

Smarter code with databases and data aware controls

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

Smarter code

for ... in with TDataSets

Would it be nice to...

```
var
  Index: TDataSetRecord;
begin
  for Index in XokumClientDataSet do
    if Assigned(Index.DataSet) then
      begin
        // ...
      end;
    end;
end;
```

Helpers

- Introduced in Delphi to support.NET
 - .NET class hierarchy differs of Win32
 - Methods, functions and properties were different or missing
- Class helpers can make these extensions
- Now also available as Record helpers

Helpers

- Helpers (class of record) function as long as the helper is visible to the user
- So:
 - Helper in the same unit,
 - or helper in a unit in the uses list

Helpers for TDataSet & TParams

```
type
  TDataSetHelper = class helper for TDataSet
  public
    //1 Returns first field that matches,
    // if no matching field then exception
    function FieldByAnyName(
      const FieldNames: array of WideString): TField;
    //1 Support for ... in loop providing TDataSetRecord
    function GetEnumerator: TDataSetEnumerator;
  end;

  TParamsHelper = class helper for TParams
  public
    function ParameterByAnyName(
      const ParameterNames: array of WideString): TParam;
  end;
```


Helpers for TDataSet (2)

```

type
  TDataSetEnumerator = class
  strict private
    FDataSet: TDataSet;
    FFirstItem: Boolean;
    FEmpty: Boolean;
  public
    constructor Create(const DataSet: TDataSet);
    function GetCurrent: TDataSetRecord;
    function MoveNext: Boolean;
    //1 Mark the for ... in
    // enumeration result as TDataSetRecord
    property Current: TDataSetRecord
      read GetCurrent;
  end;
  
