
    // read entire file into UChar array
    Read_Block Channel 5 PDFManual -1
    Close_Output Channel 5
    // write this to my datafile
    Set_Field_Value (RefTable(Product))  (RefTable(Product.PdfManual)) to PDFManual
End_Procedure
```

New Struct and Array Functions

- AppendArray()
  - Appends two arrays
- IsSameArray()
  - Returns true of two arrays are the same (including array of structs)
- IsSameStruct()
  - Returns true of two structs are the same
- BinarySearchInsertPos()
  - Returns insert position for target data for missing data in sorted list
  - Used with SearchArray() and InsertInArray() to insert data in a sorted list

New Struct and Array Functions

- Using BinarySearchInsertPos()
  - How to insert new value into a sorted list

```
// this assumes your array is already sorted

Move (BinarySearchArray(SearchData,DataArray)) to iIndex

// if key not found, add it
If (iIndex = -1) Begin
    Move (BinarySearchInsertPos()) to iIndex
    Move (InsertInArray(DataArray, iIndex,SearchData)) to DataArray
End

// iIndex will contain the position of the data
Function_return iIndex
```

Struct and Array Function Changes

- InsertInArray()
  - Passing array size appends to end of array
  - Passing -1 appends to end of array
- RemoveFromArray()
  - Passing -1 removes from end of array

Array Search and Sort Changes

- The DataFlex search and sort functions are:

```
SortArray()
SearchArray()
BinarySearchArray()
```

- These functions have two different syntaxes for working with simple data types and struct (complex) data types

```
SortArray(Target, SourceArray)
SortArray(Target, SourceArray, oSearchObject, RefFunc(MySearchFn))
```

- Working with the complex style, is more complex and slower
- Now you can use the simple syntax in more cases

Array Search and Sort Changes

- Now you can use the simple syntax with structs if:
  - The evaluation is based on the first member of the struct
  - The first member of the struct is a simple type (e.g., String)
- Using the simple syntax is not only simpler, but faster
- If these conditions are not met, you can continue to use the complex syntax
- This makes it easier and faster to perform searches and sorts on “key/data” style arrays

Data Dictionary Changes

- Data Dictionaries are faster
  - Uses a new internal cDDBuffer class
  - No code changes required

  - How will this impact your application?
    - DDOs are already pretty fast, when will you see this?
      - During DD batch update operations
      - You are processing a lot of records
      - You are processing tables with a lot of parents
      - Tables with a very large number of columns

Data Dictionary Changes

- Data Dictionary objects use less memory with large fields
  - Extended DD Fields are used for Binary and Text
  - They now only use memory needed and not the maximum possible
  - This can greatly reduce memory usage
  - No code changes required

  - This is possible because of the new UChar array changes

Data Dictionary Changes

- Cascade deletes are now more flexible
  - Allowing cascade deletes can now be defined at the child level
    - This is the child DDO that has records being deleted as part of the cascade
    - The child can define different cascade delete rules for each parent
    - This puts the cascade delete rule in the proper place

    - Set via new DD method: CascadeDeleteAllowed

```
// In CustomerNotes DD Class
Set CacadeDeleteAllowed (RefTable(Customer)) to True
:
// In OrderHea DD class
Set CacadeDeleteAllowed (RefTable(Customer)) to False
```

Data Dictionary Changes

- Cascade Deletes and Null Parents
  - A child cascade delete can now just set a null relationship instead of deleting
  - Cascade delete can now set a child's relationship to null instead of deleting the child
    - The child can define different rules for each parent
    - Applies only the parents that allow null relationships for that parent
    - Set via new DD method: CascadeDeleteParentNull

