



FGX

the Evolution of native mobile applications



CLAUDIO PIFFER

PC SOFT

CLAUDIO.PIFFER @GMAIL.COM



AGENDA

- ...partiamo da dove eravamo rimasti a giugno 2020
 - <https://youtu.be/g-c20gSG9vc>
- Installazione
- Evoluzioni della libreria e dei componenti e...
- Il futuro di FGX
 - Mobile development in Delphi should be fun!

FGX Native

→ Sviluppata da Yaroslav Brovin

- È stato molti anni dipendente Embarcadero presso San Pietroburgo come FireMonkey developer
- Conosce perfettamente pregi e difetti di FMX e da qui l'idea di sviluppare FGX Native specificatamente dedicata al mondo mobile concentrandosi solo su questa tipologia di app
- FGX native è una alternativa a FMX e non un sostituto!



FGX Native: FMX vs FGX

FMX



- (+) Più “facile” aggiungere una nuova piattaforma
- (+) l’implementazione di un controllo funziona su tutte le piattaforme sottostanti
- (-) I controlli non sono nativi
- (-) Prestazioni

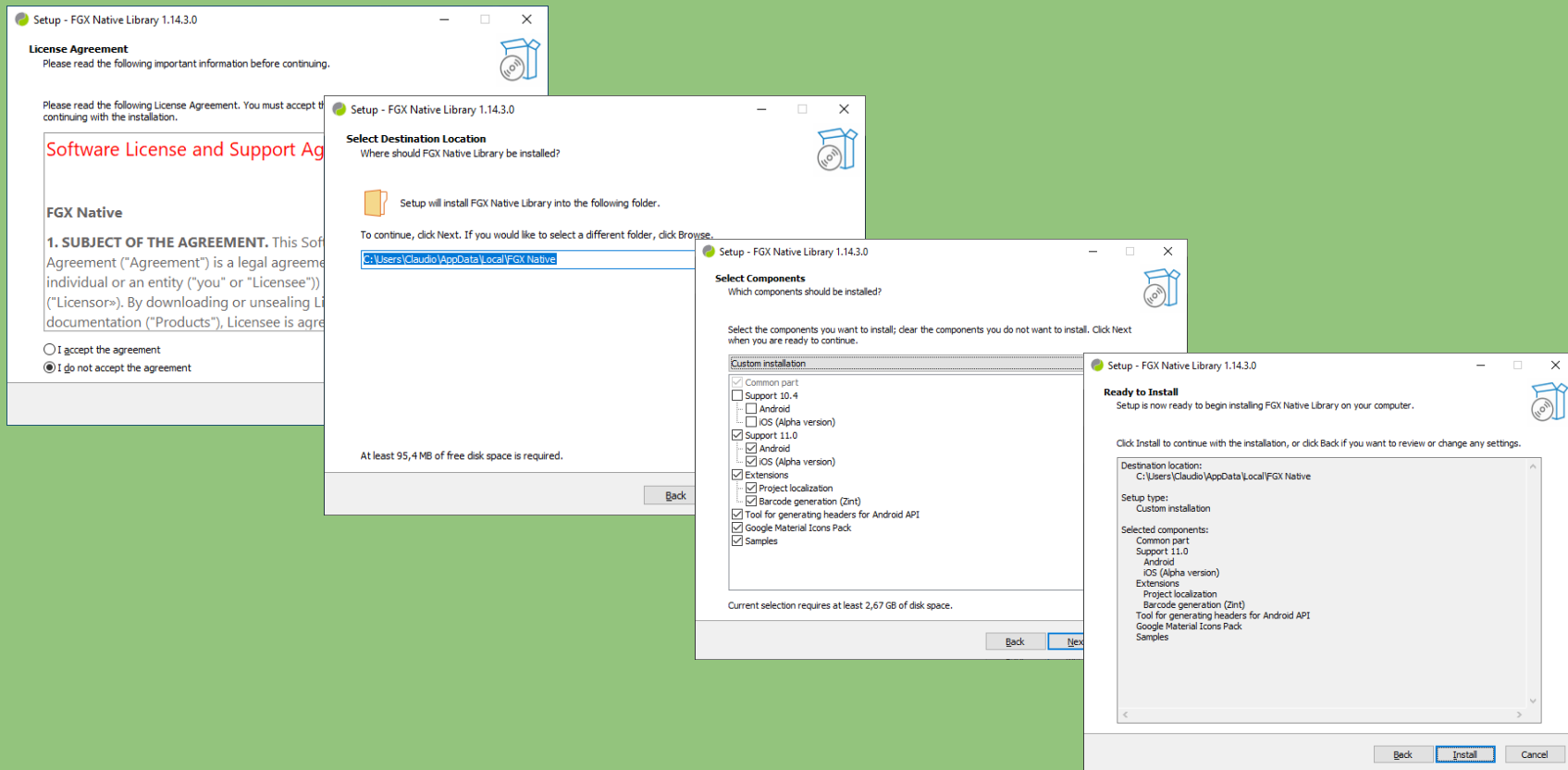
FGX Native



- (-) Più complesso aggiungere una nuova piattaforma
- (-) Implementazione separata e dedicata del controllo per ogni piattaforma
- (+) Controlli nativi
- (+) UI e animazioni ad alte prestazioni

Installazione

Installazione in Delphi 11



Evoluzione della libreria e dei componenti

Dimensionamento automatico dei componenti

- Dimensionamento automatico dei componenti implementato a basso livello
- Esposto tramite la proprietà Autosize la quale ha la responsabilità di calcolare automaticamente il dimensionamento dei componenti in termini di altezza (height) e/o larghezza (width)
- Al momento abilitato sui componenti TfgSwitch, TfgTrackBar e TfgNavigationBar
- Android e iOS utilizzano diversi tipi di carattere, dimensioni del testo e dimensione dei componenti. Questa proprietà aiuta a rendere universali le nostre applicazioni

Dimensionamento automatico dei componenti

The image shows the Qt Designer interface with three overlapping panels illustrating automatic dimensioning settings for a **TfgNavigationBar** component.

Object Inspector (Top Left): Shows the **fgTrackBar1** component. The **Properties** tab is active, and the **Autosize** property is checked, with a red box highlighting it. The **Width** and **Height** properties are also visible.

Properties Panel (Middle Left): Shows the **Properties** tab for the **TfgNavigationBarButtons** component. The **Autosize** property is checked, and the **Width** and **Height** properties are also visible.

Properties Panel (Bottom Left): Shows the **Properties** tab for the **TfgNavigationBar** component. The **Autosize** property is checked, and the **Width** and **Height** properties are also visible.

Design View (Right): Shows the **TfgNavigationBar** component in the design view. The component is a green bar with the text **TfgNavigationBar** and a status bar at the bottom showing the time **12:30**. Below the bar, a text box contains the message: "The NavigationBar height is adjusted based on the height of the system status bar."

