Companion Disk…

A run-down of what's included on the free disk with this issue

Included on our companion disk each month are all the source code and example files from the articles in that issue. In addition, we aim to use the spare space to bring you extra goodies to try out. As I explained last month, we have decided to begin including trial versions of high quality commercial software (components, libraries and utilities). This month, however, we're making an exception!

Inprise recently ran a survey on their website called the *Spirit of Delphi*. The goal was to decide who, over the years Delphi has been around, has best captured and shared the spirit of Delphi. Visitors to the website were invited to submit names and to vote for their choice. The results were announced at the recent Inprise ICon98 conference in Denver. We're delighted to report that regular contributors "Dr.Bob" Swart and Marco Cantù received honourable mentions, but there was a tie for first place. Congratulations to the joint winners: Robert Czerwinski, author of *The Delphi Super Page* (searchable via developers.href.com, a European mirror is at http://sunsite.icm.edu.pl/delphi/) and the creators of *RX Library*.

RX Library is a mammoth library of components (covering three pages on my palette!) for Delphi 1, 2, 3 and 4, plus C++Builder 1 and 3. Not only is this library well thought out, well designed and well implemented, it is free and comes with full source! In recognition of the outstanding efforts of the developers of RX Library, Fedor Koshevnikov, Igor Pavluk, and Serge Korolev, we are including the latest version of the library on this month's disk, in the RXLIB folder.

RX Library 2.50

RX Library contains very many components, objects and routines. You'll see a full listing after installing the library, but here are a few highlights…

`TRxDBLookupCombo` provides an incremental search through a lookup table by directly typing into the combo control while the lookup list is displayed. Lookup-Source can refer to a `TTable`, `TQuery`, `TRxQuery` or `TQBEQuery`. It even incrementally searches on query results and much more…

`TRxDBComboBox` is a `TDBComboBox` descendant which allows you to have different values displayed to those stored in the database without using a lookup table.

`TRxDBGrid` allows you to change the background colour and font displayed within individual cells as well as entire rows and columns. You can save and restore column order and display width in INI files and the registry. You can display icons for BLOB, memo, OLE and picture fields, select multiple records and convert column headings to buttons.

`TDBStatusLabel` displays the `DataSet` state (for all datasets) or the current record number (for dBase or Paradox tables).

`TQBEQuery` enables Delphi applications to use Paradox-style Query-By-Example (QBE) statements to query tables and perform insert and update queries.

`TRxDBFilter` encapsulates the BDE's ability to filter records locally. The component provides event on filtering and/or conditions in a `StringList` property.

`TDBProgress` displays BDE operations progress for IDAPI drivers that support callback functions.

`TRxDBRichEdit` displays and enters rich text in a memo or BLOB field (Delphi 2.0 or higher).

`TRxCalcEdit` and `TRxDBCalcEdit` (data-aware) take the display and editing of numeric data one step further: they provide a popup calculator to help you calculate the number.

`TFormPlacement` allows to save and restore form size, position and window state using an INI file or the registry. `TFormStorage` allows you to read and write published properties of virtually any component to an INI file or the registry with virtually no code.

A `TSpeedBar`, with horizontal or vertical orientation, can be placed at the top, bottom, left or right of the form. You can customise the speedbar at design-time and run-time using drag and drop. You can use flat and transparent buttons.

`TRxTimerList` component provides all the functions of `TTimer`, but uses only *one* Windows timer for up to 32,767 timing events!

`TAppEvents` is an `Application` wrapper component, which makes it easier to work with the `TApplication` object properties and events at design time.

Is there a downside? Well, the documentation is in Russian only, but the components and routines are self-explanatory and if in doubt a little digging in the source files or included demo programs should enlighten you. Enjoy!