Migrating your apps to the latest Delphi

Jeroen Pluimers
better office benelux
jpluimers@better-office.com

Target is D2009

- The 'old' slides of a D5 -> D2006/D2007 migration are still available:
  - After all the regular session slides

How 'breaking' is D2009?

- <<
  ... this release is a **breaking change** in both Delphi and C++Builder. ...main focus ... change ... to using a Unicode-based string type, appropriately named UnicodeString...
  >>
- So
  - Besides 'string' now mapping to 'UnicodeString' in stead of 'AnsiString',
    how 'breaking' are these changes?

Getting started

- Converting your projects
- Determining Delphi version

Notes from D2007->D2009

- Read these articles:
  - Delphi 2009 release notes
    - http://dn.codegear.com/article/38475
  - Marco Cantu: Delphi_and_Unicode.pdf
  - Delphi in a Unicode World,
    - Part I: What is Unicode, Why do you need it, and How do you work with it in Delphi?
      - http://dn.codegear.com/article/38437
    - Part II: New RTL Features and Classes to Support Unicode
      - http://dn.codegear.com/article/38498
    - Part III: Unicodifying Your Code
      - http://dn.codegear.com/article/38693

On-line help: topics

- What's New in Delphi and C++Builder 2009
  - ms-help://embarcadero.rs2009/devcommon/whatsnewtiburon_xml.html
- String Types
  - Base: ShortString, AnsiString, WideString, UnicodeString
    - ms-help://embarcadero.rs2009/devcommon/stringtypes_xml.html
  - Unicode (including internal memory layouts of the string types)
    - ms-help://embarcadero.rs2009/devcommon/unicodeinide_xml.html
  - Enabling for Unicode
    - ms-help://embarcadero.rs2009/devcommon/enablingunicode_xml.html
  - RawByteString (AnsiString with CodePage $ffff)
    - ms-help://embarcadero.rs2009/delphivclwin32/System_RawByteString.html
  - UTF8String (AnsiString with CodePage 65001)
    - ms-help://embarcadero.rs2009/delphivclwin32/System_UTF8String.html
- String <-> PChar conversions
  - PChar fundamentals
    - ms-help://embarcadero.rs2009/devwin32/stringdependencies_xml.html
  - Returning a PChar Local Variable
    - ms-help://embarcadero.rs2009/devwin32/passingalocalvariableasapchar_xml.html
  - Passing a Local Variable as a PChar
    - ms-help://embarcadero.rs2009/devwin32/returningapcharlocalvariable_xml.html

D2009 -> dproj files incompatible

- D2009 .dproj files are incompatible with older Delphi versions
  - Make a backup or a new version for your Delphi 2009 projects
  - Delphi 2009
    - Supports multiple configuration targets
    - Hierarchy:
      - Base
        - Debug
        - Release
    - Will convert the old .dproj to D2009 format

      The conversion targets 'Debug'
      - When switching to Release or Base, you need to re-apply all your settings manually

D2007->D2009: .dproj

- D2009 .dproj is incompatible with D2007 and before
  - Use different .dproj if you want to maintain compatibility with previous Delphi versions

{$ifdef VERxx}

- Usually best to do with include files
  - Jedi.inc is a good example
    - http://sourceforge.net/projects/jvcl/
    - http://sourceforge.net/projects/jcl/
- More info: http://tinyurl.com/6xmwn2
  - http://delphi.wikia.com/wiki/Borland_Compiler_Conditional_Defines
- We did it with these files

Defines.inc	- included in every unit
Project.inc	- included from Defines.inc
Group.inc	- included from Project.inc

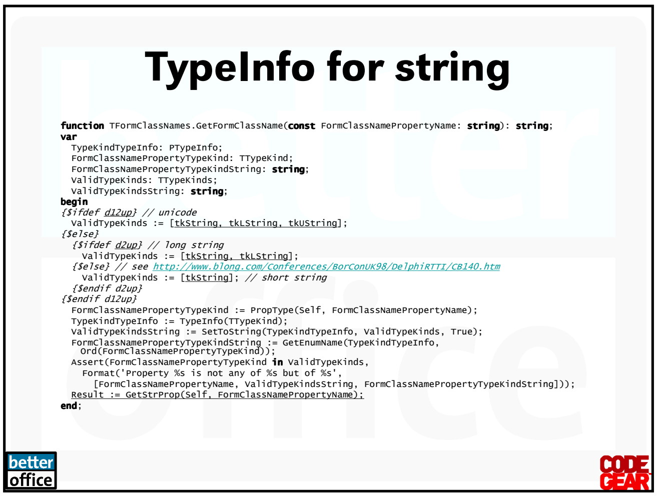
Compiling your code

TypeInfo for string

- Since 'string' is now 'UnicodeString', the underlying TypeInfo changed:
  - Property sfAdmDebitBillsFind is not any of [tkString] but of tkUString
- You need this if you want to call GetStrProp (in the TypInfo unit)

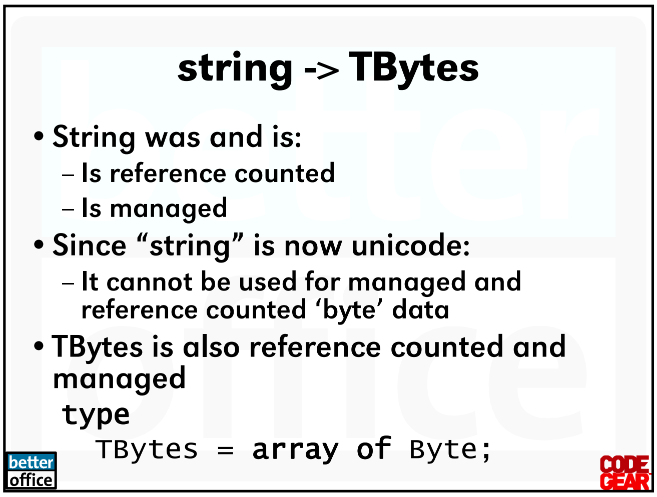
String types

- ShortString
  - Ansi (max 255)
  - Not Reference counted
- AnsiString
  - Ansi
  - Reference counted
- UTF8String
  - Unicode UTF-8
  - Reference counted
- WideString
  - Unicode UTF-16
  - Not Reference counted
- UnicodeString:
  - Unicode UTF-16
  - Reference counted