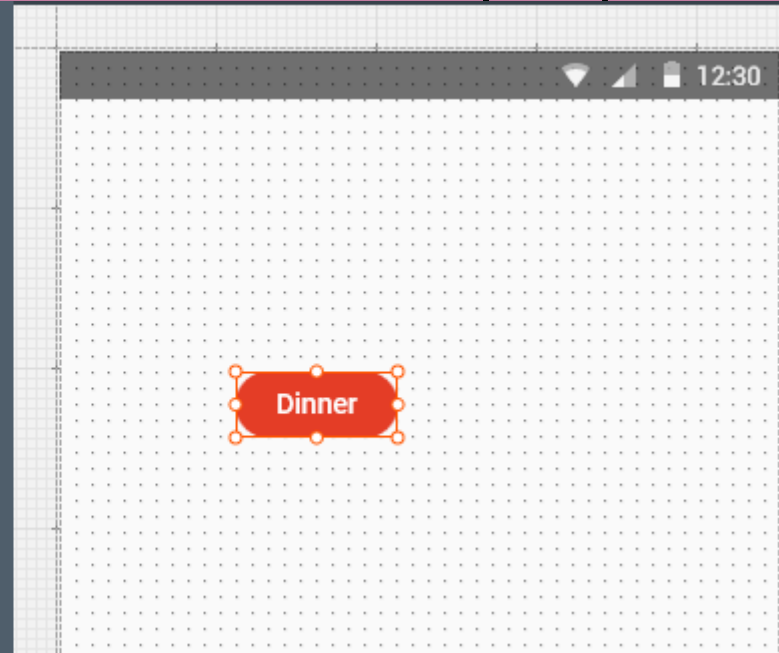
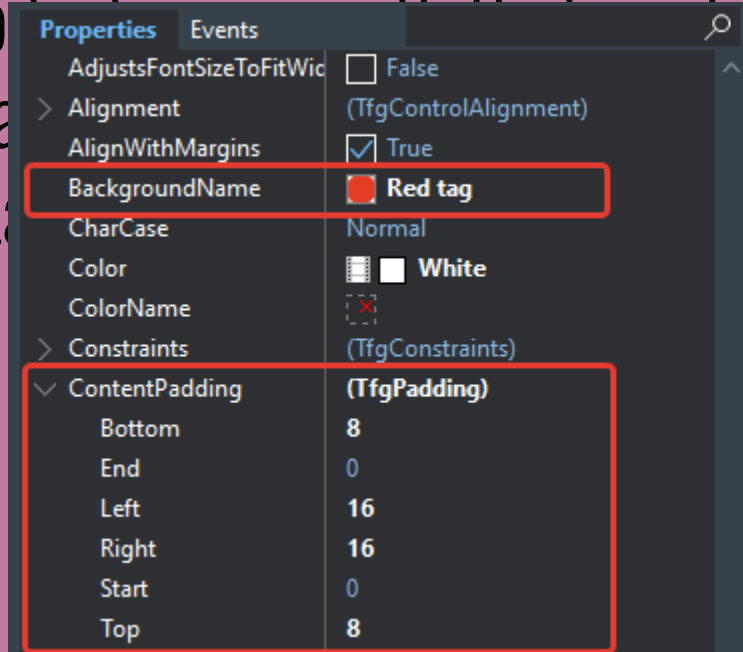
TfgLabel

→ **TfgLabel**: aggiunta la possibilità di creare dei reimpimenti

in modo

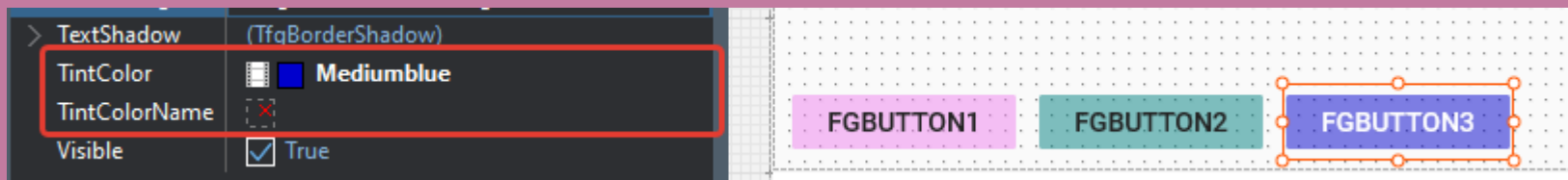
Prima

(TfgLabel)



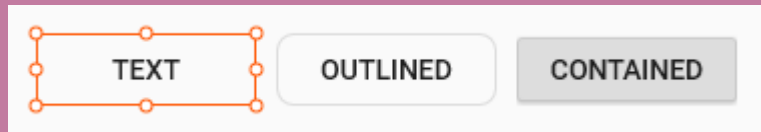
TfgButton

→ **TfgButton**: aggiunta la proprietà **TintColor** per personalizzare il colore dei pulsanti



TfgButton

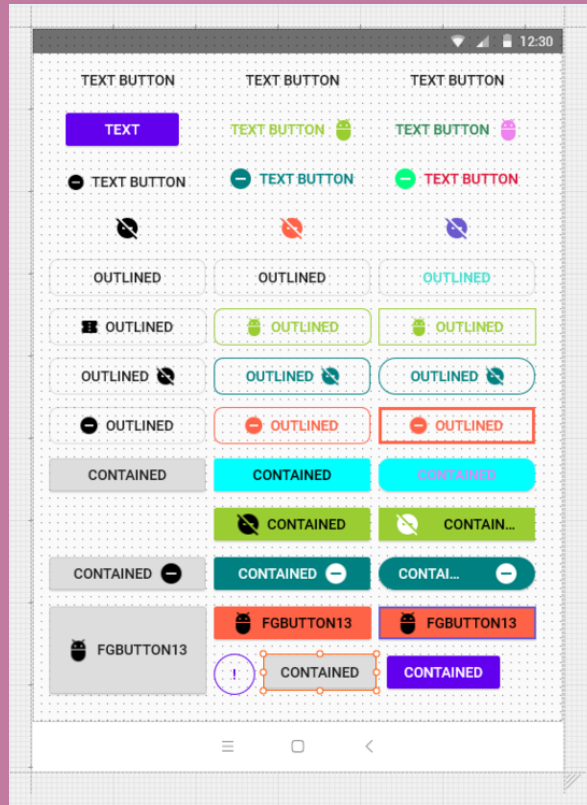
- Stili: TfgButton supporta 3 stili di base
 - **Text**: viene visualizzato solo il testo senza riempimento e bordi
 - **Outlined**: pulsante con testo e un contorno
 - **Contained**: pulsante con testo e bordo delineato
- La proprietà per settare gli stili si chiama: Appearance



TfgButton

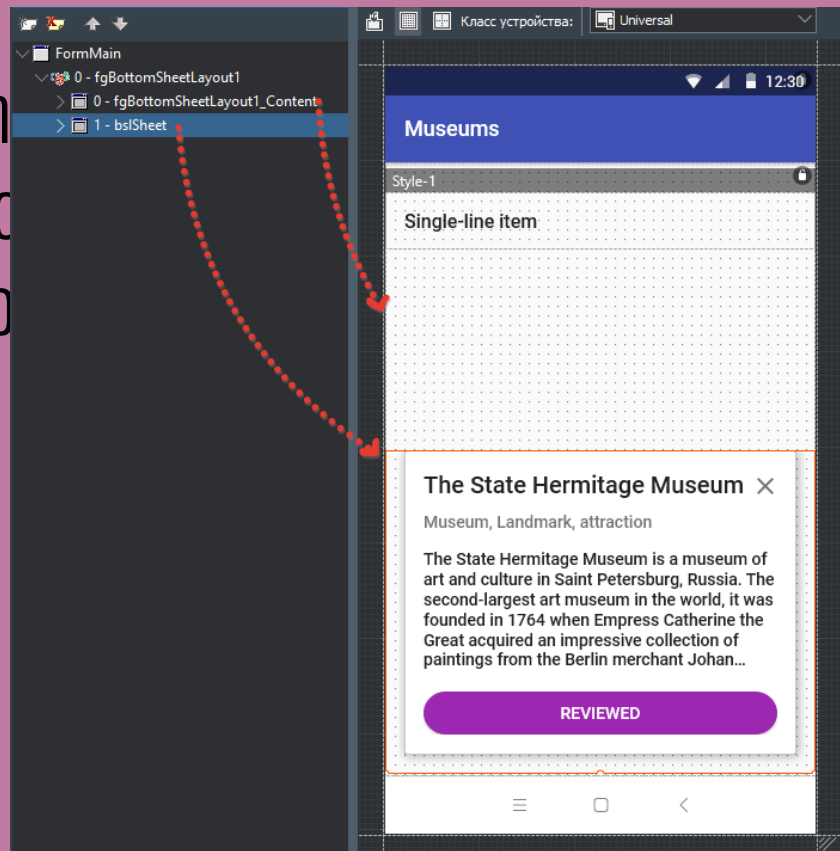
<ul style="list-style-type: none"> Appearance <ul style="list-style-type: none"> Icon <ul style="list-style-type: none"> Color ColorName Location Padding RenderMode Size Text <ul style="list-style-type: none"> Font FontColor FontColorName Shadow 	Text (TfgButtonText) <ul style="list-style-type: none"> Color ColorName Location Padding RenderMode Size Text Font FontColor FontColorName Shadow 	<ul style="list-style-type: none"> Appearance <ul style="list-style-type: none"> Border <ul style="list-style-type: none"> Color ColorName CornerRadius Thickness Icon <ul style="list-style-type: none"> Color ColorName Location Padding RenderMode Size Text <ul style="list-style-type: none"> Font FontColor FontColorName Shadow 	Outlined (TfgButtonBorder) <ul style="list-style-type: none"> Color ColorName CornerRadius Thickness Icon <ul style="list-style-type: none"> Color ColorName Location Padding RenderMode Size Text <ul style="list-style-type: none"> Font FontColor FontColorName Shadow 	<ul style="list-style-type: none"> Appearance <ul style="list-style-type: none"> Background <ul style="list-style-type: none"> FillColor FillColorName Border <ul style="list-style-type: none"> Color ColorName CornerRadius Thickness Icon <ul style="list-style-type: none"> Color ColorName Location Padding RenderMode Size Text <ul style="list-style-type: none"> Font FontColor FontColorName Shadow 	Contained (TfgButtonBackground) <ul style="list-style-type: none"> FillColor FillColorName Border <ul style="list-style-type: none"> Color ColorName CornerRadius Thickness Icon <ul style="list-style-type: none"> Color ColorName Location Padding RenderMode Size Text <ul style="list-style-type: none"> Font FontColor FontColorName Shadow
---	--	--	---	--	--

