ТЕХНОЛОГИЧНО УЧИЛИЩЕ ЕЛЕКТРОННИ СИСТЕМИ
към ТЕХНИЧЕСКИ УНИВЕРСИТЕТ - СОФИЯ

ДИПЛОМНА РАБОТА

Тема: Мобилна игра за Android.

Дипломант: *Спас Спасов*

Научен ръководител: *Мартин Григоров*

СОФИЯ

2015

Увод

Мобилната игра е видеоигра, която се играе на мобилно устройство. Тези игри се създават от специализирани софтуерни компании, често в сътрудничество с различни специалисти. В мобилни устройства с операционна система Android се разпространяват чрез онлайн магазин за приложения - Google play. Категорията с мобилни игри в Google play е най-популярна, като те надхвърлят 40% от броя на всички приложения. Мобилните игри достигнаха нови върхове в своето развитие през 2014г. и от просто „убийци“ на време се превърнаха в цели игрални вселени с десетки милиони верни фенове. Така Android вдигна високо летвата и надмина конкурента си iOS по брой на приложения и разработчици.

Android е операционна система на Google Inc. за мобилни устройства, която е една от най-широко разпространените и най-бързо развиващи се мобилни операционни системи в днешно време. За развитието на Android се грижат голям брой софтуерни разработчици, с което тя не показва никакви признаци да спре да се развива.

Има голямо разнообразие от жанрове игри и след много години на иновации е трудно и рядко да се появят изцяло нови. Драг (drag racing) игрите са публикувани за първи път през 2003 г. Основната игра в Drag racing е съвсем проста. Играчите купуват коли, които да персонализират, подобряват, тунинговат и след това се състезават с опоненти в опит да спечелят пари, които им позволяват да повишат съществуващите коли или да си купят нови. Самите състезания се провеждат на прави писти като играчът контролира газта и скоростния лост. Целта е разстоянието да се измине за минимум време.

ПЪРВА ГЛАВА

ПРОУЧВАТЕЛНА ЧАСТ. ПРЕГЛЕД НА СЪЩЕСТВУВАЩИТЕ ПРОГРАМНИ СИСТЕМИ И ПРОДУКТИ И ПРЕГЛЕД НА ИЗВЕСТНИТЕ РАЗВОЙНИ СРЕДСТВА И СРЕДИ

1.1. Преглед на съществуващи мобилни игри от жанр „Drag racing”

1.1.1. CSR Racing

Фигура 1

CSR Racing е на седмо място в раздела „Drag Racing” в Google Play, но и на това място играта не остава по-назад. Дори е в отдела „Избор на редакторите”. Само в магазина Google Play има над 10 милиона изтегляния. Играта е безплатна за изтегляне и съдържа микротранзакции. Чрез микротранзакциите потребителите имат възможност да купуват чипове с реални пари, вместо да ги събират чрез играене. Играта поддържа мултиплейър, което я прави още по-интересна. Чрез онлайн състезанието е дадена възможността потребителите да се състезават един с друг и да печелят специални автомобили. Управлението в играта не е сложно:

1.1.2. Drag Racing

Фигура 2

Играта е на първо място в раздела „Drag Racing” в Google Play и с над 100 милиона тегления. Играта е безплатна за изтегляне и съдържа микротранзакции. Потребителите имат възможността да се състезават с всяка една спортна кола, също така могат да изпробват колата преди да я купят. Играта поддържа мултиплейър. Чрез него могат да се състезават очи в очи както със свои приятели, така и с непознати хора. Управлението в играта не е сложно:
Стартиране и превключване, когато индикаторът стане син или зелен.

Развойни среди

За изработка на приложението е използвано „Unity“ (game engine). Приложението е написано на програмния език „C Sharp“.

ВТОРА ГЛАВА

ПРОЕКТИРАНЕ НА СТРУКТУРАТА НА „DRAG RACING“ МОБИЛНА ИГРА

2.1. Функционални изисквания към програмния продукт

- Реална физика за управление на колата.
- Изготвяне на 3D модели за играта.
- Направа на текстури за играта.
- „Multiplayer“ режим.
- Подобряване на автомобилите.
- Три различни писти.
- „Singleplayer“ режим.
- AI на автомобилите в „Singleplayer“ режим.
- Графичен интерфейс на играта.
- Music & Audio ефекти в играта.
- Facebook интеграция.
- Интеграция на рекламни мрежи за Android.

2.2. Избор на езика за програмиране и програмните средства

2.2.1. Програмни средства

За създаването на приложеанието са използвани гейм енджинът „Unity”, операционна система „Android”, за която е създадена самата игра, като се използва „Android SDK”.

2.2.2. Избор на език за програмиране

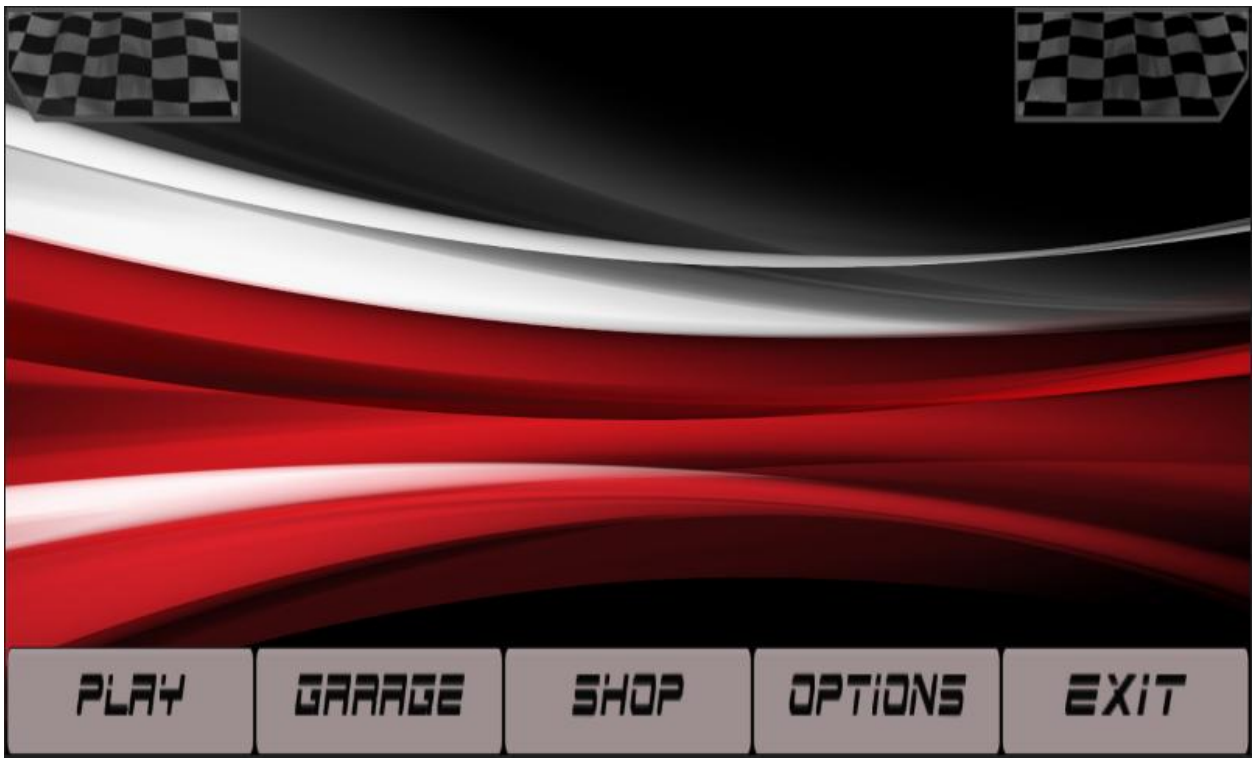
Избраният обектно-ориентиран език за програмиране е „C Sharp”. Гейм енджинът Unity предлага използването на езици за създаване на скриптове като „C Sharp“, „JavaScript“ и „Boo”.

