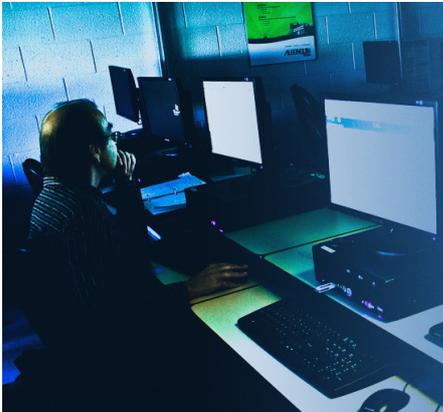


EMBARCADERO AKADEMIJA



# Embarcadero Akademija 2012: FireMonkey ogrodje za izdelavo vizualnih uporabniških vmesnikov

**Primož Gabrijelčič**

<http://thedelphigeek.com>





## Kazalo

Uvod .....	2
Strojne in programske zahteve .....	2
Primerjava z VCL.....	3
Izdelek ali razvojna različica?.....	4
Prehod .....	5
Nov projekt.....	6
Gradniki .....	8
Slogi .....	10
Animacije .....	12
Učinki .....	13
Seznami .....	14
LiveBindings.....	16
Luč, kamera, akcija! .....	17
Viri .....	19
Programi .....	19
3D .....	20
Video .....	20
Komponente.....	20

Vsi programi, omenjeni v tem dokumentu, so na voljo na naslovu  
<http://17slon.com/EA/EA-FireMonkey.zip>.

## Uvod

Programsko okolje Delphi je bilo narejeno za operacijski sistem Windows, a v svoji dolgi zgodovini je Delphi že koketiral z drugimi operacijskimi sistemi. Zagotovo ste že slišali za Kylix, različico, ki je tekla na nekaterih Linuxih, a ni bila nikoli pretirano uspešna. Eden pomembnejših razlogov za neuspeh je bila težavnost prilagajanju vedno novim različicam Linuxa, prav tako pomemben razlog pa je bila arhitektura vizualnih gradnikov v Kylixu, ki so poskušali kar natančneje oponašati platformo VCL. Ker pa se Windows in Linux bistveno razlikujeta v programskem vmesniku, je bilo to oponašanje le delno uspešno, programerji pa so se srečevali z vedno novimi težavami.

V novejših časih je Embarcadero pozornost z Linuxa preusmeril na Applov OS X in mobilne naprave z operacijskimi sistemi iOS. Problem prilagajanja hitro spreminjajočim se platformam ostaja in le leta bodo pokazala, kako uspešno so ga rešili tokrat, oponašanje platforme VCL pa so rešili na popolnoma drugačen način. Na ne-Windows operacijskih sistemih ne moremo uporabiti platforme VCL, temveč moramo programe napisati na novo z uporabo platforme FireMonkey.



FireMonkey ni popolnoma nov produkt. Embarcadero je kupil podjetje KSDev, ki je izdelovalo knjižnici za večplatformsko programiranje grafičnih aplikacij VGScene in DXScene, to tehnologijo pa so nato tesno integrirali z razvojnim okoljem RAD Studio. Rezultat je platforma, s katero lahko izdelujemo programe za Windows (v 32- in 64-bitni različici), OS X ter celo za iOS. Pri prevajanju slednjih za sedaj še pomaga odprtokodni prevajalnik FreePascal, a že letos naj bi v Delphiju XE3 dobili domorodni prevajalnik za procesorje Arm, tako da bomo lahko brez zunanje pomoči izdelovali programe za iOS in tudi za operacijski sistem Android. Skratka – s FireMonkeyem se Delphiju odpirajo vrata na povsem nove naprave!

## Strojne in programske zahteve

FireMonkey je zasnovan na popolnoma drugačen način kakor VCL. Namesto da bi izrisovanje uporabniškega vmesnika prepustil operacijskemu sistemu, vse naredi platforma sama. Zaradi tega je lahko bistveno bolj fleksibilna, programski vmesnik pa se ne spreminja glede na ciljni operacijski sistem. Tak način izrisa omogoča poleg običajnih, »ploščatih« uporabniških vmesnikov tudi izdelavo »tridimenzionalnih« aplikacij, kjer se ves izris dogaja v tridimenzionalnem prostoru.

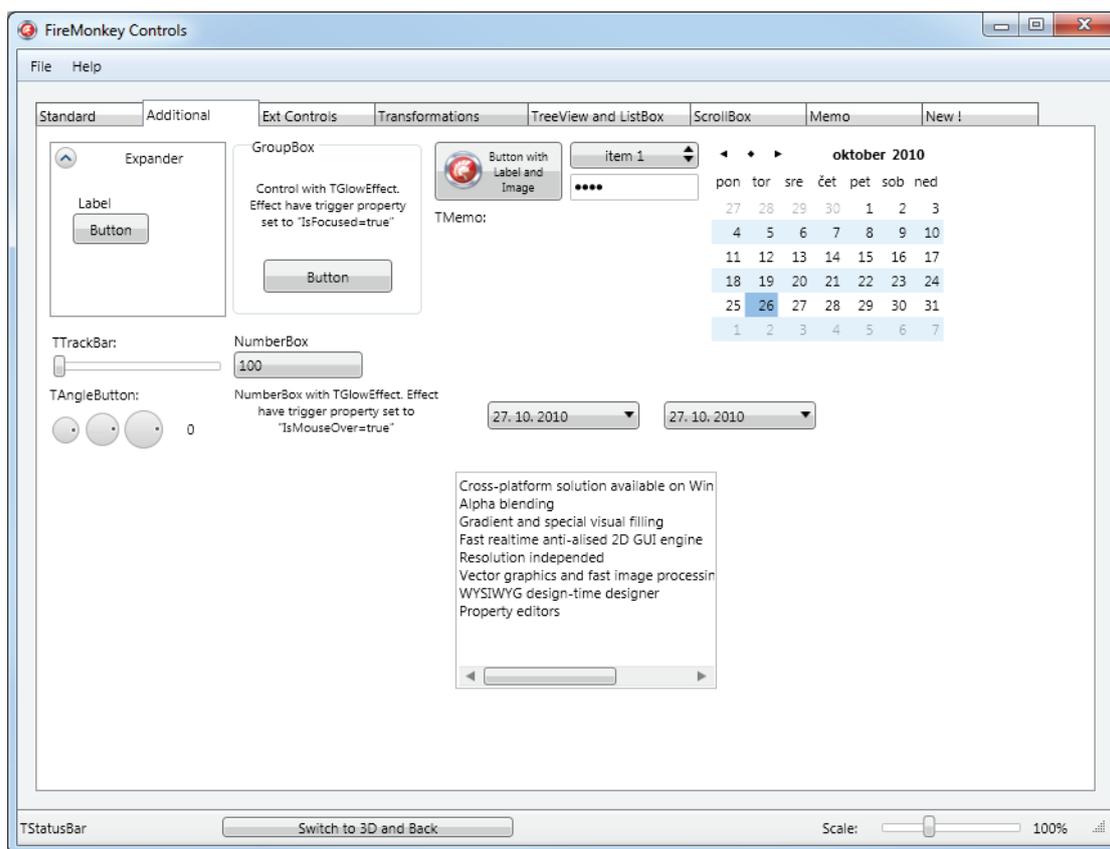
Zaradi takega načina dela je FireMonkey strojno precej bolj zahteven kakor VCL. Na srečo so tudi računalniki napredovali, tako da tudi najcenejši zmore vse, kar od njega zahteva FireMonkey. Na operacijskem sistemu Windows potrebuje vsaj različico XP ter grafično kartico, ki podpira DirectX 9.0, z operacijskim sistemom OS X lahko uporabimo poljuben računalnik, različica OS X pa mora biti 10.6 (Snow Leopard) ali 10.7 (Lion), naprave z operacijskim sistemom iOS pa morajo imeti nameščeno vsaj operacijski sistem različice 4.2.

Za izris dvodimenzionalnih grafičnih vmesnikov je na Windows uporabljena tehnologija Direct2D, na OS X in iOS pa Quartz. Za izris tridimenzionalnih vmesnikov na Windows poskrbi Direct3D, na OS X in iOS pa OpenGL. V vseh primerih je za hitrejše risanje grafičnih učinkov vprežena tudi grafična kartica (oziroma njeni procesorji GPU), kar sicer izredno pohitri izris, lahko pa prepreči delovanje, če program izvajamo v virtualiziranem okolju. Ta trenutek za poganjanje virtualnih računalnikov, v katerih bi radi razvijali in testirali aplikacije FireMonkey, priporočamo VMWare, ki ima dobro rešeno virtualizacijo GPU. V okoljih Hyper-V, VirtualBox in Citrix oponašanje grafičnih procesorjev GPU zaenkrat ne deluje, zato bodo v njih najverjetneje delovali le dvodimenzionalni programi in še pri teh utegne biti izris pomanjkljiv.

