

Fabio Falzoi

Tech lead

Qt Framework

ISIS Gobetti Volta

GENNAIO 2021

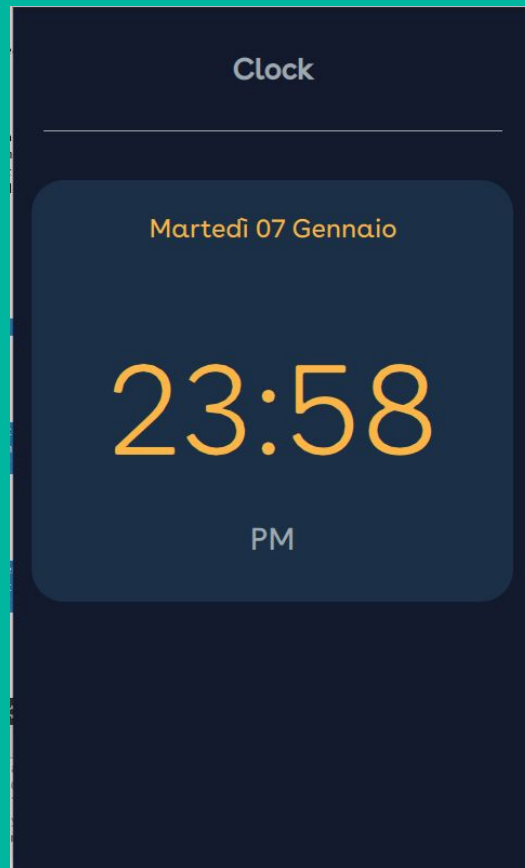
develer

LEZIONE 1 – PROGRAMMA

- Il ruolo del designer e dello sviluppatore nel processo di produzione del software
- Progetto “Clock” a partire da zero
- Uso di font personalizzati
- Montaggio della UI a partire da asset grafici
- Integrazione C++ e QML

Obiettivo della lezione 1

Orologio Digitale



Designer → **definisce** le specifiche della UI/UX

Sviluppatore → **implementa** le specifiche della UI/UX



Processo di sviluppo del software nel mondo reale

Designer → **definisce** le specifiche della UI/UX

Sviluppatore → **implementa** le specifiche della UI/UX



Progetto “Clock” a partire da zero

Come creare un progetto Qt/QML a partire da zero?

Progetto "Clock" a partire da zero

da QtCreator:

- File -> New File or Project
- Application (Qt Quick)
- Qt Quick Application (Empty)
- Scegliere directory e nome progetto

Struttura del progetto:

- file ".pro" -> descrizione del progetto Qt Quick
- Sources/main.cpp -> codice sorgente C++ contenente la funzione *main*
- Resources -> contiene le altre risorse del progetto, oltre al codice sorgente (descrizione UI tramite file QML, asset grafici, ecc.)
- main.qml -> descrizione UI in qml



Tutta la nostra UI sarà
contenuta qui!

Progetto “Clock” a partire da zero

Specifiche della UI

Descrizione tramite Adobe XD

<https://adobe.ly/2M4BS0K>

Per questa prima lezione, eviteremo di costruire i 2
button in basso *Timer* e *Alarm*

Progetto "Clock" a partire da zero

Utilizzo di Adobe XD

Descrizione tramite Adobe XD

Il secondo button "*Visualizza specifiche*" vi consente di cliccare su ogni singolo elemento per ottenere informazioni su dimensioni, colore, posizionamento, font, ecc.



