Informatik App-Projekte

Delphi XE7 ist eines der mächtigsten Programmierpakete, die es weltweit gibt. Mit diesem Paket können Windows Fenster-Anwendungen, Server-Anwendungen, Datenbank-Anwendungen und App-Anwendungen für die Betriebssysteme Android von Google und iOS von Apple geschrieben werden. Diese App-Anwendungen können für Smartphones bestimmter Größe oder auch für Tablets optimiert werden.

Wir wollen im weiteren Verlauf der AG den Schwerpunkt auf Apps für Smartphones oder Tablets legen. Dabei ist es sogar möglich, dass wir in der App einen eigenen Browser erscheinen lassen, in welchem wir Formulare ausfüllen und sie an einen Server senden.

Eigene Smartphones und Tablets können über USB angeschlossen werden. Hat man sie in den Entwickler-Status überführt, können die in Arbeit befindlichen Programme dort getestet werden.

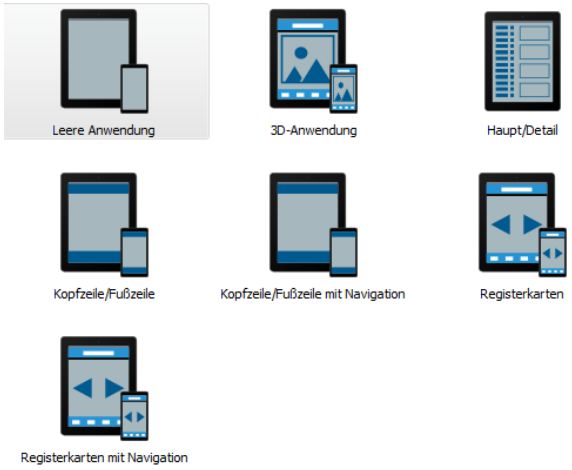
Nach dem Start von Delphi XE7 sieht der Bildschirm so aus, als befänden wir uns im Cockpit eines Düsenjets, so zahlreich sind die Informations- und Programmierfenster. Der Programmierbereich liegt in der Mitte und kann mehrere Karteikarten enthalten. Ebenso kann dort zwischen Programmtext (Code) und Formular (Design) umgeschaltet werden (auch durch F12).

Weitere Fenster werden im Laufe der ersten Anwendung „Benzin“ erklärt. Wichtig ist folgendes für den Beginn der Programmierarbeit festzuhalten:

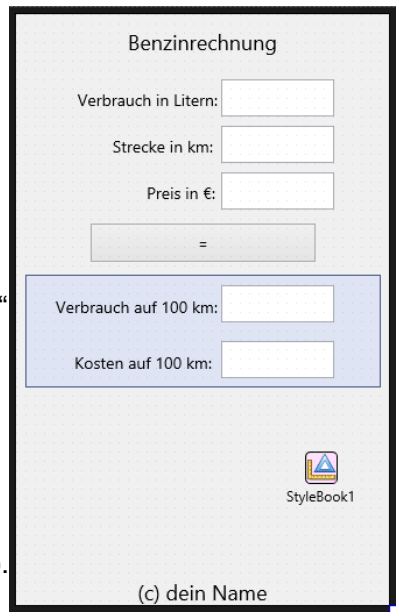
1. Delphi verwaltet eine App als Projekt. Projekte haben die Datei-Endung „.dproj“. Es werden immer nur neue Projekte erstellt, bestehende Projekte geöffnet oder in Arbeit befindliche Projekte gespeichert. Das Projekt bekommt einen sinnvollen Namen ohne eine Zahl am Ende: „Benzin“, „Bruchrechnung“, „GooglePos“ usw...
2. Gleichzeitig mit jedem neuen Projekt wird automatisch eine Pascal-Datei mit Namen „Unit1“ und Endung „.pas“ erstellt. Sie ist direkt verbunden mit dem Hauptfenster der App, dem Formular „Form1“. Zwischen ihnen kann man mit der Taste F12 umschalten. Dieses Hauptfenster der App stellt man in Höhe (hight) und Breite (width) so ein, wie man es für sein Smartphone oder Tablet haben möchte. Diese Pascal-Datei „Unit1“ wird schnellstmöglich unter dem Namen des Projekts mit Endung „1“ abgespeichert, also : „Benzin1“, „Bruchrechnung1“, „GooglePos1“ usw... Die Endung „.pas“ wird automatisch angehängt.
3. In dem Hauptfenster der App baut man nacheinander Beschriftungsfelder, Eingabefelder, Schalter usw... ein. Um die Wirkung zu prüfen, klickt man im mittleren Fenster unten auf „Design“ und kann dann oben unter Ansichten ein Smartphone oder Tablet nach Wahl auswählen. Sollen dann Objekte des Fensters gelöscht werden, muss wieder die Ansicht „Master“ gewählt werden.