TypeInfo for string

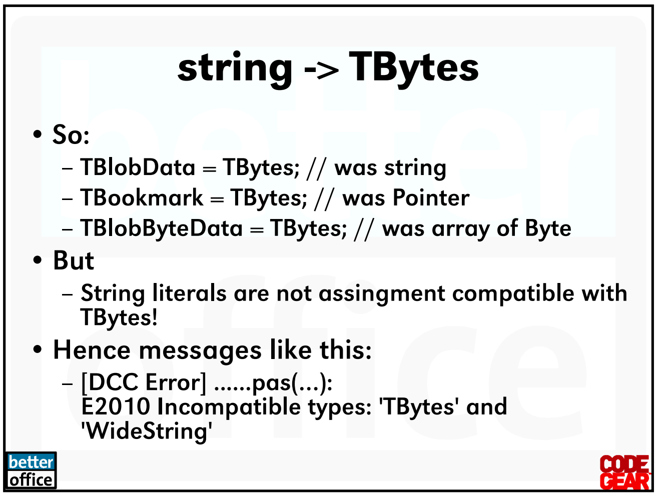
```
function TFormClassNames.GetFormClassName(const FormClassNamePropertyName: string): string;
var
  TypeKindTypeInfo: PTypeInfo;
  FormClassNamePropertyTypeKind: TTypeKind;
  FormClassNamePropertyTypeKindString: string;
  ValidTypeKinds: TTypeKinds;
  ValidTypeKindsString: string;
begin
{$ifdef d12up} // unicode
  ValidTypeKinds := [tkString, tkLString, tkUString];
{$else}
  {$ifdef d2up} // long string
    ValidTypeKinds := [tkString, tkLString];
  {$else} // see http://www.blong.com/Conferences/BorConUK98/DelphiRTTI/CB140.htm
    ValidTypeKinds := [tkString]; // short string
  {$endif d2up}
{$endif d12up}
  FormClassNamePropertyTypeKind := PropType(Self, FormClassNamePropertyName);
  TypeKindTypeInfo := TypeInfo(TTypeKind);
  ValidTypeKindsString := SetToString(TypeKindTypeInfo, ValidTypeKinds, True);
  FormClassNamePropertyTypeKindString := GetEnumName(TypeKindTypeInfo,
   Ord(FormClassNamePropertyTypeKind));
  Assert(FormClassNamePropertyTypeKind in ValidTypeKinds,
    Format('Property %s is not any of %s but of %s',
      [FormClassNamePropertyName, ValidTypeKindsString, FormClassNamePropertyTypeKindString]));
  Result := GetStrProp(Self, FormClassNamePropertyName);
end;
```



string -> TBytes

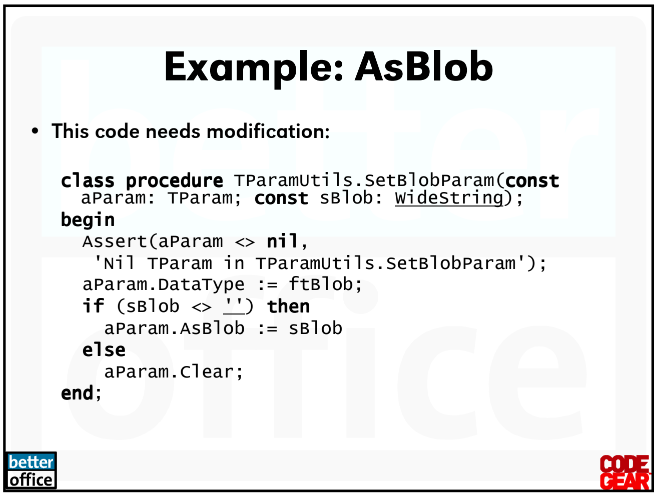
- String was and is:
  - Is reference counted
  - Is managed
- Since “string” is now unicode:
  - It cannot be used for managed and reference counted ‘byte’ data
- TBytes is also reference counted and managed

```
 type
    TBytes = array of Byte;
```



string -> TBytes

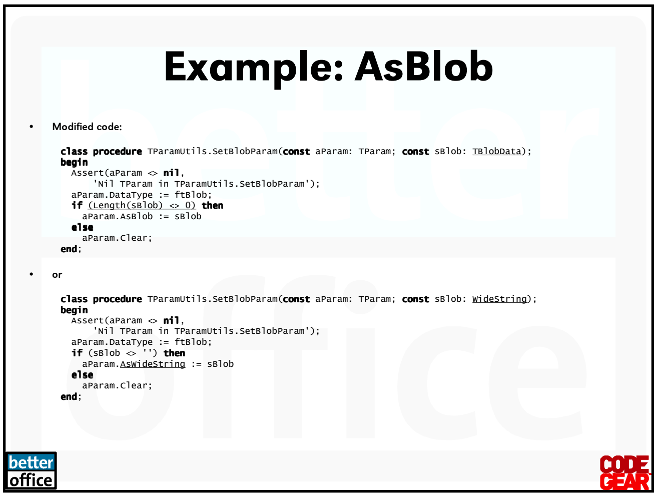
- So:
  - TBlobData = TBytes; // was string
  - TBookmark = TBytes; // was Pointer
  - TBlobByteData = TBytes; // was array of Byte
- But
  - String literals are not assingment compatible with TBytes!
- Hence messages like this:
  - [DCC Error] ......pas(...):
E2010 Incompatible types: 'TBytes' and 'WideString'