За създаването на приложението са използвани мобилната операционна система „Android”, програмният език „C Sharp“ и средата „Unity“. Unity предлага много лесно реализиране на игри към много платформи, включително и Android, също така предлага различни инструменти, които улесняват работата. Предлага три програмни езика, от които два са скриптови – „JavaScript” и „Boo”. Избран е „C Sharp”, поради предишен опит в подобни обектно ориентирани езици („C“, „C++“ и „Java“). Избрана е платформата „Android”, поради високата и използваемост, тя е най-използваната операционна система в днешни дни.

2.3. Структура на приложението

Проектът е разделен на 3 сцени и съдържа 20 скрипта.

2.3.1. Главно меню - „Main menu“



Фигура 3

В тази сцена потребителят може да избере:

- Play – зареждат се 2 бутона: „singleplayer” и “multiplayer”.
- Garage – Зарежда се „меню-гараж”.
- Shop – Зарежда се „меню-магазин“.
- Options – Зарежда се „меню-опции“.
- Exit – Приложението се изключва.

2.3.2. Меню-плей („Play“)

Фигура 4

В това меню потребителят може да избере:

- Да избере „singleplayer” (бутона от лявата страна).
- Да избере „multiplayer” (бутона от дясната страна).

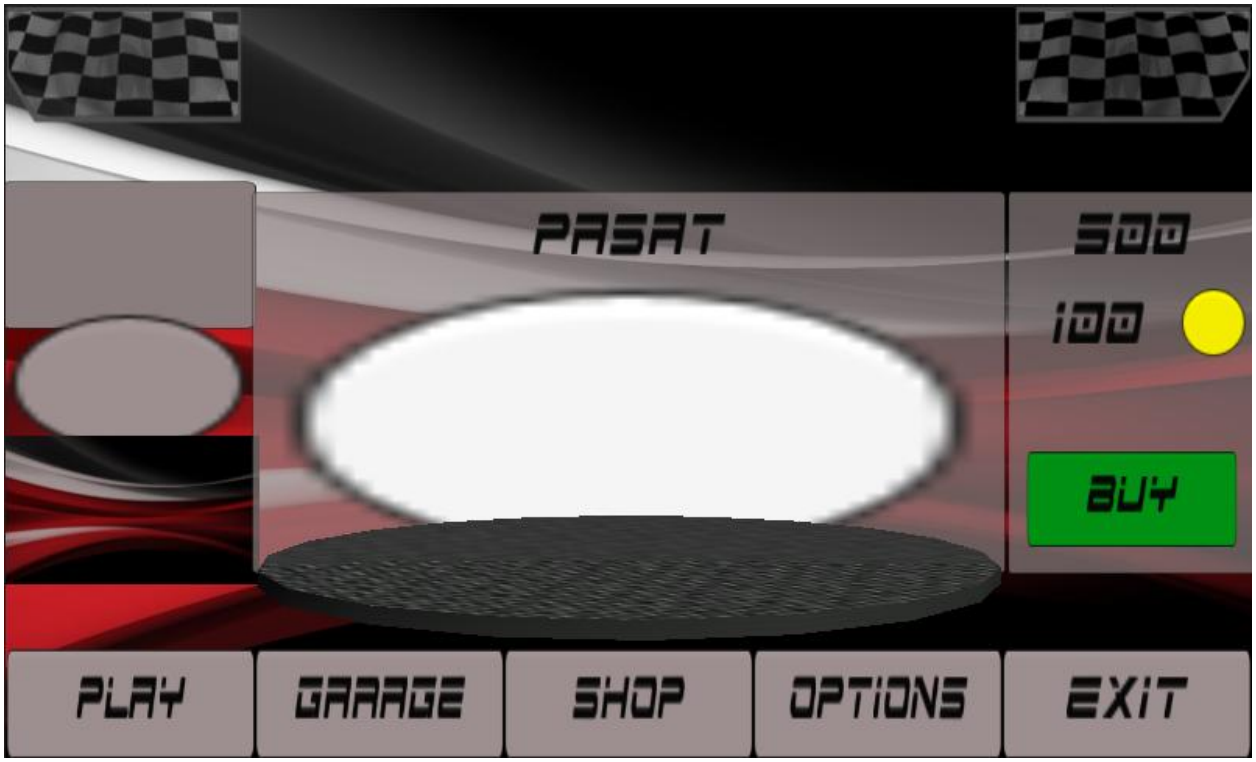
2.3.2.1. „Singleplayer“

- Зарежда сцената “singleplayer”.

2.3.2.2. „Multiplayer“

- Зарежда сцената “multiplayer”.

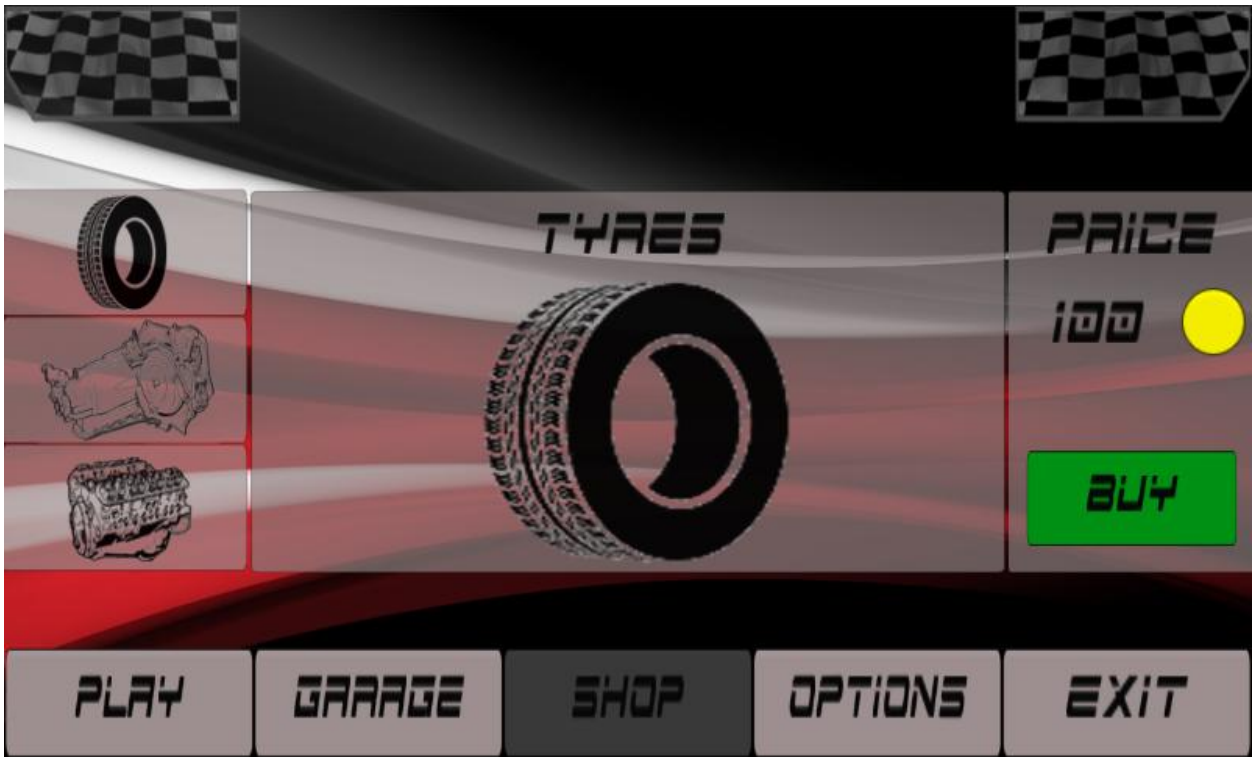
2.3.3. Гараж („Garage“)