## Primerjava z VCL

Podobnosti med platformama VCL in FireMonkey je ogromno, a tudi razlik je veliko. Izkušen Delphi programer se bo zato v FireMonkeyu kar hitro znašel, obenem pa bo (vsaj na začetku) porabil veliko časa za iskanje alternativ. Za lažji začetek smo opisali nekaj nezdržljivosti, ki programerjem pogosto povzročajo težave.

Nekateri gradniki manjkajo (TAction, TImageList, TImageView, TRichEdit, spletni brskalnik, prikaz videa, avdia), dodano pa je veliko drugih. Ogledate si jih lahko v Delphiju priloženem primeru ControlsDemo, našteji pa smo jih tudi v tem dokumentu, v razdelku Gradniki.



Gradniki nimajo lastnosti Anchors. Za razporejanje boste morali uporabiti Align, Padding, Margins ter gnezdenje z gradniki TLayout, ki nadomešča TPanel.

Višina in širina gradnikov sta po novem predstavljeni kot realni števili. Isto velja za položaj gradnikov na obrazcu. Položaj določamo z lastnostjo Position, ki ima na dvodimenzionalnih gradnikih polji X in

Y, na tridimenzionalnih pa še polje Z. Namesto Left moramo torej uporabiti Position.X, namesto Top pa Position.Y.

Lastnost Caption je v večini primerov nadomeščena z lastnostjo Text.

»Izbranost« gradnika TCheckbox označuje lastnost IsChecked.

Velikosti pisav določamo v enotah DIP (dots per inch) in ne v enotah point.

Gradnikov, ki znajo prikazovati podatke iz baz (»data-aware«), ni več. Namesto njih moramo uporabiti sistem LiveBindings.

Gradniki (razen TForm) nimajo vsak svoje predstavitve na nivoju operacijskega sistema in jim zato ne moremo več pošiljati sporočil.

Razreda Screen ni, ker je vezan na operacijski sistem Windows. Namesto njega uporabimo metode razreda Platform. Kazalko na primer spremenimo s klicem Platform.SetCursor, širino zaslona preberemo s Platform.GetScreenSize.X in podobno.

Posebnost FireMonkeya je tudi hierarhija gradnikov. Lastništvo (lastnost Owner) deluje enako kakor v VCL-u, starševstvo (lastnost Parent) pa je izvedeno popolnoma drugače. Po novem je lahko vsak gradnik otrok poljubnega drugega gradnika, poleg tega pa ni potrebno, da je izrisan znotraj njegovih meja. VCL-ov TLabeledEdit (vnosno polje z oznako) lahko v FireMonkeyu izdelamo tudi tako, da TLabel spustimo v TEdit in ga premaknemo na pravo mesto. Dobimo celoto, ki jo lahko po obrazcu premikamo kot da bi šlo za samostojen gradnik.

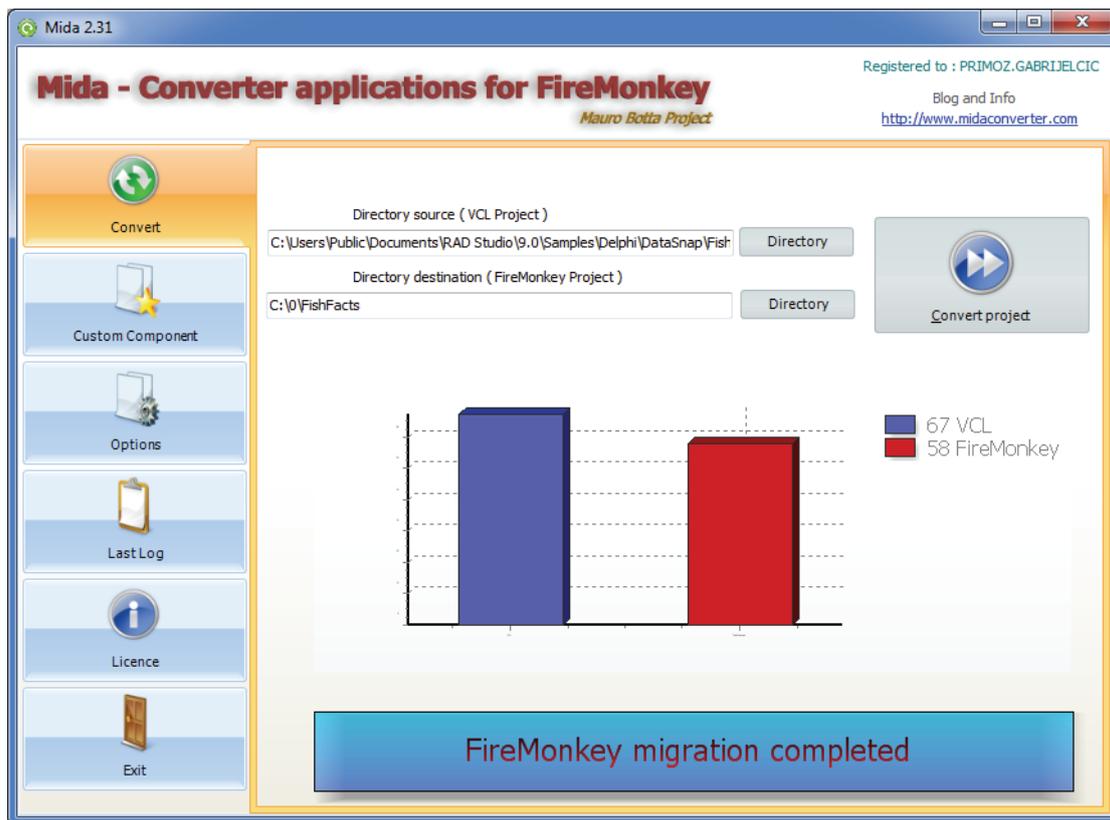


## Izdelek ali razvojna različica?

Na nekaterih področjih FireMonkey sicer deluje kot izdelek v razvojni (beta) fazi – manjka podpora tiskanju, ne obstaja rešitev za izdelavo poročil – a kar je narejeno, je zelo stabilno. Če vas omejitve ne motijo, je FireMonkey povsem primeren za izdelavo poslovnih aplikacij. Najpomembnejše – povezljivost z bazami podatkov s tehnologijo DataSnap – deluje v vseh podprtih operacijskih sistemih.

## Prehod

Vsak začetek je težak in tudi pri prehodu z VCL na FireMonkey nam ne bi škodil prijazen pomočnik, ki bi pretvoril obstoječi projekt. Zastonjske rešitve sicer (še) ni, obstaja pa plačljiv (a ne predrag) pretvornik Mida ([www.midaconverter.com](http://www.midaconverter.com)), ki opravi ogromno dela. Poleg zamenjave gradnikov VCL z gradniki FireMonkey poskrbi tudi za pretvorbo podatkovnih gradnikov v povezave LiveBindings ter pretvori nezdružljive dele izvorne kode (denimo zamenja `Screen.Width` v `Platform.GetScreenSize.X`).