Informatik Benzin-Verbrauch-App

A 1. Öffne ein neues Projekt in Delphi XE7. Datei-> neu-> geräteübergreifende Anwendung

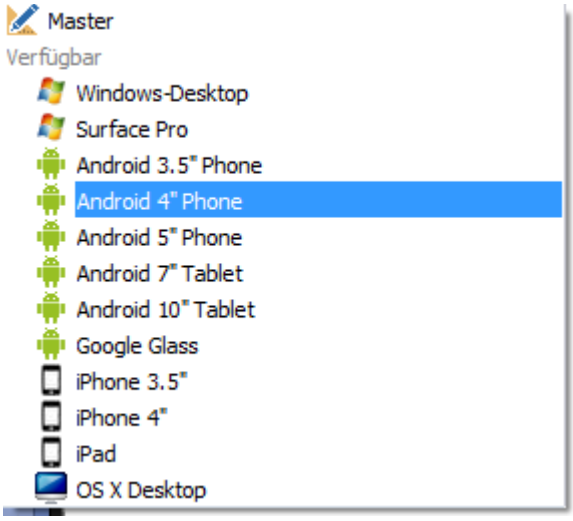


2. Wähle “Leere Anwendung“. Ein leeres Fenster erscheint. Links unten im Objektinspektor siehst du einige Eigeschaften den Fensters und kannst sie ändern. Wähle dort Height = 500 (567) und Width = 300 (384).
3. Nun wird zunächst – wie in jeder neuen App - oben eine Titelzeile und unten eine Copyright-Zeile eingefügt. Dazu legen wir zwei Objekte des Typs „tToolbar“ im Fenster ab. Diese Objekte finden wir rechts unten in der Tool-Palette, in der viele Objekte zur Verfügung stehen, die in das Fenster gezogen werden können. Nach dem Ablegen im Fenster haben sie haben bereits automatisch die Namen „Toolbar1“ und „Toolbar2“ bekommen. Wir stellen fest, dass beide „tToolbar“-Objekte am oberen Rand des Fensters erscheinen. Um das Objekt „Toolbar2“ an den unteren Rand des Fenster zu positionieren, klicken wir das Objekt an und können nun links unten im Objektinspektor viele Eigenschaften ändern. Setzen wir die Eigenschaft „align“ auf „bottom“, erscheint das Objekt sofort am unteren Rand des Fensters. In jedem dieser „tToolbar“-Objekte legen wir nun ein „tLabel“-Objekt ab. Hat man ein Label-Objekt versehentlich auf dem Formular abgelegt, nützt es nichts, es im Formular auf die Toolbar zu ziehen. Es ist notwendig, das Label-Objekt im Struktur-Fenster links oben in das Toolbar-Objekt zu ziehen. Bei beiden Label-Objekten setzen wir „align“ auf „client“ und unter „TextSettings“ die Werte für „HorzAlign“ und „VertAlign“ auf „center“. Die Beschriftung der „tLabel“-Objekte mit dem Text „Benzinrechnung“ und dem Copyright-Vermerk erfolgt unter „Caption“. Weiterhin entnehmen wir der Toolpalette rechts unter unter dem Menüpunkt „Standard“ die Objekte „tLabel“ (fünfmal), „tEdit“ (fünfmal) und „tButton“ (einmal) sowie „tStylebook“ (einmal). Wir gestalten damit das Fenster wie rechts angezeigt. Die Berechnungsergebnisse in der Mitte sind aus Gründen der Übersichtlichkeit in einem „tRectangle“-Objekt enthalten.



