



# DelphiDay

italian conference

## Modernizzare applicazioni VCL

Come affrontare il porting di vecchie applicazioni senza dover stravolgere il codice e le form originali...



# Carlo Barazzetta

**Ethea S.r.l.** - Socio fondatore e  
Responsabile progetti IT

@ carlo.barazzetta@ethea.it

X x.com/CarloBarazzetta

github.com/EtheaDev

in linkedin.com/in/carlo-barazzetta



11-12 Giugno 2024  
Piacenza





## **OPEN-SOURCE PROJECTS** **COMPONENTS**

[github.com/EtheaDev](https://github.com/EtheaDev)

**SVG Icon ImageList** <sup>306\*</sup>

[github.com/EtheaDev/SVGIconImageList](https://github.com/EtheaDev/SVGIconImageList)

**Icon Fonts ImageList** <sup>212\*</sup>

[github.com/EtheaDev/IconFontsImageList](https://github.com/EtheaDev/IconFontsImageList)

**Styled Components** <sup>127\*</sup>

[github.com/EtheaDev/StyledComponents](https://github.com/EtheaDev/StyledComponents)

## **OPEN-SOURCE PROJECTS** **COMPONENTS**

[github.com/EtheaDev](https://github.com/EtheaDev)

**Delphi GoogleMap** <sup>72\*</sup>

[github.com/EtheaDev/DelphiGoogleMap](https://github.com/EtheaDev/DelphiGoogleMap)

**Markdown Help Viewer** <sup>54\*</sup>

[github.com/EtheaDev/MarkdownHelpViewer](https://github.com/EtheaDev/MarkdownHelpViewer)

**DBAware Labeled Components** <sup>24\*</sup>

[github.com/EtheaDev/DBAwareLabeledComponents](https://github.com/EtheaDev/DBAwareLabeledComponents)

## **OPEN-SOURCE PROJECTS**

### **SHELL EXTENSIONS**

[github.com/EtheaDev](https://github.com/EtheaDev)

**SVG Shell Extensions** <sup>108\*</sup>

[github.com/EtheaDev/SVGShellExtensions](https://github.com/EtheaDev/SVGShellExtensions)

**SKIA Shell Extensions** <sup>58\*</sup>

[github.com/EtheaDev/SKIAShellExtensions](https://github.com/EtheaDev/SKIAShellExtensions)

**Markdown Shell Extensions** <sup>47\*</sup>

[github.com/EtheaDev/MarkdownShellExtensions](https://github.com/EtheaDev/MarkdownShellExtensions)

**Fattura Elettronica Explorer** <sup>22\*</sup>

[github.com/EtheaDev/FExplorer](https://github.com/EtheaDev/FExplorer)

## **OPEN-SOURCE PROJECTS**

### **OTHERS**

[github.com/EtheaDev](https://github.com/EtheaDev)

**InstantObjects** <sup>95\*</sup>

[github.com/EtheaDev/InstantObjects](https://github.com/EtheaDev/InstantObjects)

**VCL Theme Selector** <sup>54\*</sup>

[github.com/EtheaDev/VCLThemeSelector](https://github.com/EtheaDev/VCLThemeSelector)

**Kitto2** <sup>94\*</sup>

[github.com/EtheaDev/kitto2](https://github.com/EtheaDev/kitto2)

# AGENDA

---

- Come affrontare il porting di una vecchia applicazione
- Alcuni “trucchi” del mestiere
- Componenti OpenSource di Ethea
- Sostituire il BDE
- Automatizzare la modifica del codice sorgente
- Q&A

# Come affrontare il Porting di una vecchia applicazione

1

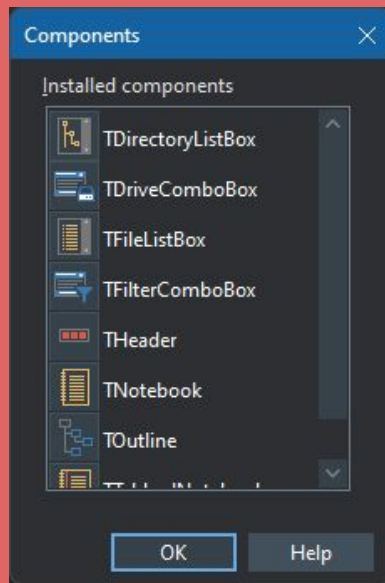
# Meglio riciclare o rifare tutto?

