TELERIK

OOP TEAM PROJECT

“CAPTAIN HOOK” TEAM

Участници:

Владислав Карамфилов

Александър Тодоров

Димитър Димитров

Ясен Андреев

Тони Терзиев

# Console Mario game

Играта Console Mario е завръщане в не толкова близкото минало, когато нашият първообраз, Super Mario, беше истински хит. Не по-малък хит очакваме и да стане наследникът му Console Mario. Целта на играта е много проста - героят „Марио“ трябва да мине и събере максимален брой пари докато се придвижва от едно ниво към друго, като накрая събраните пари се превръщат в точки. Но това няма да е толкова лесно колкото звучи, пътят до финала е осеян с безброй бездни, необятни планини и неочаквани предизвикателства. А какви са те, можете да разберете само ако се впустнете в атмосферата на Console Mario. Нашата малка игричка се състои от 3 нива. Целта на Марио в първото ниво започва със спиращата дъха бездна, ако все още е жив, неотърсил се от адреналина той ще трябва да се изправи пред следващото си предизвикателство – скок върху гигантската тръба, няколко пъти по – висока от самия него. След това той отново трябва да събере всичките си сили и да направи още един скок, този път обаче той ще бъде богато възнаграден, защото на горното ниво го очакват купища пари. И ето, че ваче забогатял, Марио ще се върне върху тръбата и застанал на точните координати ще може да мине в следващото ниво, като преди това не забравя да вземе парите и от дясната част на тръбата. В бонус нивото, несметни богатства очакват Марио, ако се изкачи на платформата и прибере парите, той ще стане толкова мега богат, че ще може да си купи замъка на края на трето ниво и да заживее щастливо за вечни времена(плюс да си намери няколко принцеси). Но между него и замъка ще се изправи последната и най – страшна заплаха – гигантска планина. Марио ще трябва да покори планината с последните си сили, преди да завърши своето пътуване и да разбере колко точки е събрал.

# How it actually works?

В основта на програмта имаме абстрактният базов клас GameObjects, който имплементира интерфейса IConsolePrintable, подсигуряващ възможността на всеки елемент да може да се принтира на конзолата. GameObjects се наследява съответно от други два абстрактни класа – MovingObject (имплементиращ IMovable) и StaticObject. MovingObject се наследява от класовете Mario и Flag, където се имплементира тяхната визуализация и функция. Класове: Cloud, Pole , Castle , Pipe, Brick, Coin от своя страна се използват за фиксираните(статични) обекти, нужни за визуализацията на играта. Абстрактният клас Level се наследява от класовете CastleLevel, BonusLevel, OutsideLevel, като по този начин се прави разграничение между различните нива. Използвани са също така енумерацийте CloudImages, Colors и MarioImages, даващи ни възможността да избираме от множество опции за визуализация на различните елементи. Интерфейсите IConsolePrintable, IEngine, IMovable, IUserInterface (имплементиран чрез евенти) ни предоставят възможностите и действията, които героят ни Марио и останалите обекти могат да извършва.

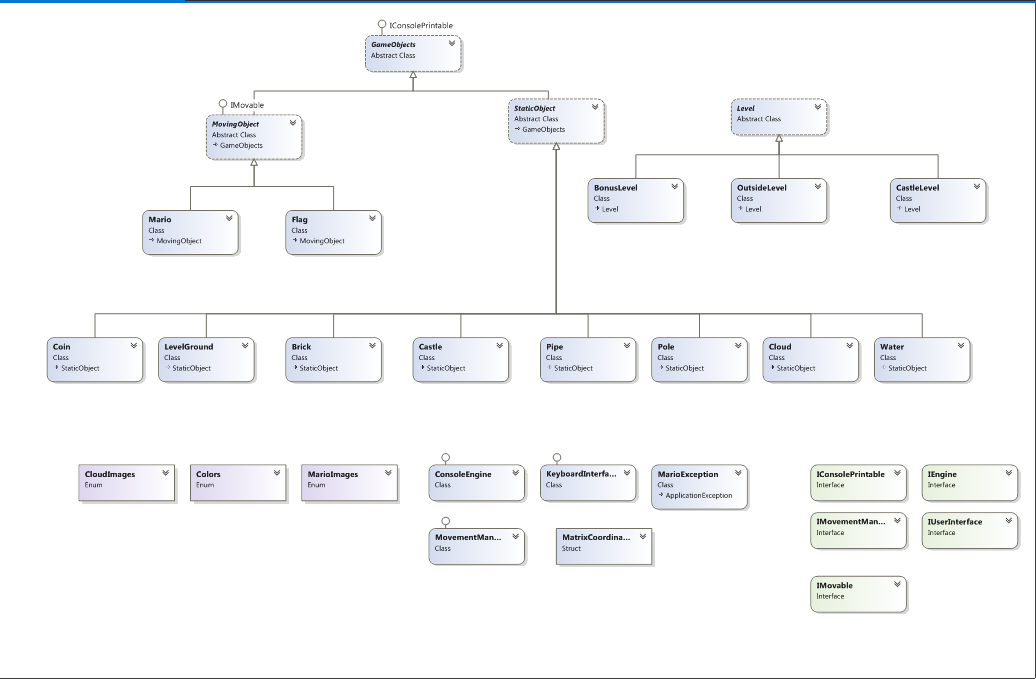


Fig.1(ClassDiagram)

# Detailed Description of Classes

## Class GameObjects

Базов, абстрактнен клас, който наследява интерфейса IConsolePrintable. Декларираме полета за MatrixCoordinates, Colors и char[,] image, които се енкапсулират в свойствата TopLeft, Image и Color, които ще ни служат за визуализация на обектите в играта. Методът PrintOnConsole се използва за рисуване в конзолата.

