TECNICHE DI PROGRAMMAZIONE

05/03/19

Primo Programma su JavaFX.

Quando si crea un nuovo progetto JavaFX, in ‘Root-Type’ noi andiamo ad indicare quale è il contenitore che andrà a contenere tutta la nostra applicazione. Noi scegliamo sempre il ‘…BorderPane’.

Una volta creato il progetto ricordarsi che:

* Il metodo ‘Main non fa nulla se non lanciare il ‘Launch’ della classe madre ‘Application’
* La classe ‘Application prima o poi chiamerà il metodo start.
* Nel Main il programma legge il file ‘nome.fxml’ che restituisce un NODO radice di tipo BorderPane.
* Nodo radice usato per creare la nuova ‘Scene’ (c’è una differenza tra il nodo radice e la scena infatti nel caso di interfacce complicate il file fxml descrive solo una parte.

Nel Main bisogna cancellare i 2 numeri nella riga: Scene scene = new Scene ( root, 400 , 400 ).

I numeri sarebbero la dimensione della finestra.

In Scene Builder:

* Sulla sinistra è presente la libreria con tutti i nodi che si dividono in Continers (contenitori) , Controls ( di controllo ) ecc…
* Sono molto utili gli Hbox ( scatola orizzontale ) e i Vbox (scatola verticale ).
* Se combino un Vbox con un Hbox ottengo un GrindPane ( griglia ).
* I continetori hanno una gerarchia e possono essere annidiati.
* FlowPane è un contenitore che si riempie da sinistra verso destra ma quando raggiunge in bordo va a capo
* In Controls i piu importanti sono: Button , Label , TextField ( 1 riga ) , TextArea ( piu righe ) , ComboBox ( usato per creare il menu a tendina ) e gli Sliders .
* Sulla sinistra, in ‘Gluon’, tutti gli elementi in blu non sono da usare, mentre quelli in nero si possono usare.
* In generale non prendere mai degli elementi da ‘Gluon’!!

Ogni nodo ha una serie di proprietà che possono essere modificate sulla destra.

Per far inserire qualcosa all’utente servono 3 cose:

- Una casella di testo in cui l’utente inserisce il nome.

- Un bottone per confermare l’inserimento.

- Un etichetta per dire all’utente cosa inserire.

Dobbiamo inserire un Vbox perche nel nostro schema vogliamo una prima parte superiori per l’immissione dei dati e uno inferiore per la visualizzazione del risultato.

Inseriamo quindi il Vbox nella parte centrale del BorderPane.

Successivamente inseriamo un Hbox dentro la Vbox in modo da ottenere la parte per l’inserimento dei 3 elementi scritti sopra ( bottone etichetta e casella di testo).

Cerco nella libreria una TextArea che ci servirà per far visualizzare i dati inseriti dall’utente.

La inseriamo nella Vbox dopo l’ Hbox.

Ricapitolando avremo un contenitore verticale ( Vbox ) che contiene un Hbox e una TextArea.

Nell’ Hbox inseriamo invece i 3 elementi che ci servono per interagire con l utente e fargli inserire i dati. Bisogna fare attenzione all’ordire in cui si inseriscono i 3 elementi nell’ Hbox per avere un BorderPane più chiaro.

Modifichiamo il nome dell’etichetta ( Nome ) e del bottone ( Inserisci ) mediante la schermata a destra delle proprietà.

Nella casella di testo invece non scriviamo niente, logicamente, in quanto sarà l’utente che dovrà scriverci. Possiamo però usare la proprietà Prompt Text per scrivere un messaggio visualizzabile dall’utente (in grigino) . Possiamo scrivere per esempio ( scrivi il tuo nome…).

Nella sezione Layout, a destra possiamo modificare le dimensioni di Vbox, Hbox ecc… Nel nostro caso allarghiamo la Vbox ( Pref Width = 500 ). Cosi facendo anche tutti gli elementi collegati alla Vbox si allargheranno di conseguenza ( BorderPane ).

Esiste una proprietà per la larghezza e la lunghezza che si chiama USE\_COMPUTED\_SIZE che fa in modo che la larghezza e la lunghezza sia quella che serve, non di piu o di meno ( calcola la dimensione che serve per contenere correttamente i nodi al suo interno ).

Usiamo quindi questo comando sia sull’ Hbox, in modo tale da non avere i nodi al suo interno schiacciati, che per la TextArea .

Sempre nel Layout c’è una proprietà che si chiama Spacing che permette di distanziare tutti i nodi presenti all’interno del nodo che stiamo considerando di un tot di pixel.

Modifichiamo lo spacing dell’ Hbox impostando 10 pixel.

Padding: comando sopra spacing , indica la distanza tra l elemento interno e il suo contenitore.

Margine: sopra padding, è il bordo tra l’elemento esterno e gli altri ad esso adiacenti.

Sostituire dei valori a questi comandi per capire bene il loro funzionamento ( per esempio nella Vbox).

