

Пловдивски университет "Паисий Хилендарски"

Факултет по Математика и Информатика

Курсов Проект

**Игра Тетрис**

по *„Програмиране на приложения*

*за мобилни приложения“*

Специалност: Софтуерни технологии и дизайн ЗБ

Изготвил: София Николаева Клочева

Фак. №: 1401682030

Курс: 3, Група 2а

**За проекта**

Проекта има за цел да се създаде нейтив мобилно приложение „Игра Тетрис“. Приложението има поле за игра, бутони за управление на фигурите, бутон за пауза на играта и изгледи за ниво и събрани точки.

За създаване на квадратите, които образуват съответни фигури, е използван TableLayout. Всеки Layout е оцветен в съответния цвят, така че да се симулира падане на фигура. Фигурите се изрисуват в процеса на играта, като формата на фигурата и нейния цвят се избира произволно. Докато падат фигурите могат да се местят: на ляво, на дясно и да се завъртат, чрез ImageButton-ни. Добавена е и функционалността за по-бързо падане, като се задържи върху формата с падащите фигури.

Полето за игра се изрисува със стартирането на играта в съответно предварително зададения размер (20x15), за да се спести писането на XML код и по-голяма гъвкавост за размер. Полето се състои от редове съдържащи TableLayout-и във формата на квадратчета оцветени с полу-прозрачен цвят, с малко разстояние помежду си, за да се различават по-добре.

Падането на фигурите и зареждането на нови се осъществява с реално време. Това е постигнато, чрез използване на два thread-а: един за падането на фигурата и пауза на играта, и друг, който следи за зареждането на нова фигура, когато предищния приключи.

Фукционалността на фогурите е направена максимално абстрактна, за да се манипулира по-лесно и да се спести писане на повторяем код. Разликите в методите са единствено в различната форма на фигурата, за това това се проверява с конкретния клас на създанения обект.

**Идея на проекта**

Целта на проекта е да се създаде лесна за ползване игра, с минимален брой бутони и изгледи. За разлика от други подобни игри, този проект е лесен за разбиране и управление, и максимално се доближава до оригиналната игра. Липсата на допълнителни библиотеки позволява по-голямата адаптивност на приложението.

**Използвани технологии**

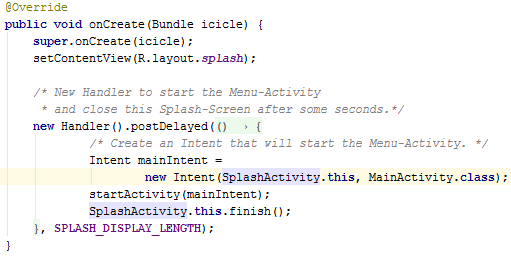
За създаване на приложението е изполвано Android Studio 2.3.1. За управление на изгледите, изрисуване, местене на фигурите и създаване на матрицата за игра е изполван Java код. Това включва Runnable интерфейси за местене на фигурите в реално време, за записване на координатите на обектите в игровото поле е използван двумерен масив. Параметрите на обектите в матрицата са за дадени в реално време, чрез задаване на LayoutParams. Изгледите и разположението на обектите в тях са използвани XML файлове. Ползвани са главно LinearLayout и TableRows. За изрисуването и дизайнът на обектите като фон, рамки и бутони са изрисувани векторно с Adobe Illustrator CS6, като главно е използван Pen tool и Live paint. Приложението е тествано локално на телефон Lenovo P70.

**Описание на приложението**

**Splash screen**

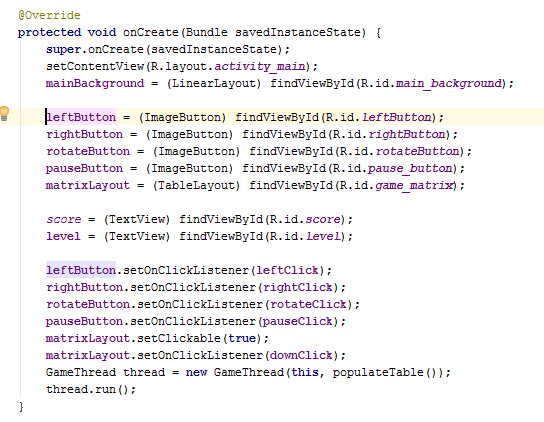
Приложението започва със **Splash** екран. Той се състои единствено от една картинка, която се появява за определено време, а след това стартира екрана с играта. Това е постигнато, чрез метода postDelayed на Handler, за отложено изпълнение на определено действие. В случая действието е зареждането на следващото activity.



