



DelphiDay
italian conference

Porting e modernizzazione vecchie 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



20 Novembre 2024
Padova



OPEN-SOURCE PROJECTS COMPONENTS

github.com/EtheaDev

SVG Icon ImageList ^{328*}

github.com/EtheaDev/SVGIconImageList

Icon Fonts ImageList ^{218*}

github.com/EtheaDev/IconFontsImageList

Styled Components ^{157*}

github.com/EtheaDev/StyledComponents

OPEN-SOURCE PROJECTS COMPONENTS

github.com/EtheaDev

Delphi GoogleMap ^{72*}

github.com/EtheaDev/DelphiGoogleMap

Markdown Help Viewer ^{65*}

github.com/EtheaDev/MarkdownHelpViewer

DBAware Labeled Components ^{29*}

github.com/EtheaDev/DBAwareLabeledComponents

OPEN-SOURCE PROJECTS

SHELL EXTENSIONS

github.com/EtheaDev

SVG Shell Extensions ^{129*}

github.com/EtheaDev/SVGShellExtensions

SKIA Shell Extensions ^{64*}

github.com/EtheaDev/SKIAShellExtensions

Markdown Shell Extensions ^{62*}

github.com/EtheaDev/MarkdownShellExtensions

Fattura Elettronica Explorer ^{23*}

github.com/EtheaDev/FExplorer

OPEN-SOURCE PROJECTS

MISC - OTHERS

github.com/EtheaDev

InstantObjects ^{98*}

github.com/EtheaDev/InstantObjects

VCL Theme Selector ^{57*}

github.com/EtheaDev/VCLThemeSelector

Kitto2 ^{96*}

github.com/EtheaDev/kitto2



AGENDA

- Come affrontare il porting di una vecchia applicazione
- Alcuni “trucchi” del mestiere
- UNICODE: un grande salto!
- Componenti OpenSource di Ethea
- Sostituire il BDE: diversi scenari
- 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:

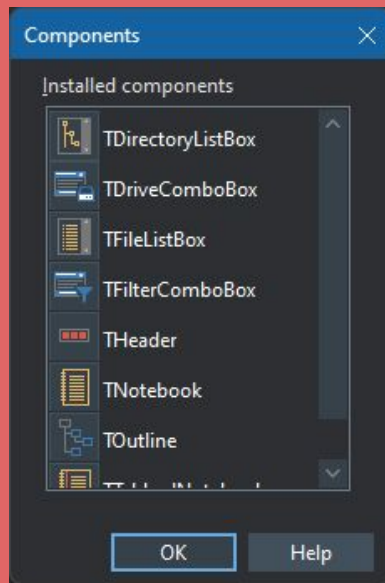
- ◆ Pro: si può riciclare parte del codice e della GUI
- Contro: richiede conoscenza del dominio applicativo

→ 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:

- Tanti sono i motivi per non “buttare” il codice:
- ◆ 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*



Ostacoli da superare:

- Alcuni ostacoli si possono superare:
 - ◆ Alcuni componenti sono “smarriti” (es. *ShellControls*)
 - ◆ I VCLStyles hanno introdotto alcuni problemi legati agli stili
 - ◆ 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 Seminario 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
 - ◆ Definire diverse \$DEFINE per hook "specifici"
- Cambiare il comportamento di "default"
- Abilitare Hooking aggiungendo le uses alle unit di Interposer

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 SVGIconImageList
 - ◆ Libreria StyleUtils, fornisce centinaia di “Hooks”
- Aggiungiamo un “tocco” moderno: Dialog animate e Google Maps

UNICODE:
un grande salto!

3

1: Cos'è UNICODE

- Da Delphi 2009 è stato introdotto il supporto a UNICODE:
 - ◆ un nuovo tipo **UnicodeString** mappa il tipo **string** che di default diventa “multibyte”, 2 byte per carattere, basato su UTF16-LE.
- UTF: Unicode Transformation Format:
 - ◆ UTF8 è il più popolare e usato in HTML e XML
 - ◆ UTF16 è usato da Windows, Java, .net
 - ◆ Il BOM (Byte Order Mark) per gestire BigEndian o LittleEndian

<https://www.embarcadero.com/images/dm/technical-papers/delphi-and-unicode-marco-cantu.pdf>

