


|   |  |                                  |       |   |              |
|---|--|----------------------------------|-------|---|--------------|
| 1Politechnika Bydgoska im. Jana i Jędrzeja Śniadeckich<br>Wydział Telekomunikacji, Informatyki i Elektrotechniki<br>al. prof. S. Kaliskiego 7, 85-796 Bydgoszcz |  |                                  |       |  <b>POLITECHNIKA<br/>BYDGOSKA</b><br>Wydział Telekomunikacji,<br>Informatyki i Elektrotechniki |              |
| Przedmiot   |  | Programowanie urządzeń mobilnych |       | Kierunek/Tryb   | IS/ST        |
| Nr laboratorium   | 10   | Data wykonania                   | 17.06 | Grupa   | 1            |
| Ocena   |  | Data oddania                     | 17.06 | Imię Nazwisko   | Cezary Tytko |
| Nazwa ćwiczenia   | Pętla gry, animacja postaci, kolizje, obsługa dotyku |                                  |       |   |              |

## 2. Cel ćwiczenia laboratoryjnego

Celem tego ćwiczenia jest stworzenie gry inspirowanej na klasycznej grze Boulderdash, realizacja zadania obejmuje pętlę gry, animacje postaci i przeciwników, tło, dźwięki, obsługę zdarzeń oraz sterowanie postacią.

## 3. Opis projektu

Na podstawie przedstawionego poniżej szkieletu aplikacji implementującego pętlę gry należy uzupełnić rozgrywkę inspirowaną się klasyczną grą Boulderdash (lista na końcu instrukcji).

## 4. Implementacja

**Kod został napisany w języku Kotlin w środowisku Android Studio.**

MainActivity.kt:

```

1. package com.example.pumlalab1
2.
3. import android.R.attr.left
4. import android.R.attr.right
5. import android.content.res.Resources
6. import android.graphics.Bitmap
7. import android.graphics.BitmapFactory
8. import android.os.Bundle
9. import android.util.Log
10. import android.widget.Button
11. import android.widget.GridLayout
12. import android.widget.ImageView
13. import androidx.appcompat.app.AppCompatActivity
14.
15.
16. class MainActivity : AppCompatActivity() {
17.
18.     private lateinit var spritesheetIdle: Bitmap
19.     private lateinit var spritesheetLeft: Bitmap
20.     private lateinit var spritesheetRight: Bitmap
21.
22.     private enum class Direction { IDLE, LEFT, RIGHT }
23.
24.     private var currentDirection = Direction.IDLE
25.     private var frameIndex = 0
26.     private val frameCount = 7
27.     private val frameWidth = 32
28.     private val frameHeight = 32
29.
30.     private lateinit var rockfordImageView: ImageView
31.
32.
33.     private lateinit var gridLayout: GridLayout
34.     private val numRows = 10

