

POLITECHNIKA ŚLĄSKA W GLIWICACH

WYDZIAŁ ELEKTRYCZNY

Instytut Elektrotechniki i Informatyki

**PROJEKT INŻYNIERSKI**

**Symulacja manewrów samochodowych wykonywanych na kursie prawa jazdy kategorii B.**

**Symulation of car maneuvers used on driving course category B.**

Student: **Grzegorz Kazimierz LISOWSKI**

Nr albumu: 210414

Studia: Stacjonarne I stopnia

Kierunek: Informatyka

Specjalność:

Prowadzący: dr inż. Maciej Sajkowski

Recenzent:

Spis treści

[1. Wstęp 5](#_Toc408246261)

[2. Cel i zakres projektu 6](#_Toc408246262)

[3. ? 7](#_Toc408246263)

[4. Opis technologii 8](#_Toc408246264)

[5. Projekt 9](#_Toc408246265)

[5.1 Menu główne 9](#_Toc408246266)

[5.2 Otoczenie 9](#_Toc408246267)

[5.3 Pojazd 10](#_Toc408246268)

[5.4 Scena pierwsza: łuk 10](#_Toc408246269)

[6. Implementacja 11](#_Toc408246270)

[7. Instrukcja użytkownika 12](#_Toc408246271)

[8. Podsumowanie i wnioski 13](#_Toc408246272)

# Przedmiot projektu

Dla młodej osoby która wchodzi w dorosłe życie kamieniem milowym jest zdanie egzaminu na prawo jazdy, chociaż istnieje dużo materiałów pozwalających przećwiczyć teoretyczną część kursu prawa jazdy a także prześledzić cały egzamin to brakuje narzędzia pozwalającego zapoznać się z wymaganymi na kursie manewrami i zapoznać się z poprawną metodą ich wykonywania.

# Cel i zakres projektu

Celem tej pracy inżynierskiej jest stworzenie narzędzia mającego pomóc osobom przygotowującym się do zdawania egzaminu na prawo jazdy poprzez interaktywną naukę oraz gamifikację manewrów wymaganych na kursie prawa jazdy. Narzędzie tego typu powinno być w stanie pomóc w przygotowaniach do kursu prawa jazdy osobom nie uprawnionym do przystąpienia do kursu prawa jazdy ze względu na nie osiągnięcie odpowiedniego wieku.

Zakres projektu obejmuje stworzenie menu głównego gry, przygotowanie otoczenia, zaprojektowanie trzech scenariuszy manewrów wykorzystywanych w nauce jazdy, implementacja narzędzi potrzebnych do weryfikacji wykonania manewrów

# Założenia projektu

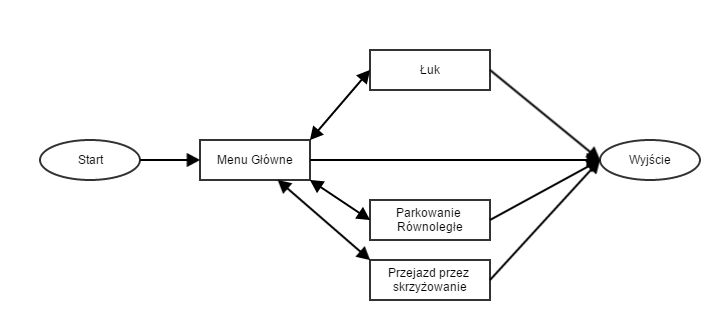
# Opis technologii

Jako silnik wybrana została technologia Unity3D w wersji 4.6 beta 20. Wybór ten spowodowany jest tym, że jest to narzędzie darmowe, pozwalające na szybkie tworzenie prototypów, dostępna jest obszerna i aktualizowana na bieżąco dokumentacja, technologia ta posiada niski próg wejścia.

# Projekt

## Menu główne

Jako menu główne zastosowano prosty interfejs użytkownika zawierający tytuł aplikacji oraz przyciski. Jako tło zastosowano trójwymiarowe otoczenie.



rys. 1 Schemat przejść między scenami

Na schemacie przejść między scenami (rys. 1) ukazano pożądany sposób działania menu głównego widać na nim, że w menu głównym należy umieścić 3 przyciski odpowiedzialne za przejście na poszczególne sceny oraz przycisk opuszczenia aplikacji.

## Otoczenie

Otoczenie w którym odbywają się sceny składa się z 3 elementów: podłoża, budynku oraz muru.

Podłoże jest wykonane jako terrain component na który nałożono teksturę betonu.