2: UNICODE – punti di attenzione

- UpperCase non funziona: usare AnsiUpperCase o ToUpper(s)
- Non usare WideString, perché non è RefCounted (come ShortString)
- AnsiString (nuovo tipo):
 - ◆ ora supporta una CodePage specifica, tramite SetCodePage
- UTF8String e RawByteString
- Warnings del compilatore per conversioni implicite (lentezza):
 - ◆ W1057 Implicit string cast from 'UTF8String' to 'string'
 - ◆ W1057 Implicit string cast from 'AnsiString' to 'string'
- Warnings del compilatore con rischio di perdita di dati:
 - ◆ W1058 Implicit string cast with potential data loss from 'string' to 'UTF8String'

3: Encoding: lettura e scrittura di files

- L'uso di TStrings per leggere e scrivere files è cambiato:
 - ◆ I metodi ReadFromFile e WriteToFile di TStrings possono essere chiamati con un encoding specifico.
- Nuova classe **TEncoding**, con sotto classi (es. **TUTF8Encoding**)
 - ◆ Se non viene specificato usa **TEncoding.Default** che tratta i file come "Ansi"
- UTF8String e RawByteString
- Warnings del compilatore per conversioni implicite (lentezza):
 - ◆ W1057 Implicit string cast from 'UTF8String' to 'string'
 - ◆ W1057 Implicit string cast from 'AnsiString' to 'string'
- Warnings del compilatore con rischio di perdita di dati:
 - ◆ W1058 Implicit string cast with potential data loss from 'string' to 'UTF8String'

4: UNICODE, Database e TField

- Con il supporto a Unicode è possibile gestire Database con campi UNICODE (es. NVARCHAR)
 - ◆ La classe del TField cambia da **TStringField** a **TWideStringField**
 - ◆ Il StringField.Value cambia da AnsiString a String (Unicode)
 - ◆ Attenzione all'uso di TField.Value (Variant) e TField.AsString (che cambia a seconda del tipo di campo)
- In caso di utilizzo di campi non Unicode il rischio è la perdita dei dati senza che il compilatore mi avvisi!
 - ◆ Se si usa "AppendRecord"
- Nelle funzioni "generiche" con TField non usare Field.Value (è un Variant) ma Field.AsString
- Si può intervenire a livello di TDataSet per decidere se creare campi TStringField o TWideStringField nel metodo:

Il supporto high-DPI

4

HIGH-DPI: la differenza si vede!


Applicazione senza High-DPI

DPI virtualization (da Windows Vista)

Species No	Graphic
90030	
Category	
Snapper	
Common_Name	
Red Emperor	
Species Name	
Lutjanus sebae	
Length (cm)	
60	
Length_In	
23,6220472440945	
Notes	

Applicazione con High-DPI

Per monitor DPI-scaling (da Windows 8.1)

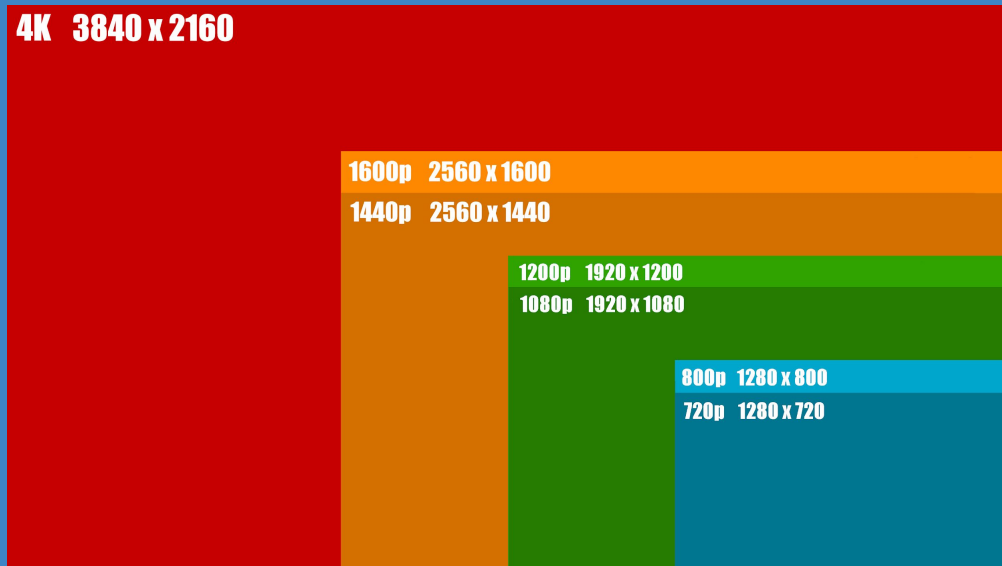
Species No	Graphic
90030	
Category	
Snapper	
Common_Name	
Red Emperor	
Species Name	
Lutjanus sebae	
Length (cm)	
60	
Length_In	
23,6220472440945	
Notes	

