ZADANIA W ANDROID STUDIO

ZADANIE 1

W pierwszym zadaniu musisz zmierzyć się z próbą napisania kodu, który sprawi że gracz będzie mógł się poruszać po naszej mapie. Postać jest stale wycentrowana – musisz zadbać o to, żeby tło się poruszało, a postać przyjmowała odpowiednią animacje. Spokojnie, to nie aż tak trudne jak się wydaje – poniżej masz dużo wskazówek 😊. Będziesz operował na dwóch klasach – GameView oraz Background. Swój efekt możesz obserwować w emulatorze Androida. Na początek sprawdź czy emulator działa -> kliknij mały zielony przycisk play w prawym górnym rogu. Po wykonaniu kodu, ujrzysz ekran początkowy gry. Kliknij „play”. Pojawi się taki ekran:

Wskazówki 1 – 3 dotyczą klasy GameView, funkcja onTouchEvent (linia 120):

- Jak widzisz funkcja onTouchEvent jako argument przyjmuje ***MotionEvent event***. Jest to nasz „nasłuchiwacz” na dotyk. Korzystając z niego, odczytaj do dwóch nowych zmiennych (int) miejsce tapnięcia na ekran. Użyj getterów z **event**.
- Teraz potrzebujemy zatrzymać ruch tła (nie zawsze jest to potrzebne – tylko w przypadku kiedy gracz dokona nowego ruchu w trakcie zmieniania miejsca postaci, lecz jej użycie gdy postać stoi nic nie zmienia). Odnajdź odpowiednią funkcję w klasie background i ja zaimplementuj w zaznaczonym miejscu.
- Następnie musimy ustawić nowy „środek” tła. Co mam na myśli ? Po dotknięciu na ekran, chcemy aby postać do niego przeszła. Jak już wcześniej wspomniałem, jest ona stale wycentrowana, czyli jedyne co musimy zrobić to przesunąć tło, tak aby punkt dotknięcia ekranu znajdywał się na jego środku. Znajdź odpowiednią funkcje w klasie background. (Potrzebuje ona dwóch argumentów).

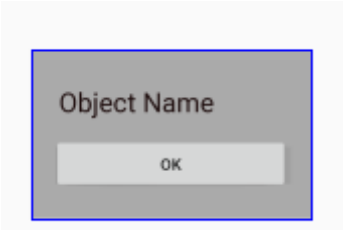
Wskazówka 4 dotyczy klasy Background (linia 63)

- Teraz musimy zająć się odpowiednim rysowaniem naszego tła. Chcąc iść w prawo, tło musi przesuwać się lewo – to stworzy właściwy i naturalny efekt dla naszego oka. Zaobserwuj kod gotowej funkcji leftScroll(). „Przesuwamy" nasze tło o stałą wartość dx (jest ona ustalona). Jednak, gdy dojdziemy do końca rozmiarów bitmapy, potrzebujemy zresetować to na początkowa wartość (czyli 0), aby uniknąć braku rysowania (na części ekranu pojawiłby się czarne plamy). Napisz analogiczne funkcje. Po ich napisaniu, uruchom emulator.

ZADANIE 2

W drugim zadaniu będziemy starali się narysować obiekt (drzewo 😊) i przypisać do niego wyskakujące okienko. Na początek zajmijmy się projektowaniem dialogu w pliku .xml. Otwórz plik res -> layout -> object_menu.xml

- Korzystając z opcji design, bądź text, zaprojektuj proste okno. Początek kodu masz już napisany. Twoim zadaniem jest dodanie jednego textView oraz button. Aby zadbać o czytelność, możesz wstawić pomiędzy nimi puste miejsce, tzw. „space".




```
<Space
    android:layout_width="match_parent"
    android:layout_height="30dp" />

<TextView
    android:id="@+id/objectName"
    android:layout_width="200dp"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:text="Object Name"
    android:textColor="#1F0F0F"
    android:textSize="24sp" />

<Space
    android:layout_width="match_parent"
    android:layout_height="15dp" />

<Button
    android:id="@+id/ok"
    android:layout_width="210dp"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:text="OK" />

<Space
    android:layout_width="match_parent"
    android:layout_height="15dp" />
```

- Przejdź do pliku Dialogs -> ObjectMenu.java. Tutaj musimy „połączyć” nasz plik .xml z klasą Javy. Masz już zadeklarowany przycisk ok. Używając funkcji findViewById i argumentu „R.id.ok” oraz „R.id.objectName”, przypisz obiekt klasy button i obiekt TextView do ich odpowiedników w pliku .xml. Spraw aby był przycisk był klikalny. Użyj setOnClickListener(this). Używając funkcji setText, ustaw nazwę wyskakującego okienka na string name. (Jest już zadeklarowany – jest to nazwa naszego obiektu, pobierana w konstruktorze). Tą nazwę ustalisz w następnym kroku.
- Przejdź do klasy GameView. Linia numer 100. Tutaj musisz stworzyć obiekt gry GameObject, wczytując plik .png, a także ustalając jego koordynaty. Przenalizuj konstruktor klasy GameObject. Początek masz dany poniżej:

```
tree = new GameObject(BitmapFactory.decodeResource(getResources(),R.drawable.tree), X
```

- Wskazówka 4 (linia 130 w GameView) jest identyczna jak wskazówka 1 w pierwszym zadaniu. Jeżeli udało ci je rozwiązać, po prostu napisz identyczny kod.
- Przejdź do 142 lini. Tutaj musisz dopisać jednego iFa. Zauważ, że każdy GameObject zawiera obiekt klasy Rect – nasze współrzędne. Używając tego obiektu i jego funkcji „contains(punktX, punktY)”, sprawdź, czy użytkownik nie dotyka na ekranie obiektu. Jeżeli tak, wyświetl okno dialogowe. (Odpowiednia funkcja znajduje się w klasie GameView).

```
if(tree.getRect()
```