---

- Buttare tutto e rifare tutto con altri linguaggi:
  - ◆ Pro: ci sono più sviluppatori e aziende disponibili
  - ◆ Contro: non si riutilizza quasi nulla di quanto fatto, molto costoso
- Buttare tutto e rifare in Delphi:
  - ◆ Pro: si può riciclare parte del codice e della GUI
  - Contro: richiede conoscenza del dominio applicativo
- Adattare l'esistenze e cambiare le tecnologie obsolete (es.BDE):
  - ◆ Pro: si ricicla gran parte del codice (80/90%)
  - ◆ Pro: può essere affidato a chi non conosce il dominio applicativo
  - ◆ Pro: non richiede formazione del personale
  - ◆ Pro: si può mantenere un approccio "parallelo" allo sviluppo

# Meglio “riciclare”, perché:

- La retrocompatibilità è uno dei cavalli di battaglia di Delphi: il codice scritto 29 anni fa compila ancora.
- Si può forzare il compilatore a comportarsi come in vecchie versioni (es. allineamento dei Record)
- I componenti ci sono ancora tutti (vedi package “Delphi 1.0 Compatibility Components”)
- I componenti “smarriti” *ShellControls*
- I VCLStyles hanno introdotto alcuni problemi!
- Alcuni componenti di terze parti sono spariti!





# Ostacoli da superare:

---

- Alcuni componenti sono “smarriti” (es. *ShellControls*)
- I VCLStyles hanno introdotto alcuni problemi...
- Se si sono usati componenti di terze che non esistono più, occorre trovare delle alternative o riadattare il codice.
- Se si è utilizzato il BDE è necessario sostituirlo.
- Se si è utilizzato un approccio “Table oriented” bisogna fare attenzione al passaggio a SQL
- *In questo talk vedremo come risolvere tanti di questi problemi...*

# Approccio “parallelo”: i vantaggi

---

- Processo di Porting “parallelo” allo sviluppo
  - ◆ Lo sviluppo applicativo non si ferma
  - ◆ Mantenere la stessa base di codice e .dfm (VersionControl)
  - ◆ Gestire un Progetto separato per la nuova versione di Delphi
  - ◆ Opzione “salvagente” (c'è sempre la vecchia versione)
- Conversione dfm da binario a txt
  - ◆ per versioning (solo da Delphi 5)
  - ◆ Gestire solo le “differenze” con diversi trucchi...che vedremo.

# Alcuni “trucchi” del mestiere

# 2

# 1° trucco: Hooking e Interposer unit

---

- Es. problema: TPanel e TForm perdono il “colore”
- Usare l’hooking abbinato all’Interposer unit:
  - ◆ Definire una singola unit di interposer per ogni unit VCL
    - es. Hooks\_Forms e Hooks\_ExtCtrls
    - Definire la stessa classe VCL nella unit di interposer
- Aggiungere a tutte le unit la unit di interposer DOPO la unit VCL
  - ◆ es. Forms, Hooks\_Forms, ExtCtrls, Hooks\_ExtCtrls
- Vediamo con un esempio come risolvere il problema di TPanel

## 2° trucco: Include/Defines/VCL Styles

- Problema: mantenere compatibile la stessa base di codice
  - ◆ Le unit .pas e .dfm devono essere retrocompatibili
- Creare una unit da includere in tutti i sorgenti (es Hooks.inc)
  - ◆ Definire quando l'hook è attivo in base alla versione di Delphi
  - ◆ Defines diverse \$DEFINE per hook "specifici"
- Cambiare il comportamento di "default"



# 3° trucco: Componenti retrocompatibili

- Problema: componenti non più disponibili:
- Creare componenti “retrocompatibili”
  - ◆ es. TPrintBarCode derivato da TPaintBox che usa ZintBarCode
- Componenti ancora disponibili:
  - ◆ Quasi tutti sono stati aggiornati
  - ◆ Es. QuickReport: è possibile fare i package nuovi...
- Usare componenti “retrocompatibili” (filosofia Ethea)
  - ◆ es. StyledComponents
  - ◆ es. LabeledDbGrid

# 4° trucco: miglioriamo la GUI

---

- Rendiamo l'applicazione più moderna:
- Abilitiamo gli stili della VCL e il supporto High-DPI
  - ◆ Attenzione al codice “owner-draw”, potrebbe non funzionare...
  - ◆ Usare un “selettore” per lo stile VCL (VCLThemeSelector)
  - ◆ Abilitare di default il tema di Windows (chiaro o scuro)
  - ◆ Sostituire le icone con VirtualImageList e ImageCollection
  - ◆ Libreria StyleUtils, fornisce centinaia di “Hooks”
- Aggiungiamo un “tocco” moderno: Dialog animate e Google Maps

# Componenti Open-Source di Ethea

3

# StyledComponents

---

- Pulsanti e Dialogs moderne (componenti “retrocompatibili”):
  - ◆ StyledButton e StyledGraphicBtns
  - ◆ StyledSpeedButton, StyledBitBtn
  - ◆ StyledCategoryButtons e StyledButtonGroup
  - ◆ StyledToolbar
  - ◆ StyledDbNavigator e StyledBindNavigator
  - ◆ StyledTaskDialog (anche con animazioni)
- Esempio: CRMDemo con “Styled” Interposer