```

better
office

CODE
GEAR

Overview of data layers

better
office

CODE
GEAR

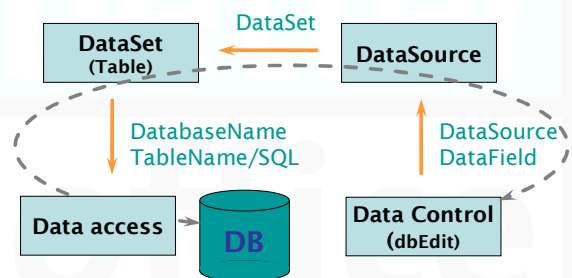
Overview of data layers

- Data Aware controls
- DataSource
- DataSet
- Data Access Solution

better
office

CODE
GEAR

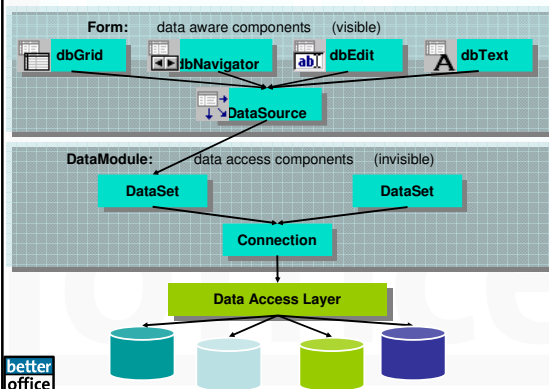
Overview of Data Layers



better
office

CODE
GEAR

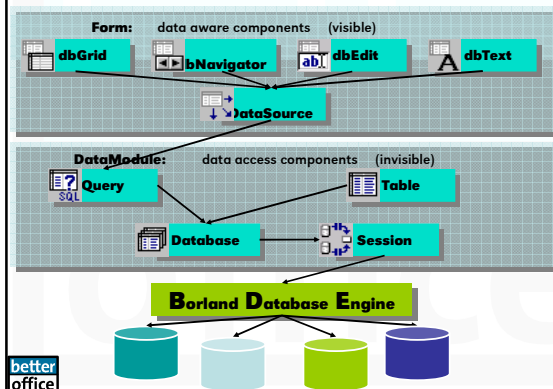
Overview – Generic application



better
office

CODE
GEAR

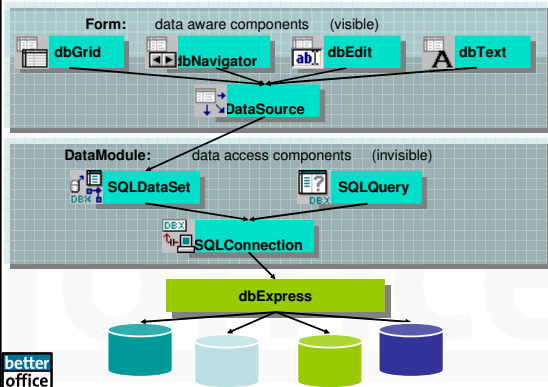
BDE application



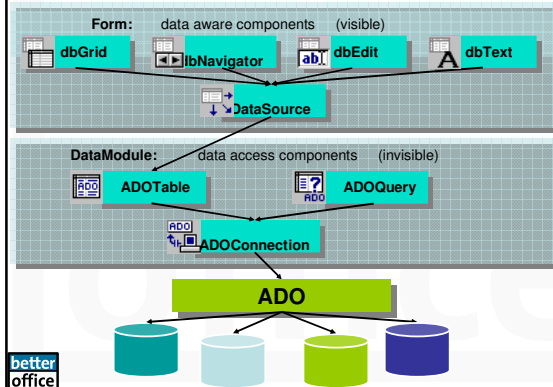
better
office

CODE
GEAR

dbExpress application



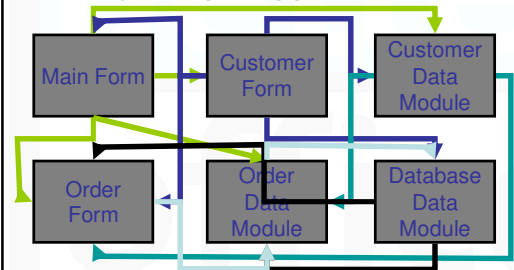
ADO application



Modularizing data layers

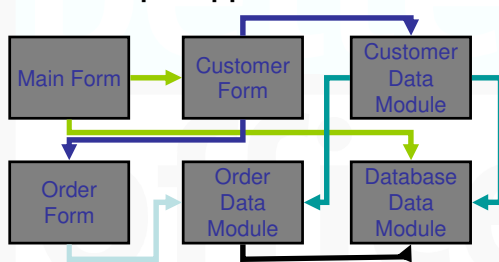
Delphi – Modularization

• Average Delphi app unit structure



Delphi – Modularization

• Good Delphi app unit structure



How come good is better than bad?

• Look for modularization in real life...

- Houses – rooms
- Cities – suburbs/blocks
- People – digestion system
- Parliament – parties

• Sometimes modularization works

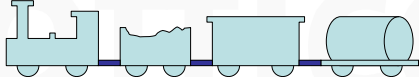
• Sometimes it doesn't 😊

Modules are everywhere...

- So why does it work?

- Internal Cohesion **HIGH**
- External Coupling **LOW**

- If also 'uniform': great!
- If also 'directional': even better!



better
office

CODE
GEAR

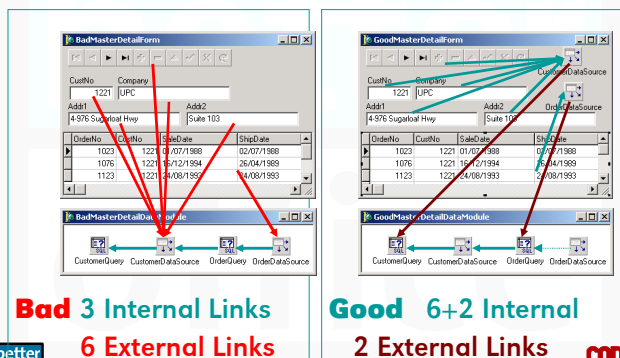
Modularization – database apps

- Let's apply this knowledge to our database apps
- Especially: where to put the DataSource...
- Where is your DataSource?

better
office

CODE
GEAR

Modularization – DataSource



better
office

CODE
GEAR

Modularization – the datasource

- DataSource has **two** goals
 - Binding GUI controls
 - Providing Master-Detail relations
- GUI binding:
put **DataSource** on **Form**
- Master-Detail relations:
put **DataSource** on **DataModule**

better
office

CODE
GEAR

Modularization – gain

- Flexibility
 - Change of GUI
 - Change of Data Access
 - Re-use of modules across projects

better
office

CODE
GEAR

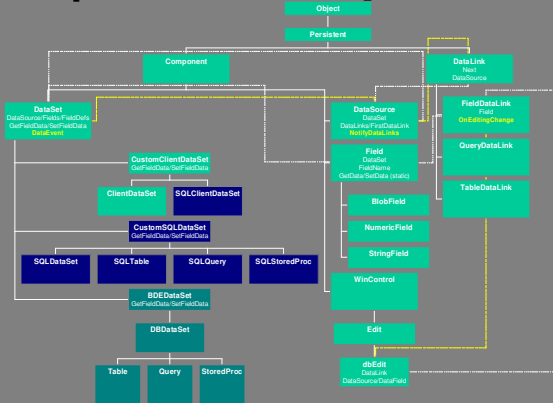
Component hierarchy

[Delphi 5 Object Hierarchy.pdf](#)

