# 12-USER INPUT animation

**User Interaction:**

* Bisogna captare le iterazioni degli utenti

Quando c’è un tocco, **ottengo un MotionEvent** contenente le informazioni che mi forniscono informazioni su cosa è avvenuto (è uno stream, devo considerarlo come una parte di una sequenza).

MotionEvent, inziialmente aveva field Action, successivamente 2 field:

* **Immagine che contiene testo, schermata, Carattere

  Descrizione generata automaticamenteActionMasked**: azione

→ notifica di ignorare

* **ActionIndex:** indice

Un **singolo tocco** è modellato dal sensore in una superficie ellittica, si assume il tocco al centro dell’ellisse

* **x, y**: centro dell’ellisse
* **touchMinor, touchMajor** : lunghezza dei mezzi assi dell’ellisse
* **size**: contiene area dell’ellisse
* **pressure**: stima della pressione applicata
* **orientation**: angolo tra l'asse maggiore dell'ellisse e l'asse -Y: quando vale 0, l'asse maggiore è verticale
  + se positivo, è inclinato verso destra; se negativo, a sinistra

**Tocchi multipli**: partendo dal primo tocco, il MotionEvent riporta le informazione circa tutte le dita a contatto con lo schermo

* nr tocchi in **pointerCount**
* ogni tocco è associato ad un id che è costante per la durata del tocco
  + accedo ad un determinato tocco con **getPointerId(index)**

### MOTION EVENT:

Immagine che contiene testo, linea, schermata, diagramma

Descrizione generata automaticamente

#### **INPUT PRECEDENTI**

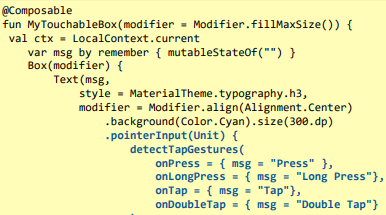
Stream di dati sul touch sono bufferizzati:

* Il dispositivo elettronico può operare ad un elevata frequenza ma non vogliamo che la nostra applicazione senta con quella frequenza
  + Il dispositivo comuncia le informazioni con una determinata frequenza in blocchi di informazione
    - Contiene l’azione attuale e un gruppo di azioni precedenti
      * getHistoricalX()
      * getHistoricalY()
      * getHistoricalOrientation()
      * …
    - Nella nostra applicazione abbiamo bisogno di un oggetto che mantenga questi dati per svolgere determinate azioni
  + Android fornisce **un gestureDetecto**r per la viewCompose e per JetpackCompose
* Lo stream di dati inizia con Activity
  + Immagine che contiene testo, schermata, linea, diagramma

    Descrizione generata automaticamenteActivity è notificata con **DispatchTouchEvent**
    - Ha la possibilità di comportarsi come filtro e scartare cose in modo tale da non propagare (es quando un ModelView appare sopra un altro (popup) e voglio sentire solo click su di lui) 🡪 posso scartare tocchi su altre parti.

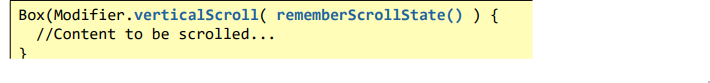
Come implementare touch in una JetbackCompose application?

* Cosa più semplice → non usarla e usare solo bottoni
* Usare modifier
  + **Modifier.clickable** contenente un parametro che è una callback che esegue determinate cose quando avviene touch
  + **Modifier.draggable** specificando orientation e state
    - Detect quando un componente viene mosso lungo uno dei due assi
  + **Modifier.pointerInput**
    - Mi permette di captare una iterazione generale, prendendomi la responsabilità di gestire la complessità
    - detectTapGesture → mi permette di detect iterazioni

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente **--codice animation**

Di default, quando usiamo compose, se non usiamo uno specifico component come lazyGrid o lazyScroll, componenti non sono scrollabili; per renderli scrollabili dobbiamo usare **verticalScroll o horizontalScroll** che devono ricordarsi la posizione in cui ci trovavamo attraverso **rememberScrollState**



Nota su gesture on scrolling:

* compose supporta due tipi di gesture basati su scrolling:
  + **scroll**: per esplorare contenuto la cui dimensione è maggiore della superficie del display in cui è contenuto
  + **swipe**: per triggerare azioni non reversibili (Es: eliminare item da una lista)

## SWIPE

C’è una serie di componenti che devono essere usati, si possono usare una serie di posizioni in cui può essere un componente:

* elemento potrà essere draggable in queste posizioni

Immagine che contiene testo, linea, Carattere, diagramma

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamenteSi può usare uno speciale draggable**: AnchoredDraggableState**

*si possono settare delle soglie di velocità e posizione in modo tale da poter gestire meglio i vari casi.*

Immagine che contiene testo, schermata, Carattere, linea

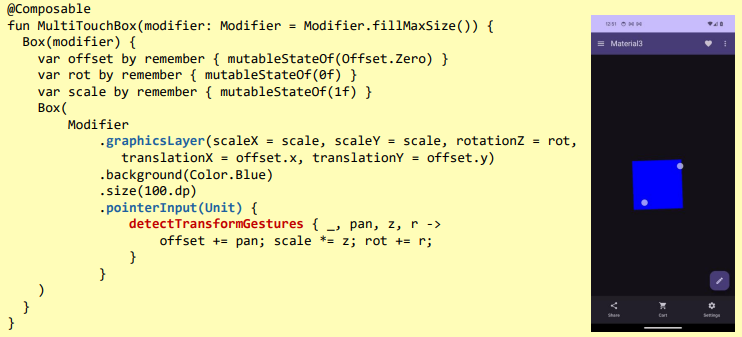
Descrizione generata automaticamente**Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamenteDragging and swiping**

* Aniimazione sul pptx

**Multi-finger gesture:** JetpackCompose supporta questo tipo di interazione (pinch) fornendo, tra i vari rilevatori in PointerInputScope, il metodo detectTransformGestures(...)

* L'utilizzo di questo metodo richiede che, nella catena di modificatori, il modificatore graphicsLayer(...) sia presente in una posizione precedente

Il metodo consente la manipolazione simultanea della rotazione, del ridimensionamento e della traslazione del componente a cui viene applicato, creando le condizioni per il comportamento pinch-to-zoom

# ANIMATION

Ci permettono di reagire alle iterazioni, guidare l’utente, evidenziare un cambiamento:

* Eliminazione elemento
* Blinking
* Mostrare relazione tra diversi elementi

Utili perché prendono attenzione dell’utente ma da usare moderatamente.  
**Nota**:

* dove abbiamo focus visivo: notiamo tutto
* contorno: notiamo che qualcosa succede ma non distinguiamo colori e etc
  + veloci

si effettua una **distinzione tra**:

* **Animazioni funzionali**: utilizzato per guidare e informare l'utente in tempo reale e può essere il risultato di un simulatore interattivo
* **Animazioni decorative**: supporto per storytelling → fornire informazioni all’utente in modo che mantenga l’attenzione

**4 maggiori aree**:

* micro-interagion
* loadring and advancent
* navigation
* story-telling

#### **Immagine che contiene testo, schermata, menu Descrizione generata automaticamente 12 principi di animation**

vedere esempi animati su pptx

Immagine che contiene testo, schermata, Carattere, logo

Descrizione generata automaticamente**Squash and stretch**: modo per fornire idea di movimento

**Anticipation**: fornisce un’anticipazione su cosa sta accadendo

* es: riduco icona e mostro cosa si avrebbe (si può fare con mouse, non con touch)

**Staging**: Il movimento viene utilizzato per dirigere l'attenzione dello spettatore ed evidenziare gli elementi chiave della scena

* usato in navigation per mostrare il nuovo screen che entra nello schermo

**Straight ahead:**

* Immagine che contiene schizzo, cerchio, modello, disegno

  Descrizione generata automaticamentepossono essere disegnate in diversi modi
* **pose-to-pose**: versione ora, cosa sta succedendo, versione finale
  + frame intermedi sono realizzati in modo automatico

**Follow Through and Overlapping Action:** quando le varie parti dell’oggetto non si muovono tutte allo stesso tempo, ma una va prima e altre seguono