Il supporto High-DPI (Documentazione)

- Come funziona? Come si attiva in Delphi?
- Documentazione MS:
<https://docs.microsoft.com/it-it/windows/win32/hidpi/high-dpi-desktop-application-development-on-windows?redirectedfrom=MSDN>
- Documentazione Embarcadero:
<https://community.idera.com/developer-tools/b/blog/posts/how-to-guide-upgrading-your-delphi-vcl-applications-to-support-4k-displays>
- Supporti differenti in base alla versione di Delphi (consigliato almeno D10.3)
 - Delphi fa quasi tutto il lavoro in automatico se:
 - `Form.Scaled = True`, `Form.ParentFont = False` (`Form.PixelsPerInch=96`)
 - Attenzione alle impostazioni “autosize”
- In Windows parliamo di “logical DPI” che non corrisponde a quella fisica!

Non solo pixel... anche dim. testo

Diverse risoluzioni, ma anche diversa % del Font, cioè DPI (DotPerInch) = PPI (PixelsPerInch)



con gli schermi 2K su notebook da 14/15 pollici e sugli schermi 4K è facile trovare impostazioni di 125% o 150% o anche 200%.

Schermo

Ridimensionamento e layout

Alcune app non rispondono alle modifiche del ridimensionamento finché non le chiudi e le riapri.

Modifica la dimensione di testo, app e altri elementi

150% (scelta consigliata)

Impostazioni ridimensionamento avanzate

Risoluzione dello schermo

3840 x 2160 (scelta consigliata)

Orientamento dello schermo

Orizzontale

Più schermi

Più schermi

Estendi questi schermi

☒ Imposta come schermo principale

2 problemi: icone e font

- I componenti scalano (forse, non benissimo...) ma le icone?
 - ◆ Le ImageList classiche non forniscono alcun supporto
 - ◆ Da Delphi 10.3 esiste VirtualImageList e ImageCollection... e prima di 10.3?
 - ◆ Diffusione di 2 standard:
 - material design
 - SVG icons
- Il font della nostra applicazione
- Si adatta al font del S.O. (Segoe UI)
- Permettiamo all'utente di personalizzarlo?

Componenti Open-Source di Ethea

5

SVGIconImageList o IconFontsImageList

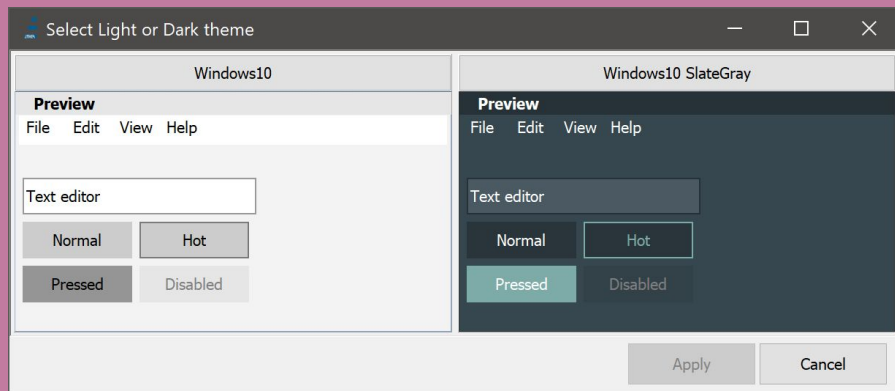
- ImageList con immagini che “scalano”:
 - ◆ Immagini scalabili (SVG o Font)
 - si adattano al cambio DPI dello schermo
 - Utilizzabili con VirtualImageList
 - ◆ Sostituzione delle ImageList già presenti
 - Occorre solo cambiare le immagini nella collection
 - IconFontsImageList è compatibile da Delphi7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

Modernizzare con i VCLStyles

- Sfruttiamo gli stili della VCL (disponibili da XE2)
 - ◆ Come si abilitano e si aggiungono
 - ◆ Disponibilità di nuovi stili in GetIt
- Proprietà: StyleElements (per “disattivarli”)
 - ◆ Il “non stile” Windows
 - ◆ Attenzione ai componenti “owner-draw”
 - ◆ Attenzione agli attributi Ctrl3D, Bevel, Border, ecc...
- Uso dei colori:
 - ◆ TStyleManager.ActiveStyle.GetStyleColor, GetStyleFontColor, GetSystemColor, GetElementColor...