Ob diese Objekte bei deinem Lieblings-Smartphone gut aussehen, kannst du nun testen, indem du oben das Dreieck rechts neben „Master“ anklickst

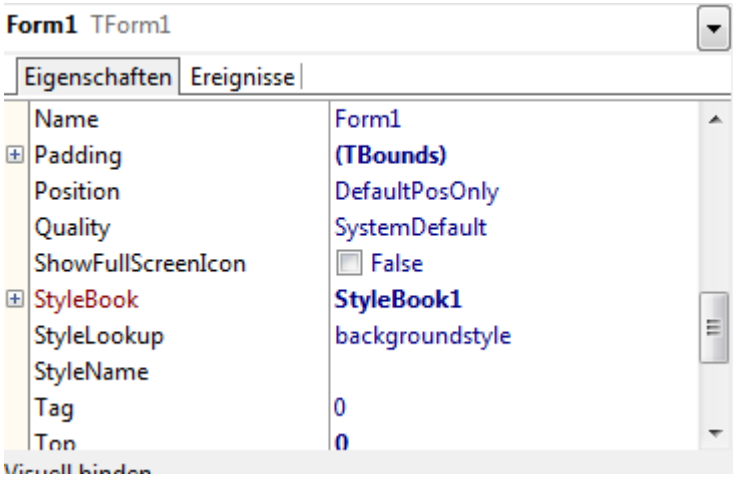
und unter



auswählst. Eine durchaus merkwürdige Neuerung gibt es in Delphi XE7: Zum Löschen von Objekten muss man wieder das Master-Formular aufrufen.

4. Nun kann auf das Symbol von Stylebook1 geklickt werden. Im mittleren Fenster ist nun das Formular veschwunden. Stattdessen erscheint oben

Original Styles können durch Drücken von „Laden“ aus dem Verzeichnis
`C:\Program Files (x86)\Embarcadero\Studio\15.0\Redist\styles` oder
`E:\Anwendg\DelphiXE7\90 Original Styles Fmx`
geladen werden. Dann unter
`E:\Anwendg\DelphiXE7\20 Eigene Styles Fmx`
abspeichern. Dazu „Speichern“ drücken und das Verzeichnis wählen. Nach dem Bearbeiten dann „Übernehmen und schließen“ drücken. Falls nun der Style nicht erscheint, liegt das daran, dass im Formular das Stylebook1 noch nicht angemeldet wurde. Dazu auf das Formular klicken und links unten im Objektinspektor die Eigenschaft „Stylebook“ rechts auf „Stylebook1“ setzen.



Dann ist der neue Style zu sehen und kann durch erneutes Anklicken des Symbols „Stylebook1“ verändert werden.

5. Will man ein Objekt (z.B. hier `Rectangle1`) unter Beibehaltung der Breite in die Mitte der Zeile

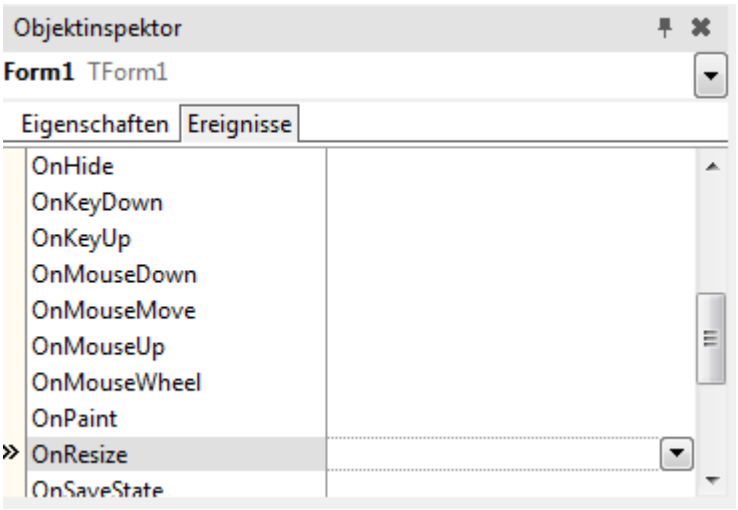
bringen, also zentrieren, so ist folgende Anweisung möglich:

```
Rectangle1.Position.X := (ClientWidth – Rectangle1.Width) / 2;
```

