

UnitUsage

Einführung

Lizenz

UnitUsage steht unter einer Lizenz, die an die ZLib Lizenz angelehnt ist:

This software is provided "as is," without warranty of any kind, express or implied. In no event shall this software or its contributors be held liable for any direct, indirect, incidental, special or consequential damages arising out of the use of or inability to use this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. Redistributions of source code must retain the above copyright notice, definition, disclaimer, and this list of conditions.
2. Redistributions in binary form (compiled executables) must reproduce the above copyright notice, definition, disclaimer, and this list of conditions in documentation and/or other materials provided with the distribution.
3. Altered versions must be plainly marked as such and must not be misrepresented as being the original source. Such altered versions also must not be misrepresented as being the original software.

Einige Teile der Software stammen nicht vom Autor. Sie wurden von der „Community“ in Foren und Sites zur Verfügung gestellt. Diese Teile der Software sind in der Regel innerhalb des Sourcecodes gekennzeichnet. Eine Liste der verwendeten Sites findet sich in der About-Box von UnitUsage. Für diese Code Teile gilt u.U. eine andere Lizenz, welche auf den entsprechenden Seiten gefunden werden kann.

Die in der Software verwendeten Icons für die Buttons stehen unter einer eigenen Lizenz. Die Icons stammen von glyFX (www.glyfx.com). Die Lizenz für diese Icons kann dort nachgelesen werden.

Zweck

UnitUsage soll dabei helfen die Unit-Abhängigkeiten eines Delphi/Borland-Pascal Projektes zu visualisieren und interaktiv „erforschbar“ zu machen.

UnitUsage verwendet hierzu die Grafikbibliothek **GraphViz**, welche von www.graphviz.org bezogen werden kann.

UnitUsage erkennt nicht, welche Units unnötigerweise in den USES Statements der einzelnen Units aufgeführt sind; UnitUsage übernimmt alle im Sourcecode referenzierten Units für die Darstellung.

Anwendung

Startet man UnitUsage erstmals, so präsentiert es sich wie folgt:

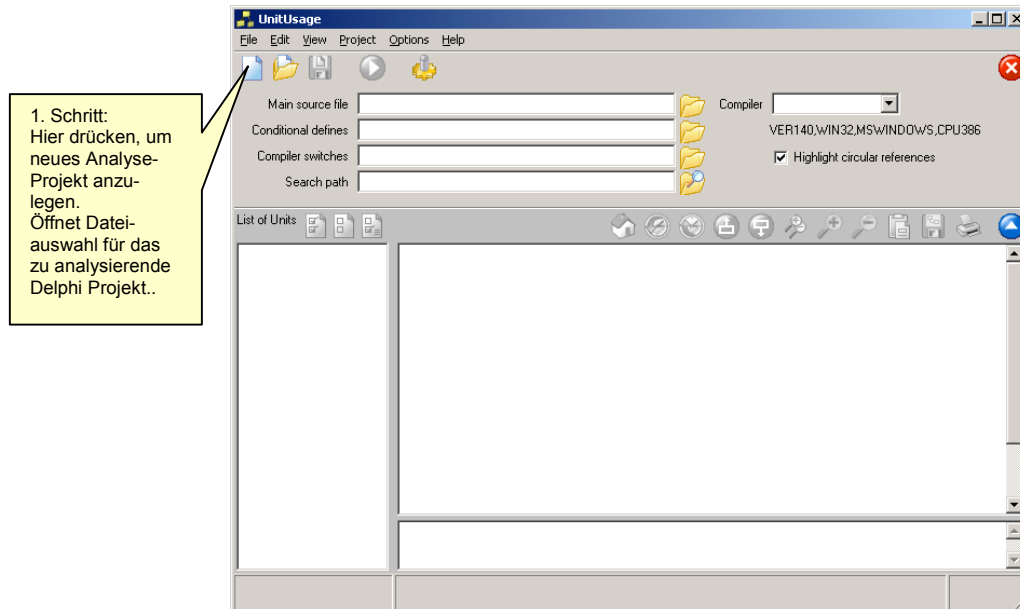


Abbildung 1 Anlegen eines neuen UnitUsage Projektes

Um ein neues UnitUsage-Projekt anzulegen drückt man einfach auf den Icon mit dem leeren Blatt links oben. Es öffnet sich ein Dateiauswahldialog zum Laden der zu analysierenden Delphi-Projektdatei. Wählen Sie hier die .dpr Datei Ihres Delphi-Projektes.