```

```

35. private val numCols = 10
36. private lateinit var map: Array<CharArray>
37. private var rockfordX = 1
38. private var rockfordY = 1
39. private lateinit var spritesheet: Bitmap
40. private lateinit var rockford: Bitmap
41.
42. override fun onCreate(savedInstanceState: Bundle?) {
43.     super.onCreate(savedInstanceState)
44.     setContentView(R.layout.activity_main)
45.
46.     gridLayout = findViewById(R.id.boardGridLayout)
47.     gridLayout.rowCount = numRows
48.     gridLayout.columnCount = numCols
49.
50.     // Inicjalizacja rockfordImageView:
51.     val displayMetrics = Resources.getSystem().displayMetrics
52.     val screenWidth = displayMetrics.widthPixels
53.     rockfordImageView = ImageView(this).apply {
54.         layoutParams = GridLayout.LayoutParams().apply {
55.             width = screenWidth / numCols // jeszcze możesz to poprawić, np. po layout pass
56.             height = screenWidth / numCols
57.         }
58.         // scaleType = ImageView.ScaleType.CENTER
59.     }
60.
61. //     gridLayout.addView(rockfordImageView) // Lub ustaw na odpowiedniej pozycji planszy
62.     animationHandler.post(animationRunnable)
63.
64.     initMap()
65.     loadSprites()
66.     drawMap()
67.
68.     findViewById<Button>(R.id.btnUp).setOnClickListener { moveRockford(0, -1) }
69.     findViewById<Button>(R.id.btnDown).setOnClickListener { moveRockford(0, 1) }
70.     findViewById<Button>(R.id.btnLeft).setOnClickListener { moveRockford(-1, 0) }
71.     findViewById<Button>(R.id.btnRight).setOnClickListener { moveRockford(1, 0) }
72. }
73.
74. private fun initMap() {
75.     map = arrayOf(
76.         "#####".toCharArray(),
77.         "#R.....#".toCharArray(),
78.         "#.....#".toCharArray(),
79.         "#....*...#".toCharArray(),
80.         "#.....#".toCharArray(),
81.         "#....O...#".toCharArray(),
82.         "#.....#".toCharArray(),
83.         "#..*.....#".toCharArray(),
84.         "#.....#".toCharArray(),
85.         "#####".toCharArray()
86.     )
87. }
88.
89. private fun loadSprites() {
90.     val options = BitmapFactory.Options().apply { inScaled = false }
91.     spritesheetIdle = BitmapFactory.decodeResource(resources, R.drawable.rockford_idle_32x32, options)
92.     spritesheetLeft = BitmapFactory.decodeResource(resources, R.drawable.rockford_left_32x32, options)
93.     spritesheetRight = BitmapFactory.decodeResource(resources, R.drawable.rockford_right_32x32, options)
94. }
95.
96. private fun drawMap() {
97.     gridLayout.removeAllViews()
98.     val displayMetrics = Resources.getSystem().displayMetrics
99.     val screenWidth = displayMetrics.widthPixels
100.    val screenHeight = displayMetrics.heightPixels
101.
102.    val tileWidth = screenWidth / numCols
103.    val tileHeight = screenHeight / numRows
104.
105.    for (i in 0 until numRows) {
106.        for (j in 0 until numCols) {
107.            if (map[i][j] == 'R' )
108.            {
109.                gridLayout.addView(rockfordImageView)
110.                continue
111.            }
112.        }
113.    }
114. }

