UNIVERSITY OF MICHIGAN
Survey Research Center

Navigating BCP with .NET

Peter Sparks
The 11th International Blaise Conference
Annapolis, Maryland
September 2007

Introduction

- BCP Overview
- Demo Program
- Code Examples

- Correction in book: p 213 Appendix B
  - Example code in .Net should be “i + 1” instead of “i – 1”

What is the BCP?

- Blaise “Engine”
- Different names
  - BCP 1.0 for Blaise 4.5
  - BCP 2.0 for Blaise 4.6
  - BLAPI4A2 for Blaise 4.7
  - BLAPI3A for Blaise 4.8

When to Use BCP?

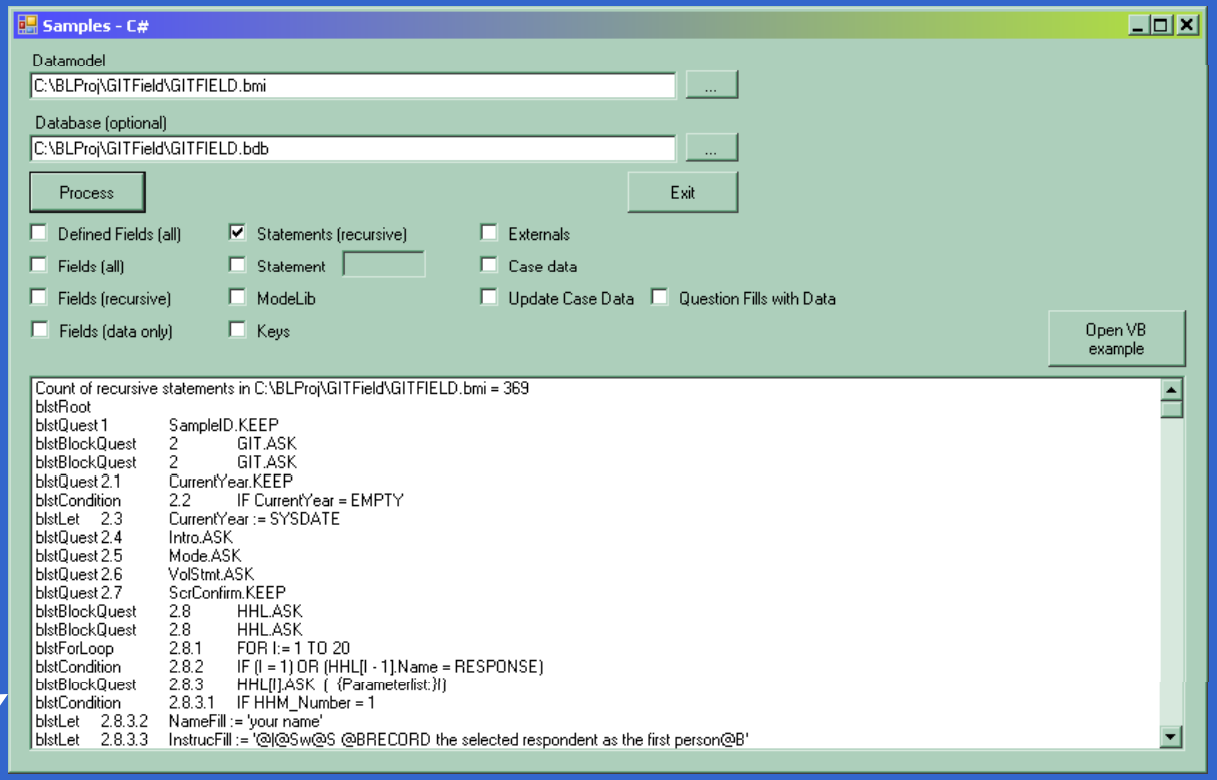
- Meta
  - Cameleon, Delta, Manipula (4.8)
- Data
  - Manipula, Maniplus
- Environment
  - Modelib, Datamodel Properties
- Complex
  - BCP

What BCP Can't Do

- Update or create datamodels.
- Compiled Manipula/Maniplus scripts
- Program comments or spacing

Companion Program

- Available via IBUC 2007 web site
  - Source code
  - Installation
  - Presented "as is."