```
// In OrderHea DD
Set ParentNullAllowed (RefTable(SalesP)) to True
Set CascadeDeleteParentNull (RefTable(SalesP)) to True
```

Data Dictionary Changes

- More flexible support for displaying committed fields
  - If a field is committed
    - All non-indexed fields are shown in DEOs as display-only
    - All indexed fields are shown in DEOs as no-put
      - This provides the flexibility to use these DEO fields for record finding
      - Some developers would prefer that these fields appear as display-only
      - You now have the choice
  - Field_CommitNoEnterOnIndex
    - This function lets you control this
    - This can be applied at the DD class or DD object level
    - This can be applied to all committed fields or selected fields

Improvements in Handling of Deleted Records

- The Problem
  - A user deletes a record that some other user has accessed
  - When a DDO cannot refind that record is raises an unhandled error
  - When this happens the end user sees one or more strange unhandled error
  - You get a support call

- This is a long standing issue. It doesn't happen often but when it does, it's confusing

Improvements in Handling of Deleted Records

- This has been significantly improved in 18.1
  - When possible it just silently adapts to the change
  - If an error is presented, it's a handled (i.e., expected) error
  - No or little programming changes required
  - Implemented on Windows and Web

Connectivity Changes

- There are a number of new changes in connectivity concerning record identity

- First we need to discuss a few things

Understanding Key Fields

- Primary Keys
  - A table should have a unique key that defines each record's identity
  - This is called the primary key
  - This key can be single segment or multi segment
  - Primary key values should not change
  - The primary key must have an index
  - Primary keys should be short as possible
  - Primary keys values should be "balanced"
- Foreign Keys
  - Primary keys are pointed to by fields in other tables
  - These are called foreign keys
  - Foreign keys are usually indexed

Types of Primary Keys

- Primary Keys can be Natural or Artificial

- Natural keys (Business keys)
  - A Natural key's value is something that makes sense in the real world
    - e.g., a State abbreviation, someone's initials
  - These are usually assigned by the user
  - They have a meaningful order

- Artificial Keys (Surrogate keys)
  - Their value has no meaning in the real world
  - They tend to be system assigned
  - They may not have a meaningful order

Types of Primary Keys

- Should you use Natural or Artificial Keys?
  - This is a very old topic
  - This can be a very touchy subject
  - You probably know all the arguments
  - We offer no recommendations, but let's talk more about artificial keys

Types of Artificial Keys

- Recnum
  - They are auto assigned, short and fast
  - Assigned when new record is saved
  - They do not require a system table
  - They are unique to the embedded database
  - Value not known until after save
  - Developers generally avoid using these as Primary Keys. Why?
    - They are fragile
    - They don’t migrate easily to other tables
    - They don’t migrate easily to other databases
    - We agree

Types of Artificial Keys

- System-Table Key Values
  - They are system table assigned, short and fast (usually integers)
  - Assigned when new the record is saved
  - Assignment generally handled by your Data Dictionary
  - It requires locks on a system table with each new save
  - They can migrate to other databases
  - Value not known until after save
  - Widely used

Types of Artificial Keys

- SQL Identity
  - They are server assigned, short and fast
  - Assigned when a new record is saved
  - They are “balanced”
  - Supported on most SQL platforms
  - Does not require a system table
  - Value not known until after save
  - Robust
  - Very widely used for Primary Keys
  - An improved recnum

Types of Artificial Keys

- SQL GUIDs
  - Are universally unique
  - They look like: 60FCEA2E-CFFA-4788-AB32-6B81F37D3FFA
  - Computer assigned
    - Can be assigned by the SQL Server or can be assigned by the application
  - They are big
    - Does this matter?
  - They are Pure
    - A GUID is a truly meaningless value
    - You'd never show it, memorize it or enter it
    - There is absolutely no order to it
  - Can be created, assigned and discarded at any time
  - No system table required
  - They are Unbalanced (but there's help for that)

RowId and Keys

- Where does RowId fit in?

  - RowId is an abstraction of a Key
  - They can be used with recnum or standard tables
  - Works with any type of Primary Key table
  - This allows you and us to write code that works with any database
  - They can completely replace the need for recnum
    - The DataFlex runtime is completely RowId based
    - You can replace all recnum references in your code with RowId references
    - This makes it easier to move to and work in an SQL Database

  - They are the “one ring to rule them all”

  - We recommend moving to full RowId usage

Connectivity Changes

- Better SQL Primary Key Support
  - Primary keys can now be assigned in Table Editor
  - Primary keys are recognized when connecting to an existing SQL Table
  - Primary keys can be assigned when converting from embedded to SQL
  - They are supported with standard and recnum tables

  - New Attribute: DF_INDEX_SQL_PRIMARY_KEY

Connectivity Changes

- Better SQL Clustered Index Support
  - What is a clustered index
  - Clustered indexes can now be assigned from Table Editor
  - Supported with standard and recnum tables
  - New Attribute: DF_INDEX_CLUSTERED

Connectivity Changes

- Better SQL Identity Column Support
  - Identity columns can be assigned in Table Editor
  - Might require changes in your Data Dictionary (remove auto-increment)
  - Standard tables only

- New Attribute: DF_FIELD_IS_IDENTITY

Connectivity Changes

- Better GUID Support
  - GUIDs can be assigned as primary Keys
    - IDs can be assigned via DD code
    - IDs can be assigned by the server via DF_FIELD_DEFAULT
      - [newid()]
      - [newsequentialid()]
  - GUIDS can be also be assigned to non-primary keys

Connectivity Changes

- Microsoft Azure SQL Support
  - Azure requires that all tables contain a clustered index
  - Even System Tables require a clustered index

  - This is now supported

Connectivity Changes

- Other
  - Connection and Conversion Wizards have been improved to support new features and be easier to use
  - Triggers and Foreign Keys are preserved during table changes
  - Cache time-outs have been fine tuned to improve performance
  - Improved large data handling
    - Get_Field_Value and Set_Field_Value UChar array changes

ValueTree Support

- ValueTrees are used to convert DataFlex data types (simple, structs and arrays) to a single format.
- That format is the ValueTree struct
- Because the ValueTree format is known, it can be saved, loaded and transmitted using a pre-defined format
- We use this extensively in Web Services (working with XML)
- We use this extenisvely in the Web Framework (working with JSON)
- This has always been a private format
- As of 18.1, it is now public. This includes:
  - The ValueTree Struct definition
  - The ValueTreeSerializeParameter command
  - The ValueTreeDeserializeParameter command
- This is an advanced technique
  - You may not directly see the benefit of this
  - You will see indirect benefits from DAW and from the community

Studio and Debugger

- Studio
  - The Web previewer is now an interactive designer
    - You can move and resize controls
    - You can drag and drop on to the designer
    - Specially designed to use “flow” design and display properties (piColumnCount, piColumnIndex, piColumnSpan)

- Debugger
  - Initializes faster / runs faster
  - Err Indicator can be used in the watch window
  - Add to Watch makes it easier to add items to the watch window
  - Char/UChar variables are better supported (display and editing)

Other Changes

- COM Anchor Fix
- Modal Dialog Fix
- Web Services Fixes
- Migration Improvements
- Documentation Improvements

- Read the What's New for more details

Web Application General Changes

- Improved theme styling with Font-Icon images
  - This scales really nicely and looks sharp on all devices
- DD Remember
  - DD Remember can now be added to your applications
- List (cWebList) changes
  - Multi-line row support
  - Fixed width columns (good for fixed images)
- Complex Web Properties are now supported
  - Web Properties now support Structs and Arrays
  - This makes it much easier to store larger amounts of data on your client
  - As easy to use as simple Web Properties (just add the WebProperty=True meta-tag)

Web Application General Changes

- Complex Web Properties are now supported
  - Web Properties now support Structs and Arrays
  - This makes it much easier to store larger amounts of data on your client
  - As easy to use as simple Web Properties (just add the WebProperty meta-tag)