TfgButton

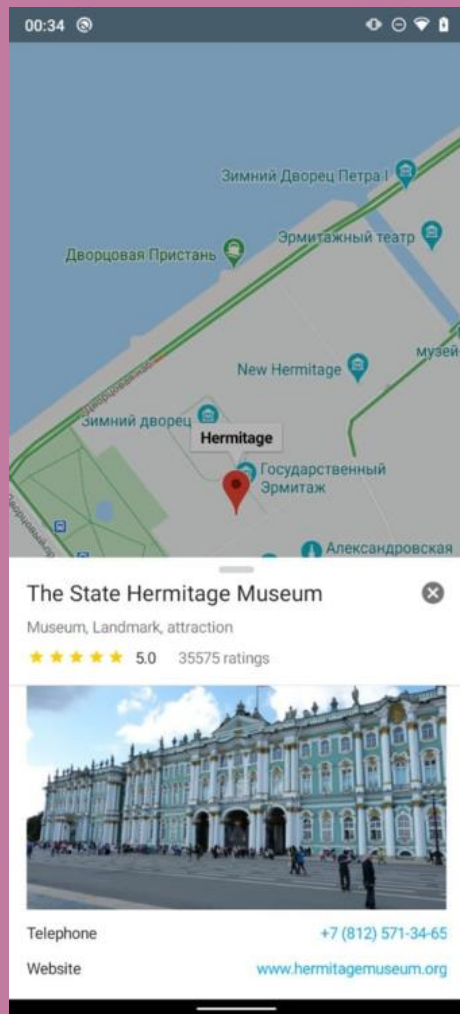


TfgBottomSheetLayout

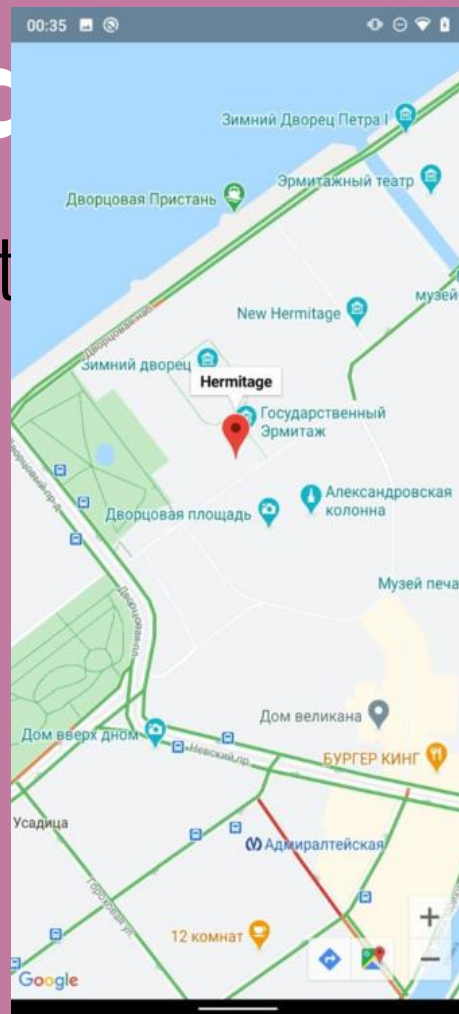
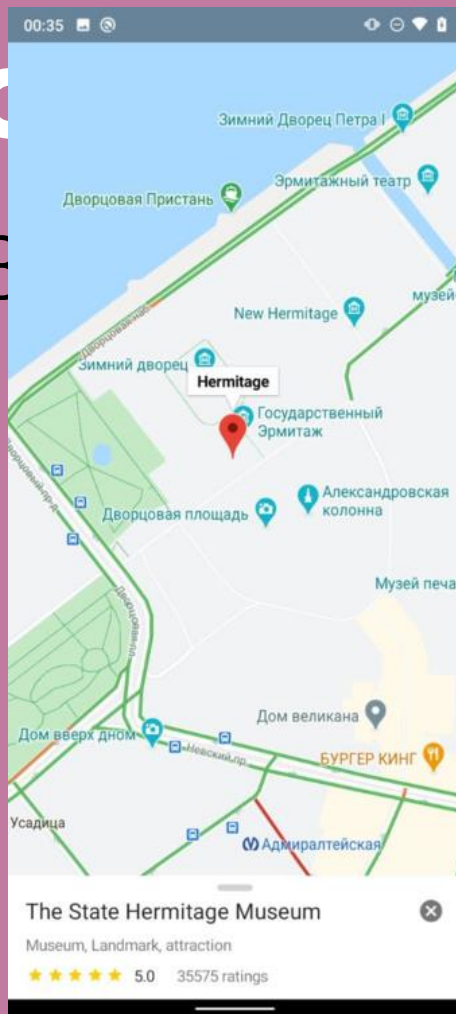
- Questo componente visualizza scorrimento
- La modalita contestuali



te la
un pannello a
armo
azione di menu

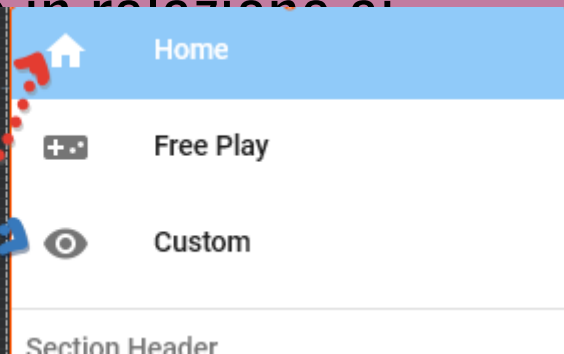
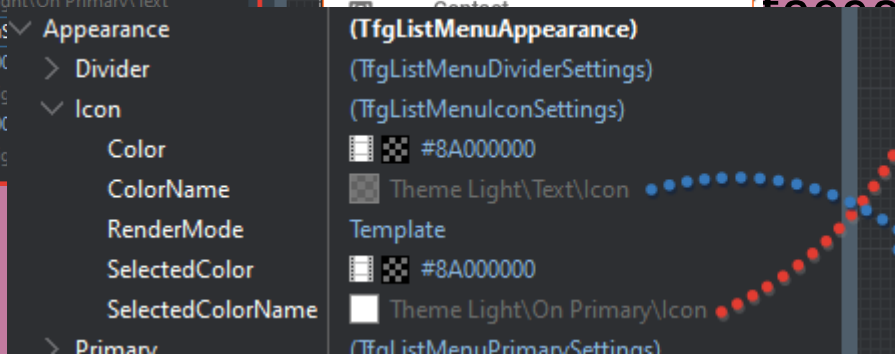
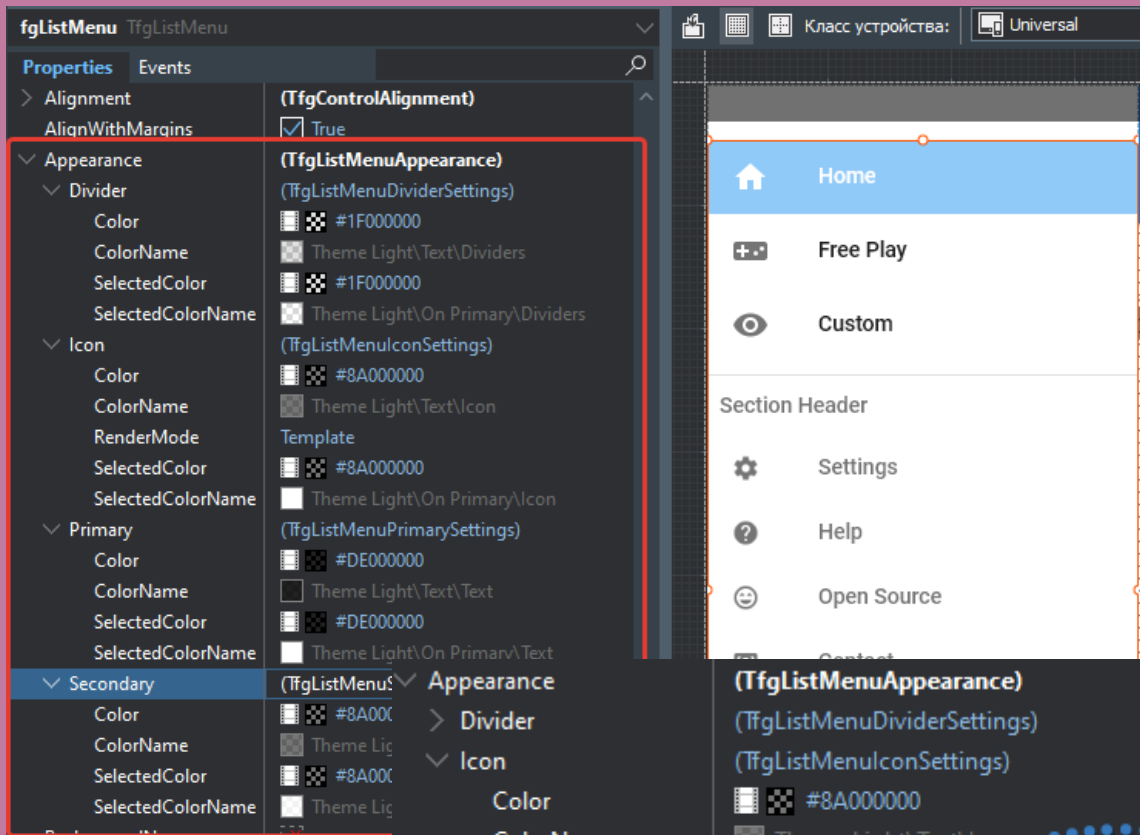