Example: AsBlob

- This code needs modification:

```
class procedure TParamUtils.SetBlobParam(const
  aParam: TParam; const sBlob: WideString);
begin
  Assert(aParam <> nil,
   'Nil TParam in TParamUtils.SetBlobParam');
  aParam.DataType := ftBlob;
  if (sBlob <> '') then
    aParam.AsBlob := sBlob
  else
    aParam.Clear;
end;
```



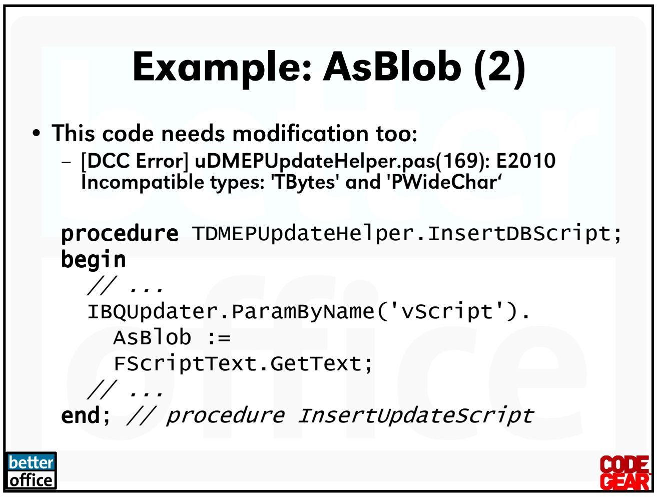
Example: AsBlob

- Modified code:

```
class procedure TParamUtils.SetBlobParam(const aParam: TParam; const sBlob: TBlobData);
begin
  Assert(aParam <> nil,
      'Nil TParam in TParamUtils.SetBlobParam');
  aParam.DataType := ftBlob;
  if (Length(sBlob) <> 0) then
    aParam.AsBlob := sBlob
  else
    aParam.Clear;
end;
```

- or

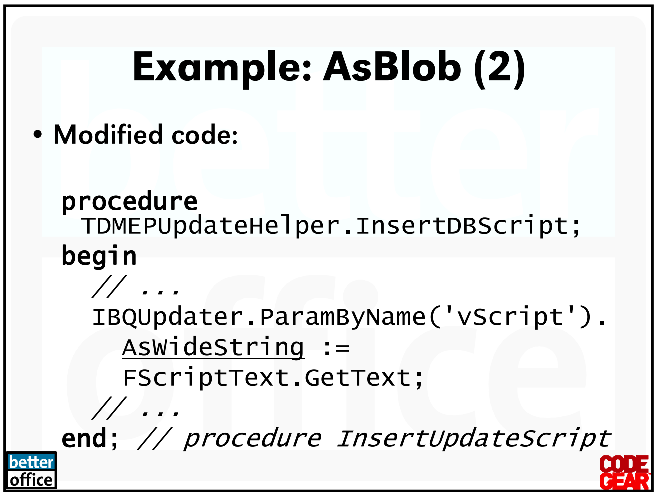
```
class procedure TParamUtils.SetBlobParam(const aParam: TParam; const sBlob: WideString);
begin
  Assert(aParam <> nil,
      'Nil TParam in TParamUtils.SetBlobParam');
  aParam.DataType := ftBlob;
  if (sBlob <> '') then
    aParam.AsWideString := sBlob
  else
    aParam.Clear;
end;
```



Example: AsBlob (2)

- This code needs modification too:
  - [DCC Error] uDMEPUpdateHelper.pas(169): E2010 Incompatible types: 'TBytes' and 'PWideChar‘

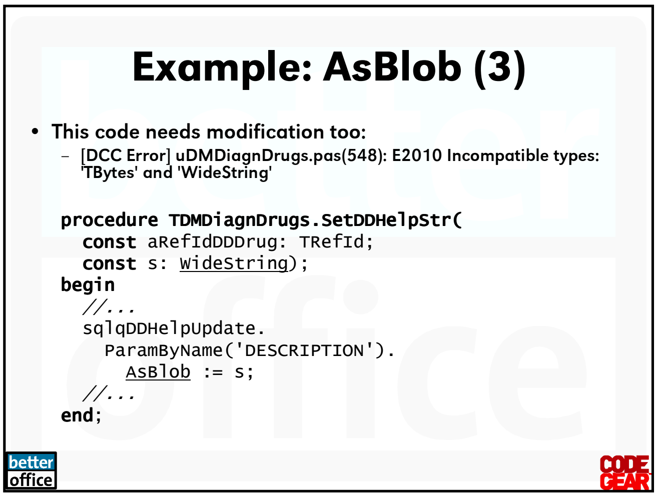
```
procedure TDMEPUpdateHelper.InsertDBScript;
begin
  // ...
  IBQUpdater.ParamByName('vScript').
    AsBlob :=
    FScriptText.GetText;
  // ...
end; // procedure InsertUpdateScript
```



Example: AsBlob (2)

- Modified code:

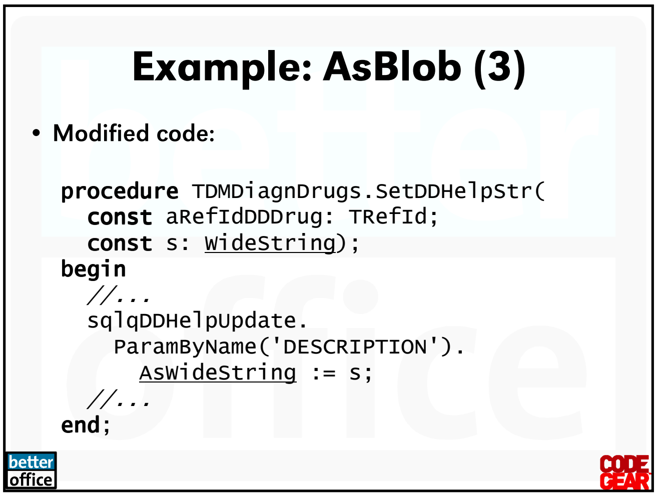
```
procedure
  TDMEPUpdateHelper.InsertDBScript;
begin
   // ...
   IBQUpdater.ParamByName('vScript').
     AsWideString :=
     FScriptText.GetText;
   // ...
end; // procedure InsertUpdateScript
```