Output from Demo Program

Steps to Using BCP

- Add the BCP's BLAPI4A2 reference to your project for Blaise 4.7
- Create a Database Manager.
  - Note: one manager for all databases.
- Create a database object for each datamodel, attach to the datamodel
- Use it
- Close the association to the datamodel
- Dispose of the objects when finished

Accessing BCP Code

C#

```
BlAPI4A2.DatabaseManager curDatabaseManager = new
    BlAPI4A2.DatabaseManager();

BlAPI4A2.Database curDatabase;
curDatabase = curDatabaseManager.OpenDatabase("");
curDatabase.DictionaryFileName = @"c:\temp\SomeDatamodel.bmi";
```

VB

```
Dim curDatabaseManager As BlAPI4A2.DatabaseManager

Dim curDatabase As BlAPI4A2.Database
curDatabase = curDatabaseManager.OpenDatabase("")
curDatabase.DictionaryFileName = "c:\temp\SomeDatamodel.bmi"
```

Demo Program Features

- All defined fields (method)
- All fields (method)
- All fields + blocks (recursive)
- Data fields only (method)
- Statements (recursive)
- Statement (ident key)
- Modelib settings
- Primary/secondary keys
- External datamodels
- Retrieve all case data
- Update a field/case
- Get question text with fills

Defined Fields

- Access
  - Get non-unique field names
  - Filter for data, generated parameters, locals, auxfields, and so forth
  - Get named parameters within the block/procedure
  - Get RulesText for a block
- Use
  - Create comparison lists

Output from Demo Program

```
blfkDataField GIT.E5.RentAmnt
blfkDataField GIT.E5.RentPeriod
blfkDataField GIT.E5.RentOth
blfkDataField GIT.MoveInYr
blfkDataField GIT.MarStat
blfkDataField GIT.HaveChldrn
blfkDataField GIT.NbrChldrn
blfkDataField GIT.R_Race[]
blfkDataField GIT.R_RaceOth
blfkDataField GIT.F4
blfkGenParameter GITFIELD GP
blfkDataField GIT.F4.BirthCity
blfkDataField GIT.F4.BirthState
blfkDataField GIT.F4.DoB_Mo
blfkDataField GIT.F4.DoB_Day
blfkDataField GIT.F4.DoB_YR
blfkExternal GIT.F4.StateFile
blfkAuxField GIT.F4.LeapYear
blfkGenAuxField GIT.F4.StateFile<8>
blfkDataField GIT.HiGrade
```

Defined Fields Code

C#

```
BlAPI4A2.Fields curFields = curDatabase.get_DefinedFields(
    (int)BlAPI4A2.BlFieldKind.blfkAll, (int)BlAPI4A2.BlFieldType.blftAll);
```

VB

```
Dim curFields As BlAPI4A2.Fields = curDatabase.DefinedFields(
    BlAPI4A2.BlFieldKind.blfkAll, BlAPI4A2.BlFieldType.blftAll)
```

All fields

- Access
  - Get unique field names
  - Filter for data, generated parameters, locals, auxfields, and so forth
- Use
  - Creating dictionaries
  - Create comparison lists

Output from Demo Program

```
blfkDataField GIT.E5.RentAmnt
blfkDataField GIT.E5.RentPeriod
blfkDataField GIT.E5.RentOth
blfkDataField GIT.MoveInYr
blfkDataField GIT.MarStat
blfkDataField GIT.HaveChldrn
blfkDataField GIT.NbrChldrn
blfkDataField GIT.R_Race[1]
blfkDataField GIT.R_Race[2]
blfkDataField GIT.R_Race[3]
blfkDataField GIT.R_Race[4]
blfkDataField GIT.R_Race[5]
blfkDataField GIT.R_Race[6]
blfkDataField GIT.R_Race[7]
blfkDataField GIT.R_Race[8]
blfkDataField GIT.R_Race[9]
blfkDataField GIT.R_RaceOth
blfkGenParameter GITFIELD GP
blfkDataField GIT.F4.BirthCity
blfkDataField GIT.F4.BirthState
```