otto
ente fgE
etLo
ut può t



Clipboard

- Implementato un nuovo servizio multiplatforma per lavorare con la clipboard:
`FGX.Clipboard.TfgClipboardService`
- Essendo un servizio la gestione è cross application

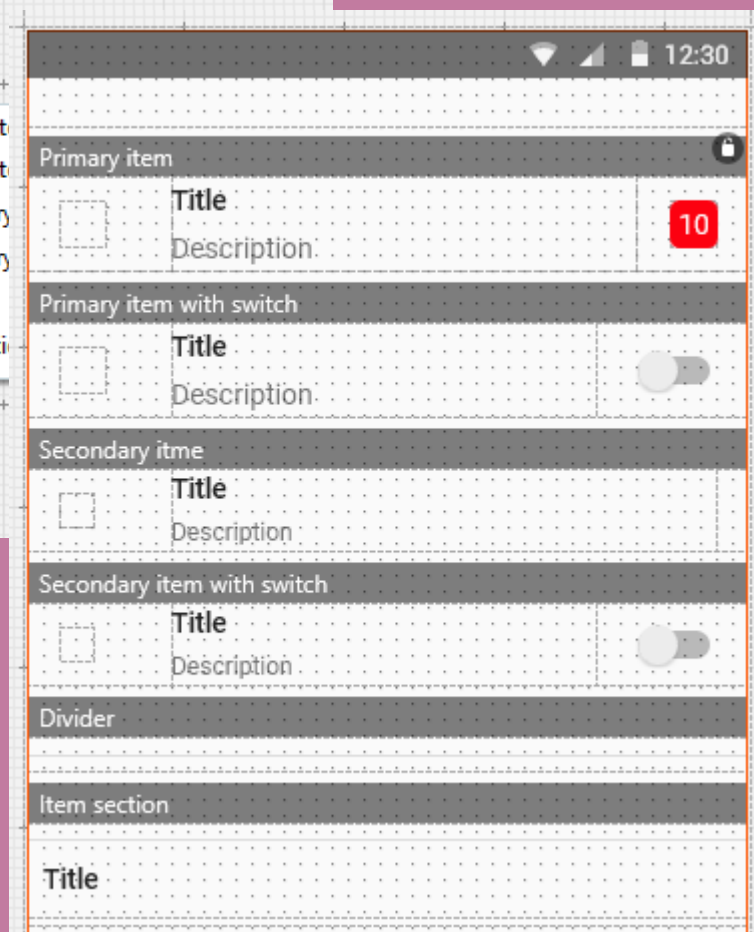
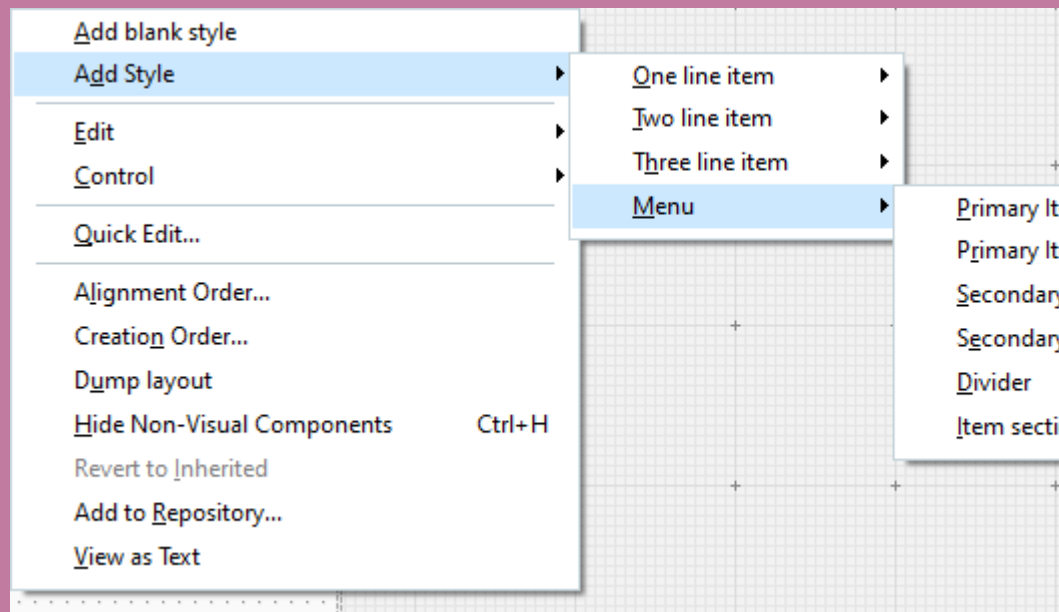