```

```

130.         val imageView = ImageView(this).apply {
131.             layoutParams = GridLayout.LayoutParams().apply {
132.                 width = tileWidth
133.                 height = tileWidth
134.             }
135.         }
136.
137.         when (map[i][j]) {
138.             '#' -> imageView.setImageResource(R.drawable.wall_32x32)
139.             '.' -> imageView.setImageResource(R.drawable.ground_32x32)
140.             '*' -> imageView.setImageResource(R.drawable.diamond_32x32)
141.             'O' -> imageView.setImageResource(R.drawable.stone_32x32)
142.             ' ' -> imageView.setImageResource(R.drawable.empty_32x32)
143.         }
144.
145.         gridLayout.addView(imageView)
146.     }
147. }
148.
149. private fun moveRockford(dx: Int, dy: Int) {
150.     val newX = rockfordX + dx
151.     val newY = rockfordY + dy
152.
153.     if (map[newY][newX] != '#') {
154.         // przesun Rockforda
155.         map[rockfordY][rockfordX] = ' ' // stare miejsce
156.         map[newY][newX] = 'R'           // nowe miejsce
157.         rockfordX = newX
158.         rockfordY = newY
159.         drawMap()
160.     }
161.
162.     if (dx < 0) currentDirection = Direction.LEFT
163.     else if (dx > 0) currentDirection = Direction.RIGHT
164.     else currentDirection = Direction.IDLE
165.     frameIndex = 0
166. }
167.
168. private fun getFrameBitmap(direction: Direction, frame: Int): Bitmap {
169.     val spritesheet = when(direction) {
170.         Direction.IDLE -> spritesheetIdle
171.         Direction.LEFT -> spritesheetLeft
172.         Direction.RIGHT -> spritesheetRight
173.     }
174.     return Bitmap.createBitmap(spritesheet, frame * frameWidth, 0, frameWidth, frameHeight)
175. }
176.
177. private val animationHandler = android.os.Handler()
178. private val animationRunnable = object : Runnable {
179.     override fun run() {
180.         Log.d("TAG", frameIndex.toString())
181.         val frameBitmap = getFrameBitmap(currentDirection, frameIndex)
182.         rockfordImageView.setImageBitmap(frameBitmap)
183.         frameIndex = (frameIndex + 1) % frameCount
184.         animationHandler.postDelayed(this, 100) // co 100ms
185.     }
186. }
187.
188. }
189.
190. }
191.
192. }
193.
194. }
195.
196. }
197.
198. }
199.
200. }
201.
202. }
203.
204. }
205.
206. }
207.
208. }
209.
210. }
211.
212. }
213.
214. }
215.
216. }
217.
218. }
219.
220. }
221.
222. }
223.
224. }
225.
226. }
227.
228. }
229.
230. }
231.
232. }
233.
234. }
235.
236. }
237.
238. }
239.
240. }
241.
242. }
243.
244. }
245.
246. }
247.
248. }
249.
250. }
251.
252. }
253.
254. }
255.
256. }
257.
258. }
259.
260. }
261.
262. }
263.
264. }
265.
266. }
267.
268. }
269.
270. }
271.
272. }
273.
274. }
275.
276. }
277.
278. }
279.
280. }
281.
282. }
283.
284. }
285.
286. }
287.
288. }
289.
290. }
291.
292. }
293.
294. }
295.
296. }
297.
298. }
299.
300. }
301.
302. }
303.
304. }
305.
306. }
307.
308. }
309.
310. }
311.
312. }
313.
314. }
315.
316. }
317.
318. }
319.
320. }
321.
322. }
323.
324. }
325.
326. }
327.
328. }
329.
330. }
331.
332. }
333.
334. }
335.
336. }
337.
338. }
339.
340. }
341.
342. }
343.
344. }
345.
346. }
347.
348. }
349.
350. }
351.
352. }
353.
354. }
355.
356. }
357.
358. }
359.
360. }
361.
362. }
363.
364. }
365.
366. }
367.
368. }
369.
370. }
371.
372. }
373.
374. }
375.
376. }
377.
378. }
379.
380. }
381.
382. }
383.
384. }
385.
386. }
387.
388. }
389.
390. }
391.
392. }
393.
394. }
395.
396. }
397.
398. }
399.
400. }
401.
402. }
403.
404. }
405.
406. }
407.
408. }
409.