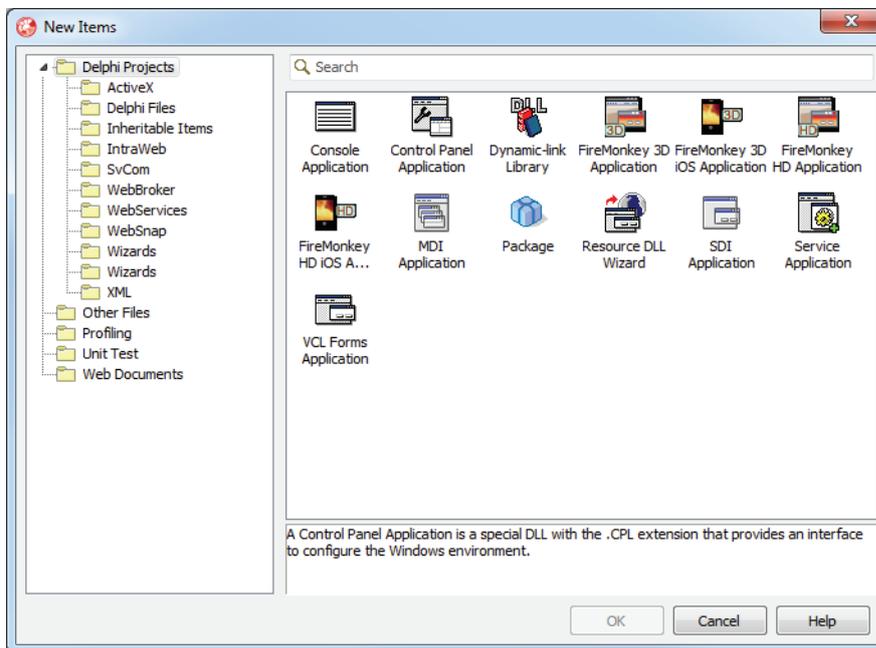


Za vse, ki so uporabljali knjižnico VGScene, poskrbi zastonjski pretvornik MonkeyGroomer ([pascalcoder.blogspot.com/search/label/MonkeyGroomer](http://pascalcoder.blogspot.com/search/label/MonkeyGroomer)).

Če bi želeli znotraj enega projekta kombinirati VCL in FireMonkey, vam priporočam ogled produkta Hydra ([www.remobjects.com/hydra/](http://www.remobjects.com/hydra/)), ki omogoča sobivanje različnih platform v isti aplikaciji. Poleg VCL in FireMonkeya podpira tudi WPF, WinForms in Silverlight.

## Nov projekt

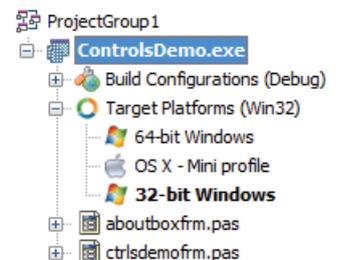
Delphi XE2 prinaša nekaj novih tipov projektov.



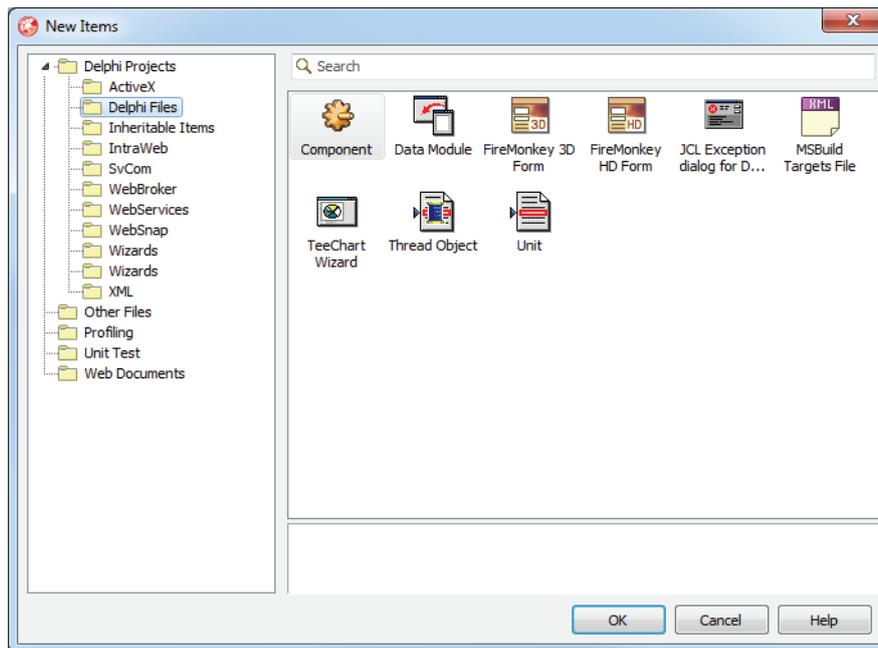
Z izbiro projekta »FireMonkey HD Application« izdelamo dvodimenzionalni program, ki lahko teče na Windows ali OS X. Podprte platforme določimo v oknu Project Manager v veji Target Platforms.

Podobno velja za »FireMonkey 3D Application«. Z izbiro te možnosti izdelamo tridimenzionalni program za Windows ali OS X.

Programi za iOS potrebujejo drugačno ogrodje, zato moramo za njih ustvariti nov projekt, ki pa lahko uporablja iste obrazce kot projekt za Windows/OS X. Uporabimo »FireMonkey 3D iOS Application« ali »FireMonkey HD iOS Application«.



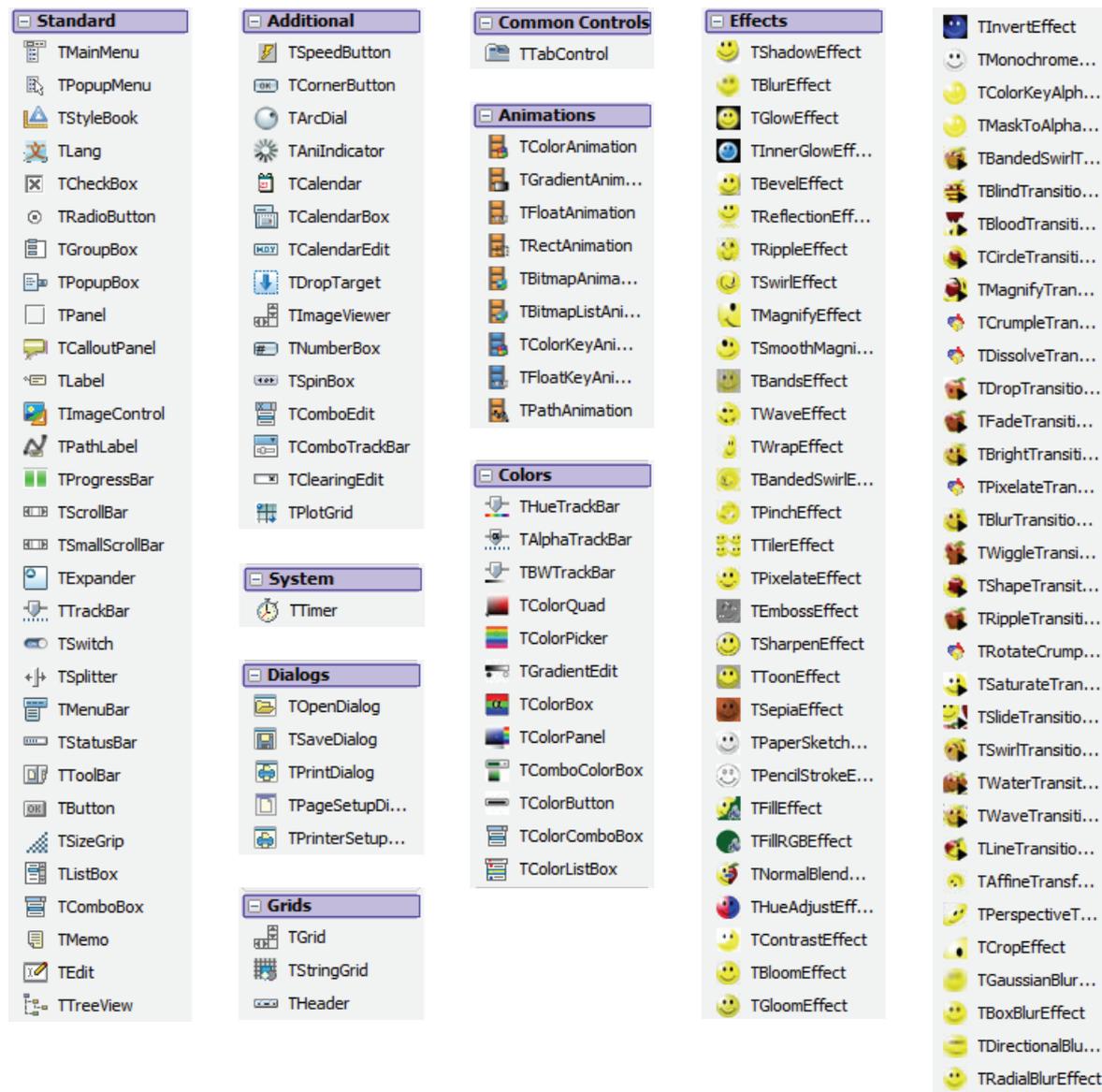
Na voljo imamo dve vrsti obrazcev – »FireMonkey HD Form« in »FireMonkey 3D Form«.