better
office

CODE
GEAR

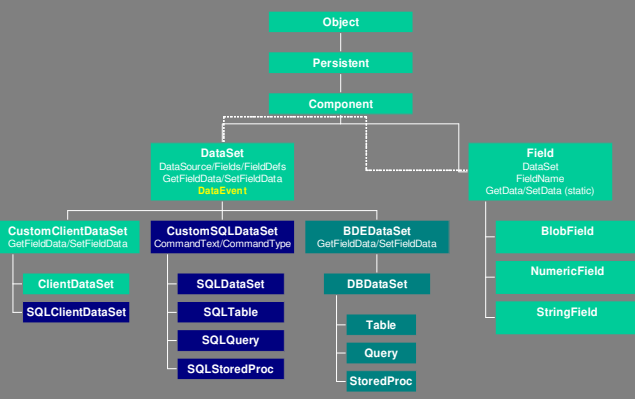
Component hierarchy – overview



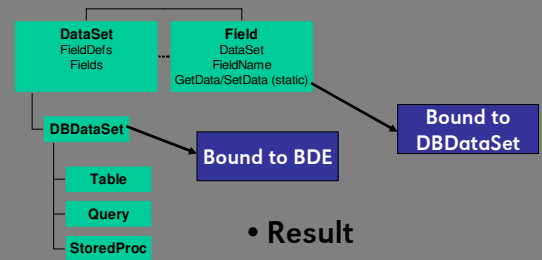
Datasets

GetField/SetField
and more...

Component hierarchy – DataSets

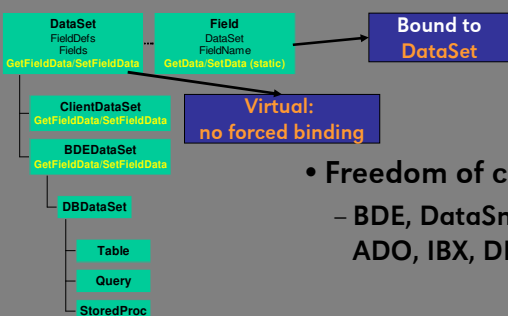


DataSet/Field – until D2



- Result
– Tight coupling
with BDE

DataSet/Field – D3 and up



- Freedom of choice
– BDE, DataSnap,
ADO, IBX, DBX, ...

GUI binding

DataLinks


Component hierarchy – Binding

```
graph TD; Object[Object] --> Persistent[Persistent]; Persistent --> Component[Component]; Persistent --> DataLink[DataLink  
Next DataSource]; Component --> DataSet[DataSet  
DataSource/Fields/FieldDefs  
GetFieldData/SetFieldData  
DataSet.next]; Component --> DataSource[DataSource  
DataSet  
DataLinks/FirstDataLink  
NotifyDataLinks]; Component --> Field[Field  
DataSet  
FieldName  
GetData/SetData (static)]; Component --> WinControl[WinControl]; DataLink --> TableDataLink[TableDataLink]; DataLink --> QueryDataLink[QueryDataLink]; FieldDataLink[FieldDataLink  
Field:  
OnEditingChange] --- Field; Edit[Edit] --- WinControl; dbEdit[dbEdit  
DataLink  
DataSource/DataField] --- WinControl;
```

The diagram illustrates the component hierarchy for binding. The root is **Object**, which leads to **Persistent**. **Persistent** branches into **Component** and **DataLink**. **Component** further branches into **DataSet**, **DataSource**, **Field**, and **WinControl**. **DataLink** branches into **TableDataLink** and **QueryDataLink**. **FieldDataLink** is shown as a specialized **Field** with an **OnEditingChange** event. **Edit** and **dbEdit** are also shown as components.



So: TDataLink is interesting

- Choose which TDataLink (or descendant to use)
 - TDataLink
 - TDataSourceLink
 - TDBCtrlGridLink
 - TDetailDataLink
 - TIBDataLink
 - TMasterDataLink
 - TFieldDataLink
 - TGridDataLink
 - THTTPDataLink
 - TListSourceLink
 - TMultiDimDataLink
 - TNavDataLink



Demo's

- TDbDisplayLabel
 - Shows the TField.DisplayLabel in a data aware control
 - Uses TFieldDataLink
 - Note: changes will not be reflected when DataSet.Active=false this is by design in the VCL (same reason a DBGrid will not show columns in that state)
- Fields editor
 - More than just adding TFields
 - Drag & Drop to other forms
 - Navigate through record



Demo's (2)

- **TDataLinkReflector**
 - Makes the virtual functions of a TDataLink available as Events
- **TRecordArrivedNotifier**
 - For people having a Paradox background
 - property DataSource
 - event OnArriveRecord

Demo's (3)

- **TDataAwareControlController**
 - Dynamically change the appearance of your data aware controls depending on the properties of the TField objects they bind to
 - Reacts on
 - ReadOnly
 - Required
 - Has scope and ownership awareness

Q & A

- Questions after the conference?

jpluimers@better-office.com