Фигура 5

В това меню потребителят може да избере кола с която да се състезава, като колите се отключват на определено достигнато ниво. В дясно са характеристиките на показаната кола. В ляво са моделите коли. Колите ще се отключват на определено достигнато ниво.

2.3.4. Магазин („Shop“)

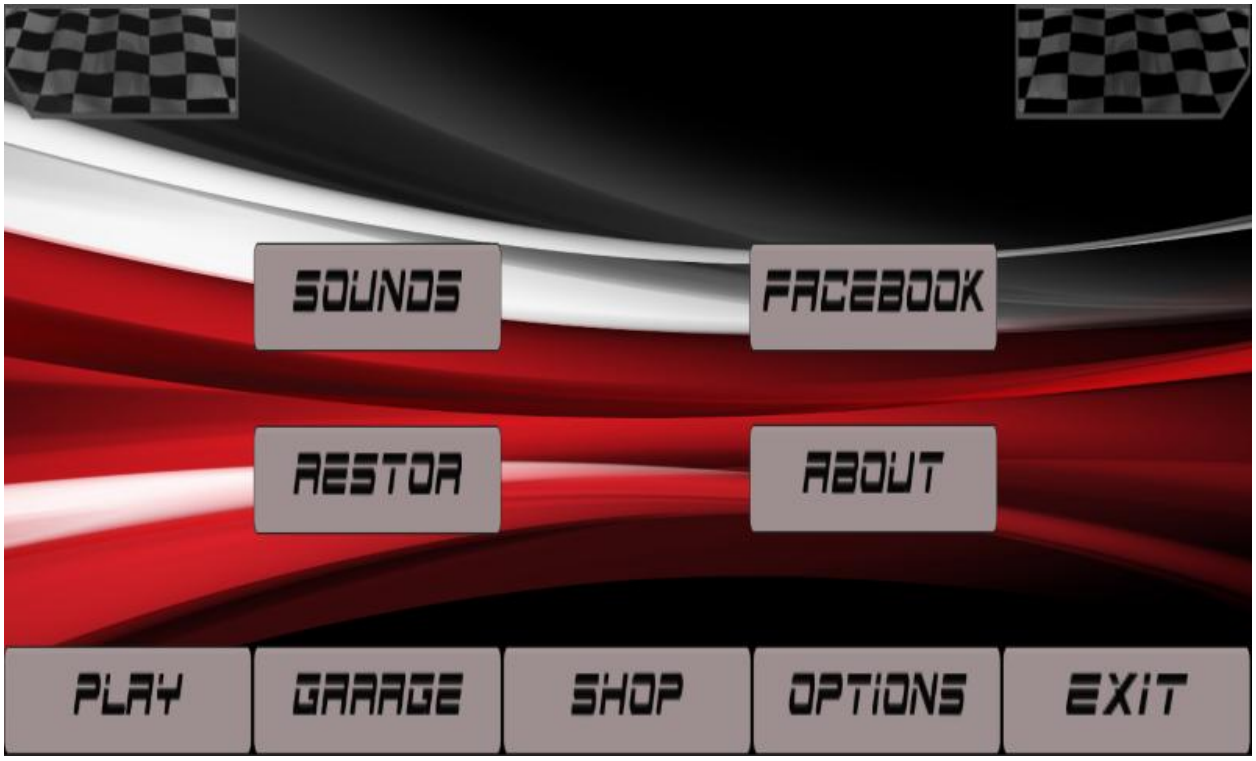


Фигура 6

В това меню потребителят може да закупи чрез виртуалните парични единици в играта, които се събират чрез играене в „Singleplayer“ режим и съответно да надгради отделните части на своята кола, подпомагащи за подобряване на времето и бързината и.

Възможните ъпгрейди са: tyres(„гуми”), gearbox(„скоростна кутия”) и engine(„двигател”). Всеки следващ ъпгрейд е на по висока цена.

2.3.5. Настройки („Options“)



Фигура 7

В това меню са настройките на играта. Потребителят може да управлява звук, може да върне играта в заводските и настройки (factory reset), да се логне в facebook акаунта си и да получи повече информация за играта.

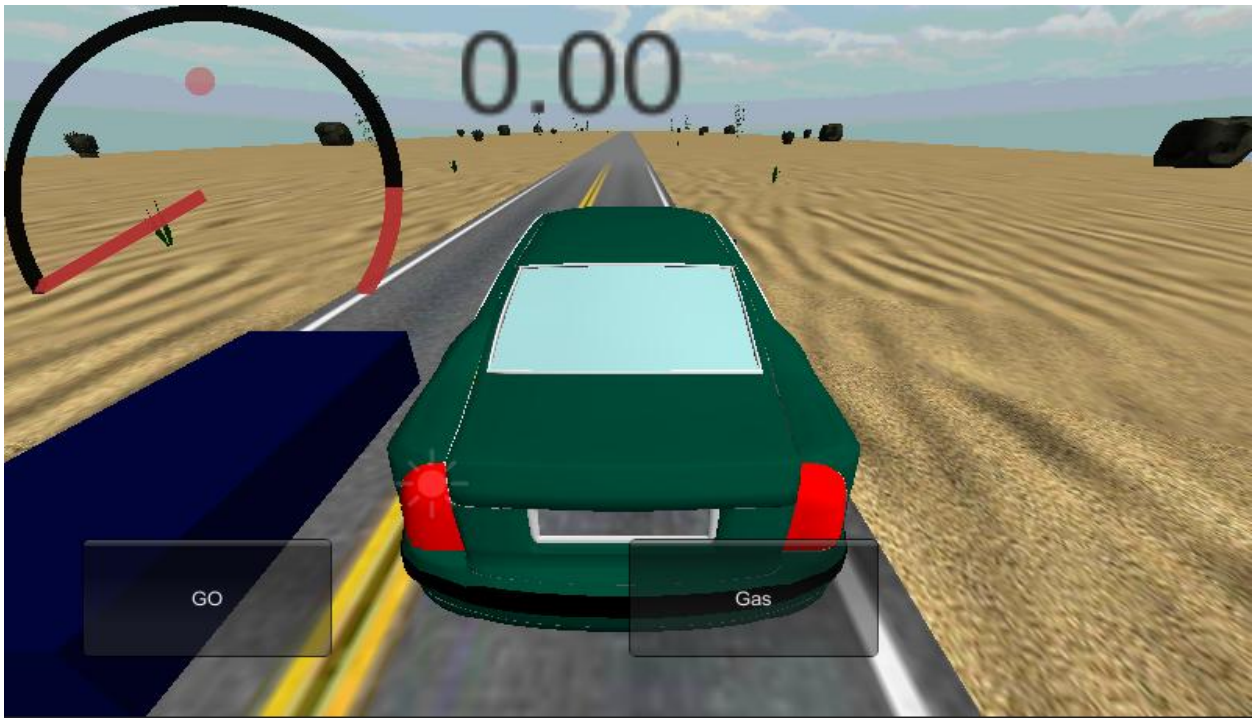
2.3.6. Статистика след завършена игра („gameover“)