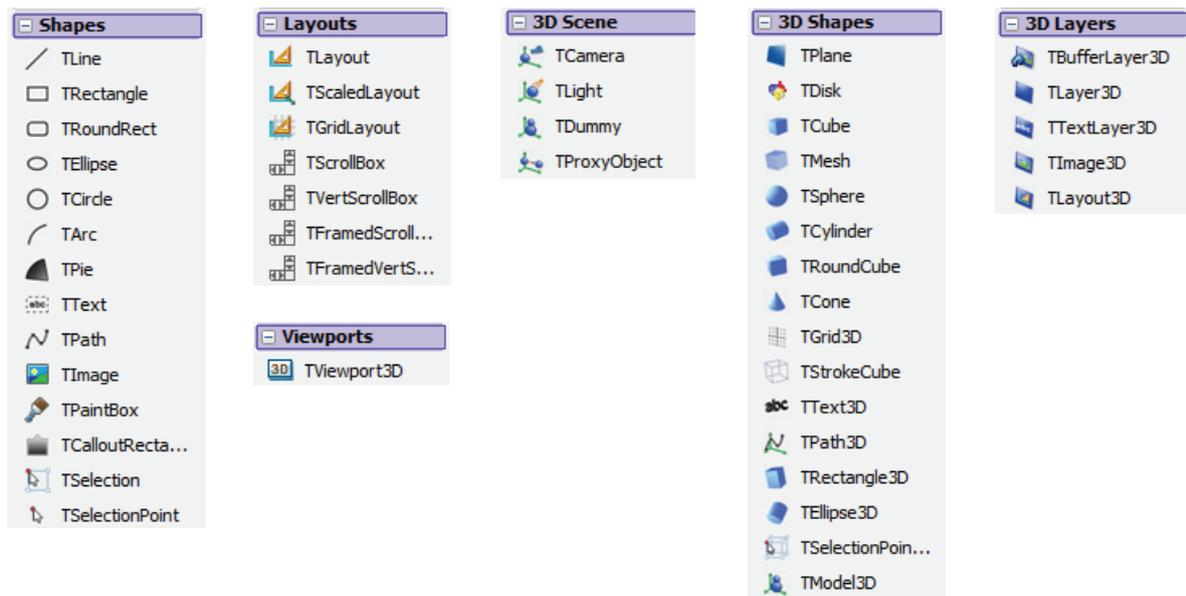


Omeniti moramo še, da razdelitev na projekte HD in 3D ni tako stroga, kot je videti na prvi pogled. Znotraj projekta lahko mešamo obrazce različnih vrst, poleg tega pa lahko na obrazec HD z uporabo gradnika TViewport3D vključimo tridimenzionalno vsebino, na obrazec 3D pa lahko z uporabo gradnika TLayer3D vključimo dvodimenzionalno vsebino.

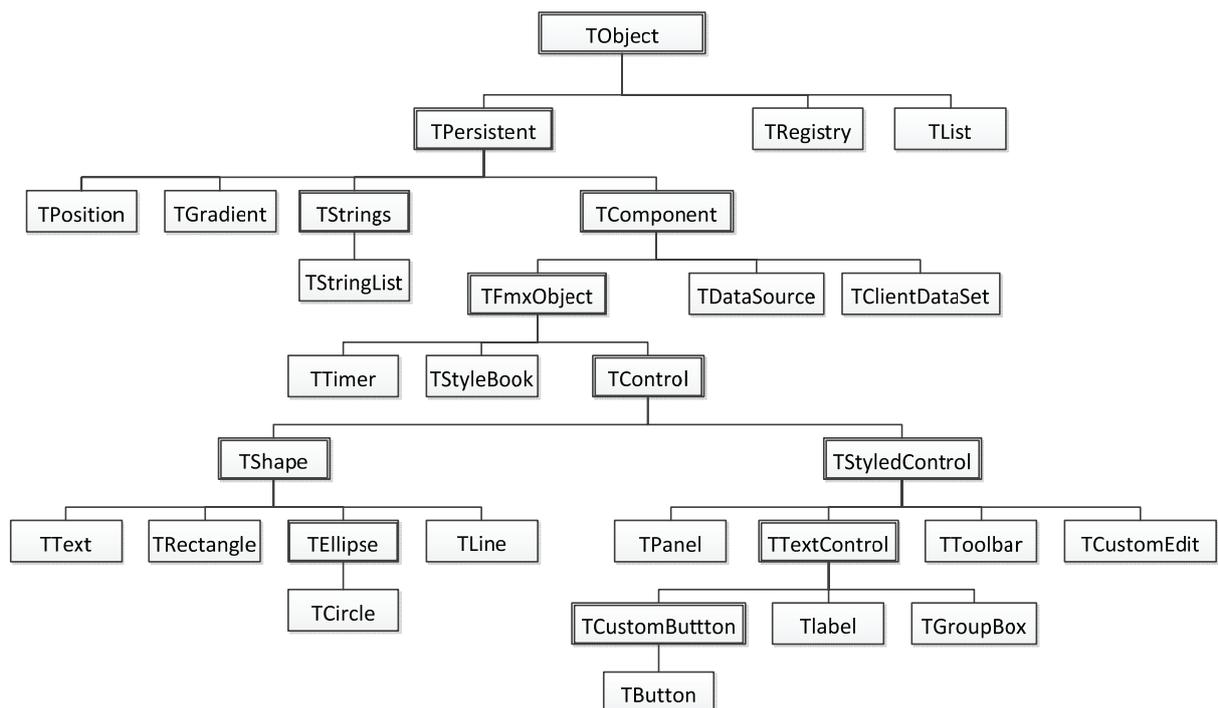
## Gradniki

Gradniki grafičnega vmesnika FireMonkey so razdeljeni v nekaj skupin: Standard, Additional, System, Dialogs, Grids, Common Controls, Animations, Colors, Effects, Viewports, Shapes, Layouts, 3D Scene, 3D Shapes in 3D Layers. Slednji trije vsebujejo gradnike tridimenzionalnih obrazcev.





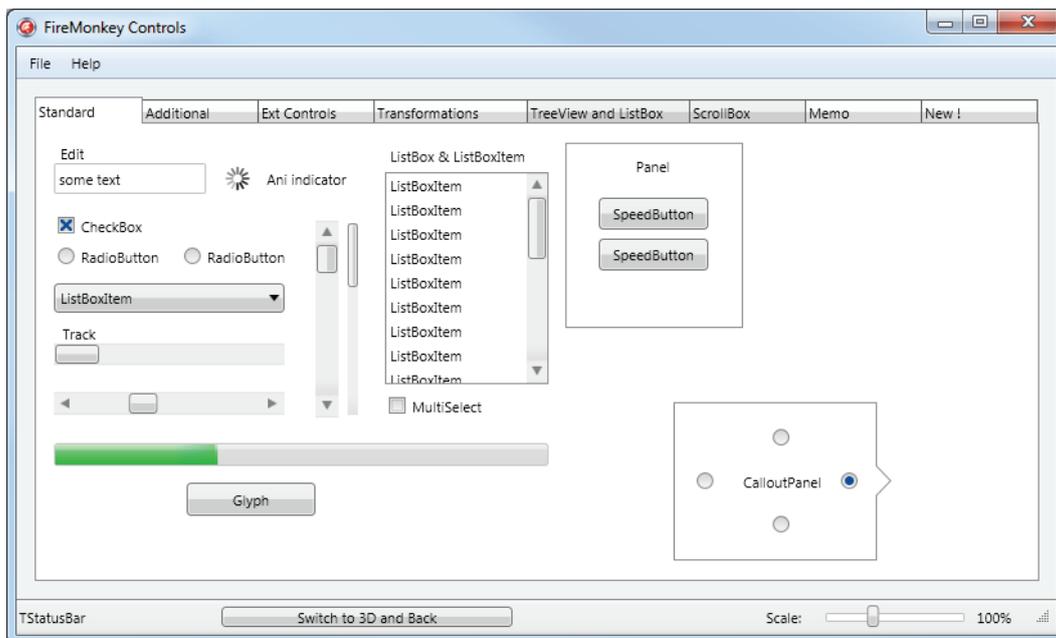
Nekateri deli platforme FireMonkey izhajajo neposredno iz razreda TPersistent (TGradient, TPosition), večinoma pa izhajajo iz razreda TFmxObject, ki je neposredni potomec razred TComponent.