All Fields Code

All Fields from the Database Level

C#

```
BlAPI4A2.Fields curFields = curDatabase.get_Fields(
    (int)BlAPI4A2.BlFieldKind.blfkAll, (int)BlAPI4A2.BlFieldType.blftAll);
```

VB

```
Dim curFields As BlAPI4A2.Fields = curDatabase.Fields(
    BlAPI4A2.BlFieldKind.blfkAll, BlAPI4A2.BlFieldType.blftAll)
```

All Fields from the Current Block Level

C#

```
 if (curField.DataType == BlAPI4A2.BlFieldType.blftBlock)
    BlAPI4A2.Fields curFields = curField.get_Fields(
            (int)BlAPI4A2.BlFieldKind.blfkAll, (int)BlAPI4A2.BlFieldType.blftAll);
```

VB

```
 If (curField.DataType = BlAPI4A2.BlFieldType.blftBlock) Then
    Dim curFields As BlAPI4A2.Fields = curField.Fields(
            BlAPI4A2.BlFieldKind.blfkAll, BlAPI4A2.BlFieldType.blftAll)
```

Statements - Retrieve

- Access
  - Get information on nearly every RULE
  - process statements in RULES order
  - Get parameters used in calling a BLOCK/ PROCEDURE
  - Step through BLOCK and PROCEDURE code
  - Retrieve specific statements via the Ident

blstBlockQuest	2.8	HHL.ASK
blstForLoop	2.8.1	FOR I:= 1 TO 20
blstCondition	2.8.2	IF (I = 1) OR (HHL[I - 1].Name = RESPONSE)
blstBlockQuest	2.8.3	HHL[I].ASK ({Parameterlist:}I)
blstCondition	2.8.3.1	IF HHM_Number = 1

blstCondition	2.55	IF OwnOrRent = PayRent
blstBlockQuest	2.56	E5.ASK
blstQuest 2.56.1	RentAmnt.ASK	
blstQuest 2.56.2	RentPeriod.ASK	
blstCondition	2.56.3	IF RentPeriod = Other
blstQuest 2.56.4	RentOth.ASK	
blstBlockQuest	2.56	E5.ASK
blstBlockQuest	2.56	E5.ASK
blstQuest 2.56.1	RentAmnt.ASK	
blstQuest 2.56.2	RentPeriod.ASK	
blstCondition	2.56.3	IF RentPeriod = Other
blstQuest 2.56.4	RentOth.ASK	
blstQuest 2.57	MoveInYr.ASK	
blstCondition	2.58	IF OwnOrRent = Neither
blstQuest 2.59	MoveInYr.ASK	
blstQuest 2.60	MarStat.ASK	
blstQuest 2.61	HaveChldrn.ASK	
blstCondition	2.62	IF HaveChldrn = Yes
blstQuest 2.63	NbrChldrn.ASK	
blstQuest 2.64	R_Race.ASK	

Statements - Limitations

- Compiled and parsed, so …
  - No blank lines or comments
  - No ENDIF, ENDLOOP statements
  - Expressions fully parenthesized

```
blstQuest 2.67.9     DoB_YR.ASK
blstCondition        2.67.10   IF ((DoB_Mo = 2) AND (DoB_Day = RESPONSE)) AND (DoB_YR = RESPONSE)
blstCondition        2.67.11   IF (DoB_YR MOD 4) = 0
blstCondition        2.67.12   IF (DoB_YR MOD 100) = 0
blstCondition        2.67.13   IF ((DoB_YR DIV 100) MOD 4) = 0
blstLet    2.67.14   LeapYear := 1
blstLet    2.67.15   LeapYear := 0
blstLet    2.67.16   LeapYear := 1
blstLet    2.67.17   LeapYear := 0
blstCondition        2.67.18   IF LeapYear = 1
blstStatement        2.67.19   CHECK
blstCheck            2.67.20   (DoB_Day <= 29) INVOLVING (DoB_YR) "@L@R@Qw@Q The leap year of Februa
blstStatement        2.67.21   CHECK
blstCheck            2.67.22   (DoB_Day <= 28) INVOLVING (DoB_YR) "@L@R@Qw@Q The non-leap year of Fe
blstCondition        2.67.23   IF (DoB_YR = RESPONSE) AND (HHL.HHL[1].BirthYr = RESPONSE)
blstStatement        2.67.24   SIGNAL
blstCheck            2.67.25   DoB_YR = HHL.HHL[1].BirthYr "@B@Sw@S Year entered is not consistent with the
blstQuest 2.68       HiGrade.ASK
blstCondition        2.69      IF ((HiGrade.ORD < 13) OR (HiGrade = DONTKNOW)) OR (HiGrade = REFUSAL)
blstQuest 2.70       HS_Diploma.ASK
```