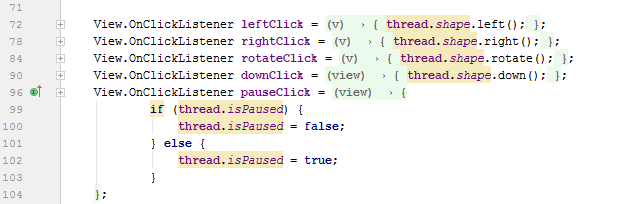


**Екран игра**

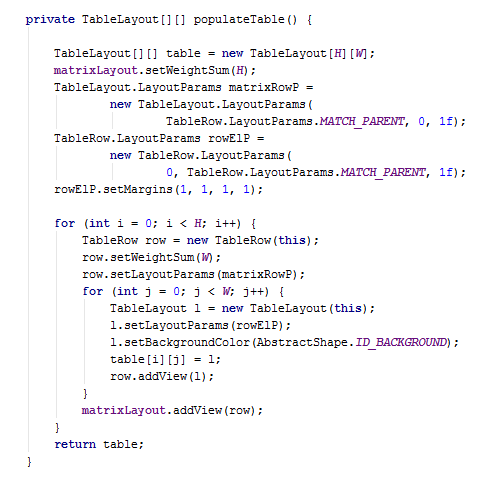
След Splash екрана се стартира MainActivity класа, който зарежда **екрана на играта**. Main Activity стартира с onCreate метода, който намира всички бутони и полетата за ниво и точки. След това задейства метода, който зарежда полето за игра (populateTable) и задейства thread-а за зареждане на фигура.



Всеки бутон има зададен OnClickListener, който при **натискане на бутон** манипулира фигурата. Бутона за пауза спира игровия thread.

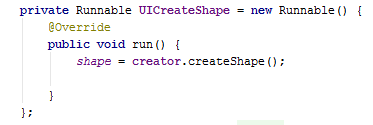


Със стартирането на екрана се зарежда **матрицата от кубчета**. Тя се състой първоначално от един празен TableLayout, като след стартиране се зареждат съответния брой кубчета на екрана. Това се постига, чрез два цикъла. Първия цикъл създава TableRow, а втория добавя към TableRow-а определения брой TableLayout-ти оцветени с полу-прозраче бял цвят. Обектите (TableLayout-и образувалите се кубчета) се записват в двумерен масив на съответния си координат в този масив, за да може тези обекти да се манипулират по-лесно в последствие.

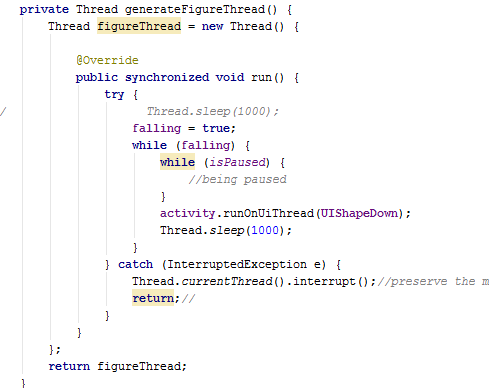
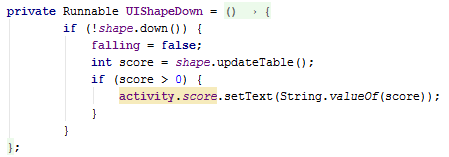


След зареждане на матрицата се стартира GameThread, който зарежда и стартира **игровия thread**. Той зарежда фигурата на thread-а на UI-а, и тя се изрисува на екрана. След това се зарежда нов thread за управление на фигурата и се изчаква да приключи, преди да се поднови цикъла на първия thread.