Фигура 8

В тази сцена потребителят може да види:

- Дали е спечелил или загубил състезанието.
- Колко виртуални парични единици е събрал от последната игра
- За колко време и спял да завърши състезанието.
- Колко виртуални парични единици има общо

2.3.7. Сингълплейър („singleplayer“)

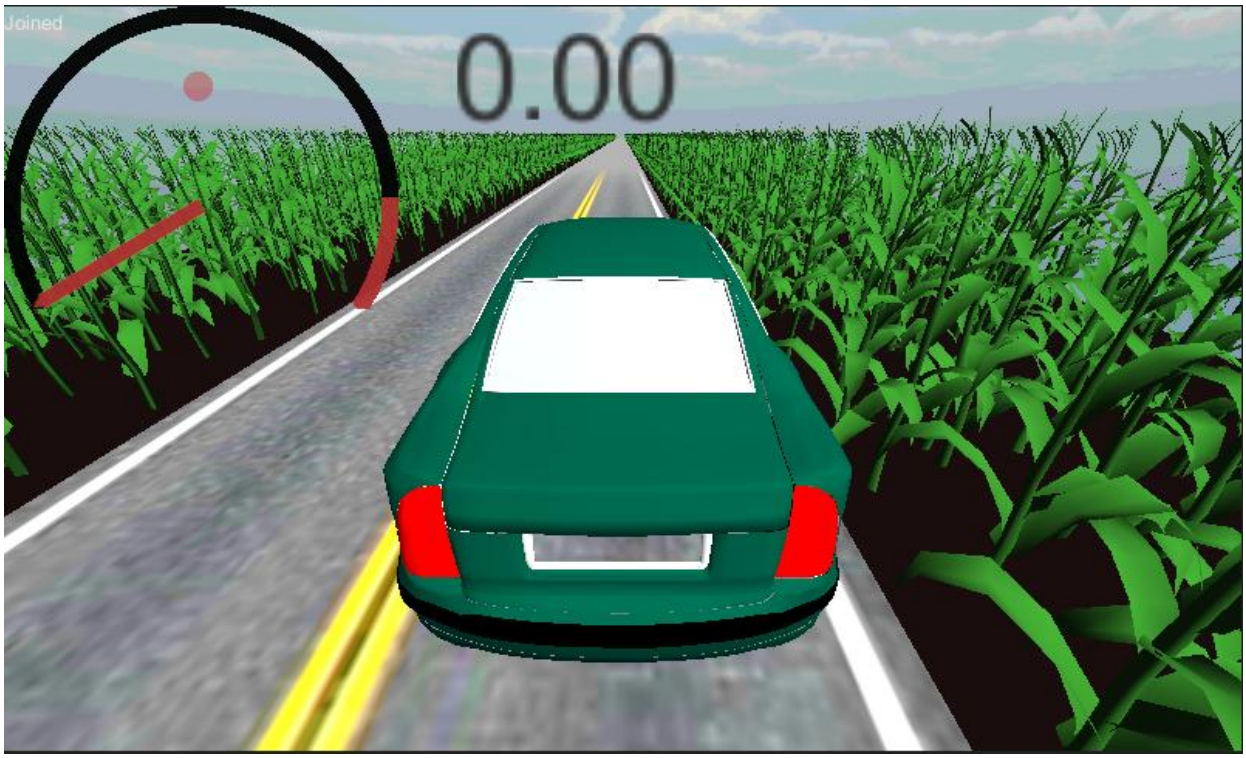


Фигура 9

В тази сцена потребителят управлява скоростите на колата в самата игра. Главната цел на играта е да се измине отсечката за възможно най-малко време, а това става, като се сменят възможно най-точно скоростите в правилния момент. Управлението е улеснено възможно най-много: Просто натискане върху екрана. Има три възможни стадия за смята на скоростта: Перфектно (Perfect), Добре (Good) и Лошо (Bad). „Perfect” дава възможно най-голяма добавка към скоростта, това става точно, като натиснеш екрана докато стрелката на скоростомера сочи средата на червената ивица.

„Good” увеличава с по малко скорост, това става, като натиснеш екрана, когато стрелката на скоростомера сочи началото или края на червената ивица. „Bad” намаля скоростта, това става като натиснеш върху екрана при всички други ситуации: докато стрелката на скоростомера сочи върху черната ивица или след достигане до края на оборотомера и непревключване навреме.

2.3.8. Мултиплейър („multiplayer“)



Фигура 10

Мултиплейърът в приложението работи на подобен принцип като сингълплейъра. След като даден потребител зареди тази сцена и се свърже чрез Photon Unity Networking, играта проверява дали има друг потребител, който чака, и ако има, двата героя тръгват. Ако няма друг потребител, играта е паузирана докато някой друг се присъедини. Печели този, който стигне първи. След това и двамата потребителя са препратени към сцената в **2.3.6.**

ТРЕТА ГЛАВА

ОПИСАНИЕ НА НАЧИНА НА РЕАЛИЗАЦИЯ НА ПРОЕКТА

Ще разгледаме програмата, описвайки работата, която бива извършвана, във всяка една от сцените на приложението, които са три на брой – „mainmenu“, „singleplayer“ и „multiplayer“ като разгледаме всички скриптове на приложението, които са двадесет на брой – „move“, „moveEnemy”, „upgrades“, „spawn“, „spawnEnemy“, „levels“, „NetworkManager“, „networkCharacter“, „mainMenu“, „Play”, „garage“, „garageCar“, „shop”, „upgrades”, „exit”, „stat”, „sounds”, „Ads”, „FBholder” и „NewShader”.

3.1. Скриптове

3.1.1. mainManu

Този скрипт се използва единствено в сцената „**mainmenu**“. Съдържа две функции:

3.1.1.1. Start

В **Start** функцията задаваме имена на бутоните и според резолюцията на устройството наместваме всички детайли.