- Text-only access to Generated RULES
- Unique identifier (Ident) numbered within a block.

```
blstLet    7.39      TotSecs := TotSecs + CurrSecs
blstGenerated        7.40      { *** GENERATED RULES *** IwerID.KEEPCallNbr.KEEPProxy.KEEPInstr_ID.KEEP *** END-GENERATED RULES *** }
blstQuest 7.41       IwerID.KEEP
```

Statements - Uses

- Use
  - Create RULES-ordered dictionaries
  - Analyze datamodels
  - Restore source code minus comments and formatting (use with FieldDef.RulesText)
  - Build cross-indexes with complex criteria (languages, assignments/fills, sub blocks)

blstLet	2.8.3.20	YearFill := 'What is your'
blstLet	2.8.3.21	SuppFill := ''
blstLet	2.8.3.22	RelToMainR := Respondent
blstQuest	2.8.3.23	RelToMainR.SHOW
blstLet	2.8.3.24	SexFill := ('(What is ' + NAMEX) + "s s
blstCondition	2.8.3.25	IF HHM_Number = 2
blstLet	2.8.3.26	RelFill := ('What is ' + NAMEX) + "s rel
blstLet	2.8.3.27	RelFill := ('(What is ' + NAMEX) + "s re
blstQuest	2.8.3.28	RelToMainR.ASK
blstStatement	2.8.3.29	CHECK
blstCheck	2.8.3.30	RelToMainR <> Responder
blstQuest	2.8.3.31	Sex.ASK
blstCondition	2.8.3.32	IF HHM_Number > 1
blstLet	2.8.3.33	AgeFill := 'is ' + NAMEX
blstLet	2.8.3.34	EngFill := 'Is ' + NAMEX
blstQuest	2.8.3.35	Age.ASK
blstCondition	2.8.3.36	IF HHM_Number > 1
blstLet	2.8.3.37	YearFill := ('What is ' + NAMEX) + "s'
blstQuest	2.8.3.38	BirthYr.ASK
blstCondition	2.8.3.39	IF (Age = RESPONSE) AND

Statements – Code C#

C#

```
retrieveRecursiveStatement(curDatabase.DictionaryAsStatement, ref totfields);
…
 for (int i = 1; i <= curStatement.Statements.Count; i++)
   {
     switch (curStatement.Statements[i].StatementType)
     {
       case (BlAPI4A2.BlStatementType.blstCondition):
           retrieveRecursiveStatement(curStatement.Statements[i].ThenStatement,
                                                    ref totStatements);
           if (curStatement.Statements[i].ElseStatement != null)
              retrieveRecursiveStatement(curStatement.Statements[i].ElseStatement,
                                                    ref totStatements);
             break;
     case (BlAPI4A2.BlStatementType.blstForLoop):
     case (BlAPI4A2.BlStatementType.blstBlockQuest):
     case (BlAPI4A2.BlStatementType.blstProcedure):
         retrieveRecursiveStatement(curStatement.Statements[i], ref totStatements);
          break;
     }
   }
```