Uso di font personalizzati

Scaricate il materiale per il corso

<https://github.com/develersrl/isis-gobetti-volta>

Uso di font personalizzati

- Spostare la directory contenente i font nella directory principale del progetto
- Aggiungere i font come risorsa del progetto (Resources -> "/" -> Add Existing Directory -> ...)
- Utilizzare il tipo QML [FontLoader](#)

```
import QtQuick 2.12
import QtQuick.Window 2.12
```

```
Window {
    width: 640
    height: 480
    visible: true
```

```
FontLoader {
    id: buenosAiresRegularFont
    source: "BuenosAires/BuenosAires-Regular.otf"
}
```

```
Rectangle {
    anchors.fill: parent
    color: "blue"
```

```
Text {
    text: "Hello World"
    color: "white"
    font.family: buenosAiresRegularFont.name
    font.pixelSize: 30
    anchors.centerIn: parent
}
```

```
}
```

Montaggio della UI a partire da asset grafici

1. Utilizzate Adobe XD per capire dimensioni e colore della finestra e del rettangolo di sfondo
2. Caricate il font custom *BuenosAires* e utilizzatelo per mostrare la stringa "Hello World"



Montaggio della UI a partire da asset grafici

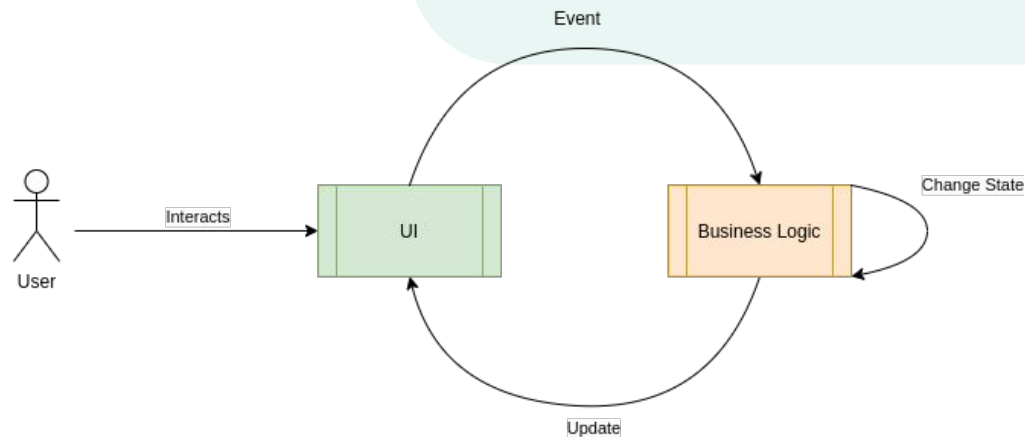
1. Completare l'interfaccia grafica come specificato su Adobe XD
2. **NON** montare i 2 button *Timer* e *Alarm*



Integrazione C++ e QML

Schema a blocchi

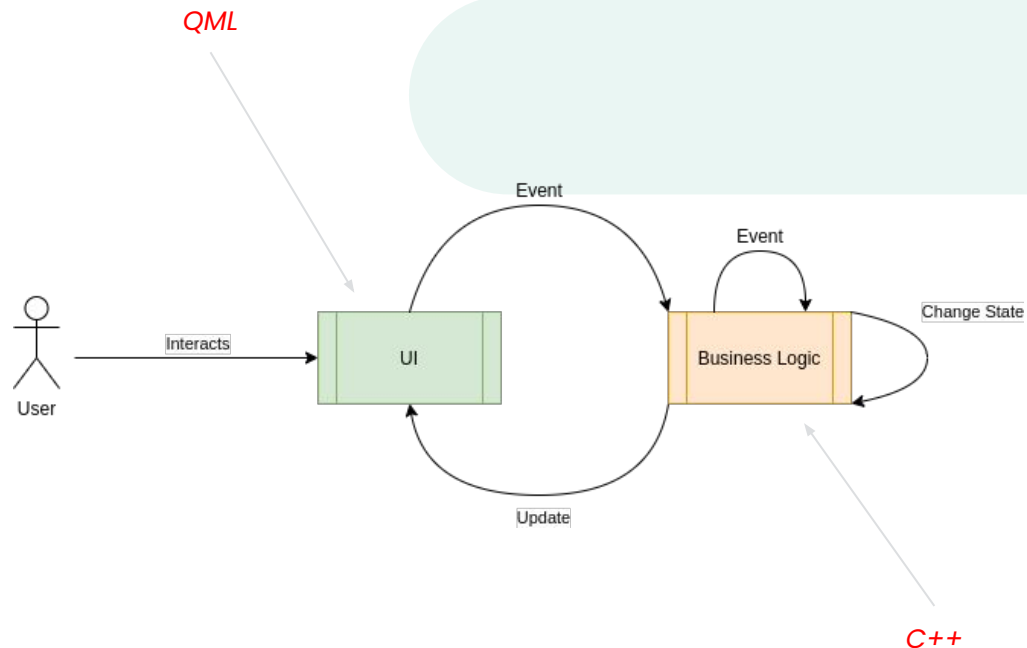
1. Interazione utente
2. Evento da UI
3. Cambio Stato
4. Aggiornamento UI