3.1.1.2. choseGame

В **choseGame** функцията задаваме кои менюта да се визуализират. И избираме да се вижда меню **2.3.2.**

3.1.1.1.

```
void Start () {

		buttonText1.text = "PLAY";

		buttonText2.text = "GARAGE";

		buttonText3.text = "SHOP";

		buttonText4.text = "OPTIONS";

		buttonText5.text = "EXIT";

		camera1.transform.position = new Vector3 (Screen.width/2f, Screen.height/2f, 0);
		input.transform.position = new Vector3 (Screen.width/2f, Screen.height/2f, 506.8f);

		FBloginButton.transform.position = new Vector3 (Screen.width/2f, Screen.height/2f-35, 506.8f);

		if(resolution==better){

			button1.transform.position = new Vector3 (Screen.width/2f-213f, Screen.height/2-128.5f, 506.8f);

			button2.transform.position = new Vector3 (Screen.width/2f-106.5f, Screen.height/2-128.5f, 506.8f);

			button3.transform.position = new Vector3 (Screen.width/2f, Screen.height/2-128.5f, 506.8f);

			button4.transform.position = new Vector3 (Screen.width/2f+106.5f, Screen.height/2-128.5f, 506.8f);

			button5.transform.position = new Vector3 (Screen.width/2f+213f, Screen.height/2-128.5f, 506.8f);

			loginBar.transform.position = new Vector3 (Screen.width/2f-220f, Screen.height/2+106f, 506.8f);

			FBpic.transform.position = new Vector3 (Screen.width/2f-148f, Screen.height/2+127f, 506.8f);

			loadingBar1.transform.position = new Vector3 (Screen.width/2f+220f, Screen.height/2+106f, 506.8f);

		}

		if(resolution==good){

			button1.transform.localScale = new Vector3 (0.41f,1.5f,1);

			button2.transform.localScale = new Vector3 (0.41f,1.5f,1);

			button3.transform.localScale = new Vector3 (0.41f,1.5f,1);

			button4.transform.localScale = new Vector3 (0.41f,1.5f,1);

			button5.transform.localScale = new Vector3 (0.41f,1.5f,1);

			button1.transform.position = new Vector3 (Screen.width/2f-200f, Screen.height/2-128.5f, 506.8f);
			button2.transform.position = new Vector3 (Screen.width/2f-100, Screen.height/2-128.5f, 506.8f);

			button3.transform.position = new Vector3 (Screen.width/2f, Screen.height/2-128.5f, 506.8f);

			button4.transform.position = new Vector3 (Screen.width/2f+100f, Screen.height/2-128.5f, 506.8f);

			button5.transform.position = new Vector3 (Screen.width/2f+200f, Screen.height/2-128.5f, 506.8f);

			loginBar.transform.position = new Vector3 (Screen.width/2f-203.5f, Screen.height/2+106f, 506.8f);

			FBpic.transform.position = new Vector3 (Screen.width/2f-132f, Screen.height/2+127f, 506.8f);

			loadingBar1.transform.position = new Vector3 (Screen.width/2f+203.5f, Screen.height/2+106f, 506.8f);

		}

}
```

3.1.1.2

```
public void choseGame(){
        singlePlayer.gameObject.SetActive (true);
        multiPlayer.gameObject.SetActive (true);
        garage.gameObject.SetActive (false);
        shop.gameObject.SetActive (false);
        stat.play = false;
        stat1.gameObject.SetActive (false);
}
```

3.1.2. move

Този скрипт се използва в сцените „**singleplayer**“ и „**multiplayer**“ и съдържа следните функции: „Update”, „OnGUI”, „OnTriggerEnter”, „Pshifts”, „Gshifts” и „Bshifts”.

3.1.2.1. Update

[4]

В **Update** функцията е логиката на движението на колата. Кога и как да се променя скоростта. Проверяваме за ъпгрейдите и при всяка скорост според превключената скорост правилно или не съответно се увеличава или намалява скоростта.

3.1.2.2. OnGUI

Чрез **OnGUI** функцията ползваме „невидим” бутон за превключването на скоростите и знаем кога е „Perfect”, „Good” или „Bad”.

3.1.2.3. OnTriggerEnter

Чрез **OnTriggerEnter** функцията разбираме кой и кога е завършил състезанието.

3.1.2.4. Pshifts, Gshifts, Bshifts

Тези функции са *IEnumerator* функции, които са по специални, в тях се извършват времеви функции(ползвам ги за yield/*пауза*/). Чрез тях се визуализира ускорението на автомобила.

3.1.2.1.1.

```
void Update () {
	if(play.SingleOrMulti == false){
		if ((NetworkManager.ttt || PhotonNetwork.countOfPlayers == 2)&& shift < 2) {
			A = true;
		}
	}
	time.text = timer.ToString("F2");
	if (finish == 1 || finish == 2) {
		PlayerPrefs.SetInt("stat",1);
		Application.LoadLevel("mainMenu");
	}
	if(PlayerPrefs.GetInt ("upgrade")==1){
		if(once){
			go = 0.65f;
			once = false;
		}
	}
```

3.1.2.1.2.

```
if(PlayerPrefs.GetInt ("upgrade")==2){
                        if(once){
                                go = 0.80f;
                                once = false;
                        }
                }
                if(PlayerPrefs.GetInt ("upgrade")==3){
                        if(once){
                                go = 1.05f;
                                once = false;
                        }
                }
```

3.1.2.1.3.

```
if (end == false) {
        if (finish == 1) {
                end = true;
                        PlayerPrefs.SetInt("subLevel", PlayerPrefs.GetInt ("subLevel")+1);
                }
                if (finish == 2) {
                        end = true;
                }
        }
        if (start) {
                if (speed < 80) {
                        if(shift==1)
                                speed += 1.5f;
                        if(shift==2)
                                speed += 1.3f;
                        if(shift==3)
                                speed += 1.1f;
                        if(shift==4)
                                speed += 0.9f;
                        if(shift==5)
                                speed += 0.7f;
                        if(shift>5)
                                speed += 0.6f;
                        if (speed < 0)
                                speed = 0;
                }
        }
```

3.1.2.1.4.

```
if (perfect) {
		second = second + Time.deltaTime;
		sec = (int)second;
		for (i=0; i<10; i++) {
			if (sec < 2) {
				transform.Translate (new Vector3 (0, 0, .001f));
			} else {
				perfect = false;
				second = 0;
			}
		}
}
if (slowDown >= 0){
		if (bad) {
			second = second + Time.deltaTime;
			sec = (int)second;
			for (i=0; i<10; i++) {
				if (sec < 2) {
					transform.Translate (new Vector3 (0, 0, -.001f));
				} else {
					bad = false;
					second = 0;
				}
			}
		}
}
```

3.1.2.1.5.

```
if (A) {
		second = second + Time.deltaTime;
		sec = (int)second;
		if(sec==5){
			B = true;
			start=true;
			shift = 1;
			A=false;
		}
}
if (speedDown == false) {
		if (speed > 0) {
			speed -= 0.5f;
		}
		if (speed < 0) {
			speed = 0;
		}
}
if(B){
		if(finish==0)
			timer += Time.deltaTime;
			transform.Translate (new Vector3 (0, 0, go));
		}
}
```