Will man ein Objekt so verbreitern, dass es zentriert ist, wählt man die Anweisung:

```
Rectangle1.Width:= ClientWidth- 2*Rectangle1.Position.X;
```

Da schon beim Start der App das Objekt zentriert positioniert werden soll, muss die Anweisung so früh wie möglich eingebaut werden. Dazu auf das Formular klicken und dann links unten im Objektinspektor die rechte Karteikarte „Ereignisse“ aufschlagen. Dann in der Zeile „OnCreate“ oder noch besser in der Zeile „OnShow“ weiter unten in das rechte Feld doppelklicken. In die sich öffnende Programmzeile ist dann der genannte Text einzutragen. Wenn auch beim Querstellen des Smartphones die Zentrierung beibehalten werden soll, schreibt man die Anweisung auch noch in die OnResize-Methode des Formulars.



6. Nun kommt der eigentliche Programmtext der App: Die Berechnung des Benzinverbrauchs und der Kosten auf 100 km. Dazu machst du einen Doppelklick auf den Schalter mit dem Gleichheitszeichen. Es öffnet sich eine leere Prozedur: die OnClick-Methode des Schalters. Dort werden über dem „begin“ die Variablen eingetragen:

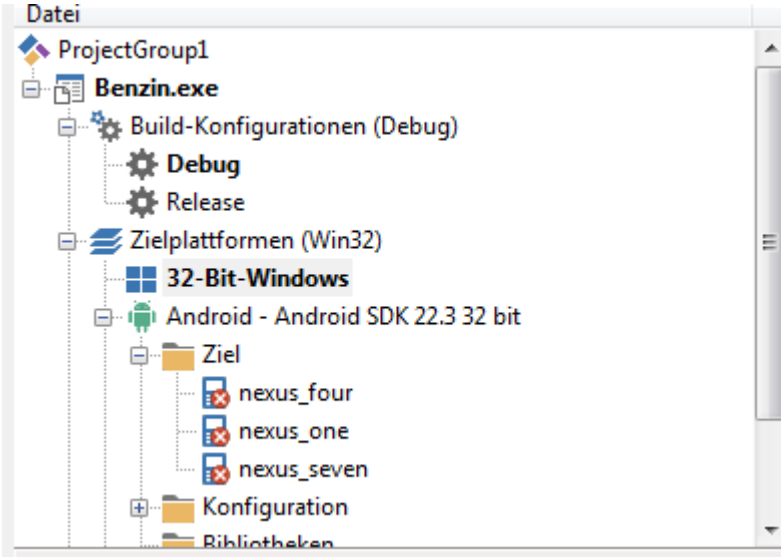
```
var Verbrauch,Strecke,Preis,Verbrauch100,Preis100:real;
```

Unter dem „begin“ werden nun folgende Zeilen eingetragen

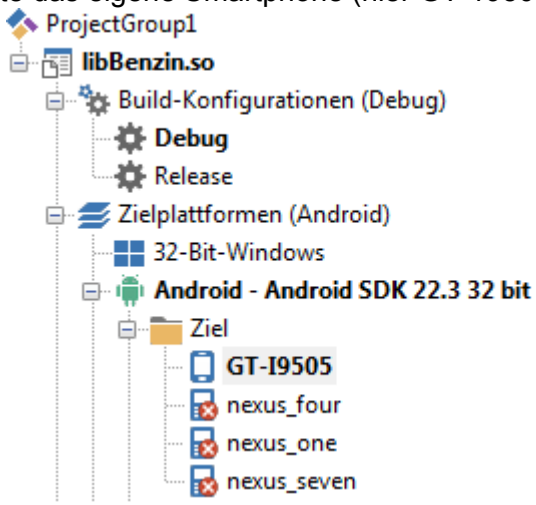
```
begin
   try
      Verbrauch:= StrToFloat(Edit1.Text);
      Strecke:= StrToFloat(Edit2.Text);
      Preis:= StrToFloat(Edit3.Text);
      if Strecke=0 then abort;
      Verbrauch100:= Verbrauch/Strecke*100;
      Preis100:= Preis/Strecke*100;
      Edit4.Text:=FloatToStrF(Verbrauch100,ffFixed,6,2)+' l';
      Edit5.Text:=FloatToStrF(Preis100,ffFixed,6,2)+' €';
   except
      Edit4.Text:='Fehler';
      Edit5.Text:='Fehler';
   end;
   Edit4.SetFocus //lässt die Tastatur des Handys wieder verschwinden
end;
```