Integrazione C++ e QML

Schema a blocchi

1. Interazione utente
2. Evento da UI
3. Evento "interno"
4. Cambio Stato
5. Aggiornamento UI



1. Creazione classe C++ *Clock*

Sources (clic destro) -> Add New -> C/C++ -> C++ Class

- Class Name: *Clock*
- Base Class: *QObject*
- *clock.cpp* e *clock.h*

Vediamo insieme il codice autogenerato

2. QTimer e QDateTime

QTimer una classe che fornisce timer periodici (*repetitive*) e *single-shot*

```
// creazione del timer
timer = new QTimer(this);

// connessione del segnale timeout allo slot opportuno
QObject::connect(timer, SIGNAL(timeout()), this, SLOT(updateClock()));

// avvio del timer (il segnale timeout verrà emesso ogni secondo)
timer->start(1000);
```

2. QTimer e QDateTime

?

```
// creazione del timer  
timer = new QTimer(this);  
  
// connessione del segnale timeout allo slot opportuno  
QObject::connect(timer, SIGNAL(timeout()), this, SLOT(updateClock()));  
  
// avvio del timer (trigger ogni secondo)  
timer->start(1000);
```

?

2. QTimer e QDateTime

[QDateTime](#) una classe che permette di manipolare date e timestamp (aggiungendo e/o sottraendo intervalli temporali, ottenendo stringhe in formati specifici, ecc.)

```
// salva la data e l'orario corrente nella variabile current  
QDateTime current = QDateTime::currentDateTime();  
  
// aggiungi un secondo a current  
current = current.addSecs(1);
```

3. Esposizione Clock a QML

In *clock.h*:

```
Q_PROPERTY(QDateTime dateTime READ currentDateTime NOTIFY dateChanged)
```

Tipo e nome della
property esportata

Metodo C++ che viene
eseguito ad ogni
lettura, da QML, della
proprietà

Segnale emesso ad
ogni modifica della
proprietà, per forzare
la UI a rileggere il
valore

3. Esposizione Clock a QML

In *main.cpp*, nella funzione *main*:

```
QQmlApplicationEngine engine;  
  
// ...  
  
// Create and expose a custom object to QML  
Clock c;  
engine.rootContext()->setContextProperty("clock", &obj);  
  
engine.load(url);
```

!!!



3. Esposizione Clock a QML

In *main.qml* posso valorizzare la proprietà *text* di un oggetto QML con i valori ottenuti dall'oggetto C++:

```
text: clock.dateTime
```

Per formattare la data o l'ora corrente, è possibile utilizzare le API fornite da Qt e utilizzabili in QML:

- `Qt.formatDate(date, format) → stringa`
- `Qt.formatTime(date, format) → stringa`

Documentazione:

[Qt.formatDate](#)

[Qt.formatTime](#)



Riuscite a trovare le
specifiche per il formato?

Un orologio funzionante!



Bonus: test manuale orologio

1. Verificare che, trascorso il 59esimo minuto, l'ora si aggiorni
2. Verificare il passaggio AM -> PM dopo mezzogiorno e PM -> AM dopo mezzanotte
3. Verificare l'aggiornamento della data dopo la mezzanotte

Come velocizzare il nostro orologio?



CONTACTS

Fabio Falzoi

fabio@develer.com

Twitter: @Pippolo84

The logo for Develer, featuring the word "develer" in a white, lowercase, sans-serif font. A horizontal orange bar is positioned above the "e" in "develer".

www.develer.com