```
{ WebProperty=True }
Property Struct[] pMyImportantData
```

- Misc Changes
  - Better Thread Stack management
    - The thread stack size has been reduced to allow many more instances
    - This is configurable
  - Larger cookies are supported
  - Lots of minor bug fixes, browser support improvements and other tweaks

- And...

Web Mobile/Touch Support

- DataFlex 18.1 provides full support for mobile/touch web devices
- It provides everything you need to build great looking mobile/touch applications using DataFlex, the technology you know

Web Mobile/Touch Overview

- Drill-down model
- Responsive technology
- Mobile style “hamburger” menus
- Action menus
- Breadcrumb control
- Touch friendly lists
  - Momentum scrolling
  - Multi-line rows
- Optimized on-screen keyboard support
- New mobile / touch theme
- Templates & Wizards

Mobile/Touch Web Applications

- Over the next three days you will hear a lot about this.
  - Harm Wibier will introduce you to all of these features in his next talk.
  - Other team members will present in-depth presentations about all of these new features

- There is a lot to see and a lot to learn

The DataFlex 2014 / 18.1

- The DataFlex development system remains
  - One Language
  - A unified and unique framework
  - Shared Data Dictionaries
  - One Studio & One Debugger
  - All focused on building business applications

- Windows Applications
- Desktop Web Applications
- Mobile/Touch Web Applications

The DataFlex 2014 / 18.1

Thank you