410. }
411.
412. }
413.
414. }
415.
416. }
417.
418. }
419.
420. }
421.
422. }
423.
424. }
425.
426. }
427.
428. }
429.
430. }
431.
432. }
433.
434. }
435.
436. }
437.
438. }
439.
440. }
441.
442. }
443.
444. }
445.
446. }
447.
448. }
449.
450. }
451.
452. }
453.
454. }
455.
456. }
457.
458. }
459.
460. }
461.
462. }
463.
464. }
465.
466. }
467.
468. }
469.
470. }
471.
472. }
473.
474. }
475.
476. }
477.
478. }
479.
480. }
481.
482. }
483.
484. }
485.
486. }
487.
488. }
489.
490. }
491.
492. }
493.
494. }
495.
496. }
497.
498. }
499.
500. }
501.
502. }
503.
504. }
505.
506. }
507.
508. }
509.
510. }
511.
512. }
513.
514. }
515.
516. }
517.
518. }
519.
520. }
521.
522. }
523.
524. }
525.
526. }
527.
528. }
529.
530. }
531.
532. }
533.
534. }
535.
536. }
537.
538. }
539.
540. }
541.
542. }
543.
544. }
545.
546. }
547.
548. }
549.
550. }
551.
552. }
553.
554. }
555.
556. }
557.
558. }
559.
560. }
561.
562. }
563.
564. }
565.
566. }
567.
568. }
569.
570. }
571.
572. }
573.
574. }
575.
576. }
577.
578. }
579.
580. }
581.
582. }
583.
584. }
585.
586. }
587.
588. }
589.
590. }
591.
592. }
593.
594. }
595.
596. }
597.
598. }
599.
600. }
601.
602. }
603.
604. }
605.
606. }
607.
608. }
609.
610. }
611.
612. }
613.
614. }
615.
616. }
617.
618. }
619.
620. }
621.
622. }
623.
624. }
625.
626. }
627.
628. }
629.
630. }
631.
632. }
633.
634. }
635.
636. }
637.
638. }
639.
640. }
641.
642. }
643.
644. }
645.
646. }
647.
648. }
649.
650. }
651.
652. }
653.
654. }
655.
656. }
657.
658. }
659.
660. }
661.
662. }
663.
664. }
665.
666. }
667.
668. }
669.
670. }
671.
672. }
673.
674. }
675.
676. }
677.
678. }
679.
680. }
681.
682. }
683.
684. }
685.
686. }
687.
688. }
689.
690. }
691.
692. }
693.
694. }
695.
696. }
697.
698. }
699.
700. }
701.
702. }
703.
704. }
705.
706. }
707.
708. }
709.
710. }
711.
712. }
713.
714. }
715.
716. }
717.
718. }
719.
720. }
721.
722. }
723.
724. }
725.
726. }
727.
728. }
729.
730. }
731.
732. }
733.
734. }
735.
736. }
737.
738. }
739.
740. }
741.
742. }
743.
744. }
745.
746. }
747.
748. }
749.
750. }
751.
752. }
753.
754. }
755.
756. }
757.
758. }
759.
760. }
761.
762. }
763.
764. }
765.
766. }
767.
768. }
769.
770. }
771.
772. }
773.
774. }
775.
776. }
777.
778. }
779.
780. }
781.
782. }
783.
784. }
785.
786. }
787.
788. }
789.
790. }
791.
792. }
793.
794. }
795.
796. }
797.
798. }
799.
800. }
801.
802. }
803.
804. }
805.
806. }
807.
808. }
809.
810. }
811.
812. }
813.
814. }
815.
816. }
817.
818. }
819.
820. }
821.
822. }
823.
824. }
825.
826. }
827.
828. }
829.
830. }
831.
832. }
833.
834. }
835.
836. }
837.
838. }
839.
840. }
841.
842. }
843.
844. }
845.
846. }
847.
848. }
849.
850. }
851.
852. }
853.
854. }
855.
856. }
857.
858. }
859.
860. }
861.
862. }
863.
864. }
865.
866. }
867.
868. }
869.
870. }
871.
872. }
873.
874. }
875.
876. }
877.
878. }
879.
880. }
881.
882. }
883.
884. }
885.
886. }
887.
888. }
889.
890. }
891.
892. }
893.
894. }
895.
896. }
897.
898. }
899.
900. }
901.
902. }
903.
904. }
905.
906. }
907.
908. }
909.
910. }
911.
912. }
913.
914. }
915.
916. }
917.
918. }
919.
920. }
921.
922. }
923.
924. }
925.
926. }
927.
928. }
929.
930. }
931.
932. }
933.
934. }
935.
936. }
937.
938. }
939.
940. }
941.
942. }
943.
944. }
945.
946. }
947.
948. }
949.
950. }
951.
952. }
953.
954. }
955.
956. }
957.
958. }
959.
960. }
961.
962. }
963.
964. }
965.
966. }
967.
968. }
969.
970. }
971.
972. }
973.
974. }
975.
976. }
977.
978. }
979.
980. }
981.
982. }
983.
984. }
985.
986. }
987.
988. }
989.
990. }
991.
992. }
993.
994. }
995.
996. }
997.
998. }
999.
1000. }

