Dokumentation - Unit fdDialog (Version 1.1)

function Msg (const txt: string; Btns: TdlgButtonSet = []): TdlgResult;
function ErrMsg (const txt: string; Btns: TdlgButtonSet = []): TdlgResult;
function WarnMsg (const txt: string; Btns: TdlgButtonSet = []): TdlgResult;
function Ask (const txt: string; Btns: TdlgButtonSet = []): TdlgResult;

Jede dieser 4 Funktionen erzeugt einen modalen Dialog. Die Funktionen unterscheiden sich im Icon, das neben dem Haupt-Dialogtext angezeigt wird. Die Dialoge können durch eine Reihe von Prozeduraufrufen konfiguriert werden, die vor dem eigentlichen Dialogaufruf stattfinden.

Txt	Message- oder Fragetext, der auch mehrzeilig sein kann.
Btns	Menge der anzuzeigenden Standard Buttons.

Der
Aufzähltyp
TdlgResult=(dlgNone,dlgTimeOut,dlgCancel,dlgAbort,dlgNo,dlgClose,dlgOK,dlgYes,dlgRetry, dlgOpen,dlgSave,dlgNoToAll,dlgYesToAll,dlgHelp,dlgIgnore,dlgBtn1,dlgBtn2, dlgBtn3,dlgBtn4,dlgBtn5,dlgBtn6,dlgBtn7,dlgBtn8,dlgBtn9,dlgBtn10);
enthält alle möglichen Rückgabewerte dieser Dialogfunktionen. Dabei entspricht die Subrange
TdlgButton=dlgCancel..dlgIgnore
den vordefinierten Standardbuttons. TdlgButtonSet ist ein set of TdlgButton. Die Standard-Buttons sind mit folgenden Texten belegt:

dlgCancel	Abbrechen	*dlgIgnore*	Ignorieren
dlgAbort	Abbrechen	*dlgClose*	Schließen
dlgNo	Nein	*dlgOK*	OK
dlgNoToAll	Alle nein	*dlgOpen*	Öffnen
dlgYes	Ja	*dlgSave*	Speichern
dlgYesToAll	Alle ja	*dlgHelp*	Hilfe
dlgRetry	Wiederholen		

Als Rückgabewert liefern die Funktionen den Wert, der dem angeklickten Button entspricht. Darüber hinaus können mit Hilfe der *procedure dlgAddButtons* zusätzliche Buttons definiert werden. Außer den Werten von TdlgButtonSet sind noch folgende Rückgabewerte möglich:

dlgTimeOut	Timeout erreicht (siehe *dlgSetTimeout*)
dlgBtn1	Erster individueller Button gedrückt (siehe *dlgAddButtons*)
...	
dlgBtn10	Zehnter individueller Button gedrückt (siehe *dlgAddButtons*)

Einige Konstante für häufig gebrauchte Buttonkombinationen sind als Konstante vordefiniert:

dlgYN	*[dlgNo,dlgYes]*
dlgYNAll	*[dlgNo,dlgNoToAll,dlgYes,dlgYesToAll]*
dlgAR	*[dlgAbort,dlgRetry]*
dlgRI	*[dlgRetry,dlgIgnore]*
dlgARI	*[dlgAbort,dlgRetry,dlgIgnore]*

Dokumentation - Unit fdDialog (Version 1.1)

function AskYN (const txt: string): boolean;

Diese Funktion erzeugt einen Dialog mit zwei Buttons und liefert ein boolean Ergebnis, je nachdem, welcher Button gedrückt worden ist. Es müssen entweder vorher mit Hilfe von dlgAddButtons zwei individuelle Buttons erzeugt worden sein, von denen automatisch der erste zum Yes-Button und der zweite zum No-Button wird, oder es darf keinen individueller Button geben – in dem Fall werden automatisch die beiden Buttons dlgYes und dlgNo erzeugt.

