Review: InfoPower 2.0

by Mike Orriss

So what is InfoPower? It is a powerful set of enhancements to the standard Delphi database components. It was first released in May 1995 for Delphi 1.0 and has been one of the most popular Delphi add-ons ever since. Before my critics (and they know who they are!) complain that this review is biased, I freely admit that I have been using the product since it was first available and that it has been used in all my database applications!

InfoPower 2.0 was released at the recent Borland Developers Conference in Anaheim, with support for Delphi 2.0 plus new features and continued Delphi 1.0 compatibility.

For this review, I will concentrate on exploring some (but not all) of the new functionality available. The features discussed below are mainly those that I have actually used that make my development easier and/or improve the quality of the finished application. Hopefully this will give you a useful flavour of the product.

Compatibility Issues

It is worth making the point that the facilities discussed are transparently available for Delphi 1.0 and 2.0 development. Delphi 2.0 now incorporates some of the features available in InfoPower 1.2 (filters for example), but InfoPower 2.0 also allows your Delphi 1.0 programs to use some Delphi 2.0 functionality (eg multiple selection of records in a `TDBGrid`).

In converting programs from InfoPower 1.02 to 2.0, I met just one problem: having to replace use of `filter` with `wwFilter`, caused by conflict with Delphi 2.0's new filter functions. I have had no problems due to InfoPower in relation to maintaining programs between Delphi 1 and Delphi 2. I wish I could say the same about certain other third party products...

One downside to the Delphi 1.0 and 2.0 compatibility is that there is no support for the `TDBGrid` columns feature, so Delphi2 programmers may find this irksome, but similar facilities are available via the `TwwTable` Design Editor.

TwwTable

There are two facilities really worth mentioning here which save me effort. The `Pack` method supports dBase and Paradox tables and there is a new `Query` property. The latter is a `TStringList` in which you place standard SQL syntax. If this `TStringList` is populated, `TwwTable.Open` will run the query instead of the standard table open. A word of caution: having built your query in the property, save it somewhere before temporarily setting the standard `TableName` property to view a table, because the `Query` property gets cleared!

Fine, but why would you want to use `TwwTable.Query` instead of a pukka query? The answer is simple: `LookupCombo` controls only accept lookup tables, not queries.

In my DTopics application, I have a set of category type fields and I wanted a drop down list showing the unique values in the field. In Delphi 1.0, this can be achieved by building a QBE file that generates a single column result table and by using this QBE file as the `TableName` in a `TTable` (eg CATEGORY.QBE). This undocumented feature works quite nicely in Delphi 1.0, but Delphi 2.0 doesn't like it at all! Replacing the QBE with an SQL statement in `TwwTable.Query` fixed the problem and, as a bonus, I no longer need the small QBE files.

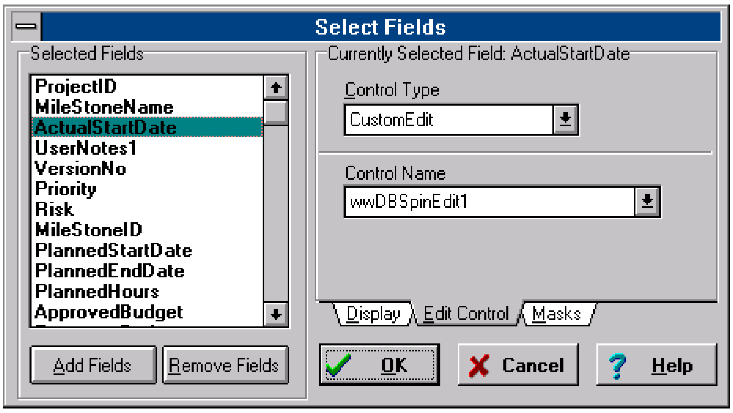
Column settings for the `TwwDBGrid` and the new Picture Mask facility are implemented in `TwwTable`. Changing the properties `ControlType`, `LookupFields`, `LookupLinks` or `PictureMasks` invokes a property editor that displays the `Select Fields` dialog, as shown in Figure 1.