Menu

le versioni ma con un
colori

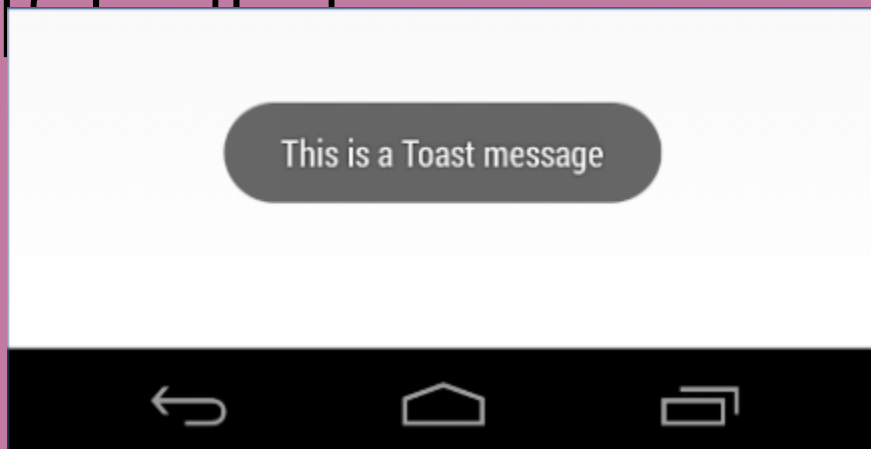
er una migliore
segno delle icone

tegg in relazione ai



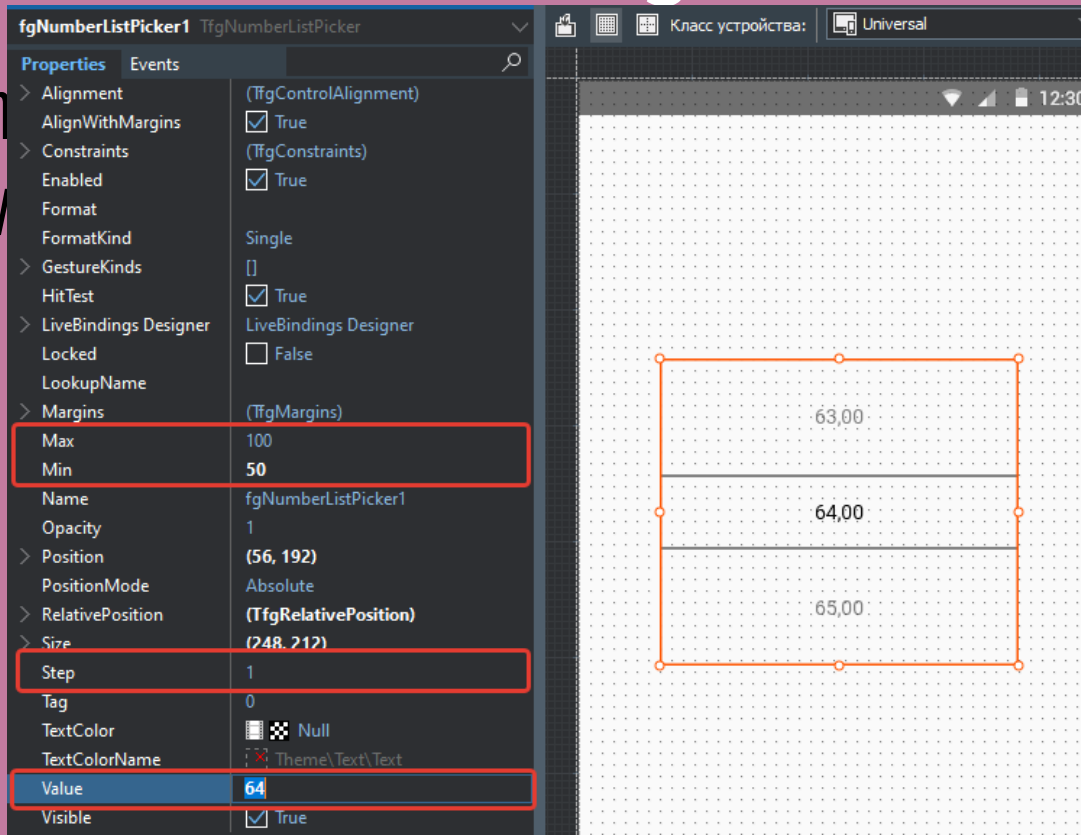
TfgToast

- Messaggio di notifica multiplatforma (in iOS sono gli avvisi popup)
- Per creare un nuovo messaggio toast utilizzare la nuova architettura TfgToastFramework



TfgVirtualListPicker e TfgNumberListPicker

→ Componente
elenco valori



valore da un

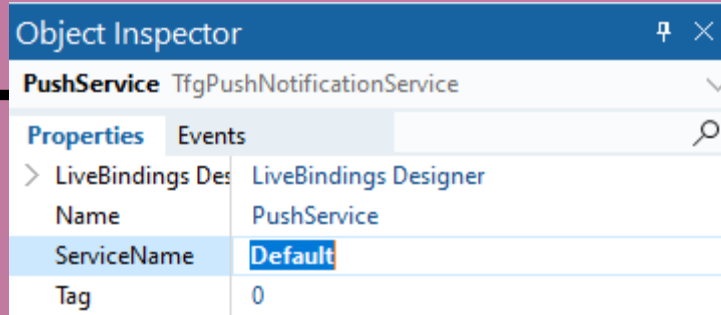
Notifiche Push

- Aggiunto un nuovo componente per la gestione delle notifiche Push multiplatforma:
`TfgPushNotificationService`
- E' un wrapper del servizio push esposto dalla RTL che semplifica l'attività di gestione delle notifiche stesse

Notifiche Push

- Il componente `TfgPushNotificationService` lavora sia con Android che con iOS. Per Android utilizza Firebase mentre per iOS il servizio Apple Push Notification Service
- La configurazione di base del componente (dopo aver configurato l'app nei relativi servizi di notifica):
 - Configurazione della proprietà `ServiceName`
 - Gestire l'evento `OnPushNotificationReceived`

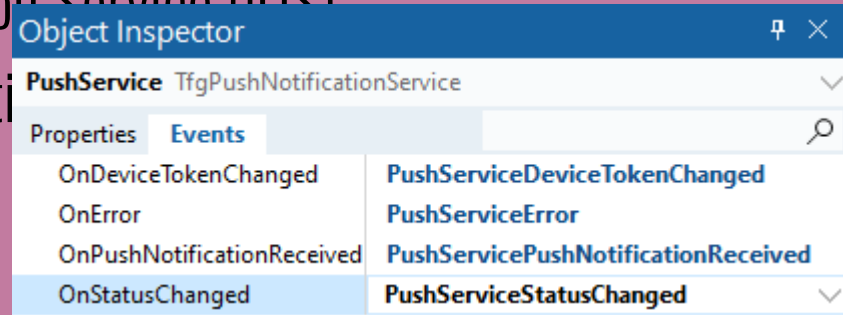
Notifiche Push



con il valore Default, può, al
guenti valori

→ **aps:** Apple Push Notification Service (iOS)

→ **Default:** selezione automatica
piattaforma



TfgPhotoPicker

- Componente multiplatforma per ottenere una foto dalla fotocamera o una o più foto dalla galleria
- Si utilizza tramite il factory: **TfgPickerPhotoFactory** che espone tre metodi
 - **PickPhotoFromCamera**: recupera una nuova foto dalla fotocamera
 - **PickPhotoFromLibrary**: recupera una foto dalla libreria
 - **PickPhotosFromLibrary**: recupera una o più immagini dalla libreria
- Su Android è necessario abilitare l'opzione «Secure File Sharing»

Animazioni

- Il supporto di FGX alle animazioni è stato notevolmente ampliato!
- Tutta la logica delle animazioni di basa su un gestore comune delle animazioni: `TfgControl.AnimationManager`
- `TfgControl.AnimationManager` è il responsabile della creazione/esecuzione/distruzione di una animazione

Animazioni

```
uses  
    FGX.Animation;  
  
var Animation := Button.AnimationManager.AddOpacityAnimation(0 {start opacity}, 1 {finish opacity}, 1000 {msec});  
Animation.Start;
```

```
uses  
    FGX.Animaton.Helpers;  
  
Button.AnimationManager.AddOpacityAnimation(0 {start opacity}, 1 {finish opacity}, 1000 {msec})  
    .SetName('fade-in');  
  
Button.AnimationManager['fade-in'].Start;
```

```
uses  
    FGX.Animation.Types, FGX.Animation.Helpers;  
  
Button.AnimationManager.AddOpacityAnimation(0 {start opacity}, 1 {finish opacity}, 1000 {msec})  
    .AddOption(TfgAnimationOption.ReleaseAnimationOnFinish)  
    .Start;
```

Animazioni

- **AnimationManager** supporta 5 tipi di animazioni base:
 - **Opacity**: per la gestione della “trasparenza” del controllo
 - **Bounds**: per la gestione della posizione e delle dimensioni
 - **Scale**: per la gestione della “scala” del componente
 - **Rotation**: per la gestione dell’angolo di rotazione
 - **Translate**: per gestire un offset del componente in relazione alla sua posizione corrente

Animazioni di gruppo

→ Per creare animazioni complesse è possibile utilizzare la animazioni di gruppo il cui compito è raggruppare più animazioni e stabilire le regole per il loro lancio

```
var
    AnimationGroup: TfgAnimationGroup;

AnimationGroup := LForm.AnimationManager.AddGroupAnimation
    .Add(LForm.AnimationManager.CreateScaleAnimation(1.125, 1, 1.125, 1))
    .Add(LForm.AnimationManager.CreateOpacityAnimation(0, 1));
```

```
Image.AnimationManager.AddGroupAnimation
    .Add(Image.AnimationManager.AddTranslationAnimation(TPointF.Create(-16, 0), TPointF.Create(16, 0), 1000))
    .Add(Image.AnimationManager.AddRotationAnimation(0, 360, 1000 )
        .SetRunOrder(TfgAnimationRunOrder.AfterPrevious)) .Start
    ;
```