Example: AsBlob (3)

- This code needs modification too:
  - [DCC Error] uDMDiagnDrugs.pas(548): E2010 Incompatible types: 'TBytes' and 'WideString'

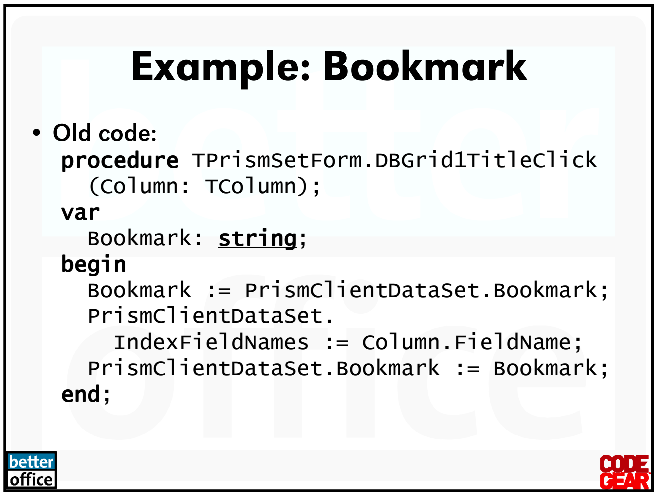
```
procedure TDMDiagnDrugs.SetDDHelpStr(
  const aRefIdDDDrug: TRefId;
  const s: WideString);
begin
  //...
  sqlqDDHelpUpdate.
    ParamByName('DESCRIPTION').
      AsBlob := s;
  //...
end;
```



Example: AsBlob (3)

- Modified code:

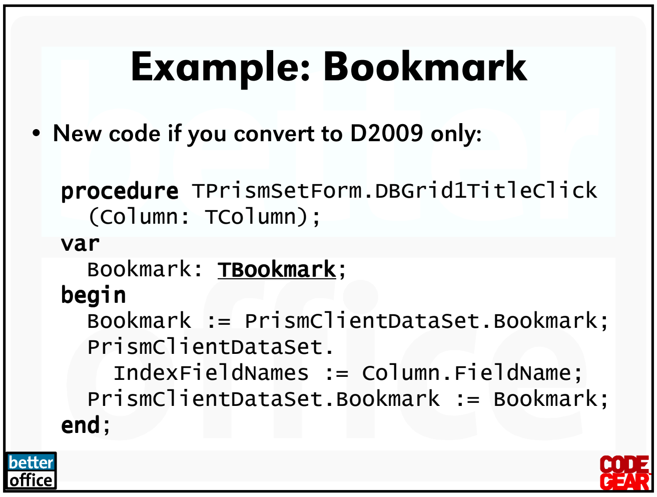
```
procedure TDMDiagnDrugs.SetDDHelpStr(
  const aRefIdDDDrug: TRefId;
  const s: WideString);
begin
  //...
  sqlqDDHelpUpdate.
    ParamByName('DESCRIPTION').
      AsWideString := s;
  //...
end;
```



Example: Bookmark

- Old code:

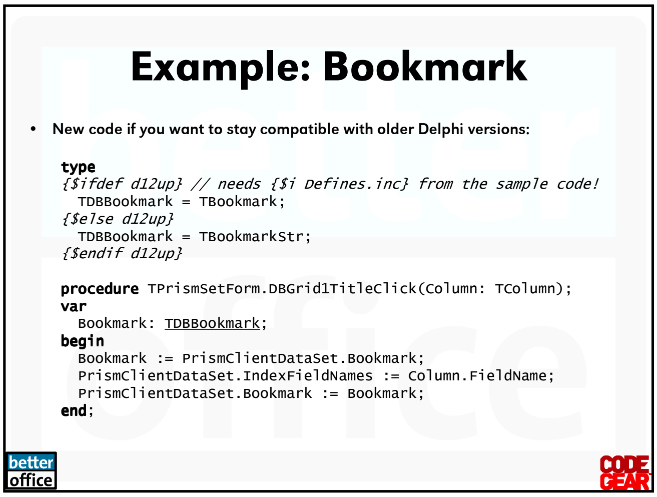
```
procedure TPrismSetForm.DBGrid1TitleClick
  (Column: TColumn);
var
  Bookmark: string;
begin
  Bookmark := PrismClientDataSet.Bookmark;
  PrismClientDataSet.
    IndexFieldNames := Column.FieldName;
  PrismClientDataSet.Bookmark := Bookmark;
end;
```



Example: Bookmark

- New code if you convert to D2009 only:

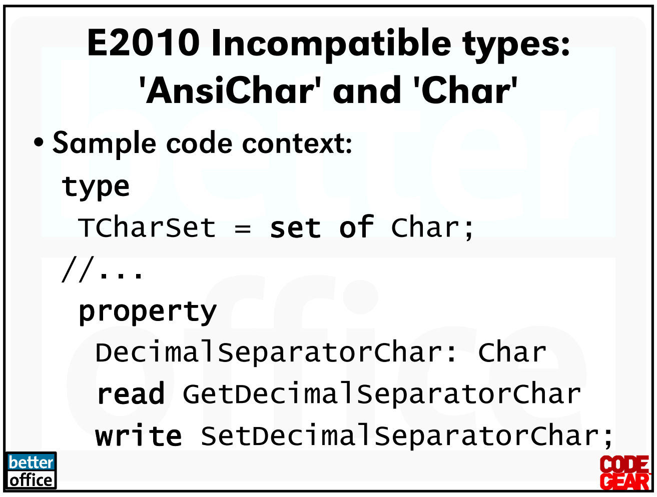
```
procedure TPrismSetForm.DBGrid1TitleClick
  (Column: TColumn);
var
  Bookmark: TBookmark;
begin
  Bookmark := PrismClientDataSet.Bookmark;
  PrismClientDataSet.
    IndexFieldNames := Column.FieldName;
  PrismClientDataSet.Bookmark := Bookmark;
end;
```



