Buse Çarık

Assignment Features:

1. I move all the game play settings into the new Config class.
2. I refactor all the user-facing Strings into string resource. For the player health and distance that is travelled, I get the value as a parameter in the string resource. (%1$d with that way in the resource file).
3. According to your videos, all distinct entities are grandchildren of Entity. I create the BitmapEntity class for the intermediary class to minimize duplication. In the BitmapEntity class, there are 3 method; loadBitmap(…) to load the bitmap and arrange its scale, render(…) and destroy() methods. Player, Star and Enemy classes are children of that class and the BitmapEntity class is the children of Entity class.
4. I improve the Star Entity class. All stars have different and random color and radius. Also, their velocity is set according to their radius. To 3D effect for stars movement, first I set the radius randomly using Game class' random variable. Then I set this random number to the star's velocity. With this way bigger starts move faster and smaller ones move slower. For random color, I create three integer variable which are 3 components of RGB values. Then again with the Game class Random generator, I set random values between 0 and 256 to the RGB values. Then in the render method, I set the star's color with setARGB(…) method to the paint object and draw the stars.
5. I create a new enemy type. For that I create ChallengingEnemy class. It is child of the Enemy class. I override the update() class to give a distinct movement to the new enemy. It moves like oscillating. When it hits to the up, it moves downward and hits to the bottom, move upward. I increase its x velocity and add y velocity to move up and downwards. I add a new variable that is direction to detect the direction of the enemy. I check the direction with method from Utils class which is findDirection(…). It takes the y value of the enemy, top and bottom values of the screen and direction as parameters and change the direction's value as true or false according to the returning value. If the direction is true, it moves upwards else it moves downwards. To movement I add or remove the y velocity to the y value according to the direction. I create one ChallengingEnemy object in the Game class.
6. For the recovery frames, I create a long timer object which is set to the current time of the device. In the checkCollisions() method, first I check the last time of crash. If the current time is bigger than the last crash time plus 2 seconds, then it is not in the recovery frame so it can crash. I update the crash time and do the other operations. To not do the same things twice because of the new enemy type, I divide the checkCollisions() method into two new method. During the recovery frames, do not enter the collision case. For the effect of the recovery frames, I change the alpha value in the paint object. I create a boolean variable which is _collision to detect if the game is in the recovery frames. In the collision

situation, I update the value of the collision and I create a new method which is checkTimer(), to check if the recovery time is finished to change the value of collision. I check the current time and the time last crash and value of the collision for the first case. If the time for recovery is finished, it updates the value of the collision variable. Also, to change the alpha value according to recovery process, I create another method which is setAlphaValue(). First, it calls the checkTimer() method, then according to the value of the collision, it updates the alpha value on paint object. And in the render() method, before render the player, I call the setAlphaValue().

7. I move the HUD rendering into a separate class. First, I create a new class; RenderHUD class. I create a paint object. I add a new font resource. To use that I used the Typeface class. I get the font from assets and set to the paint object. To arrange the text's size, align and color, I use the paint object in the constructor of the RenderHUD class. Then I create two methods to drawing the texts to the screen. One is for showing the player's health and distance that is travelled. I get the playerHealth, canvas and distance as parameters. Then I draw the texts to the canvas with drawText(…) method. In my second method, I draw the game over and restart texts with same way. During the render() method, I check the game over situation, if the game is over, I call the isGameOver() method to draw that situation, if not called isContinue() method to draw the health and distance.
8. I add sound effects for collision, death, boost and start game. I create Jukebox class for that. Mostly I use your code sample. I create a soundpool object. And for representing each situation, I create a new type of enum which is GameEvent. I store the GameEvent and id of sounds in to a HashMap with loadSoundEffects() and loadEventSound() methods. I read ids from assets with AssetFileDescriptor. I create a play() method to play the effects. In the Game class, I create a method (which is also from your code) to play the sound according to the event. It takes GameEvent as a parameter and call the play() method of the _jukebox object to play the sound. I call the boosting event in the onTouchEvent(…) method, when I check the game over situation before calling the restart() method. I have a colliding() method that is called in a colliding situation. I play crash sound in there. When I check the game over situation in the checkGameOver() method, I called onGameEvent(…) method to play the game over effect. And start game effect is called in the run method before entering the loop. I also unload the sound effects in the onDestroy() method.

Resources, Assets and Codes That Are Not Mine:

- Space ships are from https://opengameart.org/content/space-ships-side-scroller.
- Star field is from https://www.flickr.com/photos/galet09/15447326299.
- Launcher icon is from https://openclipart.org/detail/22539/rocket-icon.
- Font that I used in the project is from https://www.dafont.com/theme.font.

- The sound effects are from https://www.bfxr.net/.
- Codes in the project mostly belongs to your videos and the source code in the student portal.
- paint.setARGB(255, _colorR, _colorG, _colorB); this line of code is from https://stackoverflow.com/questions/5280367/android-generate-random-color-on- click.
- _typeface = Typeface.createFromAsset(context.getAssets(), "fonts/theme.ttf"); and _paint.setTypeface(_typeface); are from https://stackoverflow.com/questions/27588965/how-to-use-custom-font-in-a-project- written-in-android-studio.