```

## Activity\_main.xml:

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3.     xmlns:app="http://schemas.android.com/apk/res-auto"
4.     xmlns:tools="http://schemas.android.com/tools"
5.     android:id="@+id/mainLayout"
6.     android:layout_width="match_parent"
7.     android:layout_height="match_parent"
8.     android:orientation="vertical"
9.     tools:context=".MainActivity">
10.
11.     <!-- Plansza gry -->
12.     <FrameLayout
13.         android:layout_width="match_parent"
14.         android:layout_height="0dp"
15.         android:layout_weight="1">
16.
17.         <GridLayout
18.             android:id="@+id/boardGridLayout"
19.             android:layout_width="match_parent"
20.             android:layout_height="match_parent"
21.             android:padding="0dp"
22.             android:visibility="visible"
23.             tools:visibility="visible" />
24.
25.     </FrameLayout>
26.
27.     <!-- Sterowanie -->
28.     <LinearLayout
29.         android:id="@+id/controls"
30.         android:layout_width="match_parent"
31.         android:layout_height="wrap_content"
32.         android:orientation="vertical"
33.         android:gravity="center"
34.         android:padding="8dp">
35.
36.         <Button
37.             android:id="@+id/btnUp"
38.             android:layout_width="wrap_content"
39.             android:layout_height="wrap_content"
40.             android:text="↑" />
41.
42.         <LinearLayout
43.             android:layout_width="wrap_content"
44.             android:layout_height="wrap_content"
45.             android:orientation="horizontal">
46.
47.             <Button
48.                 android:id="@+id/btnLeft"
49.                 android:layout_width="wrap_content"
50.                 android:layout_height="wrap_content"
51.                 android:text="←" />
52.
53.             <Button
54.                 android:id="@+id/btnRight"
55.                 android:layout_width="wrap_content"
56.                 android:layout_height="wrap_content"
57.                 android:text="→" />
58.
59.         </LinearLayout>
60.