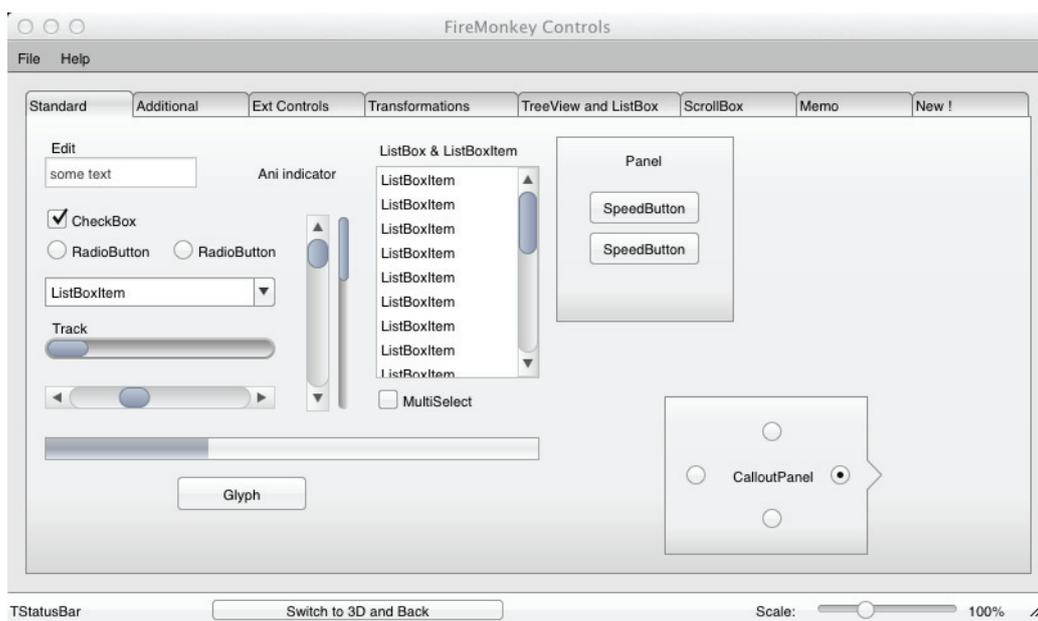
Nekaj gradnikov je izpeljanih neposredno iz razreda TFmxObject, večina pa je razdeljena v dve skupini – osnovne oblike (TShape) in gradniki s podporo slogov (TStyledControl). Le slednji podpirajo oblikovanje s slogi.

## Slogi

Bistven del platforme FireMonkey predstavljajo *slogi*. Slog (style) je skupek oblikovnih navodil, ki določa, kako bo gradnik izrisan na zaslonu. Lahko bi rekli, da slog v FireMonkeyu ustreza slogom CSS v internetnih dokumentih. S slogi so dosegli, da so programi FireMonkey videti kot domorodne aplikacije Windows in OS X. Slog lahko vpliva na obliko, barve, animacije, učinke in druge parametre prikaza.

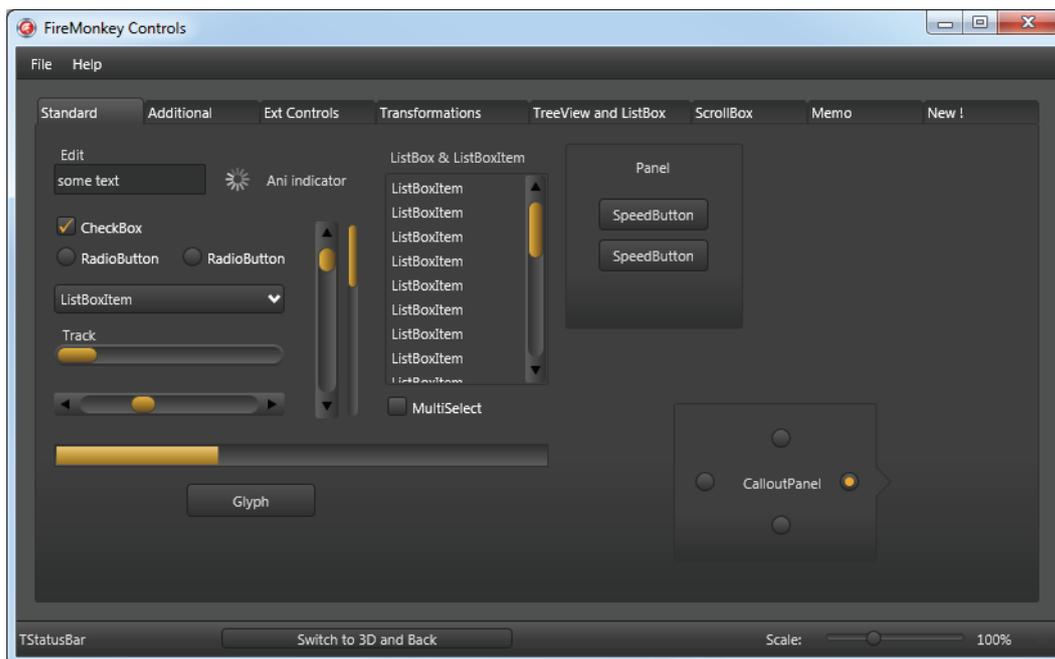


Videz programa v okolju Windows ...



... in videz programa v okolju OS X.

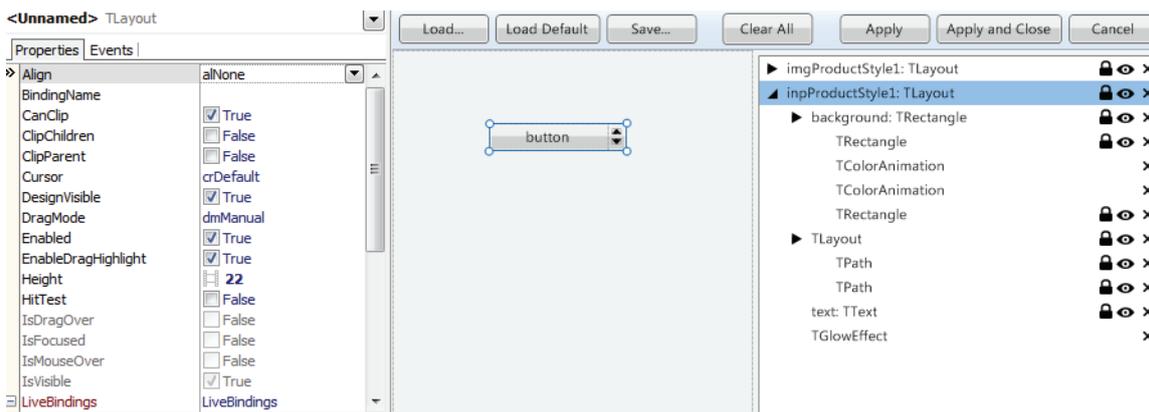
S slogi lahko tudi povsem spremenimo videz programa, tako da ne bo več podoben domorodnim programom. V katerih primerih je to smiselno uporabiti pa ostaja vprašanje dobrega okusa.



Videz programa z naloženim slogom *Dark*.

Sloge lahko naložimo iz zunanjih datotek ali pa jih vključimo v program kot sredstva (resources).

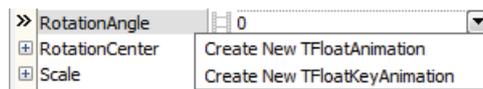
Za spreminjanje sloga posameznega gradnika le-tega kliknete z desno tipko in z menuja izberete Edit Custom Style. Nato nastavite lastnosti oblike v oknu Object Inspector in zaprete urejevalnik (Apply and Close). Privzeti slog (velja za vse gradnike istega razreda) spremenite z izbiro Edit Default Style z istega menuja.



## Animacije

Animacije v platformi FireMonkey so operacije, ki skozi čas spreminjajo lastnosti gradnika. Animiramo lahko vse vizualne lastnosti, od položaja, velikosti, rotacije do barve in prozornosti.

Lastnosti, ki jih najpogosteje animiramo, so označene z ikono filmskega traku v Object Inspectorju. Animacijo take lastnosti izdelamo enostavno tako, da kliknemo puščico desno od trenutne vrednosti in izberemo Create New TFloatAnimation.