Statements – Code VB

VB

```
 retrieveRecursiveStatement(curDatabase.DictionaryAsStatement, totfields)
...
 For i = 1 To curStatement.Statements.Count
     Select Case (curStatement.Statements(i).StatementType)
             Case (BlAPI4A2.BlStatementType.blstCondition)
                         retrieveRecursiveStatement(
                         curStatement.Statements(i).ThenStatement, totStatements)

             If Not (curStatement.Statements(i).ElseStatement Is Nothing) Then
                         retrieveRecursiveStatement(
                         curStatement.Statements(i).ElseStatement, totStatements)
             End If
             Case (BlAPI4A2.BlStatementType.blstForLoop)
             Case (BlAPI4A2.BlStatementType.blstBlockQuest)
             Case (BlAPI4A2.BlStatementType.blstProcedure)
                   retrieveRecursiveStatement(curStatement.Statements(i), totStatements)
     End Select
  Next i
```

Modelib

- Access
  - Layouts
  - Field, Info, and Grid Panes
  - Fonts
- Use
  - Recreate the look of the interviewing interface in other formats (print)
  - Could be used to create a new interface to Blaise interviewing.

```
Font index: V, Color: 536870911, Name: , Size: 10, tabstops (440,490), bold: blfsCurrent, underline: blfsCurrent, italics: blfsCurrent
Font index: W, Color: 16711680, Name: Wingdings, Size: 24, tabstops (), bold: blfsOff, underline: blfsCurrent, italics: blfsCurrent
Font index: X, Color: 16711680, Name: Wingdings 2, Size: 11, tabstops (), bold: blfsOff, underline: blfsCurrent, italics: blfsCurrent
Font index: Y, Color: 536870911, Name: , Size: 0, tabstops (), bold: blfsCurrent, underline: blfsCurrent, italics: blfsCurrent
Font index: Z, Color: 536870911, Name: , Size: 8, tabstops (), bold: blfsCurrent, underline: blfsCurrent, italics: blfsCurrent
Pages and Questions
        Layout: Interviewing
        Parallel: GITFIELD
Layout: 1, Parallel: 1, Page: 1, Question: 1: GIT.Intro
Layout: 1, Parallel: 1, Page: 1, Question: 2: GIT.Mode
Layout: 1, Parallel: 1, Page: 1, Question: 3: GIT.VolStmt
Layout: 1, Parallel: 1, Page: 2, Question: 1: GIT.HHL.HHL[1].Name
Layout: 1, Parallel: 1, Page: 2, Question: 2: GIT.HHL.HHL[1].RelToMainR
Layout: 1, Parallel: 1, Page: 2, Question: 3: GIT.HHL.HHL[1].Sex
Layout: 1, Parallel: 1, Page: 2, Question: 4: GIT.HHL.HHL[1].Age
Layout: 1, Parallel: 1, Page: 2, Question: 5: GIT.HHL.HHL[1].BirthYr
Layout: 1, Parallel: 1, Page: 2, Question: 6: GIT.HHL.HHL[1].English
Layout: 1, Parallel: 1, Page: 2, Question: 7: GIT.HHL.HHL[2].Name
Layout: 1, Parallel: 1, Page: 2, Question: 8: GIT.HHL.HHL[2].RelToMainR
Layout: 1, Parallel: 1, Page: 2, Question: 9: GIT.HHL.HHL[2].Sex
```

Modelib Code C#

C#

```
string fontName =
    curDatabase.ModeLibrary.Style.Fonts.get_CustomFont("A").Name;
int fontSize =
    curDatabase.ModeLibrary.Style.Fonts.get_CustomFont("A").Size;

BlAPI4A2.LayoutSet Layout;
BlAPI4A2.Parallel Parallel;
BlAPI4A2.StoredPage Page;
BlAPI4A2.Question Quest;

Layout = curDatabase.Screens.LayoutSetCollection[iLayout];
Parallel = Layout.ParallelCollection[iParallel];
Page = Parallel.StoredPageCollection[iPage];
Quest = Page.QuestionCollection[iQuest];
```