## Class MovingObjects

Абстрактен клас, директен наследник на GameObject и имплементиращ IMovable интерфеис. Има 1 поле- speed, което чрез енкапсулация се свързва със свойството Speed. Има два метода –Move, който не се имплементира тук и RemoveFromConsole, който се използва за чистене на обектите от конзолата.

### 2.1 Class Mario

Притежава 2 полета imageOne and imageTwo, които представляват изображението на Марио върху конзолата. Свойството Profile избира кое от двете полета да се използва при изобразяването. Методът Move се имплементира и осигурява придвижването на обекта върху игровото поле.

## Class StaticObjects

Абстрактен клас, който е директен наследник на GameObject. Използва се за логическо обособяване на всички статични(фиксирани) класове.

### 3.1 Classes Coin, LevelGround, Brick, Castle, Pipe, Pole, Cloud, Water

Наследници на StaticObject, които наследяват всички полета, свойства и методи на родителите. Всеки един от тези класове има различна визуализация имплементирана чрез поле image и свойсвто Image. Позиция върху игралното поле е запазено в полето topLeft и свойсвтото TopLeft.

## Level

Абстрактен клас използван за логическо разделение на нивата от тази игра. Притежава свойствата LevelGrid – двумерен масив от символи(chars), който представлява игралното поле, списък StaticObjects – който съдържа всички статични(фиксирани) обекти от нивото и списък MovingObjects- който всички подвижни обекти. Методи: PrintMovingObjects и PrintStaticObjects, който осигуряват изобразяването на обектите в конзолата и Load, който е отговорен за зареждане на обектите за всяко едно ниво.

### 4.1 Classes OutSideLevel, BonusLeve, CastleLevel

Наследници на базовия клас Level.Всеки един от класовете има статично поле оказващо името на нивото. Тук се зареждат всички текстури за нивото и се посочват стартови позиции за подвижните обекти.

### Level Design



Fig.2(OutSideLevel)



Fig.3(BonusLevel)

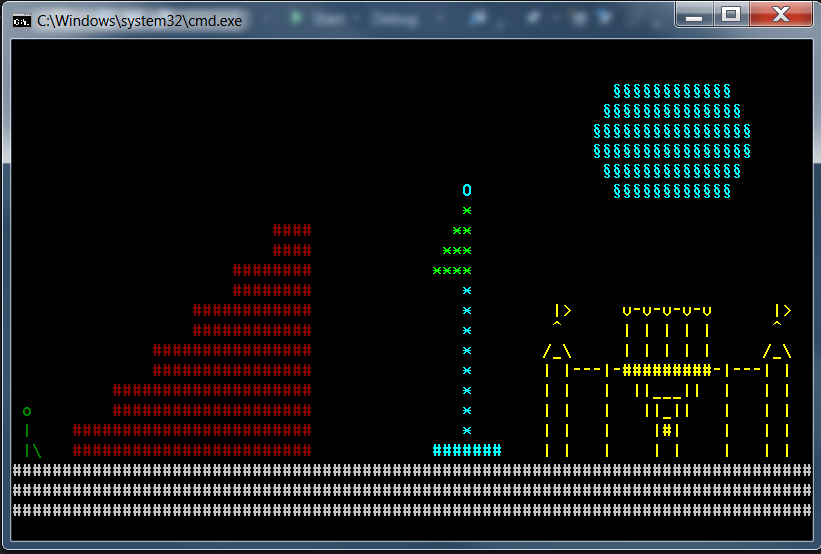


Fig.4(CastleLevel)

## ConsoleEngine

Сърцето на тази игра. Както може да се досетите от името този клас е отговорен за реализирането на цялата идея. Тук имаме свойства следящи за сегашното ниво(currentLevel), играча(Player), резултата(Score), и последен, но не на последно място MovementManager-a, без който движението на нашият храбър водопроводчик, не би било възможно.

## MovementManager

Клас отговорен за движението на главния герой. Имплементира интерфейса – IMovementManger, който реализира всички методи за движение(Up,Down,Left,Right,UpRight,UpLeft). Тук също така се резлизира проверката за collision detection и въвеждането на гравитация.

## MariоException

Custom made exception само и единствено за нашият главен герой. Наследява ApplicationException и когато се извика връща съобщение за край на играта и резултат.

## KeyboardInterface

Класът „ослушващ” се за events. Имплементира интерфейса IUserInterface и следи за натиснати клавиши. Ако някой от гейм контролите е натиснат изпраща заявка за придвижване.

## Public repository

Използвахме codeplex.com за репозитори. Линкът е:

http://caphookoop.codeplex.com/