Txt	Message- oder Fragetext, der auch mehrzeilig sein kann.
Beispiel:	*If AskYN('Wollen Sie weitermachen')...* erzeugt einen Dialog mit den Tasten Ja und Nein. Ein Klick auf Ja liefert true, ein Klick auf Nein liefert false. *dlgAddButtons('Ja, Weitermachen','Nein, Abbrechen');* *if AskYN('Wollen Sie weitermachen')...* erzeugt einen Dialog mit den individuell beschrifteten Tasten „Ja,Weitermachen“ und „Nein, Abbrechen“. Ein Klick auf Weitermachen liefert true, ein Klick auf Abbrechen liefert false.

function AskYN (const txt: string; YesBtn, NoBtn: TdlgResult): boolean;

Diese Funktion erzeugt einen Dialog mit zwei Buttons und liefert ein boolean Ergebnis, je nachdem, welcher Button gedrückt worden ist. Es dürfen vor dem Dialogaufruf keine anderen Buttons erzeugt werden als die in YesBtn und NoBtn übergebenen Buttons. Standardbuttons müssen (und können) gar nicht vorher erzeugt werden, die werden automatisch erzeugt, wenn sie in YesBtn oder NoBtn übergeben werden.

Txt	Message- oder Fragetext, der auch mehrzeilig sein kann.
YesBtn	Dieser Button entspricht der Antwort yes (=true). Falls der Button kein Standardbutton ist, muss er mit Hilfe von dlgAddButtons vor dem Dialogaufruf erzeugt werden.
NoBtn	Dieser Button entspricht der Antwort no (=false). Falls der Button kein Standardbutton ist, muss er mit Hilfe von dlgAddButtons vor dem Dialogaufruf erzeugt werden.
Beispiel:	*If AskYN('Wollen Sie weitermachen', dlgYes, dlgAbort)...* erzeugt einen Dialog mit den Tasten Ja und Abbrechen. Ein Klick auf Ja liefert true, ein Klick auf Abbrechen liefert false. *dlgAddButtons('Weitermachen');* *if AskYN('Wollen Sie weitermachen',dglBtn1,dlgAbort)...* erzeugt einen Dialog mit den Tasten Weitermachen und Abbrechen. Ein Klick auf Weitermachen liefert true, ein Klick auf Abbrechen liefert false.

Dokumentation - Unit fdDialog (Version 1.1)

procedure dlgAddButtons (Btn1: string; Btn2: string=''; Btn3: string=''; Btn4: string='';Btn5: string='');

Mit dieser Prozedur werdem dem Dialog individuell beschriftete Buttons hinzugefügt. Dabei können mit jedem Aufruf der Prozedur bis zu 5 Buttons hinzugefügt werden. Insgesamt sind maximal 10 Buttons mit individueller Beschriftung zusätzlich zu den Standardbuttons möglich. Die entsprechenden Rückgabewerte für diese Buttons sind dlgBtn1 bis dlgBtn10.

Es werden allerdings in der derzeitigen Programmversion alle Buttons in nur einer Buttonreihe angezeigt, bei sehr vielen Buttons kann dabei der Platz knapp werden. Alternativ kann man dem Dialog auch ein Panel mit zusätzlichen Buttons mitgeben.

Wenn beim Starten des Dialogs kein einziger Button (also weder ein Standardbutton noch ein individueller Button) und auch kein Timeout festgelegt wurde, wird dem Dialog automatisch der OK-Button hinzugefügt.

procedure dlgSetTimeout (ms: integer);

Mit der Prozedur wird ein Timeout (in Millisekunden) festgelegt. Nach Ablauf der Zeit wird das Dialogfenster automatisch geschlossen. In dem Fall ist der Rückgabewert der Dialogfunktion *dlgTimeout*. Wenn ein Timeout gesetzt wird, kann der Dialog auch ganz ohne Buttons bleiben. Soll ein OK Button in einem solchen Dialog angezeigt werden, dann muss der Dialogfunktion [dlgOK] als zweiter Parameter mitgegeben werden (Nur in Dialogen ohne Timeout wird dlgOK automatisch hinzugefügt, wenn es sonst keinen Button gibt).

procedure dlgSetDlgType(DlgType: TdlgType);

Bei jedem Dialog kann ein Windows-Icon (für Hinweis, Warnung, Fehler oder Frage) angezeigt werden, und zwar je nach dem Dialogtyp.
type
TdlgType=(dlgDefault, dlgNoIcon, dlgInfo, dlgWarning, dlgError, dlgQuestion);
Standardmäßig ist der Dialogtyp des Fensters dlgDefault, dann hängt das angezeigte Icon von der Aufruffunktion ab:

Msg	*dlgInfo*
WarnMsg	*dlgWarn*
ErrMsg	*dlgError*
Ask	*dlgQuestion*
AskYN	*dlgQuestion*

Mit dem Aufruf *dlgSetDlgType(dlgNoIcon)* wird die Anzeige des Icons ganz abgeschaltet. Auch wenn der Messagetext leer ist, wird kein Dialogicon angezeigt.