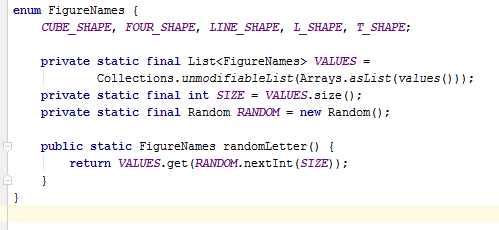


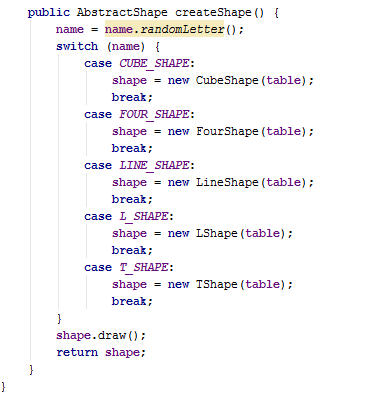
Със стартирането на **втория thread** за управлението на фигурата (figureThread) се задейства цикъл, който се върти докато фигурата не стигне до дъното (методът за местене надолу трябва да върне false). Докато този цикъл не е прекъснат на всяка една секунда се задейства метода runOnUiThread, който изпълнява падане надолу и преизчисляване на точките, ако има запълнен ред. Този thread също следи дали не е натистнат бутона за пауза, ако е то се задейства втория празен цикъл, който се върти докато не се продължи играта.



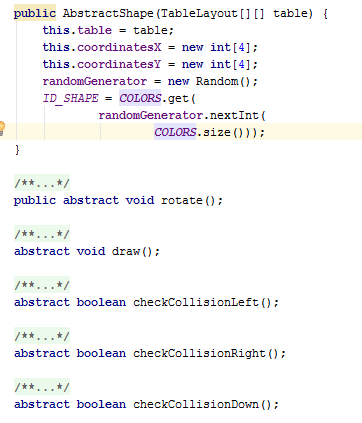
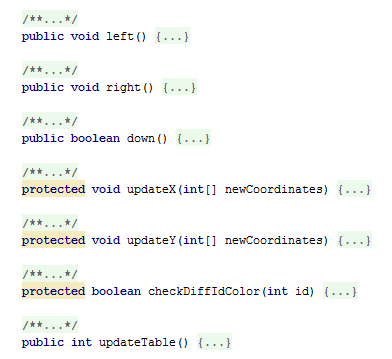
**Създаване на фигурите**

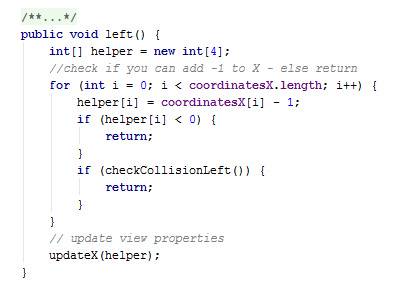
Фигурата се зарежда произволно, чрез **генератор произволни имена** на фигурите от играта. Това се постига, чрез енъм и метод за получане на произволно име от възможните стойности.



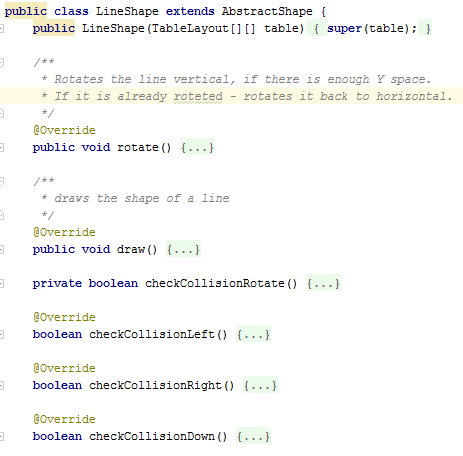
С помощта на Factory method се инстанциранконкретната фигура. Метода зарежда произволното име и спрямо него връша конкретен клас от фигурите.

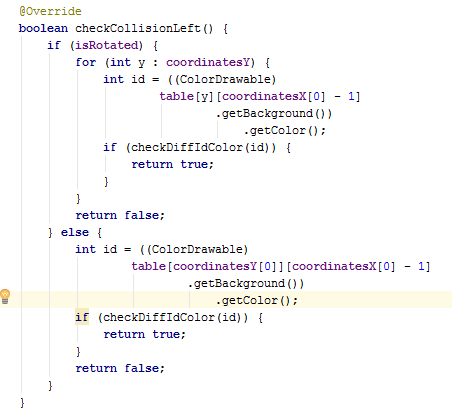
По-голямата част от функционалността е описана в **абстрактния клас** AbstractShape, той съдържа рамката от методи на всяка фигура и имплементиера някой от методите им, но не изцяло. Например има имплементиран метод за местене на фигурата на ляво, но този метод извиква метод от клас на конкретната фигура, който проверява за колизии с други фигури. След изпълнението на всеки един метод, който променя фигурата се задейства метод за обновяване на координатите на фигурата в матрицата.



