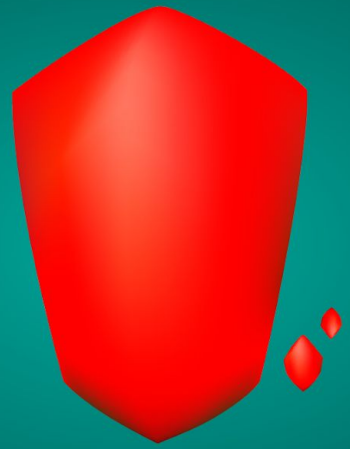
Hochschule Bremen
Mobile Computing SS2017
Touch Gestures in Mobile Apps



Crystal Creatures

Denise Kirschner
Nicole Przybycin
Pawan Basnet

Structure

- Our App: Crystal Creatures
- Common Gestures
- Drag & Drop Gesture
- Custom Gestures
  - Gesture Builder
  - DIY
- Crystal Creatures Demo
- Sources

Our App: Crystal Creatures

- Our idea:
  - Create a game with focus on raising a creature
  - The user is provided option between three creature to choose from
  - The creature you choose hatches as a baby and matures as you interact with it
  - We implemented the use of gesture to interact with the creature
- The game
  - After you are finished with the selection and naming the creature, game enters home screen with baby version of the creature selected
  - Creature has to be fed and taken care of, so it evolves into next stage
  - Inversely if not few and taken care, the creature will die and you will have to start again