TwwDBEdit

This `TDBEdit` descendant is new to InfoPower 2.0. It supports InfoPower's picture masks (see the section below), word wrap and automatic date detection. It can be embedded in a `TwwDBGrid` (see below). The word wrap support is very useful, particularly for long description fields that can really mess up your carefully designed layout.

The automatic date detection is used to support an autofill facility via the spacebar. The end-user can just fill in as much of the date as is required and tap the spacebar until the date is complete.

For even greater control than that supplied by picture masks, the `OnCheckValue` event handler passes a boolean parameter, `PassesPictureTest`, where you can call your bespoke routines for invalid input or do further tests, calling `SysUtils.Abort` if the tests define that the input is bad.



Select Fields

Selected Fields

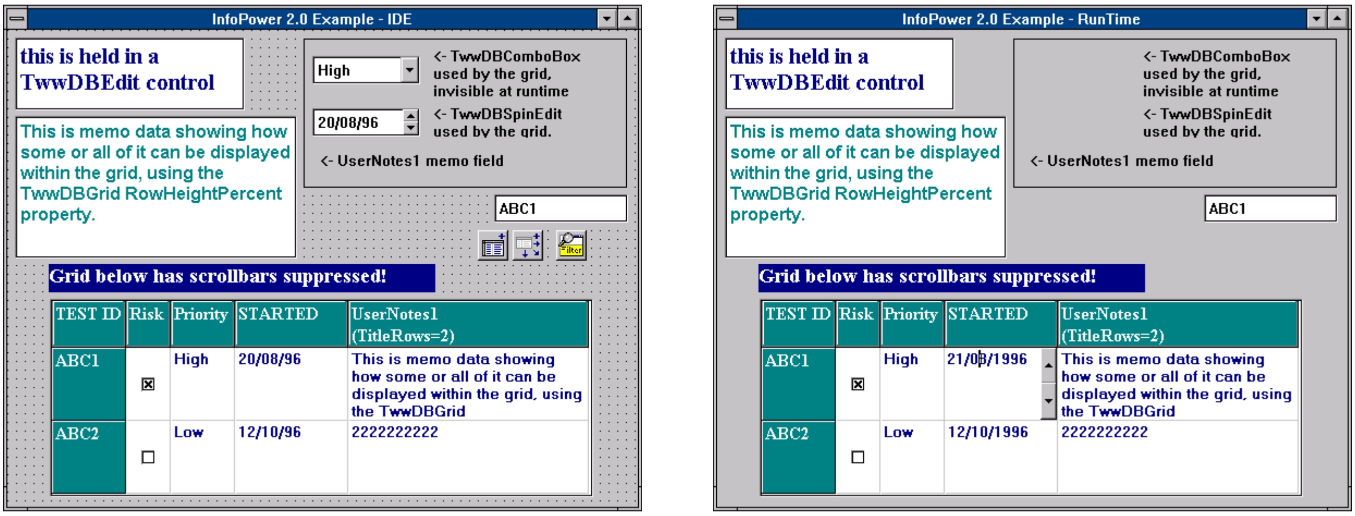
ProjectID
MileStoneName
ActualStartDate
UserNotes1
VersionNo
Priority
Risk
MileStoneID
PlannedStartDate
PlannedEndDate
PlannedHours
ApprovedBudget

Currently Selected Field: ActualStartDate

Control Type: CustomEdit

Control Name: wwDBSpinEdit1

➤ *Figure 1*



InfoPower 2.0 Example - IDE

this is held in a TwwDBEdit control

This is memo data showing how some or all of it can be displayed within the grid, using the TwwDBGrid RowHeightPercent property.

High <- TwwDBComboBox used by the grid, invisible at runtime

20/08/96 <- TwwDBSpinEdit used bv the grid.

<- UserNotes1 memo field

ABC1

Grid below has scrollbars suppressed!

TEST ID	Risk	Priority	STARTED	UserNotes1 (TitleRows=2)
ABC1	☒	High	20/08/96	This is memo data showing how some or all of it can be displayed within the grid, using the TwwDBGrid
ABC2	□	Low	12/10/96	2222222222

InfoPower 2.0 Example - RunTime

this is held in a TwwDBEdit control

This is memo data showing how some or all of it can be displayed within the grid, using the TwwDBGrid RowHeightPercent property.