Modelib Code VB

VB

```
Dim fontName As String =
    curDatabase.ModeLibrary.Style.Fonts.CustomFont("A").Name
Dim fontSize As Integer =
    curDatabase.ModeLibrary.Style.Fonts.CustomFont("A").Size

Dim Layout As BIAPI4A2.LayoutSet
Dim Parallel As BIAPI4A2.Parallel
Dim Page As BIAPI4A2.StoredPage
Dim Quest As BIAPI4A2.Question

Layout = curDatabase.Screens.LayoutSetCollection(iLayout)
Parallel = Layout.ParallelCollection(iParallel)
Page = Parallel.StoredPageCollection(iPage)
Quest = Page.QuestionCollection(iQuest)
```

Primary/Secondary Keys

- Access
  - All defined keys for a datamodel
- Use
  - Dictionaries
  - Creating scripts
  - Special processing of Blaise databases

```
Keys
Name: PRIMARY, Kind: blkkPrimary, using SampleID
Name: Secondary, Kind: blkkSecondary, using Complete
```

Primary/Secondary Keys Code

C#

Name - curDatabase.Keys[i + 1].Name
Kind - curDatabase.Keys[i + 1].Kind
Parts - curDatabase.Keys[i + 1].InvolvedFields[j + 1].IndexedName

VB

Name - curDatabase.Keys(i + 1).Name
Kind - curDatabase.Keys(i + 1).Kind
Parts - curDatabase.Keys(i + 1).InvolvedFields(j + 1).IndexedName

External Datamodels

- Access
  - All information the same as the main datamodel
- Use
  - Verify external data via the main datamodel
  - Create multiple dictionaries at the same time.

```
Externals
Database: C:\BLProj\GITField\statelist.bdb, Columns = StateCode, ST, STATENAME
1, AK, ALASKA
2, AL, ALABAMA
4, AR, ARKANSAS
3, AZ, ARIZONA
5, CA, CALIFORNIA
6, CO, COLORADO
7, CT, CONNECTICUT
8, DE, DELAWARE
9, FL, FLORIDA
10, GA, GEORGIA
11, HI, HAWAII
15, IA, IOWA
12, ID, IDAHO
13, IL, ILLINOIS
14, IN, INDIANA
16, KS, KANSAS
17, KY, KENTUCKY
18, LA, LOUISIANA
```

External Datamodels Code C#

C#

```
BlAPI4A2.Fields oFields =
    curDatabase.get_DefinedFields((int)BlAPI4A2.BlFieldKind.blfkExternal,
    (int)BlAPI4A2.BlFieldType.blftAll);

DBExt.StorageFormat =
    oFields[i + 1].FieldDef.ExternalInformation.StorageFormat;
DBExt.AsciiFileSettings.Separator =
    oFields[i+1].FieldDef.ExternalInformation.AsciiFileSeparator;
DBExt.AsciiFileSettings.Delimiter =
    oFields[i+1].FieldDef.ExternalInformation.AsciiFileDelimiter;
DBExt.DictionaryFileName =
    oFields[i+1].FieldDef.ExternalInformation.DictionaryFileName;
 DBExt.DataFileName =
    oFields[i+1].FieldDef.ExternalInformation.DataFileName;
DBExt.Connected = true;
DBExt.ReadNextRecord();
DBExt.ActiveRecord[k + 1].Value;
DBExt.Connected = false;
DBExt = null;
```

External Datamodels Code VB

VB

```
Dim oFields As BlAPI4A2.Fields =
    curDatabase.DefinedFields(BlAPI4A2.BlFieldKind.blfkExternal,
    BlAPI4A2.BlFieldType.blftAll)

DBExt.StorageFormat =
    oFields(i+1).FieldDef.ExternalInformation.StorageFormat
DBExt.AsciiFileSettings.Separator =
    oFields(i+1).FieldDef.ExternalInformation.AsciiFileSeparator
DBExt.AsciiFileSettings.Delimiter =
    oFields(i+1).FieldDef.ExternalInformation.AsciiFileDelimiter
DBExt.DictionaryFileName =
    oFields(i+1).FieldDef.ExternalInformation.DictionaryFileName
DBExt.DataFileName =
    oFields(i+1).FieldDef.ExternalInformation.DataFileName
DBExt.Connected = True
DBExt.ReadNextRecord()
DBExt.ActiveRecord(k + 1).Value
DBExt.Connected = False
DBExt = Nothing
```