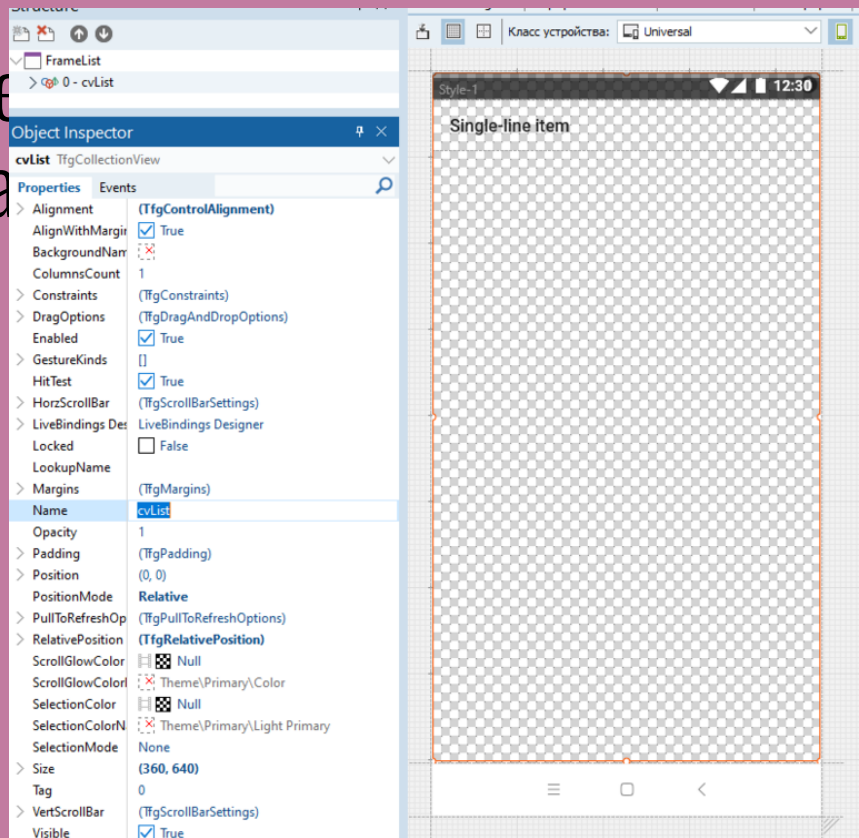
Animazioni: template

- Importando la unit `FGX.Animation.Templates` è possibile utilizzare dei modelli già impostati.
- Ad esempio se voglio «agitare» (shake) una immagine:
 - `Image.Shake;`
- Se voglio personalizzare i parametri:
 - `Image.Shake(TfgShakeAnimationParameter.Default.SetDuration(2000).SetRepeatCount(10));`

TfgPopup

→ Componente
discesa legale

menu a

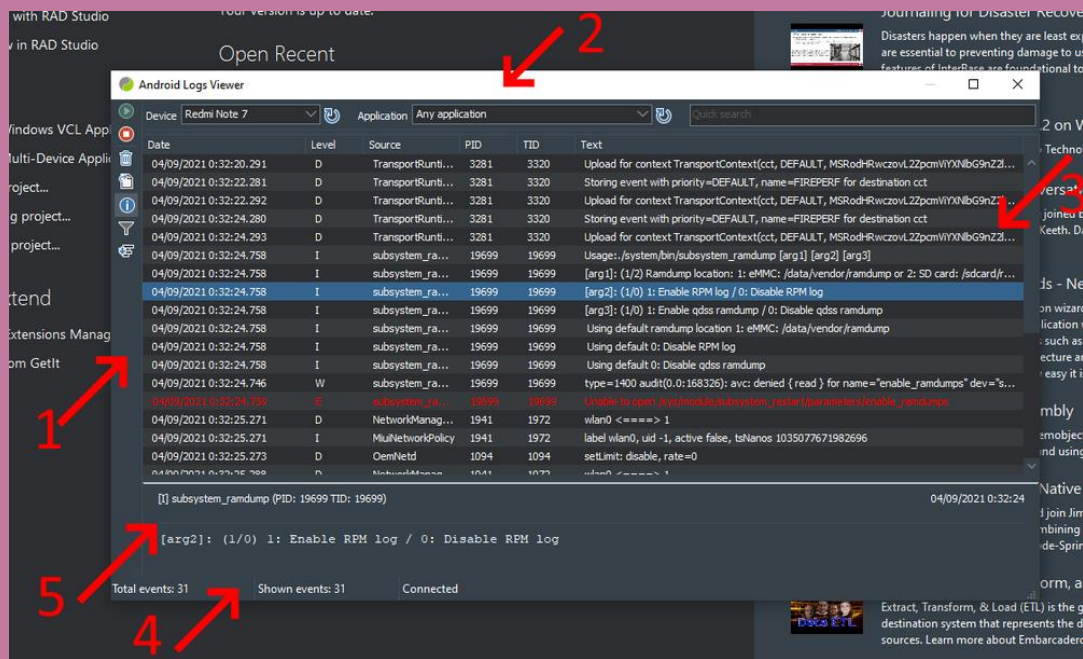


TfgLog

- La classe TfgLog è una classe multiplatforma che permette di effettuare il log applicativo
- Il log applicativo è uno strumento che si affianca al debugger
 -e diciamo che lo aiuta in quanto alcune volte il debugger fa i capricci ;-)
- I livelli di registrazione del log sono:
 - TfgLogLevel = (Trace, Debug, Info, Warning, Error, Fatal)

TfgLog - Android

→ FGX Android Log Viewer è un tool grafico che permette di accedere al log senza utilizzare logcat o installare Android Studio



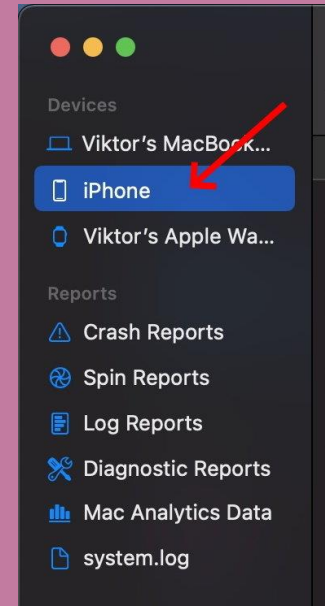
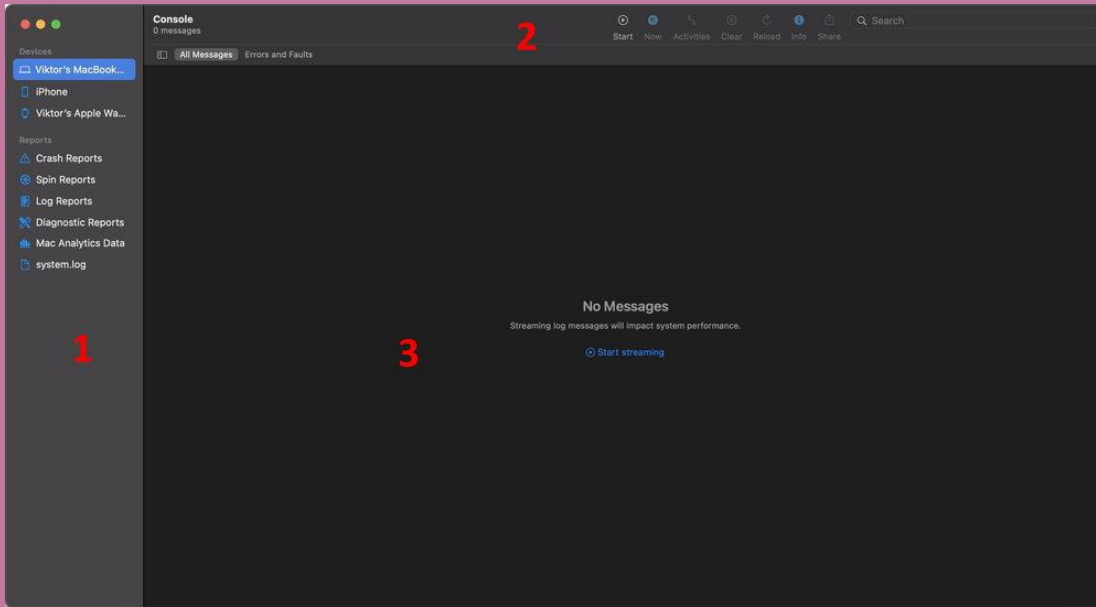
- 1: barra degli strumenti utili per avviare/interrompere acquisire il log, cancellare l'elenco, esportarlo....
- 2: barra degli strumenti superior utilizzata per applicare i filtri
- 3: finestra del log
- 4: barra di stato
- 5: dettaglio dell'evento selezionato