3.1.2.2.1.

```
void OnGUI () {
	if (speed < 69) {
		GUI.DrawTexture (new Rect (position.x, position.y, bar1.width, bar1.height), bar1);
	} else {
		GUI.DrawTexture (new Rect (position.x, position.y, bar2.width, bar2.height), bar2);
	}
	if (speed == 70) {
		GUI.DrawTexture (new Rect (position.x, position.y, bar1.width, bar1.height), bar1);
	}
	if (speed == 71) {
		GUI.DrawTexture (new Rect (position.x, position.y, bar2.width, bar2.height), bar2);
	}
	if (speed == 72) {
		GUI.DrawTexture (new Rect (position.x, position.y, bar1.width, bar1.height), bar1);
	}
	if (speed == 73) {
		GUI.DrawTexture (new Rect (position.x, position.y, bar2.width, bar2.height), bar2);
	}
	if (speed == 74) {
		GUI.DrawTexture (new Rect (position.x, position.y, bar1.width, bar1.height), bar1);
	}
	if (speed == 75) {
		GUI.DrawTexture (new Rect (position.x, position.y, bar2.width, bar2.height), bar2);
	}
	if (speed == 76) {
		GUI.DrawTexture (new Rect (position.x, position.y, bar1.width, bar1.height), bar1);
	}
```

3.1.2.2.2.

```
Vector2 centre = new Vector2 (position.x + bar1.width / 2, position.y + bar1.height / 2);
var savedMatrix = GUI.matrix;
var speedFraction = speed / topSpeed;
var needleAngle = Mathf.Lerp (stopAngle, topSpeedAngle, speedFraction);
partSpeed = needleAngle / 100;
//GUIUtility.RotateAroundPivot(needleAngle, centre);
for (int i=0; i<100; i++) {
        //StartCoroutine(Wait());
        GUIUtility.RotateAroundPivot (partSpeed, centre);
}
GUI.DrawTexture (new Rect (centre.x, centre.y - arrow.height / 2, arrow.width, arrow.height), arrow);
GUI.matrix = savedMatrix;

if(GUI.Button(new Rect(Screen.width-(Screen.width-50),Screen.height-120,160,80), "GO")) {
        shift++;
        if(shift<=6){
//perfect
                if(speed >= 73 && speed <= 76){
                        Debug.Log("PERFECT!!!!!!!!!!!!!!!");
                        perfect = true;
                        perfectShift = true;
                        perfectShifts+=1;
                        //go+=0.1f;
                        StartCoroutine(Pshifts());
                        if(slowDown<=5){
                                slowDown++;
                        }
                }
```

3.1.2.2.3.

```
        //good
                if(speed > 69 && speed < 72 || speed >77 && speed <79){
                        Debug.Log("GOOD!!!!!!!!!!!!!!!");
                        perfect = true;
                        goodShift = true;
                        goodShifts++;
                        StartCoroutine(Gshifts());
                        if(slowDown<=5){
                                slowDown++;
                        }
                }
                //bad
                if(speed<=69 || speed >= 79){
                        Debug.Log("BAD!!!!!!!!!!!!!!!");
                        bad = true;
                        if(slowDown>=0){
                                StartCoroutine(Bshifts());
                        }
                        slowDown--;
                }

                speed-=50;

                }
        }
}
```

3.1.2.3.

```
void OnTriggerEnter(Collider other){
	if (other.gameObject.tag == "FinishLine") {
		if(first){
			finish = 1;
			first = false;
		}
	}
}
```

3.1.2.4.

```
IEnumerator Pshifts() {
	for(int i=0;i<3;i++){
		yield return new WaitForSeconds(0.33f);
		go+=0.033f;
	}
}
```

```
IEnumerator Gshifts() {
	for(int i=0;i<3;i++){
		yield return new WaitForSeconds(0.33f);
		go+=0.0166f;
	}
}
```

```
IEnumerator Bshifts() {
	for(int i=0;i<3;i++){
		yield return new WaitForSeconds(0.33f);
		go+=0.033f;
	}
}
```

3.1.3. moveEnemy

Този скрипт се използва единствено в сцената „**singleplayer**“. Има за цел да движи бота в сингълплейъра по конкуриращ начин. Съдържа две функции: „Update” и „OnTriggerEnter”.

3.1.3.1. Update

В **Update** функцията проверяваме на какво ниво е потребителят и според това опонента се движи спрямо него.

3.1.3.2. OnTriggerEnter

Както и преди: Разбираме кой, кога е свършил /първи, втори/.

3.1.3.1.

```
void Update () {
if(PlayerPrefs.GetInt ("subLevel")==1){
	if(once){
		speed = Random.Range(0.54f, 0.56f);
		once = false;
	}
}
if(PlayerPrefs.GetInt ("subLevel")==2){
	if(once){
		speed = Random.Range(0.64f, 0.66f);
		once = false;
	}
}
if(PlayerPrefs.GetInt ("subLevel")==3){
	if(once){
		speed = Random.Range(0.69f, 0.71f);
		once = false;
	}
}
if(PlayerPrefs.GetInt ("subLevel")==4){
	if(once){
		speed = Random.Range(0.79f, 0.81f);
		once = false;
	}
}
if(PlayerPrefs.GetInt ("subLevel")==5){
	if(once){
		speed = Random.Range(0.84f, 0.86f);
		once = false;
	}
}
if(PlayerPrefs.GetInt ("subLevel")==6){
	if(once){
		speed = Random.Range(0.94f, 0.96f);
		once = false;
	}
}
if(PlayerPrefs.GetInt ("subLevel")==7){
	if(once){
		speed = Random.Range(1.09f, 1.11f);
		once = false;
	}
}
if(playerScript.shift>0)
	start = true;

	if (start)
	transform.Translate (new Vector3 (0, 0, speed));
	if (playerScript.perfectShift) {
		speed+=Random.Range(0.04f, 0.06f);
		playerScript.perfectShift=false;
	}
	if (playerScript.goodShift) {
		speed+=Random.Range(0f, 0.2f);
		playerScript.goodShift=false;
	}

}
```

3.1.3.2.

```
    void OnTriggerEnter(Collider other){
        if (other.gameObject.tag == "FinishLine") {
            if(move.first){
                move.finish = 2;
                move.first = false;
            }
        }
    }
```

3.1.4. Shop

Този скрипт се използва единствено в сцената „**mainmenu**“. Има за цел да съдържа инструменти с които играчите да подобряват автомобила си.Съдържа пет функции: „Start”, „shopMenu”, „tyres”, „gearbox” и „engine”.

3.1.4.1. Start

В **Start** функцията подреждаме бутоните на менюто – както в другите скриптове.

3.1.4.2. shopMenu

В **shopMenu** функцията разрешаваме при натискането на дадения бутон да отвори менюто.

3.1.4.3. Tyres

Това е най-малкият ъпгрейд - добавя най-малко скорост.

3.1.4.4. Gearbox

Това е вторият ъпгрейд - добавя повече скорост.

3.1.4.5. Engine

Това е третият ъпгрейд – дава възможно най-много скорост.