Delphi bo izdelal nov gradnik tipa TFloatAnimation, ki mu nastavimo lastnosti animacije. Določimo trajanje (Delay), animacijo lahko ponavljamo (Loop), določimo začetne in končne vrednosti (Start, Stop) ter prožilec (Trigger). Animacijo lahko sprožimo tudi v kodi s klicem metode Start.

Vsaka animacija spreminja le vrednost ene lastnosti. V primeru na desni je bila to lastnost RotationAngle, ki določa nagib gradnika. Hkrati lahko poženemo več animacij, ki spreminjajo različne lastnosti.

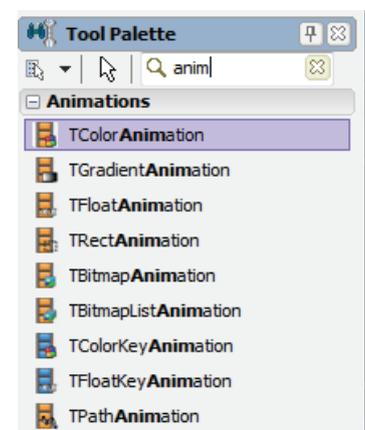
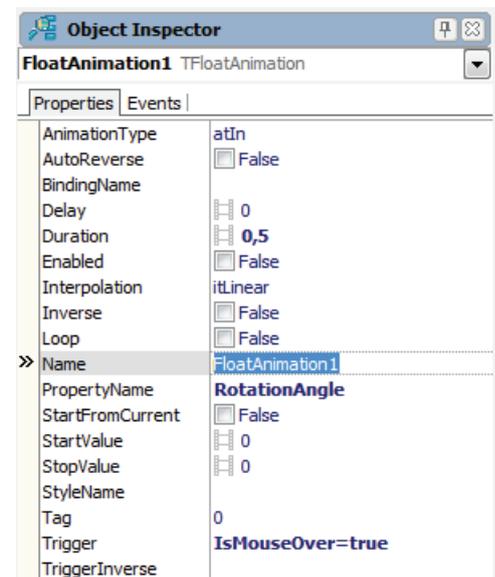
Animacijo lahko naredimo tudi programsko, tako da na objektu, ki ga želimo animirati, pokličemo metodo AnimateFloat.

Spodnji primer bo v 0,5 sekunde spremenil vodoravni položaj gumba btnAdd iz trenutnega položaja na odmik 120, hkrati pa bo spremenil tudi navpični položaj iz trenutnega na odmik 99.

```
btnAdd.AnimateFloat('Position.X', 120, 0,5);
btnAdd.AnimateFloat('Position.Y', 99, 0,5);
```

Animacijo lahko izdelamo tudi z izbiro animacijskega gradnika s palete. Animacijski gradnik odložimo znotraj objekta, ki ga želimo animirati, tako da postane njegov otrok.

Na voljo imamo animacijske gradnike za animiranje barv, prelivov, poljubnih realnih vrednosti, pravokotnikov in bitnih slik. Na voljo sta tudi dve različici, ki jemljeta vrednosti iz seznama, namesto da bi jih izračunavali z interpolacijo (TColorKeyAnimation in TFloatKeyAnimation) ter animacija, ki premika gradnik po določeni poti (TPathAnimation).

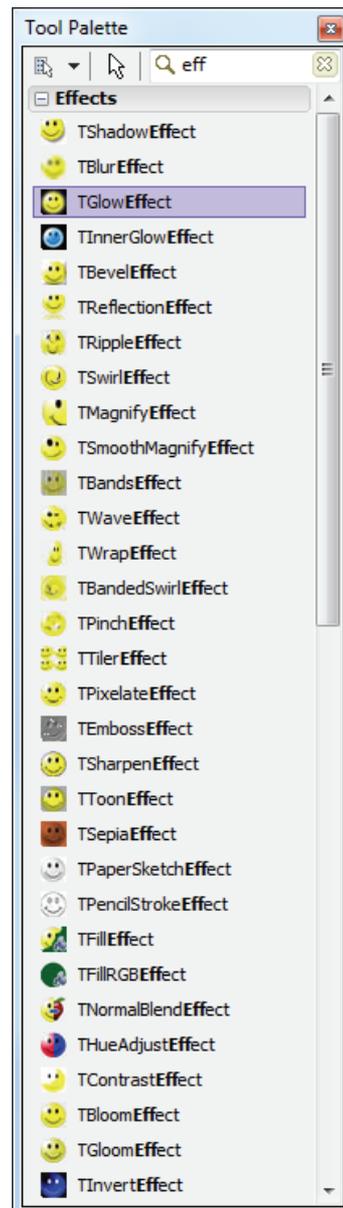
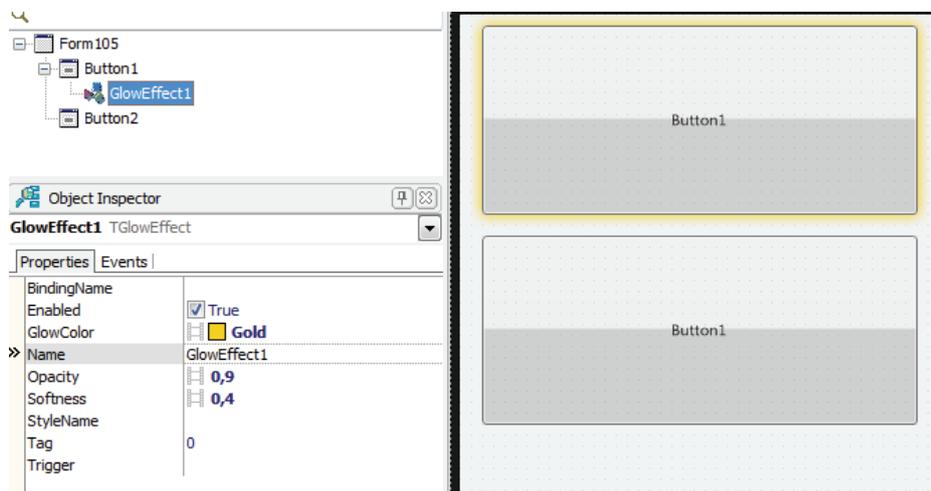


## Učinki

FireMonkey vsebuje množico grafičnih učinkov, ki vplivajo na prikaz slike na zaslonu. Uporabimo jih lahko na poljubnih gradnikih, ne samo na slikah, ter na tak način olepšamo uporabniški vmesnik. Tipičen primer rabe je učinek »glow« na izbranem gumbu ali na vnosnih poljih.

Vsi učinki se izračunajo z uporabo grafičnega procesorja (GPU), tako da ne vplivajo na hitrost izrisa programa. Nekatere parametre učinkov lahko celo animiramo.

Učinek uporabimo, tako da izberemo primeren gradnik s seznama ter ga odložimo znotraj gradnika, na katerem bi radi uporabili ta učinek. Nato nastavimo lastnosti učinka. V spodnjem primeru smo učinek TGlowEffect odložili v gumb. Gumb je s tem pridobil zlat sij. Za primerjavo je pod njim viden še navaden gumb brez sija.



## Seznami

Gradnik TListBox je le od daleč podoben svojemu soimenjaku s platforme VCL. V njega sicer lahko dodajamo nize znakov, a to je le majhen del njegovih zmožnosti.

Polnost zmogljivost TListBoxa dosežemo, če vanj dodamo gradnike TListBoxItem, v vsakega pa postavimo poljubno kombinacijo drugih gradnikov FireMonkey. Testni program, ki ga lahko prenesete z interneta (povezava je zapisana pod kazalom) izkorišča to možnost za prikaz slik in besedila.

Testni program vsebuje enoto ProductDescription, ki definira obrazec frmProductDescription. Na obrazcu so slika (TImageControl) ter dve oznaki (TLabel). Vsi trije gradniki počivajo na nevidnem gradniku TLayout.



Glavni obrazec vsebuje gradnik lbProducts tipa TListBox. Da bi vanj dodali novo vrstico, moramo izvesti nekaj enostavnih korakov.