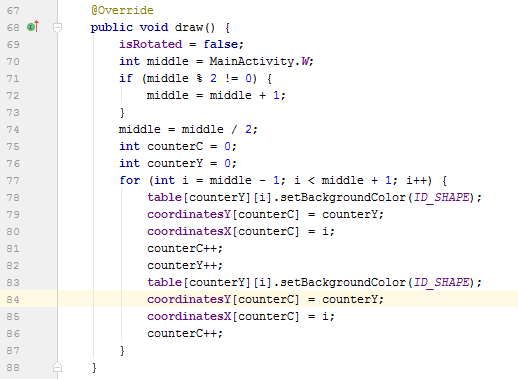


Всяка **фигура** имплементира методите наследени от абстрактната фигура, тези методи са строго специфични за конкретната фигура, която се рисува. Това са: изрисуване, завъртане и проверяване за колизии.

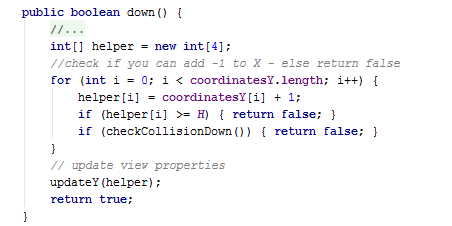




Всяка фигура се има различен алгоритъм за **изрисуване** на полето за игра. Най-общо се намира се горната среда на матрицата и конкретната фигура се изрисува, като всяко квадратче от фигурата се оцветява в солиден цвят. Стартовите координати се записват в класа. След изрисуванети фигурата започва да се мести надолу и също се обновяват координатите й. Това е конкретен пример за изрисуване на „И“ фигурата.

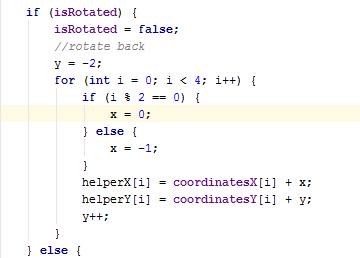
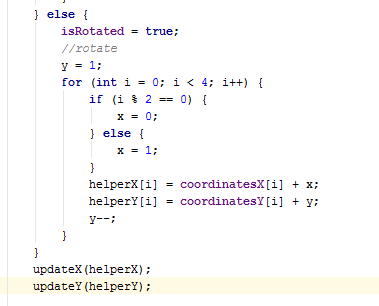


След изрисуването, фигурата започва да „пада“, като през определено време се задейства метода за местене **надолу**. Това става с помощен масив, в който се записват новите координати на фигурата. Ако някоя координата не може да бъде преместена и настъпи колизия – методът прекъсва, а ако не се намери колизия, в случая Y координатата на фигурата, се заместра с помощния масив и цветовете на квадратчетата на двата масива координати се променят.



Алгоритмите за местене на фигурата са аналогични на алгоритъма за **падане** надолу. Метода за **завъртане** е изцяло различен за всяка фигура и се задейства метода в своя клас на фигурата. В него отново се ползва помощни масиви с новите координати и се обновяват и двата масива X и Y.

Това е пример на завъртане от метода на „И“ фигурата.



**Заключение**

Приложението е нейтив и може да работи на всякакъв вид Android устройства. Това разположение на обектите на екрана е постигнато, чрез задаване на съответните стойности на WeigthSum и layout weight, така че големината на обектите да не зависи от предварително зададени стойности, а да се променя спрямо големината на екрана устойството, на което се ползва. За изрисуването и дизайна не са ползвани допълнителни библиотеки, което позволява на приложението да се адаптира за почти всяка версия Android. Дизайнът на кода е направен така, че лесно да се разширява фукциалността на приложението. Това значимо увеличава неговата аудитория. Приложението е инсталирано и тествано на телефон Lenovo P70.