61.         <Button
62.             android:id="@+id/btnDown"
63.             android:layout_width="wrap_content"
64.             android:layout_height="wrap_content"
65.             android:text="↓" />
66.
67.     </LinearLayout>
68. </LinearLayout>
```

## 5. Funkcje kluczowe

### Wyświetlanie planszy gry w siatce (GridLayout)

- Graficzna reprezentacja planszy 10x10 z różnymi obiektami (ściany, ziemia, diamenty, kamienie, puste pola).

### Sterowanie postacią Rockford za pomocą przycisków kierunkowych

- Możliwość poruszania się w górę, dół, lewo i prawo z aktualizacją pozycji na planszy.

### Wczytywanie i animowanie sprite'ów Rockforda

- Wczytywanie trzech różnych arkuszy sprite'ów (idle, lewo, prawo) i animowanie ich co 100 ms w zależności od kierunku ruchu.

### Dynamiczne odświeżanie planszy po ruchu

- Każdy ruch Rockforda powoduje przerysowanie całej planszy, z uwzględnieniem nowej pozycji.

## 6. Testowanie

### Test ruchu i kolizji:

- Sprawdzono, czy postać Rockforda nie przechodzi przez ściany (#) oraz czy poprawnie przemieszcza się po dostępnych polach (., ).
- Przetestowano zachowanie przy próbie wyjścia poza granice planszy.

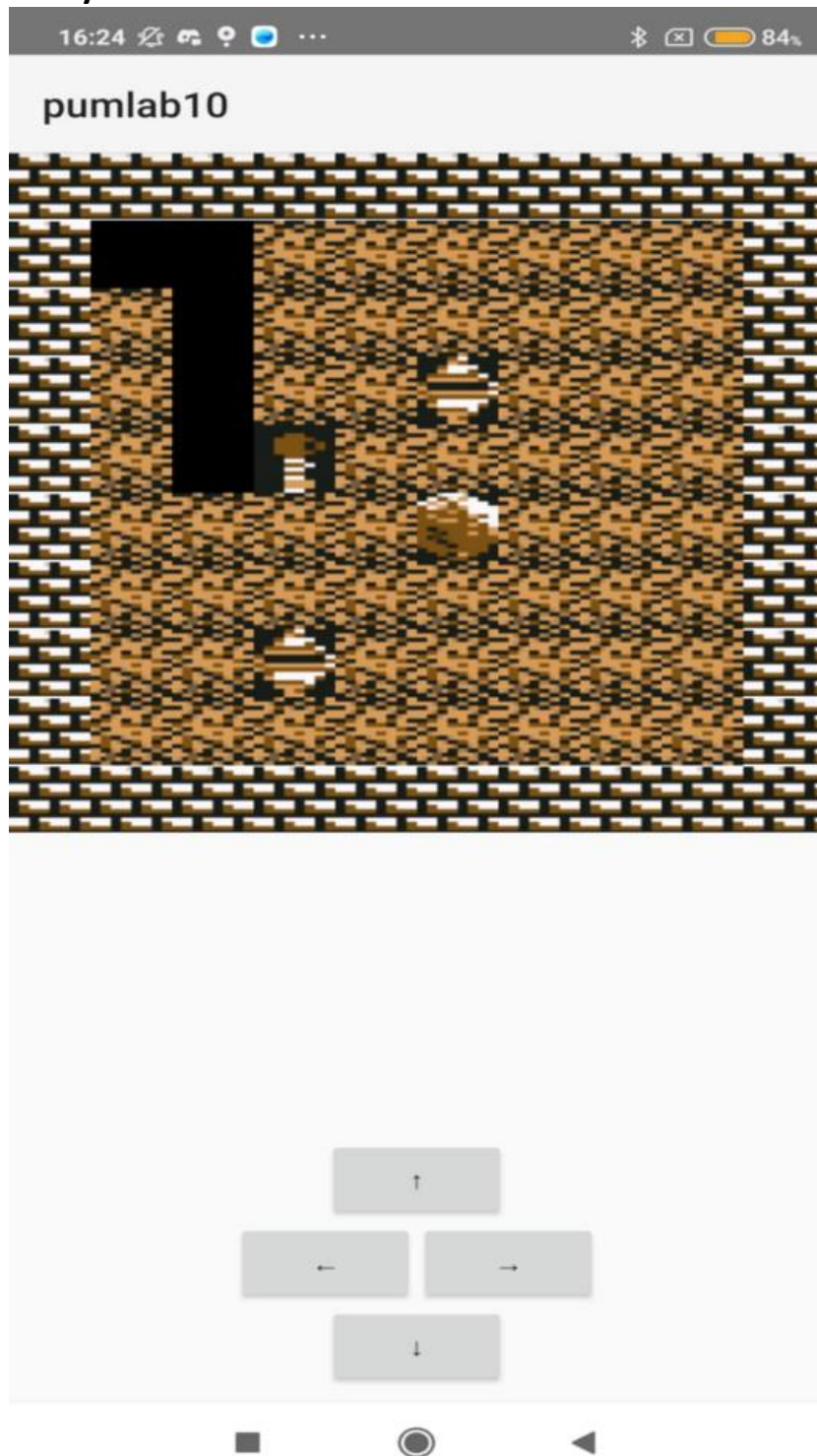
### Test animacji postaci:

- Zweryfikowano, czy animacje są zgodne z kierunkiem ruchu (idle, lewo, prawo).
- Upewniono się, że klatki animacji zmieniają się płynnie i w równych odstępach czasu.

### Test interfejsu i reakcji na przyciski:

- Przetestowano działanie przycisków kierunkowych – sprawdzono, czy każdy klik powoduje odpowiedni ruch postaci.
- Oceniono responsywność interfejsu przy szybkim i wielokrotnym naciskaniu przycisków.

## 7. Wyniki



## 8. Podsumowanie

Projekt realizuje przedstawienie gry inspirowanej na klasycznej grze Boulderdash. Do zbudowania planszy i stworzenia animacji wykorzystano dostarczone w instrukcji zasoby graficzne.

## 9. Trudności i błędy

- Nie wystąpiły żadne trudności ani błędy.

## 10. Źródła i odniesienia

- Nie korzystano ze źródeł i odniesień innych niż ta instrukcja.

## 11. Dodatkowe materiały

- Nie korzystano z dodatkowych materiałów.