Damit nach dem Drücken der Berechnungstaste beim Ablauf der App die Eingabetastatur auf deinem Handy verschwindet, musst du noch für die Editfelder Edit4 und Edit5, die nur der Ausgabe dienen sollen, die Eigenschaft `readonly` auf `true` setzen.

7. Jetzt wird das Programm unter 32-Bit-Windows getestet. Dazu im Fenster rechts oben die passende Einstellung wählen und in der obersten Zeile den grünen Startknopf drücken.



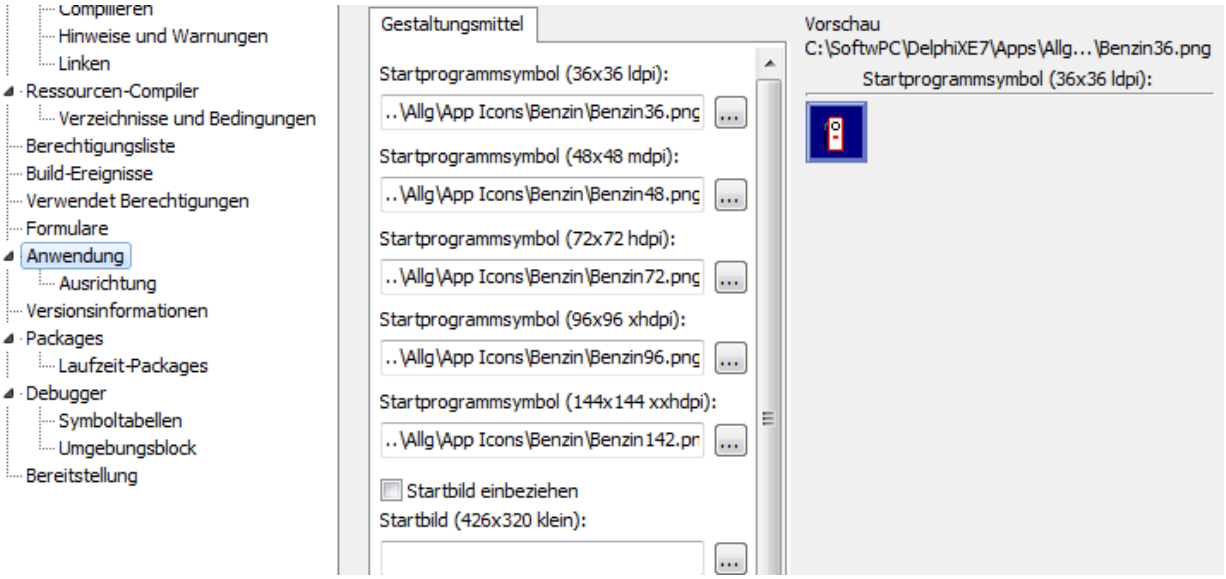
8. Dann wird unter Android getestet. Dazu das eigene Smartphone über USB-Stick anschließen und – falls noch nicht geschehen – in den Entwicklermodus versetzen. (Einstellungen->Optionen->Geräteinformationen->Build-Nummer sechsmal antippen, damit die Entwickleroptionen zugreifbar werden, dann unter Einstellungen->Optionen->Entwickleroptionen->USB-Debugging Option einschalten und auch auch „Wach bleiben“ einschalten). Nach dem Anschließen das Smartphone entsperren und „Computer vertrauen“ bestätigen. Klickt man nun im oben angezeigten Fenster mit der rechten Maustaste auf „Ziel“ (aktualisieren), sollte das eigene Smartphone (hier GT-19505) angezeigt werden:



Dies ist deshalb wichtig, da die angezeigten Emulatoren (nexus_four usw...) auf den Schulcomputern nicht zufriedenstellend laufen. Um keine unnötigen Compilierfehler zu bekommen, sind folgende Einstellungen nötig: Geht man über Projekte -> Optionen, erhält man unter Anwendung eine ganze Liste von Programmsymbolen. Sie können später mit dem Bildeditor in fünf Versionen 36x36, 48x48, 72x72, 96x96 und 144x144 Bildpunkte gezeichnet, dann als png-Datei abgespeichert und in Delphi XE7 eingelesen werden. Zunächst sind Default-Symbole vorhanden. Falls ein Symbol fehlt, erscheint eine Fehlermeldung des

Compilers. Der Haken vor „Startbild einbeziehen“ ist zu entfernen, andernfalls ergibt sich ebenfalls eine Fehlermeldung des Compilers.

9. Im Verzeichnis der Anwendung ist nun z.B. unter



`E:\Anwendg\DelphiXE7\Klasse9\BenzinApp\Android\Debug\Benzin\bin\` die Datei `Benzin.apk` (android package file) zu finden. Sie kann nun auf einer Internetseite zum Download angeboten werden.

10. Ergänzungen: Sollen die drei getätigten Eingaben auch nach dem Ausschalten und anschließendem Einschalten der App wieder zu sehen sein, müssen sie beim Schließen des Programms abgespeichert und beim Öffnen des Programms gelesen werden. Zum Speichern werden im OnClose-Ereignis des Formulars die Zeilen zunächst in ein nicht sichtbares Memofeld übertragen und dann abgespeichert. Dazu muss ein Memo-Objekt ins Formular gezogen werden und die Eigenschaft `visible` auf `false` gesetzt werden. Außerdem muss die Unit `System.IOUtils` unter `uses` hinzugefügt werden, in der sich die Prozedur `TPath.GetDocumentsPath` befindet.

```
Memo1.Text:='';
Memo1.Lines.Add(EditVerbrauch.Text);
Memo1.Lines.Add(EditStrecke.Text);
Memo1.Lines.Add(EditPreis.Text);
Memo1.Lines.SaveToFile(System.IOUtils.TPath.GetDocumentsPath+'/Benzindaten');
```

Zum Einlesen wird im OnShow-Ereignis des Formulars zunächst das Memofeld geladen und dann der Text des Memofeldes in die Zeilen in die Editfelder verteilt:

```
if fileExists(System.IOUtils.TPath.GetDocumentsPath+'/Benzindaten') then begin
  Memo1.Text:= '';
  Memo1.Lines.LoadFromfile(System.IOUtils.TPath.GetDocumentsPath+'/Benzindaten');
  EditVerbrauch.Text:= Memo1.Lines[0];
  EditStrecke.Text:= Memo1.Lines[1];
  EditPreis.Text:= Memo1.Lines[2];
end;
```

11. Ergänzungen: Soll bei der Eingabe nach Drücken der Returntaste in das nächste Eingabefeld gesprungen werden wie beim Drücken der Tabulatortaste, so ist in der OnKeyDownMethode des ersten Editfeldes einzugeben:

```
procedure TForm1.EditVerbrauchKeyDown(Sender: TObject; var Key: Word;
  var KeyChar: Char; Shift: TShiftState);
begin
```

```
    if Key = vkReturn then begin
       Key := vkTab;
       KeyDown(Key, KeyChar, Shift);
    end;
end;
```

12. Ergänzungen: Soll bei der Eingabe lediglich Ziffern und Komma möglich sein, so kann im OnShow-Ereignis durch folgende Anweisung ein Eingabefilter eingebaut werden:

```
Edit1.FilterChar := '0123456789,';
```