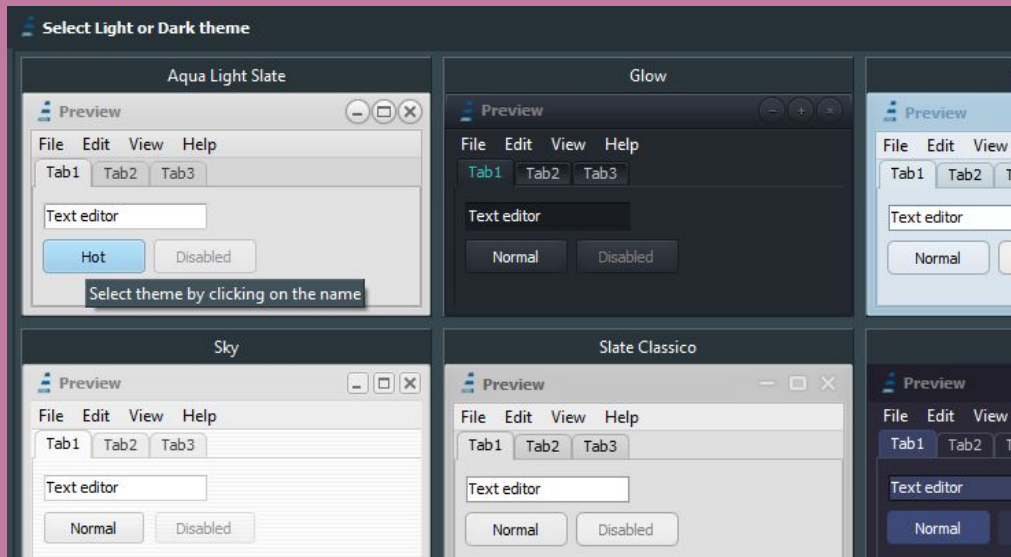
Tema chiaro/scuro

- Sfruttiamo gli stili della VCL:
 - ◆ Attenzione alla proprietà StyleElements (per disattivarli).
 - ◆ Usiamo il preview/selector TVCLThemeSelector per semplificare la scelta
 - ◆ Escludere o meno il “non stile” Windows
 - ◆ Autoridimensionamento (MaxCols e MaxRows)
 - ◆ Come si integra (provare il VCLThemeSelectorLauncher)
- Basato su:
 - ◆ VCLStylePreview della libreria VCLStyleUtils ma con supporto High-DPI



VCLThemeSelector da Delphi 10.4

- Da Delphi 10.4, VCLTheme Selector sfrutta “per control-style”
- ◆ E’ possibile utilizzare più stili contemporaneamente
- ◆ Il preview/selector TVCLThemeSelector è stato ridisegnato
- ◆ Utilizzo di una form di preview con oggetti “reali” e “interattivi”
- Non necessita più di VCLStylePreview



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

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 Borland Database Engine (BDE)

6

Diversi metodi e possibilità

- Prima opzione: non sostituirlo, esiste ancora!
- “ElevateDB/DBISAM” per file Paradox
- Utilizzo di NexusDB per file DBF
- Replace “components” with FireDAC
 - ◆ https://docwiki.embarcadero.com/RADStudio/Athens/en/Migrating_BDE_Applications_to_FireDAC
- metodo Ethea:
 - ◆ Utilizzare InstantBDEExpress (obsoleto)
 - ◆ Utilizzare InstantBDE2FireDAC (in Beta)

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” ma con driver Devart è ancora utilizzabile oggi (con buone performance)
- 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:

- Prodotto in Beta: disponibile entro la fine dell'anno

Automatizzare la modifica del codice sorgente

7

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

8

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>



Q&A
THANK YOU