<- TwwDBComboBox used by the grid, invisible at runtime

<- TwwDBSpinEdit used bv the grid.

<- UserNotes1 memo field

ABC1

Grid below has scrollbars suppressed!

TEST ID	Risk	Priority	STARTED	UserNotes1 (TitleRows=2)
ABC1	☒	High	21/08/1996	This is memo data showing how some or all of it can be displayed within the grid, using the TwwDBGrid
ABC2	□	Low	12/10/1996	2222222222

➤ *Above: Figures 2 and 3*
The same form at
Design Time and Run Time

```
procedure TForm1.GridTitleButtonClick(Sender: TObject;  AFieldName: String);
var len: integer;
    fld: TField;
begin
  with Sender as TwwDBGrid,DataSource.DataSet do begin
    SetActiveField(AFieldName);
    FieldByName(AFieldName).Index := FixedCols;
  end;
end;
```

➤ *Listing 1*

TwwDBSpinEdit

This, like TwwDBEdit, descends from TwwDBCustomEdit and supports the facilities described above, with the exception of word wrap. I find the control most useful for date fields. In addition to the autofill support mentioned above, clicking on the up or down button will change day, month or year, depending on the current caret position within the text portion of the control. This is very similar to the way that the Windows Control Panel supports date changes.

Incidentally, that reminds me. There is some confusion about the date format displayed by data-aware controls when in edit or display mode and, with the millennium changes upon us, it is important to get this right. When displaying data, the format is taken from the underlying TField's DisplayFormat or, if that is not set, from the SysUtils.ShortDateFormat global variable. When editing, the format is always taken from the current ShortDateFormat value.

So, it is simplest to leave DisplayFormat blank for TDateField controls and always use ShortDateFormat and, because some Luddite may reset the Windows system variable to a 2 character year, it is advisable to set your own ShortDateFormat. I do this by loading a value from an INI file or the registry, which can be done in Form.Create or, in Delphi 2.0, in a form unit's Initialization section. Note that this technique fails when running from the IDE if any tables are left open in the IDE, as the edit display mask seems to get set when the session opens.

Back to TwwDBSpinEdit. I was a bit disappointed that, for non-integer values, a button click always changes the value by one. I would have liked a similar facility to dates so that I could increment by 0.01 (say) – Roy please note!

TwwDBComboBox

There are two very useful enhancements here over the standard Delphi control. Incremental searching of the dropped-down list is implemented: a very end-user friendly feature that previously required you to roll your own or buy a third party enhancement.

There is also a new MapList property. This is a TStringList object with a custom property editor that encapsulates the SetText and GetText methods of its underlying TField. This means, for example, that you can set the combobox to display High, Medium or Low but store H, M or L. However, Woll2Woll have missed a small opportunity here: I would like a similar property editor available for the standard TDBRadioGroup. It is inconvenient to edit the Items and Values properties separately, with the obvious danger of their values getting out of step. Still, I can always add a TmoDBRadioGroup to my to-do list!

TwwDBGrid

There have been quite a few improvements here. In particular, I like the new Boolean TitleButtons property. When set to True, it activates the linkage to the OnTitleButtonClick event handler. As an example, the code in Listing 1 moves the clicked column next to the first non-fixed column.

There are other new features that make life easier. The ability to display columns using most of InfoPower's controls, being able to increase row heights and word wrap facilities mean that the grid can satisfy a much greater range of data entry requirements. How many of you have coded a pop-up data entry form just because there are a couple of fields that the grid can't handle?

One item for the wish list though. The RowHeightPercent property affects the whole grid. I would like to see a boolean VariableRowHeight property together with an associated MaxRowHeightPercent.

Picture Masks

InfoPower supports a data entry template, or mask, for the values that can be entered into a field displayed on the screen (Figure 4). This functionality duplicates the Picture function available in Paradox, but with a couple of enhancements for Delphi. First, it is available for all database types, not just Paradox, and second, it offers more control. Validation is performed before posting the record and, optionally via the InfoPower data-aware controls' UsePictureMask property, on exiting a control. Note that UsePictureMask has to be set to False though, if you are using Tfield GetText and SetText methods to vary the displayed or stored data, because the mask relates to stored data, but is checked against the displayed data. There is a similar incompatibility when using a TwwDBComboBox with MapList set to True.