- Najprej naredimo nov objekt listItem razreda TListBoxItem. Njegov lastnik naj bo kar obrazec (Self), ki bo poskrbel za pravočasno uničenje objekta.
- Nato naredimo nov objekt prodItem razreda TfrmProductDescription. Njegov lastnik je pravkar generirani listItem. S tem smo naredili nov obrazec, ki pa ni prikazan na zaslonu, ker nima določenega starša.
- Nastavimo višino in širino objekta listItem.
- Priredimo vrednosti lastnostim objekta prodItem (nastavimo sliko ter obe oznaki).
- Na koncu nastavimo starša (Product je zgoraj omenjeni nevidni gradnik, ki vsebuje sliko in obe oznaki) ter objekt listItem dodamo v TListBox.

```
var
  listItem: TListBoxItem;
  prodItem: TfrmProductDescription;
begin
  listItem := TListBoxItem.Create(Self);
  prodItem := TfrmProductDescription.Create(listItem);
  listItem.Width := Trunc(lbProducts.ClientWidth);
  listItem.Height := prodItem.ClientHeight;
  prodItem.Image.Bitmap.Assign(imgProduct.Bitmap);
  prodItem.Description.Text := inpProduct.Text;
  prodItem.Quantity := Trunc(inpQuantity.Value);
  prodItem.Product.Parent := listItem;
  lbProducts.AddObject(listItem);
end;
```

Rezultat je lepo oblikovan seznam s slikami in opisi.

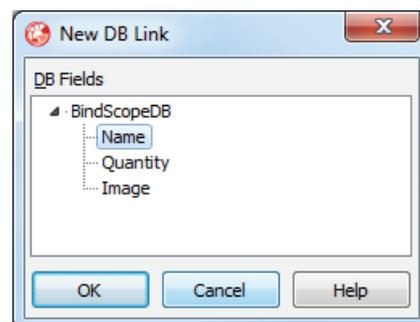
	3D obesek Quantity: 1
	prstan Quantity: 1
	ogrlica z uhani Quantity: 3
	uhani Quantity: 2

## LiveBindings

Kot nadomestek za podatkovno osveščene (data-aware) gradnike vsebuje FireMonkey tehnologijo LiveBindings. Ta je vgrajena tudi v novi VCL in omogoča bistveno več, kot le povezavo podatkovnih virov z zaslonskimi gradniki. Ker pa je tema zelo obširna, se bomo tu posvetili le osnovam, ki jih potrebujete za izgradnjo podatkovne aplikacije s FireMonkeyem.

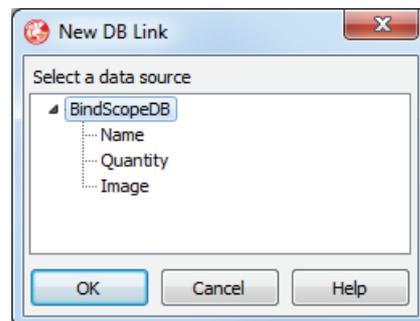
Denimo, da na obrazcu (ali data modulu) že imamo komponento TDataSource. V našem testnem primeru (enota TrgovinaDBBrowser) je TDataSource povezan na TClientDataSet; v vašem primeru bo morda na kaj drugega.

Podatkovno osveščeno vnosno polje naredimo, tako da na obrazec odložimo navadno vnosno polje TEdit. Nato kliknemo puščico desno od lastnosti LiveBindings in iz menija izberemo Link to DB Field. Pojavi se izbirnik s polji iz podatkovnega vira. Izberemo pravo polje in – to je to! Dobili smo podatkovno osveščeno vnosno polje. Delphi je spotoma na komponenti postavil dve polji – TBindingsList (seznam akcij LiveBindings) in TBindScopeDB (povezava med TBindingsList in podatkovnim virom).

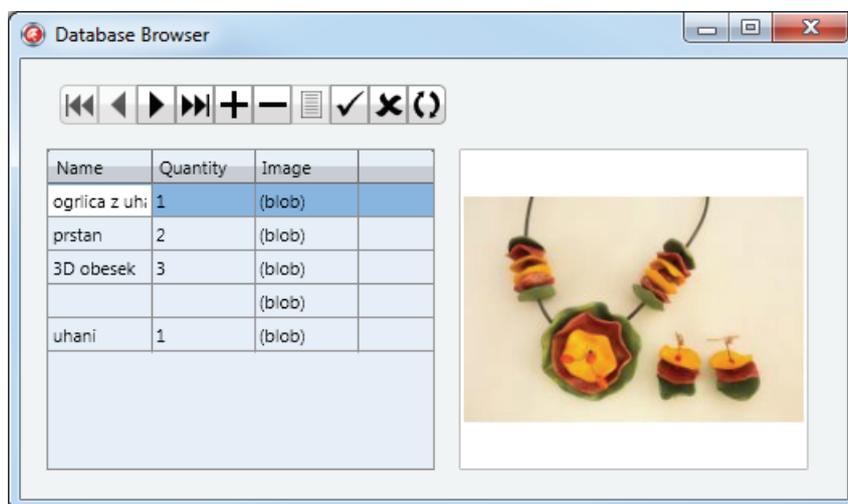


Na enak način lahko slikovni gradnik TImageControl povežemo s poljem, ki vsebuje sliko.

Izdelava podatkovno osveščene tabele se rahlo razlikuje od zgoraj opisanega postopka. Najprej na obrazec odložimo gradnik TStringGrid. Kliknemo na puščico desno od njegove lastnosti LiveBindings in iz menija izberemo Link to DB DataSource. Izberemo želeni podatkovni vir in TStringGrid bo prikazoval podatke.



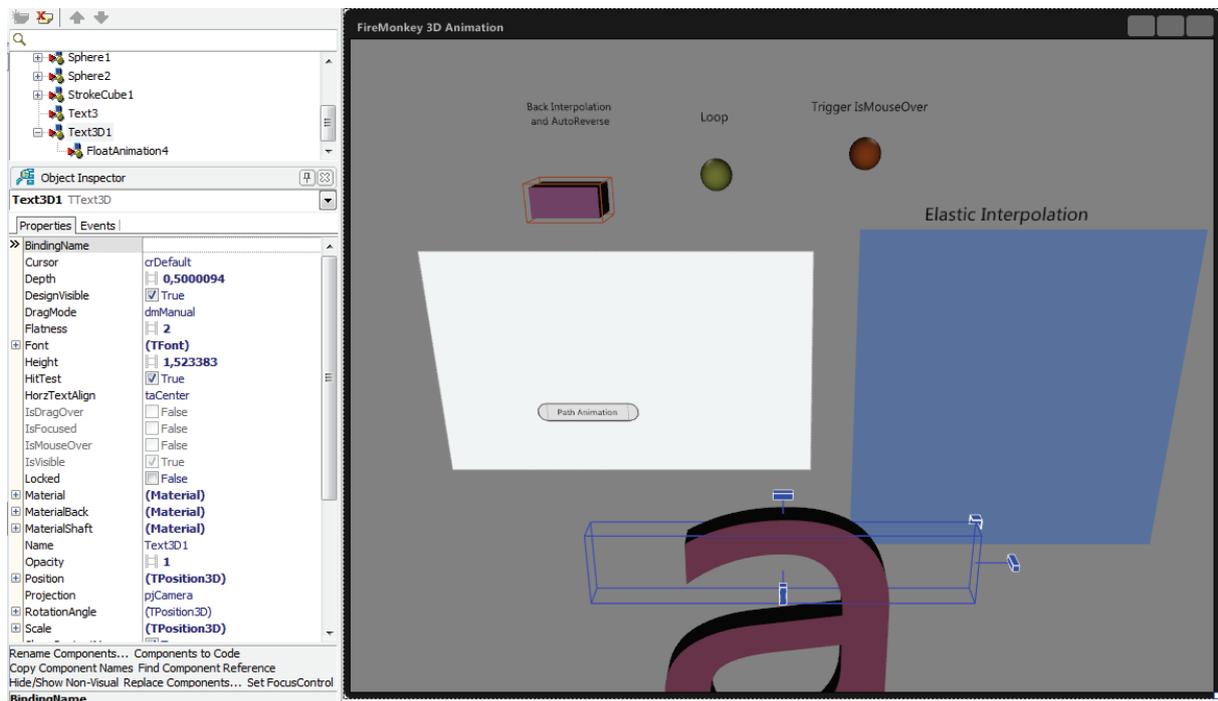
Navigacijo po podatkovnem viru izvedemo z gradnikom TBindNavigator. Nastaviti moramo le lastnost BindScope, ki mora kazati na gradnik TBindScopeDB, ki ga je Delphi ustvaril v prvem koraku zgoraj.