Common Gestures

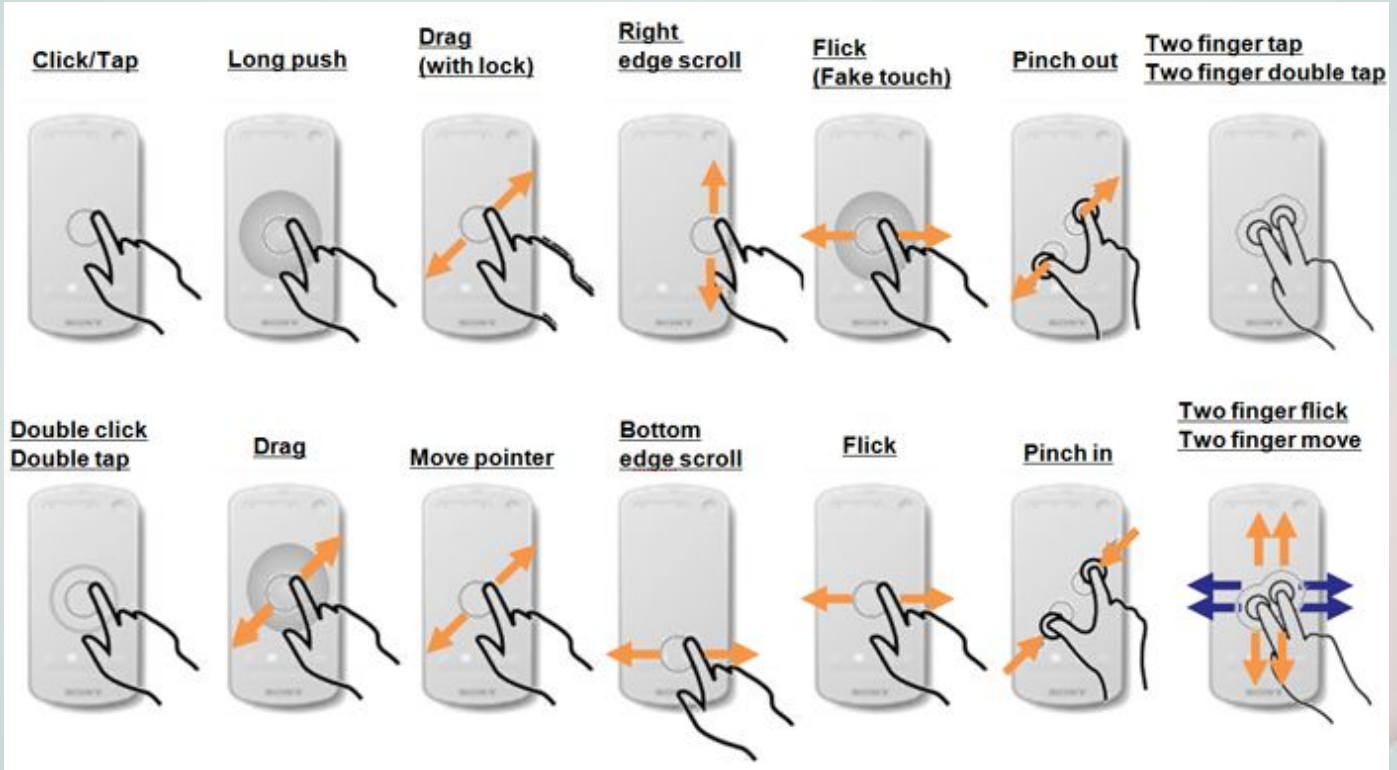


Illustration 1

Common Gestures

- Common gesture are included with android studio
- To detect gestures in you must first import gesture compat
- To use the common gesture, import and implement gesturedetector
  - gesture listener for gesture like swipe
  - double listener for double tap gestures

```
import android.view.GestureDetector;

import android.support.v4.view.GestureDetectorCompat;

public class theActivity extends Activity implements
GestureDetector.OnGestureListener, GestureDetector.OnDoubleListener
```

Gesture Compat and onTouchEvent

- You need to create GestureDetectorCompat object to use gesture compat class
- Then override the onTouchEvent class to call the gesture method you wrote when that gesture is detected

```
@Override
public boolean onTouchEvent(MotionEvent event) {
    this.mDetector.onTouchEvent(event);
    // Be sure to call the superclass implementation
    return super.onTouchEvent(event);
}
```

Gesturedetector

- Gesturedetector lets you implement methods such as onLongPress, onScroll
- Overriding these method change what they do when these gesture are performed

```
public boolean onFling(MotionEvent event1, MotionEvent event2,
                       float velocityX, float velocityY) {
    Log.d(DEBUG_TAG, "onFling: " + event1.toString() + event2.toString());
    float distance = event1.getX() - event2.getX();
    if (distance < -200) {
        Log.d(DEBUG_TAG, "Fling was to the right.");
        iterate = (iterate + 1) % creatures.length;
        creSwitcher.setImageResource(creatures[iterate]);
    } else if (distance > 200) {
        Log.d(DEBUG_TAG, "Fling was to the left.");
        iterate = (iterate + 1) % creatures.length;
        creSwitcher.setImageResource(creatures[iterate]);
    }return true;
}
```

Drag & Drop

- View.onDragListener
- create your own Listener method @Override onDrag(view, event)
- switch(event.getAction()) - fill cases with behaviour
  - ACTION_DRAG_STARTED
  - ACTION_DRAG_ENTERED
  - ACTION_DRAG_EXITED
  - ACTION_DRAG_LOCATION
  - ACTION_DROP
  - ACTION_DRAG_ENDED
- Move to another location: onDrop -> set new X/Y location for view
- Move back to old location: onDrop
  -> save starting location and set old X/Y location for view

Custom Gestures - Gesture Builder

- Install Gesture Builder on a physical or the AVD (Android Virtual Device)
  - If you use the AVD, add SD Card support
- Create gestures in the Gesture Builder, so a text file will be generated with all your gestures
- Open the Android project, create a new directory in /app/res and put the generated file from the Gesture Builder in the new directory
- Add a GestureOverlayView to the layout

```
<android.gesture.GestureOverlayView
     android:id="@+id/gesture Overlay"
     android:layout_width="match_parent"
     android:layout_height="match_parent"
 </android.gesture.GestureOverlayView>
```