# SVGIconImageList o IconFontsImageList

- ImageList con immagini che “scalano”:
  - ◆ Immagini scalabili (SVG o Font)
    - si adattano al cambio DPI dello schermo
    - Utilizzabili con VirtualImageList
  - ◆ Sostituzione delle ImageList già presenti
    - Occorre solo cambiare le immagini nella collection
      - IconFontsImageList è compatibile da Delphi7
      - SVGIconImageList è compatibile da Delphi XE3



# Number Box e DbGrid Avanzata

---

- Componenti DBAwareLabeledComponents:
- NumberBox Dataware:
  - TDbNumberBox
- Advanced DbGrid:
  - ◆ Retrocompatibile con DbGrid
    - Alternate Row Colors, Sort, Row Margins, Lines Per Row (memo), custom colors, incremental search

# Delphi Google Maps

---

→ Componente per integrazione facile di Google Maps:

- ◆ Vai alla posizione:
  - per indirizzo, Lat/Lon o “Course”
- ◆ Routing (Drive, Walk, Bicycle, Transit)
  - Per indirizzi o per Lat/Lon
- ◆ Preview di stampa
- ◆ Position Markers e custom Markers
- ◆ GUI control on/off (satellite, Street View, Zoom)

# Sostituzione del BDE

4

# Diversi metodi e possibilità

---

- Prima opzione: non sostituirlo, esiste ancora!
- Utilizzo di “ComponentACE”
- Utilizzo di NexusDB (per DBF)
- Replace “components” with FireDAC
  - ◆ [https://docwiki.embarcadero.com/RADStudio/Athens/en/Migrating\\_BDE\\_Applications\\_to\\_FireDAC](https://docwiki.embarcadero.com/RADStudio/Athens/en/Migrating_BDE_Applications_to_FireDAC)
- metodo Ethea:
  - ◆ Utilizzare InstantBDEExpress (obsoleto)
  - ◆ Utilizzare InstantBDE2FireDAC (coming soon)

# Utilizzo di InstantBDExpress

---

→ Molte aziende lo utilizzano dal 2006!

## ◆ Pro:

- Comodo per un porting trasparente da BDE a DbExpress
- Utilizzabile in modalità Interposer o con nuovi componenti
- Collaudato da più di 18 anni

## ◆ Contro:

- Tecnologia vecchia: usa “DbExpress”
- SQL oriented (non è compatibile con DBF o Paradox)



# Utilizzo di InstantBDE2FireDAC

---

→ Stessa filosofia di InstantBDEExpress ma con più vantaggi:

◆ Pro:

- Utilizza tecnologia moderna (FireDAC)
- Comodo per un porting trasparente da BDE a FireDAC
- Utilizzabile in modalità Interposer o con nuovi componenti
- Può utilizzare anche le tabelle DBF e Paradox (via ODBC)

◆ Contro:

- Disponibile entro la fine dell'anno

**Automatizzare  
la modifica del  
codice sorgente**

5

# Modificare il codice esistente

---

→ Sfruttare le nuove funzionalità del linguaggio:

◆ Helpers

- es. TDataSetHelper

◆ Generics, Anonymous Methods

- es. TRecord<R: record>
- es. ForEachRecord

→ Sostituire più facilmente il codice esistente

# Sfruttare e automatizzare RegEx

---

- Quando il codice da sostituire è tanto e “variabile”:
  - ◆ RegEx fornisce la soluzione per ricerche e sostituzioni complesse:
    - Per l'analisi: <https://regex101.com/>
- Sostituire più facilmente il codice esistente:
  - ◆ SourceManager (di Ethea)
    - Sostituzione massiva di codice
    - Lavora a “blocchi” di funzioni/procedure Delphi

# Esempi e risorse

# 6

# Esempi usati e citati nel Talk

- I componenti “fantasma” ShellControls:
  - ◆ <https://github.com/EtheaDev/DelphiShellControlsPackages>
- Il piccolo progetto di esempio “PanelsDemo” per spiegare la tecnica “hook-interposer”:
  - ◆ <https://github.com/carloBarazzetta/InterposerDemo>
- Il componente DelphiGoogleMap:
  - ◆ <https://github.com/EtheaDev/DelphiGoogleMap>
- Il componente TLabeledDbGrid e TDBNumberBox:
  - ◆ <https://github.com/EtheaDev/DBAwareLabeledComponents>
- Gli “StyledComponents” e le dialog animate (con Skia4Delphi):
  - ◆ <https://github.com/EtheaDev/StyledComponents>
- I componenti SVGIconImageList per le icone SVG scalabili:
  - ◆ <https://github.com/EtheaDev/SVGIconImageList>
- La demo di “interposer” per modernizzare la Windows10CRMDemo:
  - ◆ [https://github.com/EtheaDev/StyledComponents/wiki/Interposer-Unit-\(Vcl.StyledComponentsHooks\)](https://github.com/EtheaDev/StyledComponents/wiki/Interposer-Unit-(Vcl.StyledComponentsHooks))
- La demo di una applicazione moderna e del selettore di Stili della VCL:
  - ◆ <https://github.com/EtheaDev/VCLThemeSelector>



GRAZIE