Example: Bookmark

- New code if you want to stay compatible with older Delphi versions:

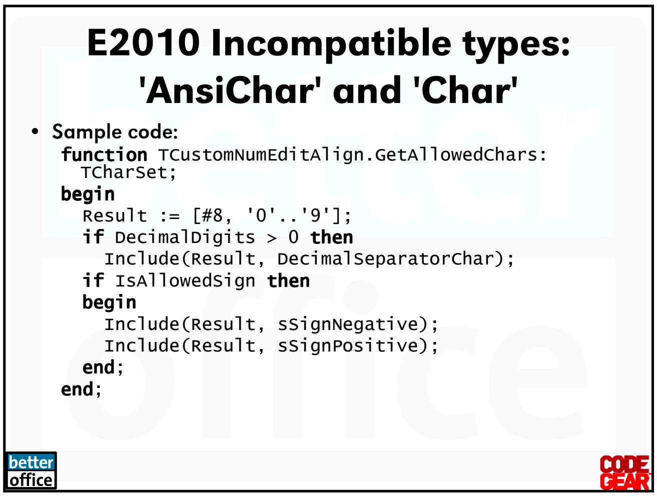
```
type
{$ifdef d12up} // needs {$i Defines.inc} from the sample code!
  TDBBookmark = TBookmark;
{$else d12up}
  TDBBookmark = TBookmarkStr;
{$endif d12up}

procedure TPrismSetForm.DBGrid1TitleClick(Column: TColumn);
var
  Bookmark: TDBBookmark;
begin
  Bookmark := PrismClientDataSet.Bookmark;
  PrismClientDataSet.IndexFieldNames := Column.FieldName;
  PrismClientDataSet.Bookmark := Bookmark;
end;
```



E2010 Incompatible types: 'AnsiChar' and 'Char'

- Sample code context:

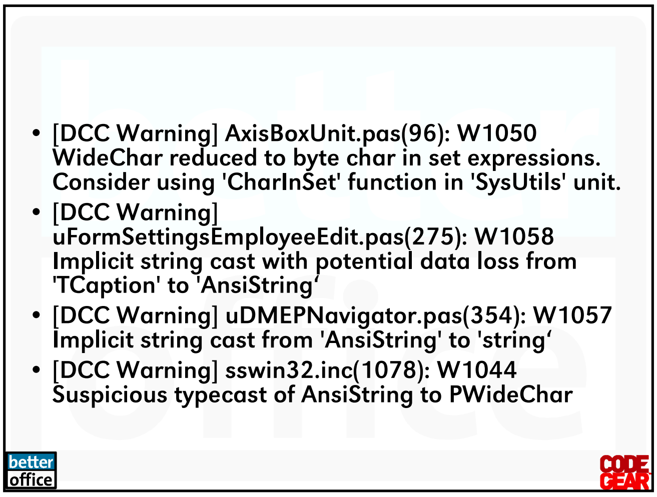
```
type
  TCharSet = set of Char;
//...
  property
    DecimalSeparatorChar: Char
    read GetDecimalSeparatorChar
    write SetDecimalSeparatorChar;
```



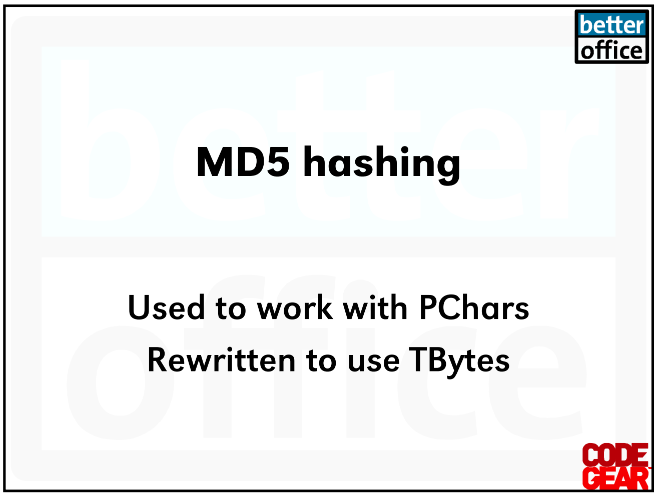
E2010 Incompatible types: 'AnsiChar' and 'Char'

- Sample code:

```
function TCustomNumEditAlign.GetAllowedChars:
  TCharSet;
begin
  Result := [#8, '0'..'9'];
  if DecimalDigits > 0 then
    Include(Result, DecimalSeparatorChar);
  if IsAllowedSign then
  begin
    Include(Result, sSignNegative);
    Include(Result, sSignPositive);
  end;
end;
```



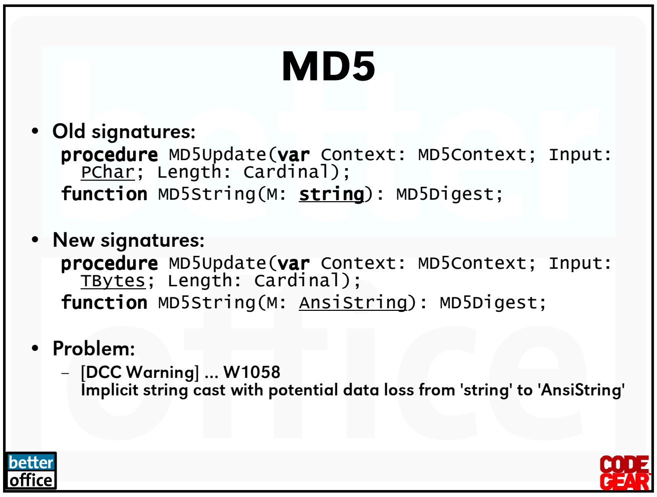
- [DCC Warning] AxisBoxUnit.pas(96): W1050 WideChar reduced to byte char in set expressions. Consider using 'CharInSet' function in 'SysUtils' unit.
- [DCC Warning] uFormSettingsEmployeeEdit.pas(275): W1058 Implicit string cast with potential data loss from 'TCaption' to 'AnsiString'
- [DCC Warning] uDMEPNavigator.pas(354): W1057 Implicit string cast from 'AnsiString' to 'string'
- [DCC Warning] sswin32.inc(1078): W1044 Suspicious typecast of AnsiString to PWideChar