Custom Gestures - Gesture Builder

- In the activity declare a variable of type GestureLibrary
- In the onCreate() method load the library from the Gesture Builder's file
- Add the GestureOverlayView

```
public class MainActivity extends AppCompatActivity {
       private GestureLibrary  lib;
        protected void onCreate(Bundle savedInstanceState) {
                    …
             lib = GestureLibraries. fromRawResource(this, R.raw.gestures);
             if( !lib.load() )
                       finish();
          GestureOverlayView gesture = (GestureOverlayView)
findViewById(R.id.gestureOverlay);
     }
}
```

Custom Gestures - Gesture Builder

- Register the OnGesturePerformedListener event listener on the GestureOverlayView an implement the onGesturePerformed Method

```
protected void onCreate(Bundle savedInstanceState) {
                …
        @Override
        public void onGesturePerformed(GestureOverlayView gestureOverlayView,
Gesture gesture) {
            ArrayList<Prediction> predictionArrayList =  lib.recognize(gesture);
            for(Prediction prediction : predictionArrayList) {
                if (prediction.score > 1.0)
                /* do something if the gesture was recognized,
                 prediction.name equals the gesture name from the Gesture Builder*/
            }
        }
    });
```

Custom Gestures - DIY

- onDragListener / onTouchListener
- create your own Listener method @Override onDrag/onTouch
- switch(event.getAction()) - fill cases with behaviour
- save finger movement in array

- ACTION_DRAG_STARTED and ACTION DOWN
- ACTION_DRAG_LOCATION /ACTION_MOVE
- ACTIONS_DRAG_ENDED/ACTION_DROP/ ACTION_CANCEL

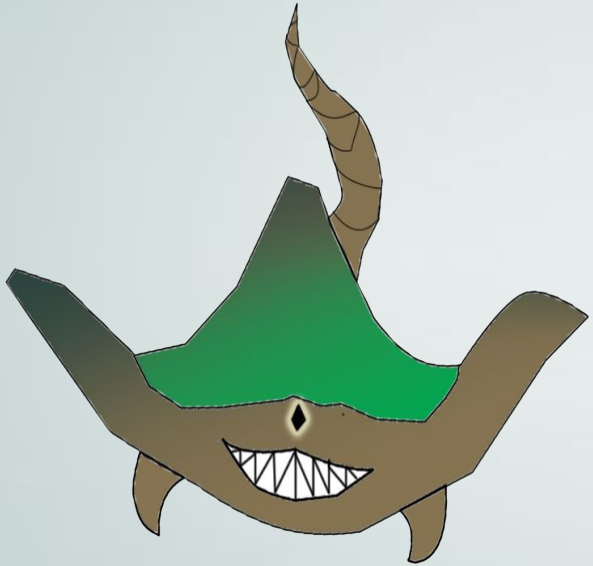
Custom Gestures - DIY

- Example: Zig-Zag Pattern
- view.setOnDragListener(new View.OnDragListener(){ *class body* }
  - create a java.List for the X and Y coordinate
  - add individual variables
    - @Override public boolean onDrag(View v, DragEvent event){ *method body*}
    - switch(event.getAction()){ *switch body* }
    - case DragEvent.ACTION_DRAG_LOCATION:

```
list.add(X); list.add(Y);
if(pastNewPos > pastOldPos)& if(currentNewPos < currentOldPos) = leftTurn
if(leftTurn)& if(pastNewPos < pastOldPos)& if(currNewPos >
currentOldPos)=rightTurn
if(leftTurn&rightTurn){ counter++ } & if(counter=3){ doSomething}
then leftTurn = false; rightTurn = false;
```

    - case DragEvent.ACTION_DRAG_ENDED:
      - reset counter and booleans

Crystal Creature Demo



Sources

- Web sources:
  - Custom gestures with Gesture Builder (10th June 2017, 16:23)
- Videos:
  - Android Studio Tutorial - Gesture Overlay View (10th June 2017, 14:31)
- Images:
  - Illustration 1 (6th June 2017, 10:23)
- Tools
  - Gesture Builder App (11th June 2017, 21:42)

Thank you for your attention!