Character	Description
#	Any digit (0-9)
?	Any letter (a-z or A-Z)
&	Any letter (a-z or A-Z – convert to upper case)
~	Any letter (a-z or A-Z – convert to lower case)
@	Any character (letters convert to upper case)
!	Any character
;	Next character is to be used literally
*	Repeat the following character any number of times; for instance *& means convert any number of characters to upper case; to specify a specific number of times, follow the * with a number, so to specify 5 numbers in a row you would use *5{#}
[abc]	Optional sequence of characters abc that do not need to be entered by the end-user
{a,b,c}	Grouping operator: the end-user must choose either a, b or c; for instance if you have three valid choices of Auto, Automobile and Car, set the picture mask to {Auto[mobile],Car}
,	Separator allowing multiple masks
If you enter any other character in a picture mask, InfoPower treats the character as a constant. InfoPower automatically fills-in (enters or types) the constant for the end-user, provided that the AutoFill property setting is True	

➤ *Figure 4: InfoPower Picture Mask elements*

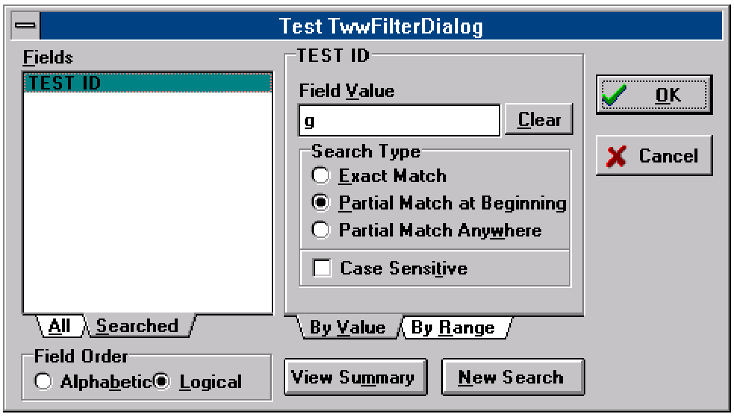
TwwFilterDialog

I haven't actually used this component yet, having already rolled my own, but I must include it in this review because of its potential power. This non-visual control gives the end-user a visual filter facility, with the SQL required being constructed behind the scenes. The example (Figure 5) was created by dropping the control on the form, setting its DataSource property and placing the following code in a button OnClick event:

```
TwwFilterDialog1.Execute
```

Versatility is assured by the developer being able to restrict the field list displayed and the availability of the OnInitDialog and OnExecuteSQL event handlers.

➤ *Figure 5*



Test TwwFilterDialog

Fields: TEST ID

TEST ID

Field Value: g

Search Type

○ Exact Match

● Partial Match at Beginning

○ Partial Match Anywhere

□ Case Sensitive

Field Order

○ Alphabetic ● Logical

Wrinkles And Gotchas

Well, it doesn't come free in the Delphi box!

Summary

This brief review does not cover all the components and functionality available, but should give you some idea of the smoothness and professionalism of this product. Although InfoPower 2.0 is an evolution rather than a revolution, the latest version just enhances its reputation.

The biggest new feature is the Picture Mask support, which has already saved me hours of work. Other notable enhancements include the end-user filter dialog, enhanced grid display options and the new SpinEdit control.

Is it worth the money? If you develop database apps, then generally the answer is yes, but a Delphi 2.0 only developer who does relatively little database work might find the cost difficult to justify. For myself, as a database developer supporting Delphi 1.0 and Delphi 2.0 apps, this is now even more of a *must have* than the previous release.

InfoPower is produced by Woll2Woll Software. You may contact them via Compuserve (GO WOLL2WOLL which takes you to WINAPI, Section 5) or via the Web at http://woll2woll.com. Demo and trial versions are available (INFOTRL.ZIP and INFODEMO.ZIP). Any software tools retailer worth his or her salt will stock InfoPower and quote you a competitive price (check the ads in this issue).