3.1.4.1.

```
void Start () {
        if (resolution == good) {
        button1.transform.position = new Vector3 (Screen.width/2-201, Screen.height/2+51, 506.8f);
        button2.transform.position = new Vector3 (Screen.width/2-201, Screen.height/2, 506.8f);
        button3.transform.position = new Vector3 (Screen.width/2-201, Screen.height/2-51, 506.8f);
        panel1.transform.position = new Vector3 (Screen.width/2, Screen.height/2, 506.8f);
        panel1.transform.localScale = new Vector3 (0.59f, 0.48f, 1);
        panel2.transform.position = new Vector3 (Screen.width/2+202, Screen.height/2, 506.8f);
        }
        if (resolution == better) {
        button1.transform.position = new Vector3 (Screen.width/2-218, Screen.height/2+51, 506.8f);
        button2.transform.position = new Vector3 (Screen.width/2-218, Screen.height/2, 506.8f);
        button3.transform.position = new Vector3 (Screen.width/2-218, Screen.height/2-51, 506.8f);
        panel1.transform.position = new Vector3 (Screen.width/2, Screen.height/2, 506.8f);
        panel2.transform.position = new Vector3 (Screen.width/2+219, Screen.height/2, 506.8f);
        }
}
```

3.1.3.2.

```
        public void tyres(){
                tyersPic.gameObject.SetActive (true);
                gearboxPic.gameObject.SetActive (false);
                enginePic.gameObject.SetActive (false);
                Price.text = "100";
                tyres1 = true;
                gearbox1 = false;
                engine1 = false;
        }
```

3.1.3.3.

```
public void gearbox(){
        tyersPic.gameObject.SetActive (false);
        gearboxPic.gameObject.SetActive (true);
        enginePic.gameObject.SetActive (false);
        Price.text = "500";
        tyres1 = false;
        gearbox1 = true;
        engine1 = false;
}
```

3.1.3.4.

```
public void engine(){
        tyersPic.gameObject.SetActive (false);
        gearboxPic.gameObject.SetActive (false);
        enginePic.gameObject.SetActive (true);
        Price.text = "1000";
        tyres1 = false;
        gearbox1 = false;
        engine1 = true;
}
```

3.1.5. Garage

Този скрипт се използва единствено в сцената „**mainmenu**“. Има за цел да съдържа различни коли, които се отключват на достигнато ниво, с които играчите да се състезават. Съдържа пет функции: „Start”, „garageMenu”, „car1a”, „car2a” и „car3a”.

3.1.5.2. garageMenu

В **garageMenu** функцията разрешаваме при натискането на дадения бутон да отвори менюто.

3.1.5.3. car1a, car2a и car3a

Визуализиране на различните коли. При всяко натискане на даден бутон да се визуализира дадената а другите не.

3.1.5.2.

```
public void garageMenu(){
        PlayerPrefs.DeleteKey ("stat");
        statMenuGameObject.gameObject.SetActive (false);
        shopMenuGameObject.gameObject.SetActive (false);
        garageMenuGameObject.gameObject.SetActive (true);
        garageScript.gameObject.SetActive (true);
        disName.gameObject.SetActive (true);
        singlePlayer.gameObject.SetActive (false);
        multiPlayer.gameObject.SetActive (false);
}
```

3.1.5.3.

```
public void car1a(){
        car1.gameObject.SetActive (true);
        car2.gameObject.SetActive (false);
        car3.gameObject.SetActive (false);
        Price.text = "100";
        tyres1 = true;
        gearbox1 = false;
        engine1 = false;
}
```

```
public void car2a(){
        car1.gameObject.SetActive (false);
        car2.gameObject.SetActive (true);
        car3.gameObject.SetActive (false);
        Price.text = "500";
        tyres1 = false;
        gearbox1 = true;
        engine1 = false;
}
```

```
public void car3a(){
        car1.gameObject.SetActive (false);
        car2.gameObject.SetActive (false);
        car3.gameObject.SetActive (true);
        Price.text = "1000";
        tyres1 = false;
        gearbox1 = false;
        engine1 = true;
}
```

3.1.6. Stat

Този скрипт се използва единствено в сцената „**mainmenu**“. Появява се меню с данни само след финала. Има за цел да информира за състезанието. Съдържа три функции: „Start”, „Update” и „statMenu”.

3.1.6.2. Update

Скрипта **Update** връща меню с характеристика за състезанието.

3.1.6.2. Update

Чрез **statMenu** знаем кога да се визуализира менюто.

3.1.6.2.

```
void Update () {
        if (PlayerPrefs.GetInt ("stat") == 1)
        if(play)
                statMenu ();
        if(move.finish == 1){
        myTimer.text = move.timer.ToString("f2");

        enemyTimer.gameObject.SetActive (false);

        myUpgr.gameObject.SetActive (false);

        enemyUpgr.gameObject.SetActive (true);
        WinOrLose.text = "You Win";
        prize1.text = "1000";
        prize2.text =
(move.perfectShifts*10).ToString();
        prize3.text = (move.goodShifts*5).ToString();
        playerName.color = Color.green;
        enemyName.color = Color.red;
        }
if(move.finish == 2){
        enemyTimer.text = move.timer.ToString("f2");
        myTimer.gameObject.SetActive (false);

        enemyUpgr.gameObject.SetActive (false);
        myUpgr.gameObject.SetActive (true);
        WinOrLose.text = "You Lose";
prize2.text = (move.perfectShifts*10).ToString();
        prize3.text = (move.goodShifts*5).ToString();
        enemyName.color = Color.green;
        playerName.color = Color.red;
}

playerName.text = PlayerPrefs.GetString ("name");
}
```

3.1.6.3. statMenu

```
public void statMenu(){

        statMenuGameObject.gameObject.SetActive (true);

        disName.gameObject.SetActive (true);

}
```

3.1.7. networkCharacter

Този скрипт се използва единствено в сцената „**multiplayer**“. Обработване на позицията спрямо времето. Съдържа две функции: „Update” и „OnPhotonSerializeView”.

3.1.7.1. Update

Обработваме времето и мястото на съперника.

3.1.7.2. OnPhotonSerializeView

Препращаме данните на позиция и време.

3.1.7.1.

```
void Update (){

        if (!photonView.isMine){

                timeToReachGoal = (currentPacketTime-5) / lastPacketTime;

                currentTime += Time.deltaTime;

                transform.position = Vector3.Lerp(positionAtLastPacket, realPosition, (float)(currentTime /
                timeToReachGoal));

        }

}
```

3.1.7.2.

```
void OnPhotonSerializeView(PhotonStream stream, PhotonMessageInfo info){
        if (stream.isWriting){
                stream.SendNext((Vector3)transform.position);
        }
        else{
        currentTime = 0.0;
        positionAtLastPacket = transform.position;
        realPosition = (Vector3)stream.ReceiveNext();
        lastPacketTime = currentPacketTime;
        currentPacketTime = info.timestamp;
        }
}
```

3.1.8. NetworkManager

Този скрипт се използва единствено в сцената „**multiplayer**“. Трансфер на данни чрез ***PhotonNetwork***. Съдържа девет функции: „Start”, „Connect”, „OnGUI”, „OnJoinedLobby”, „OnPhotonRandomJoinFailed”, „OnPhotonRandomJoinFailed”, „OnPhotonPlayerConnected”, „Update” и „SpawnMyPlayer”.

3.1.8.1. Start,Connect,OnGUI,OnJoinedLobby,OnPhotonRandomJoin Failed,OnPhotonRandomJoinFailed,OnPhotonPlayerConnected

Имат за цел да създадат стая ако няма съществуваща, и да обедени играчите на едно място.

3.1.8.2. SpawnMyPlayer

Spawn-ва играч на определена позиция спрямо това кой поред се е свързал.