Dokumentation - Unit fdDialog (Version 1.1)

procedure dlgSetCanClose (Canclose: TcanCloseFunc);

Dem Dialog kann eine Prüfroutine mitgegeben werden, die bei ungültigen Eingaben durch den Rückgabewert *false* verhindern kann, dass der Dialog geschlossen wird. Die Prüfroutine sollte eine Methode des Hauptformulars sein:
type TCanCloseFunc = function(Btn: TdlgResult): boolean of object
Wenn einer der Buttons *dlgCancel* und *dlgAbort* geklickt wurde, dann wird die Prüfroutine nicht aufgerufen, beim Klick auf diese Buttons (falls vorhanden) wird der Dialog auf jeden Fall beendet. Auch bei Erreichen eines gesetzten Timeout wird der Dialog ohne Aufruf der Prüffunktion beendet.

procedure dlgSetposition (Left,Top: integer; Relative: boolean=true);

Die Anzeigeposition am Bildschirm kann manuell gesetzt werden. Dabei beziehen sich die Koordinaten *Left* und *Top* normalerweise auf den linken oberen Eckpunkte des aufrufenden Formulars. Wird *Relative=false* übergeben, dann sind *Left* und *Top* die absoluten Bildschirmkoordinaten.
Wird diese Funktion nicht aufgerufen, dann erscheint der Dialog in der Mitte des vorher aktiven Fensters. Weitere rekursiv aufgerufene Dialoge werden dann bei jeder weiteren Rekursionsstufe etwas nach rechts und nach unten versetzt angezeigt.

procedure dlgSetFontsize (size: integer);

Mit der Prozedur kann die Schriftgröße der Buttons und des Messagetexts für den nächsten Dialog verändert werden (der Standardwert für alle Dialoge wird mit *dlgSetStandardFontsize* gesetzt). Die Prozedur sollte vor dem Aufruf von *dlgAddButtons* ausgeführt werden, da die Schriftgröße der individuellen Buttons sonst nicht verändert wird.

procedure dlgSetFont (Font: TFont: integer);
Mit der Prozedur kann die Schriftart der Buttons und des Messagetexts für den nächsten Dialog verändert werden (der Standardwert für alle Dialoge wird mit *dlgSetStandardFont* gesetzt). Wenn man den Font nicht explizit setzt, wird der Standardfont des Formulars hergenommen. Die Prozedur muß vor dem Aufruf von *dlgAddButtons* ausgeführt werden, damit die Schriftart dieser Buttons davon beeinflusst wird.

procedure dlgAddControl (co: TWinControl);

Durch diese Prozedur bekommt der Dialog seine Flexibilität. Dem Dialog kann damit ein beliebiges WinControl (typischerweise ein Eingabefeld wie *TEdit*, oder ein *Tpanel*, auf dem mehrere Eingabefelder angeordnet sind) mitgegeben werden. Dieses Control gestaltet man im Designer, direkt im Formular der Unit irgendwo in einem freien Bereich, und im Object inspector setzt man *visible=false*, sodaß das Control im Fenster der Unit nicht angezeigt wird. Wenn der Dialog aufgerufen wird, wird dann dieses Control im Dialogfenster zwischen dem Dialogtext und den Buttons angezeigt.

Dokumentation - Unit fdDialog (Version 1.1)

procedure DlgSetDefaultCancel (Default, Cancel: TDlgResult);

Mit dieser Prozedur können der Default und der Cancel Button manuell gesetzt werden. Beim Drücken der Eingabetaste und der Escapetaste wird der Dialog geschlossen und der Wert der Default-Taste bzw. der Cancel-Taste zurückgegeben.
Wenn keine Buttons manuell ausgewählt wurden, sucht das Programm nach „geeigneten“ Standard Tasten für Default und Cancel:
Geeignete Tasten für Default sind dlgOK, dlgYes, dlgRetry, dlgOpen, dlgSave.
Geeignete Tasten für Cancel sind dlgCancel, dlgAbort, dlgNo, dlgClose
Wenn Sie keinen keinen Cancel-Button wünschen, übergeben Sie als Parameter den Wert dlgNone.

procedure dlgReset;

Alle für den nächsten Dialog bereits getätigten Einstellungen werden, ohne dass eine der Dialogfunktionen aufgerufen wird, wieder zurückgesetzt. dlgReset muß normalerweise nicht aufgerufen werden, denn es wird automatisch nach dem Aufruf jeder der Dialogfunktionen ausgeführt.