MD5 hashing

Used to work with PChars

Rewritten to use TBytes



MD5

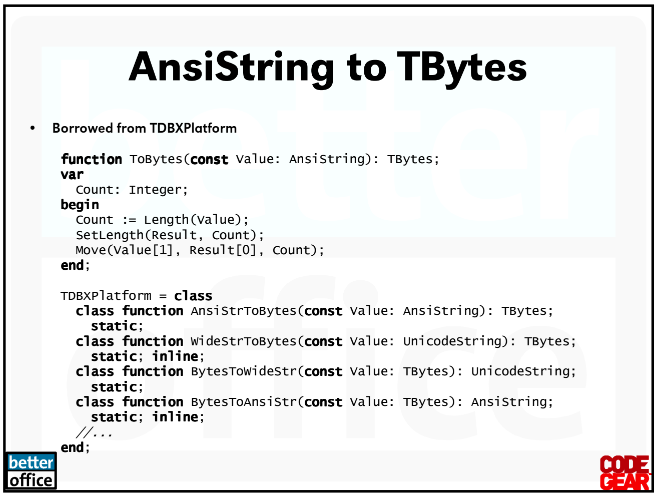
- Old signatures:

```
procedure MD5Update(var Context: MD5Context; Input:
  PChar; Length: Cardinal);
function MD5String(M: string): MD5Digest;
```

- New signatures:

```
procedure MD5Update(var Context: MD5Context; Input:
  TBytes; Length: Cardinal);
function MD5String(M: AnsiString): MD5Digest;
```

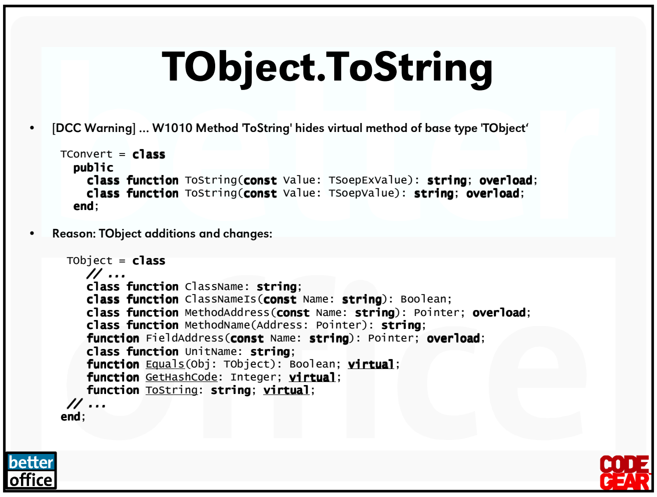
- Problem:
  - [DCC Warning] … W1058 Implicit string cast with potential data loss from 'string' to 'AnsiString'



AnsiString to TBytes

- Borrowed from TDBXPlatform

```
function ToBytes(const Value: AnsiString): TBytes;
var
  Count: Integer;
begin
  Count := Length(Value);
  SetLength(Result, Count);
  Move(Value[1], Result[0], Count);
end;

TDBXPlatform = class
  class function AnsiStrToBytes(const Value: AnsiString): TBytes;
    static;
  class function WideStrToBytes(const Value: UnicodeString): TBytes;
    static; inline;
  class function BytesToWideStr(const Value: TBytes): UnicodeString;
    static;
  class function BytesToAnsiStr(const Value: TBytes): AnsiString;
    static; inline;
  //...
end;
```



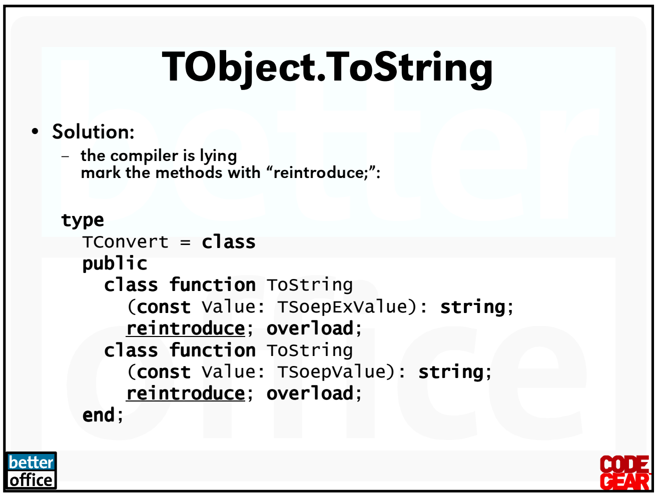
TObject.ToString

- [DCC Warning] … W1010 Method 'ToString' hides virtual method of base type 'TObject'

```
TConvert = class
  public
    class function ToString(const Value: TSoepExValue): string; overload;
    class function ToString(const Value: TSoepValue): string; overload;
  end;
```

- Reason: TObject additions and changes:

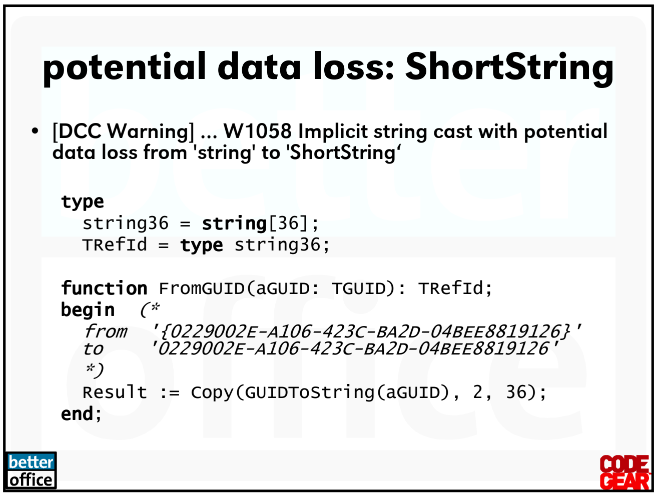
```
 TObject = class
    // ...
    class function ClassName: string;
    class function ClassNameIs(const Name: string): Boolean;
    class function MethodAddress(const Name: string): Pointer; overload;
    class function MethodName(Address: Pointer): string;
    function FieldAddress(const Name: string): Pointer; overload;
    class function UnitName: string;
    function Equals(Obj: TObject): Boolean; virtual;
    function GetHashCode: Integer; virtual;
    function ToString: string; virtual;
 // ...
end;
```



TObject.ToString

- Solution:
  - the compiler is lying
    mark the methods with “reintroduce;”:

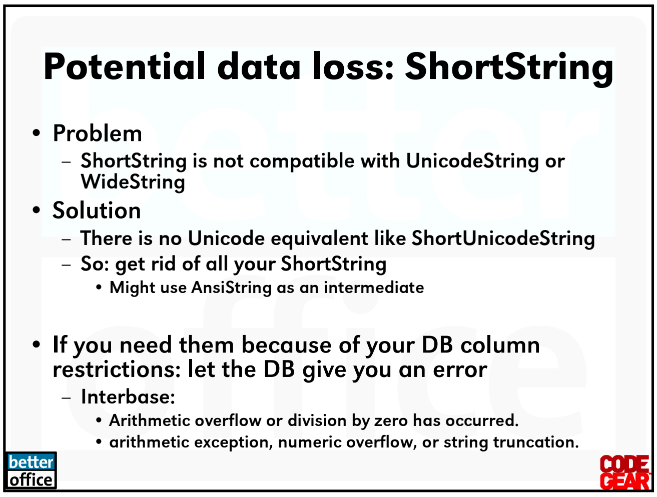
```
type
  TConvert = class
  public
    class function ToString
      (const Value: TSoepExValue): string;
      reintroduce; overload;
    class function ToString
      (const Value: TSoepValue): string;
      reintroduce; overload;
  end;
```



potential data loss: ShortString

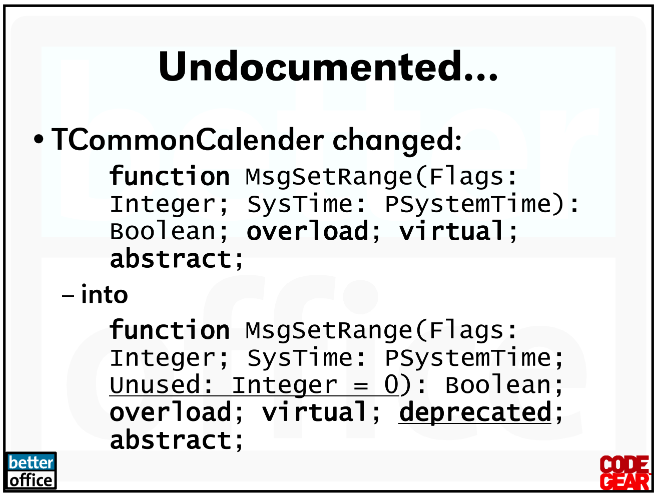
- [DCC Warning] … W1058 Implicit string cast with potential data loss from 'string' to 'ShortString'

```
type
  string36 = string[36];
  TRefId = type string36;

function FromGUID(aGUID: TGUID): TRefId;
begin  (*
  from  '{0229002E-A106-423C-BA2D-04BEE8819126}'
  to     '0229002E-A106-423C-BA2D-04BEE8819126'
  *)
  Result := Copy(GUIDToString(aGUID), 2, 36);
end;
```



Potential data loss: ShortString

- Problem
  - ShortString is not compatible with UnicodeString or WideString
- Solution
  - There is no Unicode equivalent like ShortUnicodeString
  - So: get rid of all your ShortString
    - Might use AnsiString as an intermediate
- If you need them because of your DB column restrictions: let the DB give you an error
  - Interbase:
    - Arithmetic overflow or division by zero has occurred.
    - arithmetic exception, numeric overflow, or string truncation.



Undocumented…

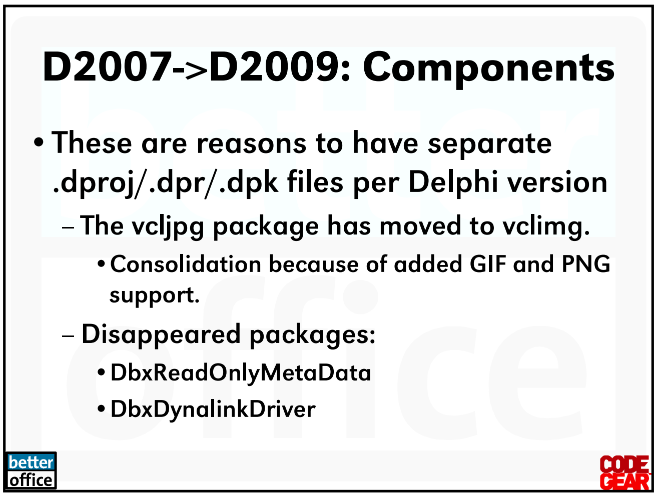
- TCommonCalender changed:

```
function MsgSetRange(Flags:
Integer; SysTime: PSystemTime):
Boolean; overload; virtual;
abstract;
```