3.1.8.3. Update

Изчислява колко хора са се свързали.

3.1.8.1.

```
void Connect() {
        PhotonNetwork.ConnectUsingSettings ("Shift");
}
```

```
void OnGUI() {
        GUILayout.Label (PhotonNetwork.connectionStateDetailed.ToString ());
}
```

```
void OnJoinedLobby() {
        PhotonNetwork.JoinRandomRoom ();
}
```

```
void OnPhotonRandomJoinFailed() {
        PhotonNetwork.CreateRoom (null);
}
```

```
void OnJoinedRoom(){
        SpawnMyPlayer ();
}
```

```
void OnPhotonPlayerConnected(PhotonPlayer connectedq){
        ttt = true;
}
```

3.1.8.2.

```
void SpawnMyPlayer() {

        GameObject myPlayerGO = (GameObject) PhotonNetwork.Instantiate ("PlayerController4",new
        Vector3(position,0,0),Quaternion.identity,0);

        connected++;//"PlayerController"

        myPlayerGO.transform.FindChild ("Main Cam").gameObject.SetActive (true);

        }
```

3.1.8.3.

```
        void Update() {

                connected = PhotonNetwork.countOfPlayers;

                if(connected == 2)

                        position = -2.4f;

                Debug.Log ("connected: " + connected);

        }
```

3.1.9. FBholder

Този скрипт се използва единствено в сцената „**mainmenu**“. Интеграция на facebook. Съдържа девет функции: „Awake”, „SetInit”, „OnHideUnity”, „FBlogin”, „AuthCallback”, „User”, „ProfilePic”, „UserName” и „Share”.

```
        void Awake(){

                FB.Init (SetInit, OnHideUnity);

        }
```

```
private void SetInit(){
	Debug.Log ("FB Init done.");
	if (FB.IsLoggedIn) {
		Debug.Log("FB Logged in");
	}
	else{
	}
}
```

```
private void OnHideUnity(bool isGameShown){
	if (!isGameShown) {
		Time.timeScale = 0;
	}else{
		Time.timeScale = 1;
	}
}
```

```
public void FBlogin(){
	FB.Login ("email", AuthCallback);
}
```

```
void AuthCallback(FBResult result){
	if (FB.IsLoggedIn) {
		Debug.Log("FB login worked!");
	}
	else{
		Debug.Log("FB Login fail");
	}
}
```

```
public void User(){
        FB.API (Util.GetPictureURL ("me", 128, 128), Facebook.HttpMethod.GET, ProfilePic);
        FB.API ("/me?fields=id,first_name",Facebook.HttpMethod.GET,UserName);
}
```

```
void ProfilePic(FBResult result){
        if (result.Error != null) {
                Debug.Log("Problem get pic");
                FB.API (Util.GetPictureURL ("me", 128, 128), Facebook.HttpMethod.GET, ProfilePic);
                return;
        }
        Image UserAvatar = FBPicture.GetComponent<Image> ();
        UserAvatar.sprite = Sprite.Create (result.Texture, new Rect (0, 0, 128, 128), new Vector2 (0, 0));
}
```

```
void UserName(FBResult result){
        if (result.Error != null) {
                FB.API ("/me?fields=id,first_name",Facebook.HttpMethod.GET,UserName);
                return;
        }
        profile = Util.DeserializeJSONProfile (result.Text);
        Text UserMsg = FBUserName.GetComponent<Text> ();
        UserMsg.text = profile["first_name"];
}
```

```
public void Share(){

	FB.Feed (

		linkCaption: "Can you beat me?",

		picture: "http://www.brandsoftheworld.com/sites/default/files/styles/logo-thumbnail/public/052012/fifthgear_logo.jpg",

		linkName: "I beat them.",

		link: "http://apps.facebook.com/" + FB.AppId + "/?challenge_brag=" + (FB.IsLoggedIn ? FB.UserId : "guest")

		);

}
```

3.1.10. Ads

Този скрипт се използва единствено в сцената „**mainmenu**“. Интеграция на реклами. Съдържа две функции: „Awake” и „Update”.

3.1.10.1. Awake

Проверяваме дали се подържа платформата.

3.1.10.2. Update

Логиката кога да се пуска рекламата ако има достъп до такава.

На всеки три игри се пуска реклама.

3.1.10.1.

```
	void Awake() {

		if (Advertisement.isSupported) {

			Advertisement.allowPrecache = true;

			Advertisement.Initialize ("24046", false);

		} else {

			Debug.Log("Platform not supported");

		}

	}
```

3.1.10.2.

```
void Update(){
        //PlayerPrefs.DeleteKey ("showAds");

        if (move.finish == 1 || move.finish == 2) {
                if(once){
                        PlayerPrefs.SetInt("showAds", PlayerPrefs.GetInt ("showAds")+1);
                        Debug.Log("THISSS: "+PlayerPrefs.GetInt ("showAds"));
                        once = false;
                }
        }

        if (PlayerPrefs.GetInt ("showAds") == 3 && Advertisement.isReady()) {
                Advertisement.Show(null, new ShowOptions {
                        pause = true,
                        resultCallback = result => {
                                Debug.Log(result.ToString());
                                PlayerPrefs.SetInt("showAds", 0);
                        }

                });

        }
}
```

ЧЕТВЪРТА ГЛАВА

РЪКОВОДСТВО ЗА ПОТРЕБИТЕЛЯ

3.1. Изисквания на приложението

За използването на приложението е нужно наличието на мобилно устройство с операционна система Android - смартфон или таблет с версия на операционната система 2.3.1 и нагоре.

3.2. Инсталация на приложението

За инсталацията на приложението е необходимо потребителят да разполага с APK файл на приложението в мобилното си устройство. Когато потребителят отвори този файл, той трябва да се съгласи със следните изисквания на приложението:

- Промяна или изтриване на данни
- Пълен достъп до мрежата
- Преглед на мрежовите връзки
- Тестване на достъп до защитени данни

Ако потребителят е съгласен с изискванията, трябва да натисне бутона “Install”. При натискането на този бутон, инсталацията на приложението започва. Когато инсталацията е успешна, се изписва “Application installed”. Тогава потребителят има избор – да натисне бутона „Done“ и да затвори текущия прозорец или да натисне бутона „Open“ и да отвори приложението. Отворено, приложението въвежда

потребителя в главното меню. В това меню потребителят има избор от следните бутони:

- Shop – при натискане потребителят отива в магазина на играта, от където може да надгради отделните части на своята кола;
- Garage – при натискане потребителят отива в гаража, където може да смени колата си, ако е достигнал необходимото ниво;
- Options – при натискане потребителят отваря меню с настройките на играта;
- Play – при натискане потребителят може да избере между Single и Multiplayer;
- Exit – приложението се затваря.

В противен случай потребителят може да откаже инсталацията на приложението.

ИЗПОЛЗВАНА ЛИТЕРАТУРА

1. Unity Script Reference, http://docs.unity3d.com/ScriptReference/
2. UnityAds Knowledge Base, https://unityads.unity3d.com/help/index
3. Documentation for building software with Facebook, https://developers.facebook.com/docs
4. Photon Unity Networking, http://doc-api.exitgames.com/en/pun/current/pun/doc/general.html
5. Unity Communit, http://forum.unity3d.com

Съдържание