TfgLog iOS

→ La visualizzazione
dalla console

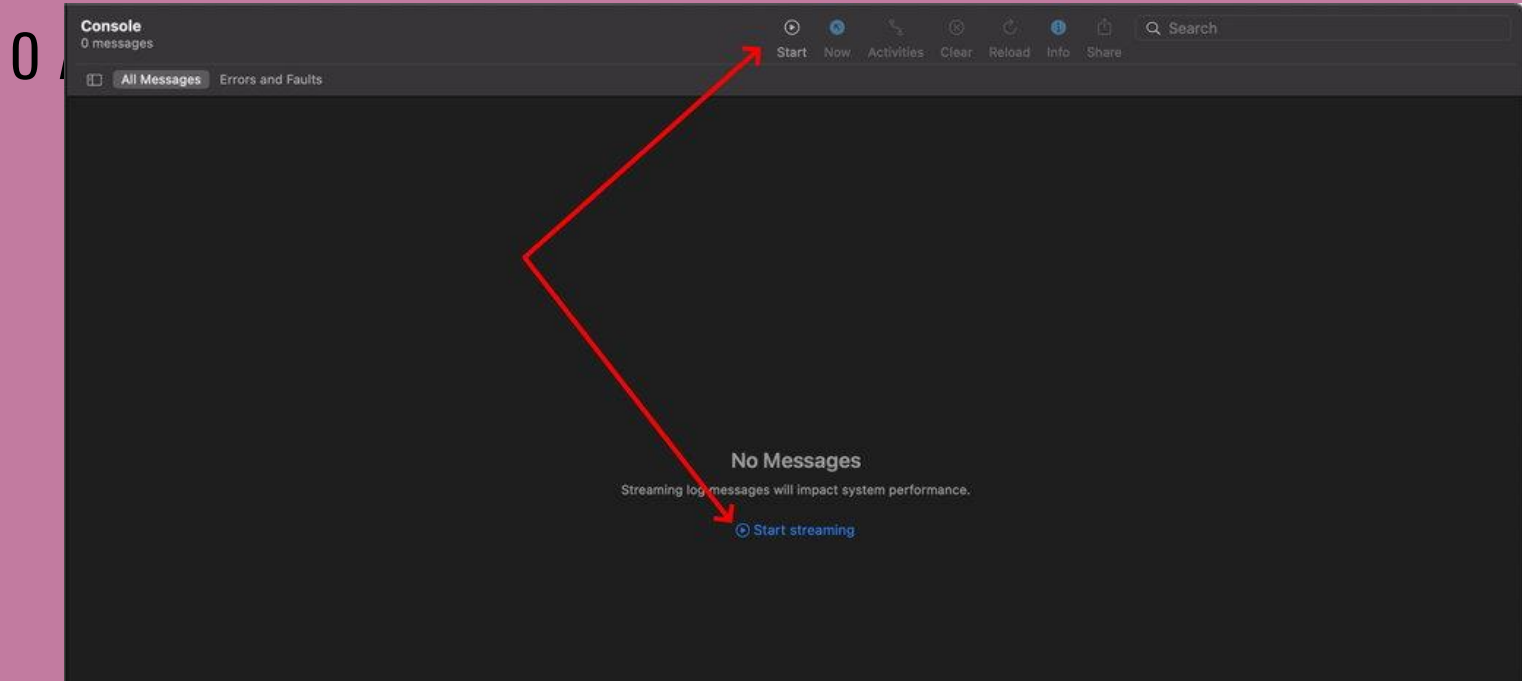


TfgLog iOS

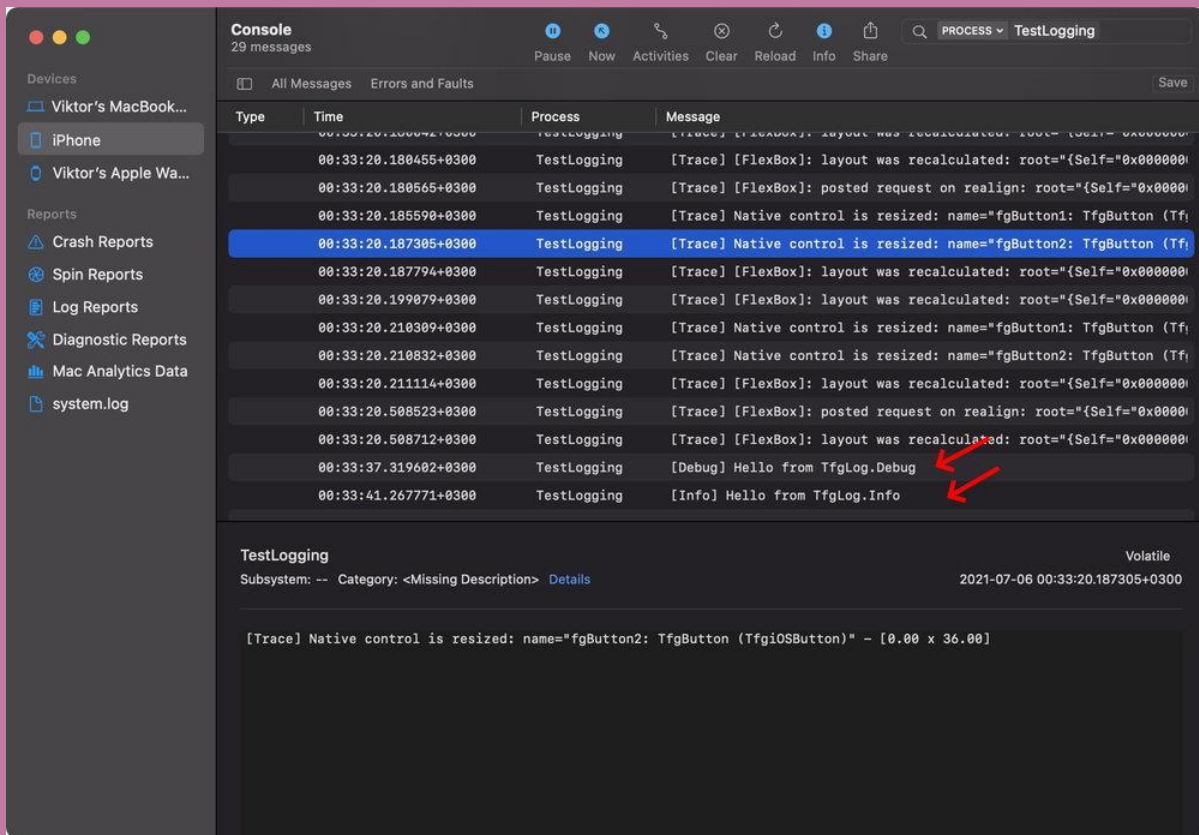


TfgLog iOS

→ L'avvio del log viene effettuato premendo il pulsante Start



TfgLog iOS



Console
29 messages

Pause Now Activities Clear Reload Info Share

PROCESS TestLogging

All Messages Errors and Faults Save

Type	Time	Process	Message
	00:33:20.180455+0300	TestLogging	[Trace] [FlexBox]: layout was recalculated: root="{Self="0x00000000"
	00:33:20.180455+0300	TestLogging	[Trace] [FlexBox]: layout was recalculated: root="{Self="0x00000000"
	00:33:20.180565+0300	TestLogging	[Trace] [FlexBox]: posted request on realign: root="{Self="0x00000000"
	00:33:20.180565+0300	TestLogging	[Trace] [FlexBox]: posted request on realign: root="{Self="0x00000000"
	00:33:20.185590+0300	TestLogging	[Trace] Native control is resized: name="fgButton1: TfgButton (Tfg
	00:33:20.187305+0300	TestLogging	[Trace] Native control is resized: name="fgButton2: TfgButton (Tfg
	00:33:20.187794+0300	TestLogging	[Trace] [FlexBox]: layout was recalculated: root="{Self="0x00000000"
	00:33:20.199079+0300	TestLogging	[Trace] [FlexBox]: layout was recalculated: root="{Self="0x00000000"
	00:33:20.210309+0300	TestLogging	[Trace] Native control is resized: name="fgButton1: TfgButton (Tfg
	00:33:20.210309+0300	TestLogging	[Trace] Native control is resized: name="fgButton1: TfgButton (Tfg
	00:33:20.210832+0300	TestLogging	[Trace] Native control is resized: name="fgButton2: TfgButton (Tfg
	00:33:20.210832+0300	TestLogging	[Trace] Native control is resized: name="fgButton2: TfgButton (Tfg
	00:33:20.211114+0300	TestLogging	[Trace] [FlexBox]: layout was recalculated: root="{Self="0x00000000"
	00:33:20.508523+0300	TestLogging	[Trace] [FlexBox]: layout was recalculated: root="{Self="0x00000000"
	00:33:20.508712+0300	TestLogging	[Trace] [FlexBox]: posted request on realign: root="{Self="0x00000000"
	00:33:20.508712+0300	TestLogging	[Trace] [FlexBox]: layout was recalculated: root="{Self="0x00000000"
	00:33:37.319602+0300	TestLogging	[Debug] Hello from TfgLog.Debug
	00:33:41.267771+0300	TestLogging	[Info] Hello from TfgLog.Info

TestLogging
Subsystem: -- Category: <Missing Description> Details
2021-07-06 00:33:20.187305+0300

[Trace] Native control is resized: name="fgButton2: TfgButton (TfgiOSButton)" - [0.00 x 36.00]

FGX con

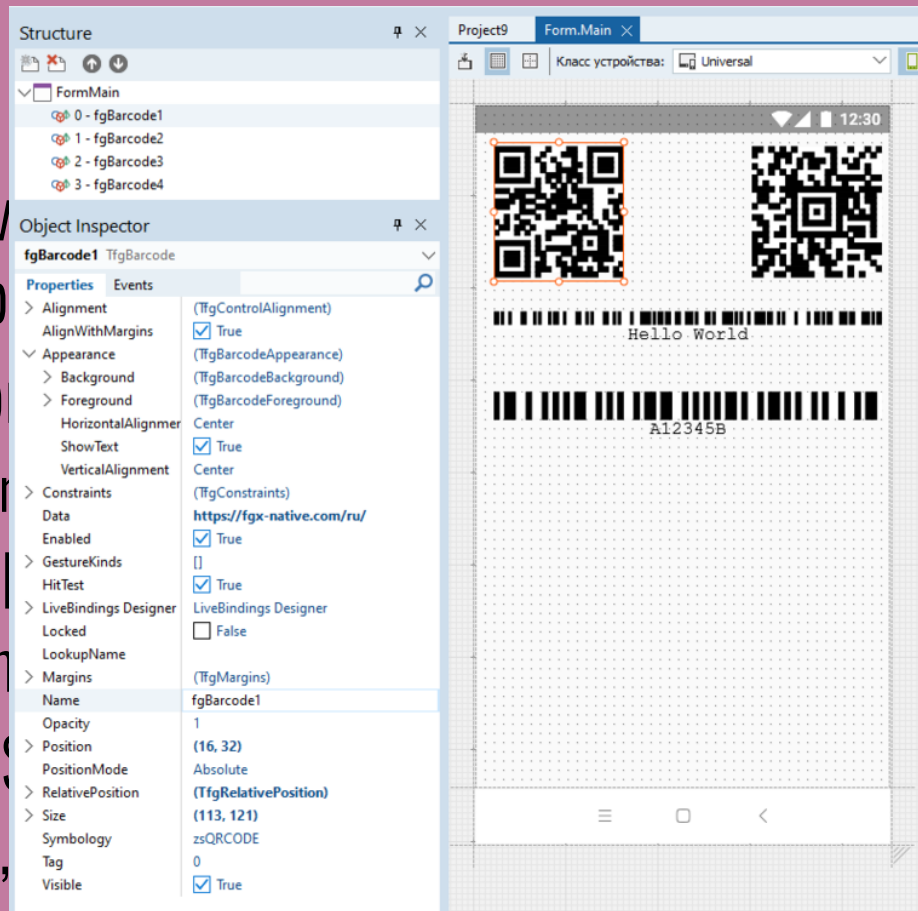
→ Il team di sviluppo
a librerie open source

→ La prima libreria

→ Ma ne arr

→ Zint è una libreria
utilizza si che

→ TfgBarcodeS
forocamera,



en source

nsioni native FGX

oper FGX che la

con la

n codice a barre