Case Data

- Access
  - All data fields in a datamodel
  - Retrieve cases filtered by status or special criteria
  - Retrieve cases by primary/secondary keys
  - Retrieve question text with fills completed
- Use
  - Special filtering of data
  - Convert to other formats not easily done by Cameleon/Manipula
  - Combine with non-Blaise data sources
  - Update case data uniquely

Record #30, Key=109165, Data=109165						
Record #31, Key=110183, Data=110183	11	11				
Record #32, Key=110184, Data=110184						
Record #33, Key=110185, Data=110185						
Record #34, Key=111203, Data=111203						
Record #35, Key=111204, Data=111204						
Record #36, Key=111205, Data=111205						
Record #37, Key=112223, Data=112223	11	11	121	6	4	0

Case Data Code

C#

Record # - curDatabase.ActiveRecord.FormID + ",
Key - curDatabase.PrimaryKeyFieldValues
Data - curDatabase.DictionaryAsField.Text

VB

Record # - curDatabase.ActiveRecord.FormID
Key - curDatabase.PrimaryKeyFieldValues
Data - curDatabase.DictionaryAsField.Text

Update Case Data Code

C#

```
curDatabase.KeyValue = key;
curDatabase.ReadRecord();
curDatabase.DataEntryBehaviour =
        BlAPI4A2.BlDataEntryBehaviour.bldbUncheckedEditing;
curDatabase.get_Field(field).Text = data;
curDatabase.CheckRecord();
curDatabase.UpdateRecord();
```

VB

```
curDatabase.KeyValue = key
curDatabase.ReadRecord()
curDatabase.DataEntryBehaviour =
        BlAPI4A2.BlDataEntryBehaviour.bldbUncheckedEditing
curDatabase.Field(field).Text = data
curDatabase.CheckRecord()
curDatabase.UpdateRecord()
```

Get question text with fills

- Requires a connection to a database
- Requires choice of a language
- Can retrieve the question text without fills.

```
Record #2, Field=GIT.HHL.HHL[1].RelToMainR, Original Text=^RelFill, Filled Text=
Record #2, Field=GIT.HHL.HHL[1].Sex, Original Text=^SexFill, Filled Text=@BASK if necessary: @B What is SALLY HARPER's sex?
Record #2, Field=GIT.HHL.HHL[1].Age, Original Text=How old ^AgeFill? @/@/@|@Sw@S @BIf less than one year old, enter [0]@B
Record #2, Field=GIT.HHL.HHL[1].BirthYr, Original Text=^YearFill year of birth?, Filled Text=What is your year of birth?
Record #2, Field=GIT.HHL.HHL[1].English, Original Text=^Engfill speak English?, Filled Text=Do you speak English?
Record #2, Field=GIT.HHL.HHL[2].Name, Original Text=@>@B[F1]--Help@B @>@BCurrent HH Member #^HHM_Number@B@< @
```

Get question text with fills Code

C#

```
curDatabase.DataEntryBehaviour =
      BlAPI4A2.BlDataEntryBehaviour.bldbStrictInterviewing;

if (!string.IsNullOrEmpty(curFields[i].get_DefinedQuestionText(1)))
    if (curFields[i].get_DefinedQuestionText(1).Contains("^"))

Original_Text = curFields[i].get_QuestionText(1, false);
Filled_Text = curFields[i].get_QuestionText(1, true);
```

VB

```
curDatabase.DataEntryBehaviour =
      BlAPI4A2.BlDataEntryBehaviour.bldbStrictInterviewing
 If Not (String.IsNullOrEmpty(curFields(i).DefinedQuestionText(1))) Then
    If (curFields(i).DefinedQuestionText(1).Contains("^")) Then

Original_Text = curFields(i).QuestionText(1, False)
Filled_Text = curFields(i).QuestionText(1, True)
```

Last Observations

- BCP is Complex!
- Recursion is sometimes necessary
- Multiple ways of getting similar data
- Text parsing becomes a big issue
- Sound programming principals helpful

Questions

- Contact info:

Peter Sparks

zebulon@isr.umich.edu

+01 734.647.6127