UnitUsage präsentiert sich dann wie folgt:

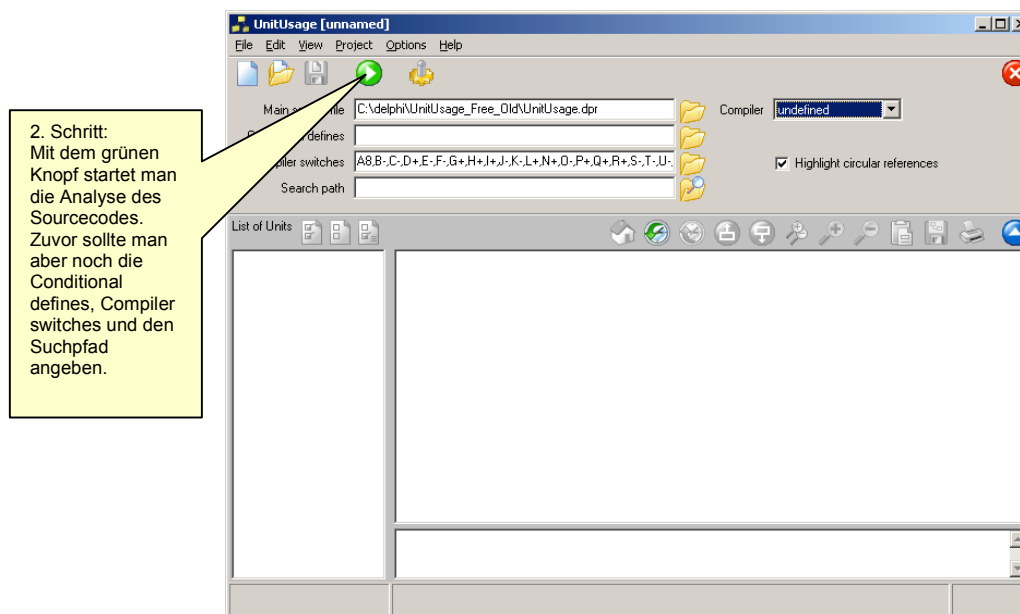


Abbildung 2 Start der Code-Analyse durch Drücken des grünen Knopfes

Conditional defines und Compiler switches werden automatisch aus der Konfigurationsdatei gelesen, falls vorhanden. Sie können aber selber Daten eingeben oder andere Konfigurationsdateien importieren. (Verwenden Sie hierzu die Icons mit dem gelben Ordner Symbol jeweils rechts neben den Eingabefeldern).

Mit Hilfe der Compiler Listbox lassen sich noch bequem die Default-Compiler Direktiven für diverse Delphi Compiler hinzunehmen. Diese werden dann unter der Listbox angezeigt und bei der Analyse des Sourcecodes berücksichtigt.

Nachdem nun das Projekt eindeutig bestimmt ist, drücken Sie den grünen Knopf (Abbildung 2), um die Analyse des Sourcecodes vorzunehmen.

Falls die Analyse ohne (größere) Fehler durchgeführt werden konnte, wird UnitUsage das Programm *dot.exe* aufrufen, um die Grafik zu berechnen. Der Pfad zu *dot.exe* kann nötigenfalls im Optionen-Dialog eingestellt werden.

Im Memo-Fenster rechts unten wird die Code-Analyse übrigens dokumentiert und etwaige Fehler ausgegeben.

Voila, der erste Graph ist sichtbar:

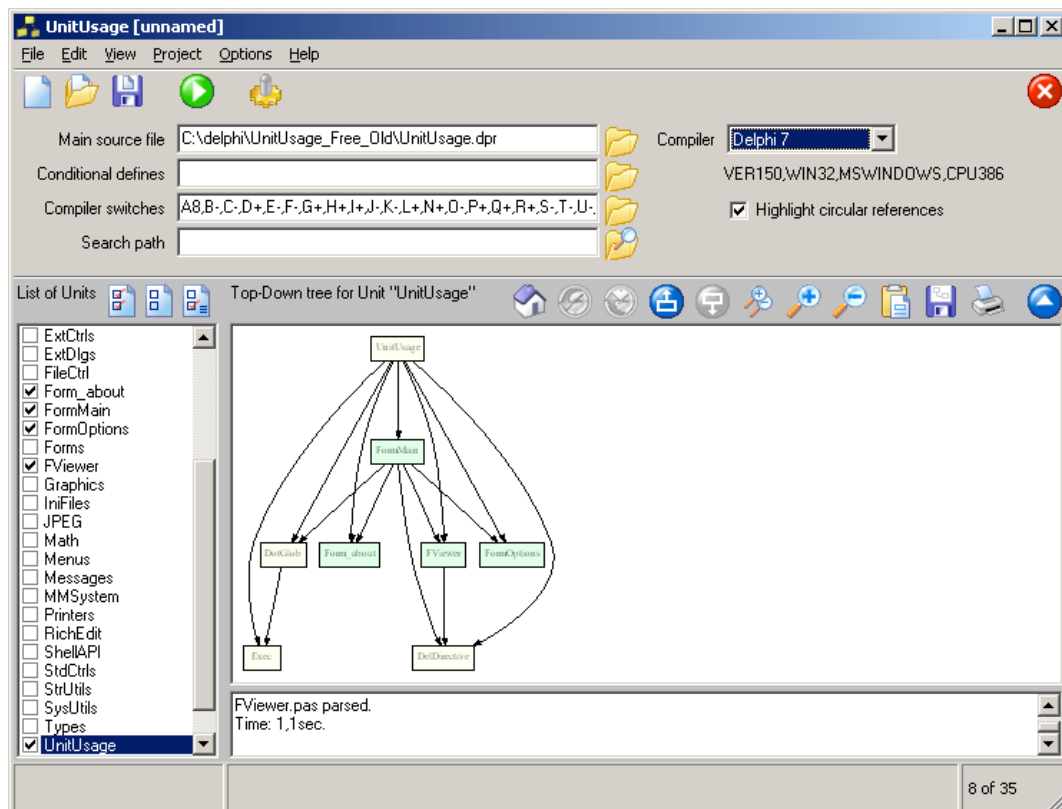


Abbildung 3 Der erste Graph mit der Delphi-Projektdatei als Ausgangspunkt.

Um Platz für die Grafik zu machen kann man übrigens das Panel mit den Projektdaten einklappen (blauer Knopf ganz rechts).

Bewegt man nun den Mauszeiger über die Grafik, so werden die Verbindungen zu/von der Unit unterm Cursor grafisch hervorgehoben (Abbildung 4).

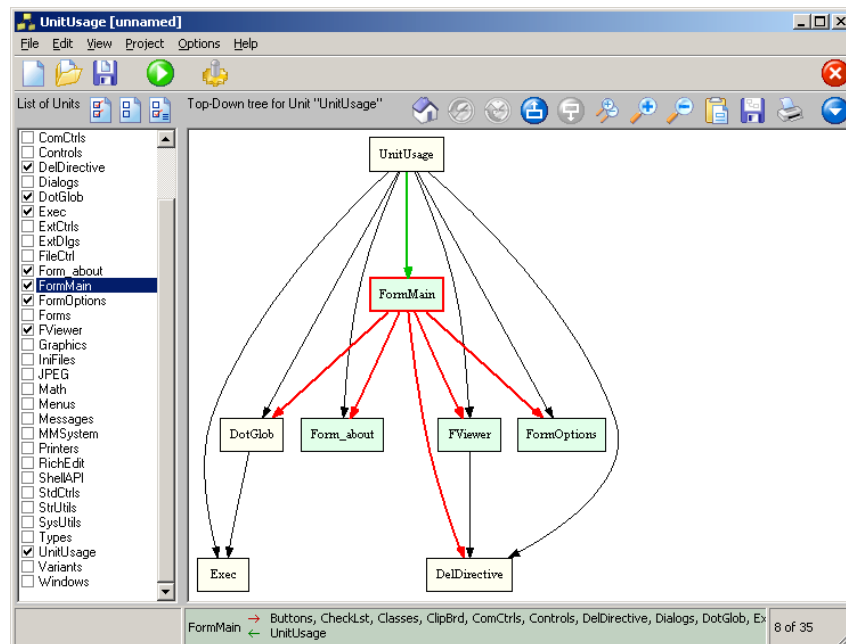


Abbildung 4 Elemente unterm Mauszeiger werden samt Verbindungen hervorgehoben

Über die “List of Units” an der linken Seite kann man angeben, welche Units mit in die Grafik aufgenommen werden sollen. Dabei unterscheiden die Checkboxes drei Zustände:

- Ausgeschlossen (Checkbox leer)
- Eingeschlossen, aber ohne “Kinder” (Checkbox ausgegraut)
- Eingeschlossen mit allen Kindern (Checkbox “gecheckt”)

Über der Liste befinden sich drei Schalter für die Schnellauswahl von

- Alle wählen
- Keine wählen
- Nur Units mit Sourcecode wählen (default)

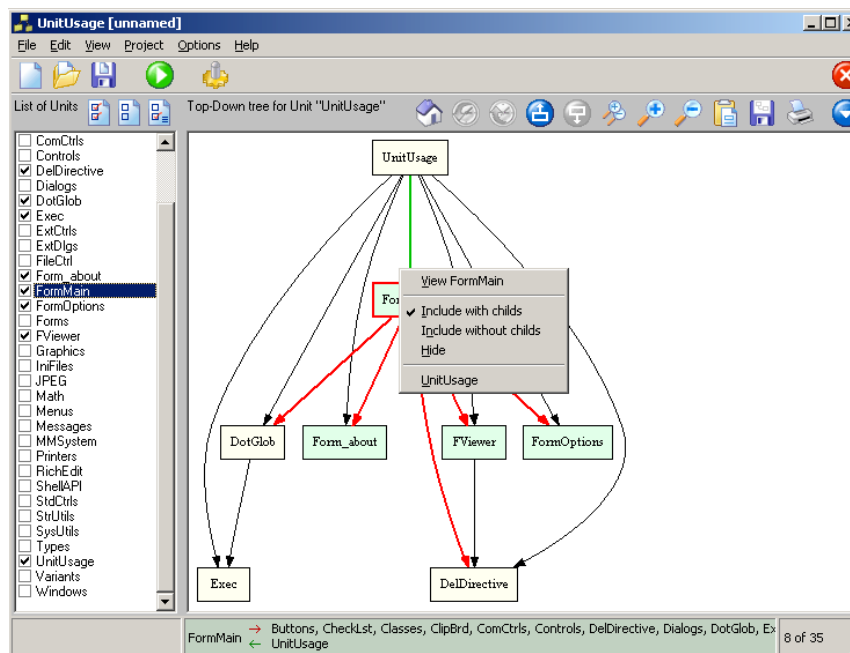


Abbildung 5 Kontextmenü im Graphen

Die Einstellung, ob eine Unit mit in den Graph eingebaut werden soll, mit oder ohne Kinder etc. kann auch über das Kontextmenü erfolgen. Dazu klickt man mit der rechten Maustaste auf das entsprechende Unit-Symbol. (Siehe Abbildung 5)

Nun wird der Graph bei großen Projekten schnell unübersichtlich. Hier ist es dann sinnvoll, sich nur einen Teil des Graphen anzusehen. Man kann jederzeit einen neuen Ausgangspunkt für den Graphen wählen, in dem man eine andere Unit mit Doppelklick im Graphen oder in der Unit-Liste anwählt:

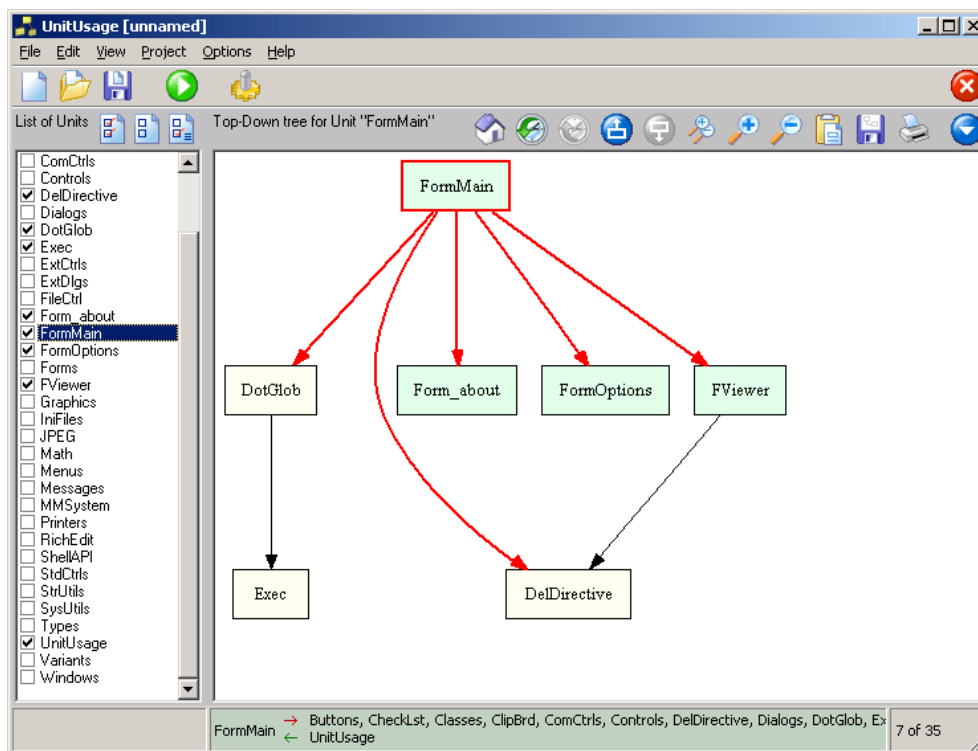


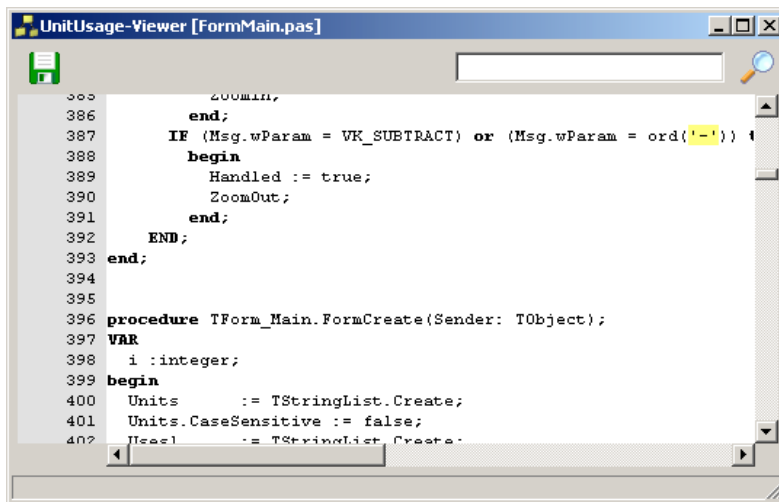
Abbildung 6 Mittels Doppelklick auf FormMain wurde dieses als Basis für den Graphen ausgewählt.

Auf diese Art kann man sich rasch im Baum nach unten durch-klicken. Nach oben ist es etwas unbequemer: Die “Eltern” Units einer Unit erhält man über das Kontextmenü (rechter Mausklick).

Die Grafik kann automatisch skaliert werden. Mit der + und – Taste und den zugehörigen Zoom-Schaltern kann jederzeit die Größe variiert werden.

Die Grafiken können sowohl als Bitmap als auch als Vektor-Graphik exportiert werden. Letzteres bietet die uneingeschränkte Skalierbarkeit der Grafik in anderen Anwendungen und beim Drucken.

Über das Kontext Menü kann man sich auch den Sourcecode anzeigen lassen:



```
383      ZoomIn;
386      end;
387      IF (Msg.wParam = VK_SUBTRACT) or (Msg.wParam = ord('-')) then
388      begin
389          Handled := true;
390          ZoomOut;
391      end;
392  END;
393 end;
394
395
396 procedure TForm_Main.FormCreate(Sender: TObject);
397 VAR
398     i :integer;
399 begin
400     Units      := TStringList.Create;
401     Units.CaseSensitive := false;
402     Ucsel      := TStringList.Create;
```

Abbildung 7 Sourcecode Anzeige

Diese Anzeige erlaubt nicht, den Sourcecode zu editieren. Im Gegensatz zu anderen Editoren aber berücksichtigt dieser Viewer die gesetzten Compiler-Directiven. Nicht verwendeter Code wird wie Kommentar ausgegraut dargestellt.