  - into

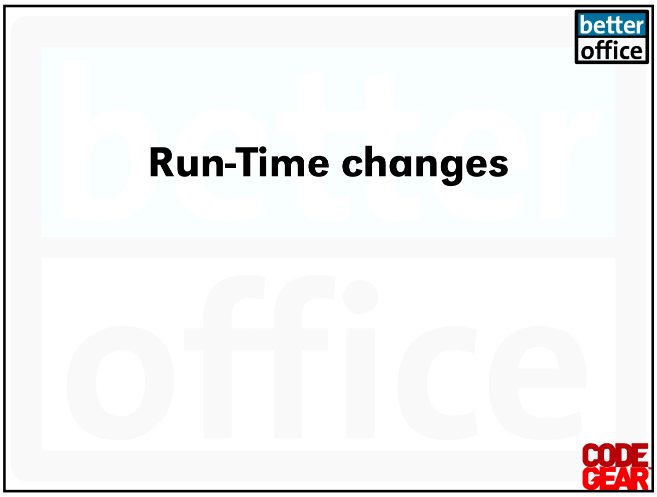
```
function MsgSetRange(Flags:
Integer; SysTime: PSystemTime;
Unused: Integer = 0): Boolean;
overload; virtual; deprecated;
abstract;
```

Building your components

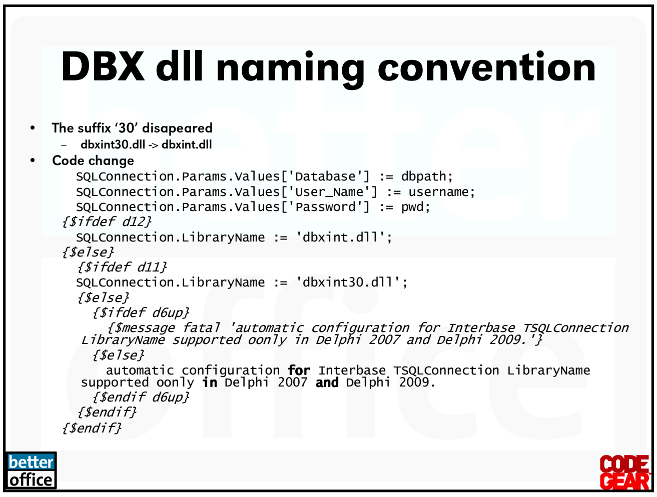


D2007->D2009: Components

- These are reasons to have separate .dproj/.dpr/.dpk files per Delphi version
  - The vcljpg package has moved to vclimg.
    - Consolidation because of added GIF and PNG support.
  - Disappeared packages:
    - DbxReadOnlyMetaData
    - DbxDynalinkDriver



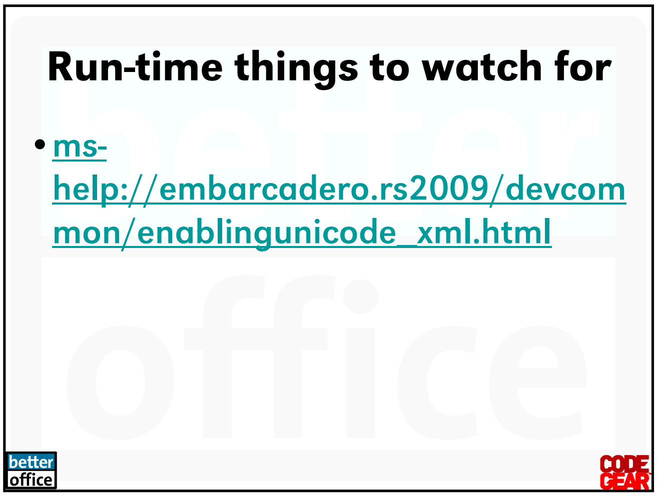
Run-Time changes



DBX dll naming convention

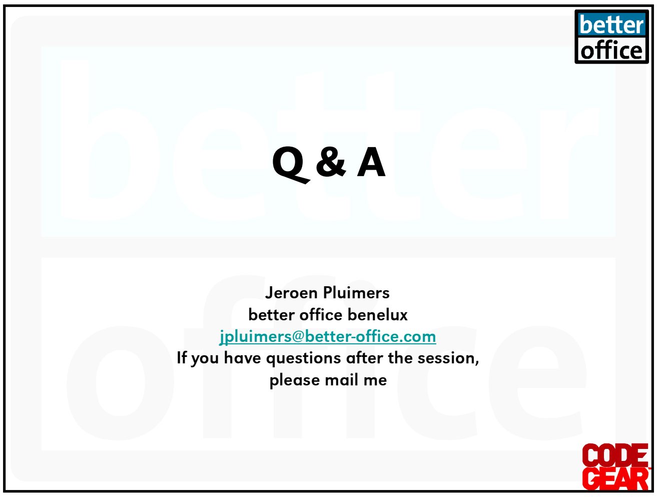
- The suffix '30' disapeared
  - dbxint30.dll -> dbxint.dll
- Code change

```
    SQLConnection.Params.Values['Database'] := dbpath;
    SQLConnection.Params.Values['User_Name'] := username;
    SQLConnection.Params.Values['Password'] := pwd;
  {$ifdef d12}
    SQLConnection.LibraryName := 'dbxint.dll';
  {$else}
    {$ifdef d11}
    SQLConnection.LibraryName := 'dbxint30.dll';
    {$else}
      {$ifdef d6up}
        {$message fatal 'automatic configuration for Interbase TSQLConnection
    LibraryName supported oonly in Delphi 2007 and Delphi 2009.'}
      {$else}
        automatic configuration for Interbase TSQLConnection LibraryName
    supported oonly in Delphi 2007 and Delphi 2009.
      {$endif d6up}
    {$endif}
  {$endif}
```



Run-time things to watch for

- ms-help://embarcadero.rs2009/devcommon/enablingunicode_xml.html



Q & A

Jeroen Pluimers
better office benelux
jpluimers@better-office.com
If you have questions after the session,
please mail me