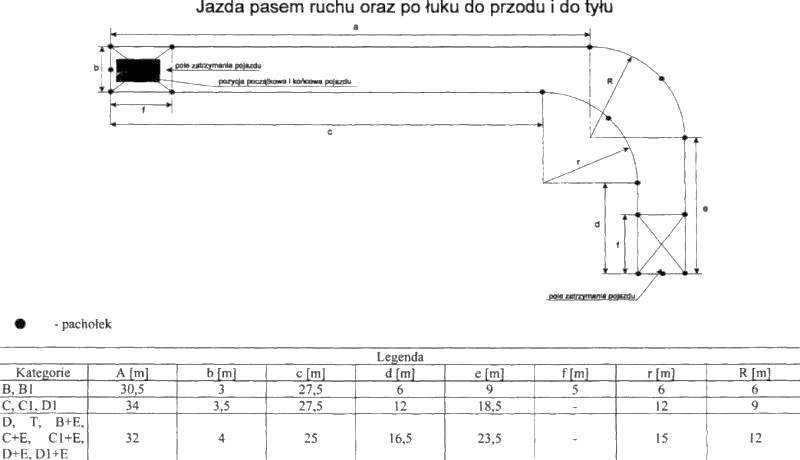
Budynek jest stworzony z modularnych płaskich elementów natomiast mur jest obiektem 3D oba te elementy mają nałożone na siebie tekstury oraz normal mapy.

## Pojazd

Jako pojazd wykorzystany został skyCar z paczki new sample assets beta będącej częścią unity 5.0, w paczce tej znajdują się wszystkie komponenty potrzebne do stworzenia pojazdów w unity wraz z przykładowym modelem.

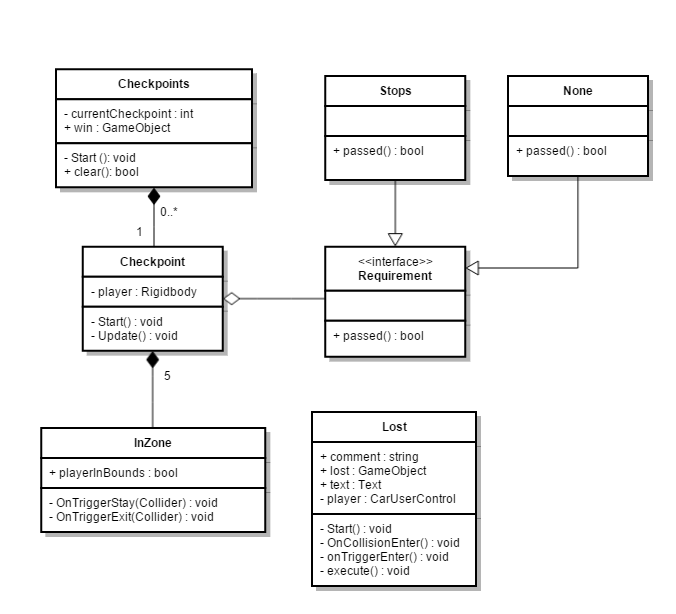
## Scena pierwsza: łuk

Pierwszym manewrem rozpatrywanym w programie będzie jazda po łuku. W tym celu należy stworzyć tor o wymiarach podanych w rys. 2 ukazującym długości poszczególnych linii oraz rozmieszczenie słupków.



rys. 2 rozmieszczenie linii i pachołków na łuku (źródło: prawko.com)

W celu weryfikacji poprawnego wykonania manewru konieczne jest wykrywanie kolizji ze słupkami oraz z liniami, należy też wykryć czy pojazd znalazł się i zatrzymał się w poprawnej pozycji.



rys. 3 Diagram klas.

Rys. 3 przedstawia diagram klas. Klasa Checkpoints przechowuje referencje na wszystkie należące do niej skrypty klasy Checkpoint dba o to żeby aktywowane były one po kolei oraz o to aby wyświetlić menu wygranej po oczyszczeniu ostatniego checkpointa. Klasa Checkpoint sprawdza czy pojazd znalazł się we właściwej pozycji otrzymując informacje od pięciu obiektów klasy InZone oraz czy jest spełniony warunek podany przez interfejs Requirement jeśli obie te rzeczy są spełnione Checkpoint inforuje klasę Checkpoints ze należy przejść do kolejnego punktu, po skończeniu wszystkich punktów należy wyświetlić wiadomość o wygranej. Klasa InZone ustawia playerInBounds jeśli wykryje, że pojazd znalazł się w danej strefie, natomiast przy jej opuszczeniu playerInBounds przyjmuje wartość false. Interfejs Requirement realizuje wzorzec projektowy strategii, gdzie wymiennymi strategiami są klasy Stops oraz None. None pozwala na skończenie checkpointu poprzez najechanie w odpowiednie miejsce. Stops pozwala zakończyć checkpoint dopiero po zatrzymaniu się w wyznaczonej strefie. Ostatnią klasą jest klasa Lost dokładana jest ona do każdego obiektu i każdej strefy której naruszenie powoduje przegranie rozgrywki, po przegranej gracz straci kontrole nad samochodem oraz wyświetlane jest menu z powodem przegranej.

# Implementacja

# Instrukcja użytkownika

[1]

# Podsumowanie i wnioski

# Bibliografia

|  |  |
| --- | --- |
| [1] | O. s. k. prawko. [Online]. Available: http://www.prawko.com/. [Data uzyskania dostępu: 6 Styczeń 2014]. |
| [2] | D. Rahmel, Joomla! Profesjonalne tworzenie stron WWW, Helion, 2010. |