TfgVirtualPagerLayout

- **TfgVirtualPagerLayout**: è un componente di base che permette di caricare/scaricare controlli (TfgControl).
- **OnLoadPage** è l'evento che viene invocato quando c'è da caricare una nuova pagina. Per pagina si intende qualsiasi componente visuale
- **OnUnloadPage** è l'evento che viene invocato quando una pagina viene «scaricata»
- **OnGetPageCount** è l'evento che viene invocato per determinare il numero delle pagine

TfgVirtualPagerLayout

- In caso di pagine “dinamiche”, dove quindi il count impostato all’inizio può variare si possono invocare i seguenti metodi:
 - **Reload** : metodo che serve per notificare al componente che l’elenco delle pagine è stato modificato
 - **NotifyPageInserted** : notifica al componente che è stata inserita una nuova pagina nell’indice specificato
 - **NotifyPageRemoved** : notifica al componente che la pagina specificata come indice deve essere rimossa

Lottie file

- Dalla versione 1.13.5 FGX ha il pieno supporto ai lottie file
- Lottie è una libreria open source di Airbnb per l'esecuzione di immagini animate create con Adobe After Effects
- Lottie file è un file di testo in JSON contente le istruzioni per creare l'immagine vettoriale con animazione
- **TfgLottieImage**: è il componente che supporta questo tipo di file
- Da questo link <https://lottiefiles.com/> è possibile cercare e scaricare immagini in questo formato

Servizi di autenticazione

- In FGX sono stati implementati e verranno aggiunti i client di accesso ai seguenti servizi di autenticazione
 - ID Apple e Apple Pay
 - Facebook login
 - Google Sign-in e Google Pay
 - VK Login
 - ...

Habit Builder

→ Prototipazione di

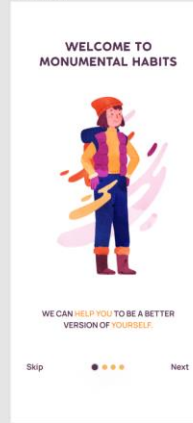
→ <https://www.pixelskit.com/login>

Onboarding

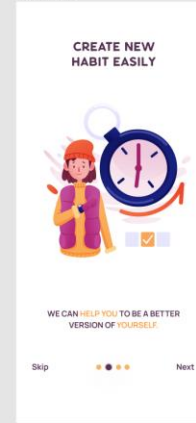
Splash Screen



Introduction 1



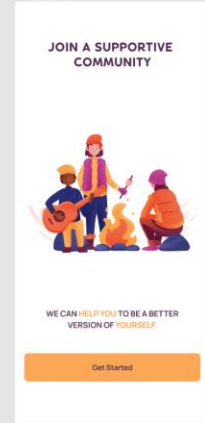
Introduction 2



Introduction 3



Introduction 4

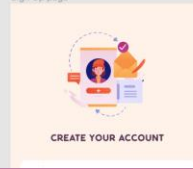


Sign up and Log in

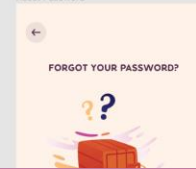
Login Page



Sign Up page



Reset Password



Futuro...

Roadmap

- FGX è una libreria in costante evoluzione! Escono nuove versioni ogni 2-3 settimane. Queste versioni correggono bug ma soprattutto aggiungono funzionalità/componenti nuovi alla libreria
 - Finalizzazione della libreria su iOS
 - Frame
 - Styles
 - Authentication services
 - Swipe per TfgCollectionView
 - Charts....

Link

→ Pagina ufficiale del prodotto

→ <https://fgx-native.com/en/>

→ Documentazione del prodotto

→ <https://fgx-native.com/en/introduction.html>

→ Forum

→ <https://forum.fgx-native.com>



THANK YOU