## Luč, kamera, akcija!

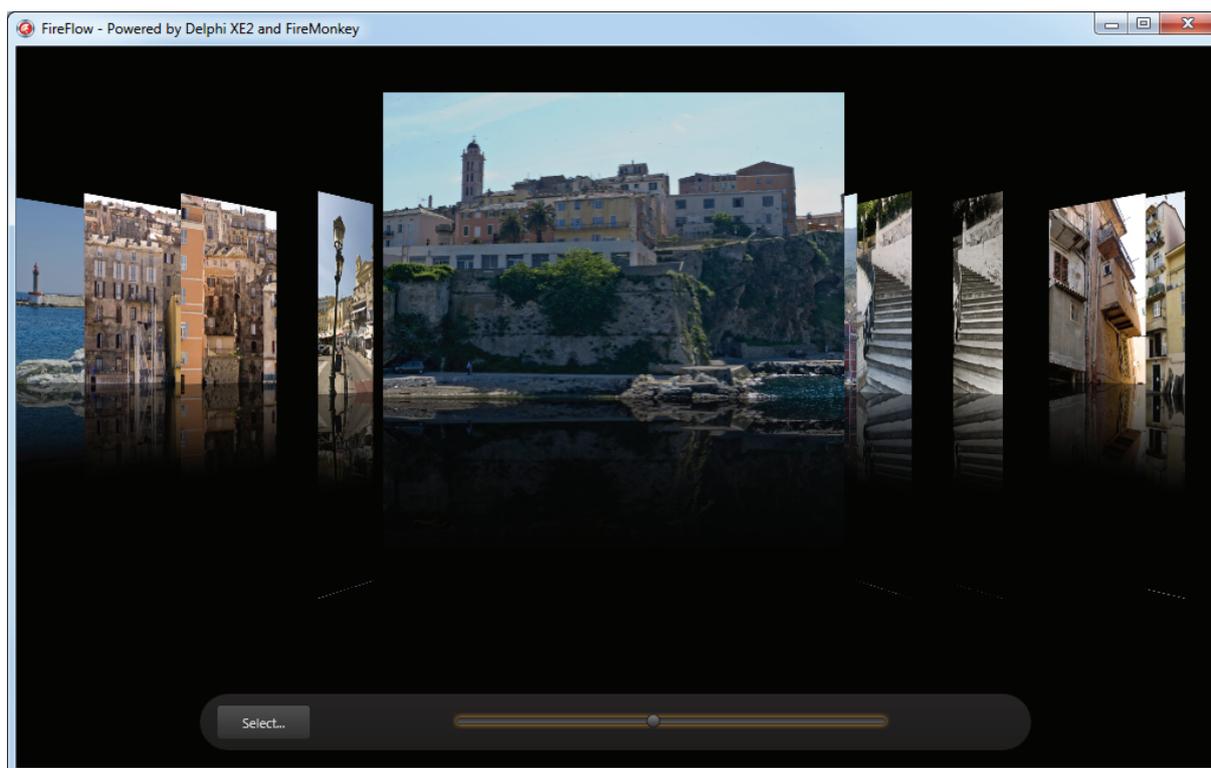
Tridimenzionalne obrazce gradimo iz osnovnih tridimenzionalnih teles (kvader, krogla, valj, stožec), tridimenzionalne mreže in tridimenzionalnega besedila. Na ta telesa nalepimo teksture, jih osvetlimo z lučmi, postavimo kamero na pravo mesto in si ogledamo rezultat. V prostor lahko postavimo tudi dvodimenzionalne površine, na katere nalepimo sliko ali poljubne dvodimenzionalne gradnike (lahko kar cel dvodimenzionalni obrazec).

V tridimenzionalnem urejevalniku so gradniki okrašeni s tremi ročicami za vrtenje v vseh treh prostorskih oseh. V smereh osi X in Y premikamo gradnik z miško, za premikanje v smeri osi Z pa držimo pritisnjeno tipko Ctrl in premikamo miško.



Zapletene tridimenzionalne modele lahko izdelamo v zunanjem urejevalniku 3D objektov, ga izvozimo v zapis COLLADA ter uvozimo v gradnik TViewport3D. Ta možnost je predstavljena v Delphiju priloženem projektu COLLADAModelViewer.

Tridimenzionalni obrazci omogočajo enostavno izdelavo vmesnikov tipa »CoverFlow«.



V programu FireFlow, ki je priložen Delphiju, je vsaka slika postavljena na gradnik TLayer3D, ti gradniki pa so razpostavljeni v prostoru. Za izračun projekcije in animacijo ob prehodu poskrbi FireMonkey.

## Viri

*FireMonkey Application Platform (dokumentacija)*

[http://docwiki.embarcadero.com/RADStudio/en/FireMonkey\\_Application\\_Platform](http://docwiki.embarcadero.com/RADStudio/en/FireMonkey_Application_Platform)

*Customizing FireMonkey Applications with Styles*

[http://docwiki.embarcadero.com/RADStudio/en/Customizing\\_FireMonkey\\_Applications\\_with\\_Styles](http://docwiki.embarcadero.com/RADStudio/en/Customizing_FireMonkey_Applications_with_Styles)

*FireMonkey Fonts and Native Look & Feel*

<http://itinerantdeveloper.blogspot.com/2012/03/firemonkey-fonts-and-native-look-feel.html>

*Delphi XE2: Applying a Style to FireMonkey*

<http://blog.analogmachine.org/2011/09/01/applying-a-style-to-firemonkey/>

*FireMonkey Animation Effects*

[http://docwiki.embarcadero.com/RADStudio/en/FireMonkey\\_Animation\\_Effects](http://docwiki.embarcadero.com/RADStudio/en/FireMonkey_Animation_Effects)

*LiveBindings – Displaying database data in a FireMonkey application*

<http://members.adug.org.au/2011/12/29/livebindings-01/>

*Creating a FireMonkey Component*

[http://docwiki.embarcadero.com/RADStudio/en/Creating\\_a\\_FireMonkey\\_Component](http://docwiki.embarcadero.com/RADStudio/en/Creating_a_FireMonkey_Component)

*A TBitButton equivalent for FireMonkey*

<http://members.adug.org.au/2012/03/02/a-tbitbutton-equivalent-for-firemonkey/>

*How to Create Your Own FireMonkey Image Effect*

<http://members.adug.org.au/2011/12/15/how-to-create-your-own-firemonkeyimage-filtereffect-to-use-with-firemonkey/>

*Useful tips for fireMonkey and Delphi XE2*

<http://blog.analogmachine.org/2011/09/04/useful-tips-for-firemonkey-and-delphi-xe2/>

## Programi

*MonkeyGroomer – pretvorba projektov VGScene v FireMonkey*

<http://pascalcoder.blogspot.com/search/label/MonkeyGroomer>

*Mida – pretvorba projektov VCL v FireMonkey*

<http://www.midaconverter.com/>

*MonkeyMixer – souporaba VCL in FireMonkey obrazcev v istem projektu*

<http://www.simonjstuart.com/2011/10/19/world-meet-monkeymixer-use-firemonkey-forms-directly-in-vcl-projects/>

*MonkeyStyler – izdelava novih slogov*

<http://monkeystyler.com/styler>

## 3D

*FireMonkey 3D*

[http://docwiki.embarcadero.com/RADStudio/en/FireMonkey\\_3D](http://docwiki.embarcadero.com/RADStudio/en/FireMonkey_3D)

*FireMonkey 3D Text Editor*

<http://www.andreanolanusse.com/en/firemonkey-3d-text-editor-delphi-source-code-available/>

*Visualizing wave interference using FireMonkey and C++Builder XE2*

<http://edn.embarcadero.com/article/42115>

## Video

*FireMonkey FireStarter*

<http://cc.embarcadero.com/Item/28563>

*Building Rich Business Applications with FireMonkey – video in odličnem priročniku v obliki pdf*

<http://www.embarcadero.com/rad-in-action/firemonkey>

*24 Hours of Delphi*

<http://edn.embarcadero.com/article/42070>

*FireMonkey 3D Controls: New Ideas for Visualizations*

<http://www.youtube.com/watch?v=MSk4LqFgNFQ>

*FireMonkey Styles with Eugene Kryukov*

<http://blogs.embarcadero.com/vsevolodleonov/2012/03/12/firemonkey-styles-with-eugene-kryukov-webinar-recording/>

*31 Days of FireMonkey*

<http://www.youtube.com/playlist?list=PL19268CFB728C1EFF>

## Komponente

*Arcana*

<http://arcana.sivv.com/apesuite/>

*TMS*

<http://www.tmssoftware.com/site/products.asp?t=fm>