Ogni volta che aggiungiamo qualcosa su SceneBuilder, il contenuto del file Nome.fxml su Eclipse si aggiorna automaticamente.

Proviamo a far partire il Main ( tasto play Verde in alto ) . Il programma non funziona ancora come vorremmo noi. Dobbiamo cambiare, mediante dei codici, il comportamento di ciascun nodo in modo che ogni elemento faccia ciò che vogliamo noi. ( frase inutile ).

Nel nostro caso, noi dobbiamo intervenire solo quando l’utente preme il bottone; negli altri nodi, l utente può gia fare quello che vuole ( scrivere, cancellare, copiare e incollare ecc…).

Per modificare l’azione del bottone torniamo su SceneBuilder e dopo aver premuto sul bottone premiamo sul CODE nel menu a tendina a destra ( quello delle proprietà e del layout ).

Qui abbiamo due cose che ci interessano:

* Fx:id 🡪 l’identita del bottone verrà convertita in una variabile java all’interno del mio programma. Se per esempio scrivo ‘pippo’ , Java avrà accesso ad una variabile chiamato pippo di tipo Button e che punta esattamente a quell’ oggetto ( il bottone ).
* Una serie di comandi che iniziano tutti con On e che ci indicano tutte le azioni che possono avvenire su quel determinato oggetto. All’interno di questi comandi inserisco dei metodi java. Questi ultimi verranno chiamati quando avviene quello indicato dal nome del codice. Per esempio in ‘ On Action ‘, il metodo verrà richiamato ogni volta che avviene un azione sul bottone. Inseriamo in On Action il metodo ‘handleinserisci’ .

Diamo un id al bottone ( bottone ) , al TextField ( nome ) e alla TextArea ( messaggi ). Questi sono i nomi delle variabili Java collegati agli elementi dell’interfaccia.

In basso a sinistra c’è la voce Controller ( ci servirà in seguito ) che ci indica la classe Controller (it.polito.tdp.primo.PrimoController nel nostro caso) e i nomi delle variabili java.

Nel menu View in alto, la voce ‘ Show Sample Controller Skeleton ‘ crea un controller che contiene le variabili scritte nel fxml che vorremmo manipolare tramite il codice java. Ci da inoltre il nome del metodo del bottone handleinserisci.

Copiamo il contenuto di questo file e lo andiamo a inserire in Eclipse nella classe PrimoController (cancellare prima le due righe vecchie ).

Ricordarsi di salvare ogni tanto.

IMPORTANTE : se eseguo qualche modifica con SceneBuilder, per esempio aggiungo variabili, metodi o in generale qualsiasi cosa, quando torno su Eclipse, prima di runnare il programma, premere sulla tendina del file fxml in alto. Cosi facendo Eclipse aggiorna le modifiche fatte. Se ci si dimentica di farlo, è come se non avessimo apportato nessuna modifica ( Eclipse tonto cit.)

Il nostro obbiettivo è quello di, una volta premuto il bottone , visualizzare nella TextArea il nome presente nella TextField preceduto da Buongiorno. Come facciamo?

Andiamo in PrimoController e sotto la voce:

void handleInserisci (ActionEvent event)

inseriamo :

String nomeScritto = nome.getText();

Creiamo quindi una variabile nomeScritto che conterrà ciò che inseriamo nel TextField. Sarebbe stato pippo.getText() se in SceneBuilder avessimo deciso di usare pippo come fx:id del TextField.

Quindi fino ad adesso abbiamo detto: ‘ Quando premi il bottone Inserisci, crea una variabile nomeScritto che prende il nome scritto nella textField’.

Ora però dobbiamo ancora visualizzarlo nella TextArea con Buongiorno prima.

Aggiungiamo sotto:

messaggi.setText(“Buongiorno, “ +nomeScritto+ “\n”);

n.b: \n per andare a capo.

Se runniamo il programma adesso dovrebbe andare.

Se al posto di setText avessimo messo appendText, nel caso avessimo aggiungo altri nomi, li mette uno dopo l’altro; con set sosituisce ( provare per capire ).

Nome.clear(); pulisce la casella di testo dove l’utente inserisce il nome.

In SceneBuilder, se vado nella TextArea e vado nelle sue proprietà, se tolgo la spunta alla voce Editable, non permetto all’utente di modificare l’area di testo.

In PrimoController, dopo:

String nomeScritto = nome.getText();

aggiungiamo un if che ci controlla che l’utente abbia inserito qualcosa nella TextField ( evito che prema il bottone senza aver inserito in nome e che quindi esca scritto Buongiorno, niente ).

If(nomeScritto.lenght()==0) {

Messaggi.appendText (“Errore: devi inserire un nome!”); }

Else{

Se scrivo:

nomeScritto = nomeScritto.trim() ;

elimino gli spazi prima e dopo il nome quando l’ utente digita il nome.

FINE