Standardwerte setzen: Im Gegensatz zu den bisher beschriebenen Prozeduren, die nur für den nächsten Dialogaufruf wirken, werden mit den im Anschluss beschriebenen Prozeduren Standardwerte für alle folgenden Dialoge permanent verändert.

procedure dlgSetStandardFont (Font: TFont);

setzt die Schriftart, die im Fragetext und in allen Buttons des Dialogs verwendet wird. Die Schrift von Komponenten, die in zusätzlichen Controls verwendet werden, werden von diesem Aufruf aber nicht beeinflusst.

procedure dlgSetStandardFontSize (size: integer);

setzt die Schriftgröße im Fragetext und in den Buttons, die Schriftart bleibt dabei unverändert. Die Schriftgröße in hinzugefügten Controls bleibt von diesem Prozeduraufruf unberührt.

procedure dlgSetStandardText(dlgButton: TdlgButton; Const Txt: string);
Mit dieser Prozedur können die Texte der Standardbuttons verändert werden. Sollen mehrere Standardbutton-Texte geändert werden, dann muss diese Prozedur für jeden zu ändernden Button aufgerufen werden.

procedure DlgSetStandardSpaces(Border, Top, Bottom, Hdistance, VDistance: integer);
Mit dieser Prozedur kann der rechte, linke, obere und untere Rand sowie die Mindestabstände zwischen den Buttons eingestellt werden. Ist der Dialog breiter, dann werden die Buttons unabhängig vom Mindestabstand gleichmäßig über die Breite des Dialogs verteilt.

Border	Linker und rechter Rand
Top	Oberer Rand
Bottom	Unterer Rand
HDistance	Mindestabstand horizontal zwischen den Buttons
VDistance	Abstand vertikal zwischen den Dialogelementen

procedure DlgSetStandardColors (info,warn,error,question: Tcolor);

Mit dieser Prozedur kann die Farbe des Rahmensfür die vier Dialogtypen festgelegt werden. Standardmäßig werden alle Rahmen in *clDkGray* eingefärbt, außer den Fehlermeldungen. Fehlermeldungen werden standardmäßig mit rotem Rahmen gezeichnet.

procedure DlgSetStandardIcons (ShowIcon: boolean);

Mit dieser Prozedur kann die Anzeige des Icons neben dem Nachrichten- bzw. Fragetext für alle folgenden Dialoge ein – und ausgeschaltet werden.

Dokumentation - Unit fdDialog (Version 1.1)

Hinweis:

Die Klasse Tdialog ist ein „Beinahe-Singleton“. Normalerweise wird nur eine Klasseninstanz erzeugt, sobald ein Dialog vorbereitet wird, und diese Instanz wird nach dem Aufruf der Fragefunktion bzw. beim Aufruf von d*lgReset* wieder gelöscht. Wenn allerdings während des Dialogs in der Prüffunktion *CanClose* oder in einer Ereignisbehandlungsroutine eines Elements des mittels *dlgSetControl* hinzugefügten Controls ein weiterer Dialog quasi rekursiv aufgerufen wird, dann wird eine weitere, zusätzliche Instanz des Dialogfensters erzeugt. Und das kann prinzipiell beliebig oft geschachtelt gemacht werden.

Ein direkter Zugriff auf eine Klasseninstanz sollte nie erfolgen. Dialoge sollten nur über die sechs Dialogfunktionen erzeugt werden.

Plattformen:

Die Klasse Tdialog ist für Windows entwickelt worden. Sie sollte auch auf allen anderen Plattformen laufen, allerdings miteinem Schönheitsfehler. Die Dialoge haben keine Titelleiste, die macht bei kurzen Zwischenfragen wenig Sinn. Derzeit ist es nur unter Windows möglich, den Dialog mit der Maus an eine andere Stelle des Bildschirms zu ziehen. Man klickt dazu den Dialog an einer beliebigen Stelle an, an der kein Control angezeigt wird, und kann ihn an die gewünschte Stelle ziehen. Die Funktion, mit der das bewerkstelligt wird, ist aber nicht plattformübergreifend verfügbar und existiert nur unter Windows.