**Slow in and slow out**: Come un oggetto richiede tempo per accelerare e decelerare mentre si muove

**Arc**: Le cose si muovono secondo il moto dell'arco, più velocemente qualcosa si muove, più piatto è l'arco e più ampia è la curva

**Secondary Actions**: complementare o evidenziare l'azione principale che si svolge in una scena

**Timing**: Descrive la velocità con cui qualcosa si muove e per quanto tempo rimane fermo

**Exaggeration**: ci sono alcune situazioni in cui si vuole operare sulla parte emotiva del cervello

* intrattenere persone e mostrare elementi

**Solid drawing**: per rappresentare alcune icone che sembrano tridimensionali

**Appeal**: si possono usare le animazioni per creare Appeal → rendere le mie cose interessanti

* raccontare una storia

**Come farlo?**

Quando si vuole cambiare la gerarchia:7



Quando si vuole alterare qualcosa che sia già sullo schermo bisogna usare ricomposizione:

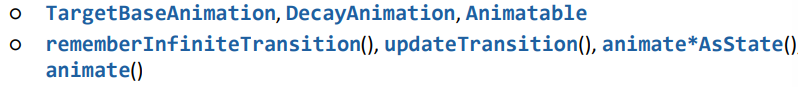


Immagine che contiene testo, diagramma, design, origami

Descrizione generata automaticamente

## Animated visibility

Elemento modulare che mostra/nasconde il contenuto in base ad un valore booleano

* permette di cambiare il comportamento dell’animazione per mostrare/nascondere il contenuto attraverso i parametri enter:EnterTransition e exit:ExitTransition
* all’interno della lambda functione che racchioude il contenuto da mostrare, è possibile definire ulteriori transizioni che risultano collegate allo stato visibile
  + questo permette di creare effetti di animazioni concatenate

Immagine che contiene testo, schermata, Carattere, documento

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamente

#### Immagine che contiene testo, Carattere, schermata, documento Descrizione generata automaticamenteCrossfade

### AnimatedContent

È possibile definire transizioni specializzate attraversi elementi composable AnimatedContent(..) .

* fornisce il parametro transitionSpec che è responsabile per definire quali transizioni devono essere usate per far comparire e scomparire componenti basandosi sulla coppia formata dllo stato iniziale e dallo stato finale del cambiamento
* **Immagine che contiene testo, Carattere, schermata, linea

  Descrizione generata automaticamente**il valore preso da questo parametro è il composition di 2 transizioni usando l'operatore infisso con with

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

### Animatable

La funzione Animatable ritorna un container che incapsula un singolo valore di un tipo generico T che può evolvere nel tempo

* Offre il metodo  **animateTo(..)** che permette di animare il valore corrente verso quello specificato come destinazione, specificando il tipo di interpolazione da adottare e l'eventuale velocità iniziale
  + This method is of the suspend fun type and can only be invoked in the context of a coroutine

Immagine che contiene testo, schermata, Carattere, documento

Descrizione generata automaticamente

🡪animazione su pptx

### Animate \*AsState

Insieme di funzioni composanble che permetto di aniamre un singolo valore dall’indicatore del valore da ottenere

* Numeri, colori, dimensioni
* La durata dell’animazioni e il suo progresso può essere controllata attraverso il parametro opzionale di tipo AnimationSpecù
  + Immagine che contiene testo, schermata, Carattere

    Descrizione generata automaticamenteVarious animation types are supported: spring, tween, keyframes, repeatable, infiniteRepeatable, snap

### AnimationSpec

Interfaccia che rappresenta i parametri relativi all'animazione:

* Tipo del dato animato
* Flag che indica se animazione è infinita o no
* Altri dettagli

Funzioni che gestiscono animationSpec:

* **Tween**: anima un valore, per una data durata, possibilmente introducento un ritardo iniziale, easing è la curva che definisce il valore durante l’interpolazione
* **Spring**: crea un animazione basandosi sul modello fisico di una molla
* **Keyframes**: anima un valore basandosi su posizione intermedie ad un dato tempo
  + Immagine che contiene testo, Carattere, schermata

    Descrizione generata automaticamenteÈ possibile specificare una curva di interpolazione per definire la modalità di interpolazione in ogni intervallo

Immagine che contiene testo, schermata, Carattere

Descrizione generata automaticamente