**2019/07/19 実験実習**

**目的**

この演習においてはオブジェクト指向プログラミングの理解を深める。

**装置/ツール**

* Visual Studio
* Windows 10 Proト

**実験**

**問題11.1**

FighterクラスのInitializeメソッドを参考にして図11.10中に問題11.1追加箇所とコメントされた箇所に三角形のサイズである定数MAX\_XおよびMAX\_Yと、適切に命令を追加して、敵オブジェクトを逆三角形として表示されるようにしなさい。また、上記ソースコードを実行してスクリーンショットを示しなさい。

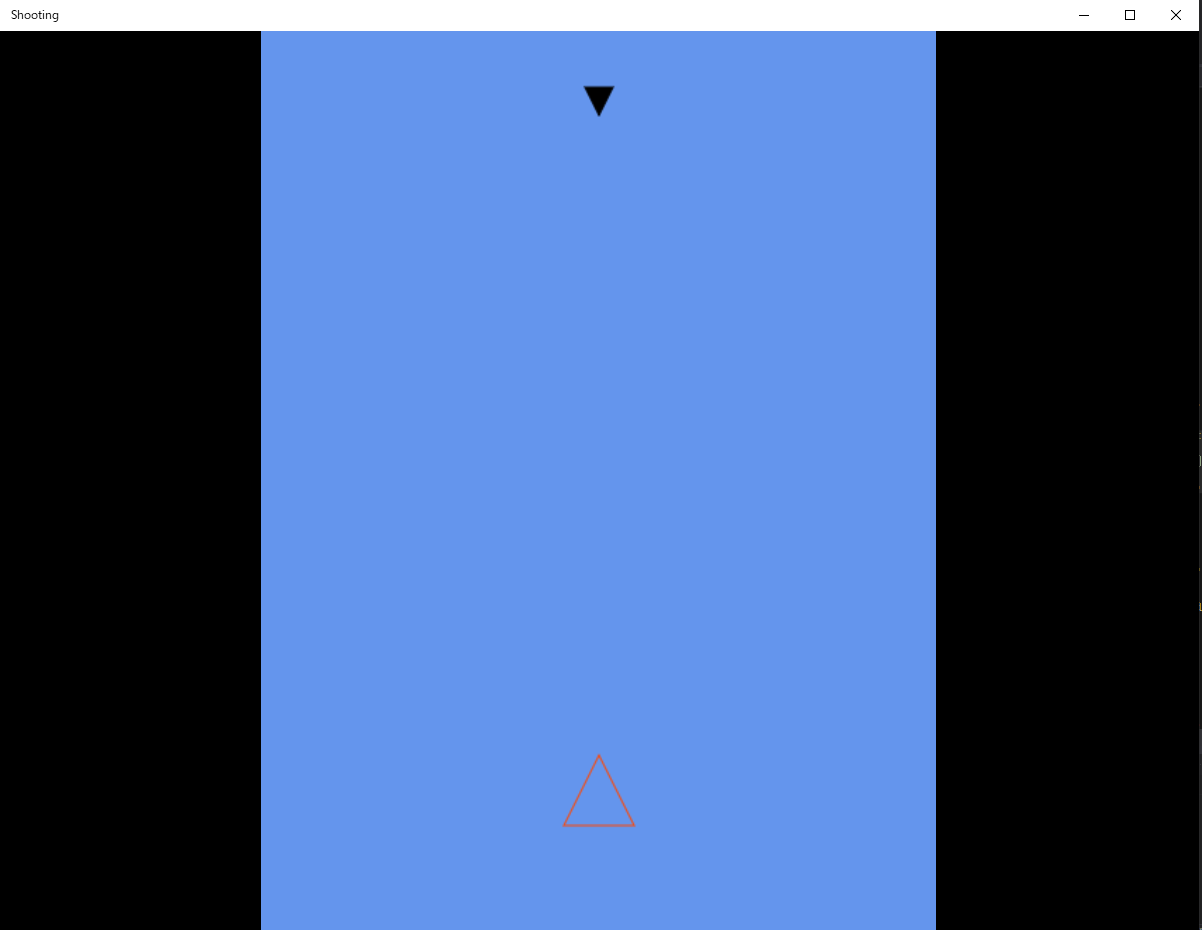
実行結果を図11.1に示す。  


図11.1 敵オブジェクトを表示した

**問題11.2**

すべての変更を追加してプログラムのスクリーンショットを報告しなさい。

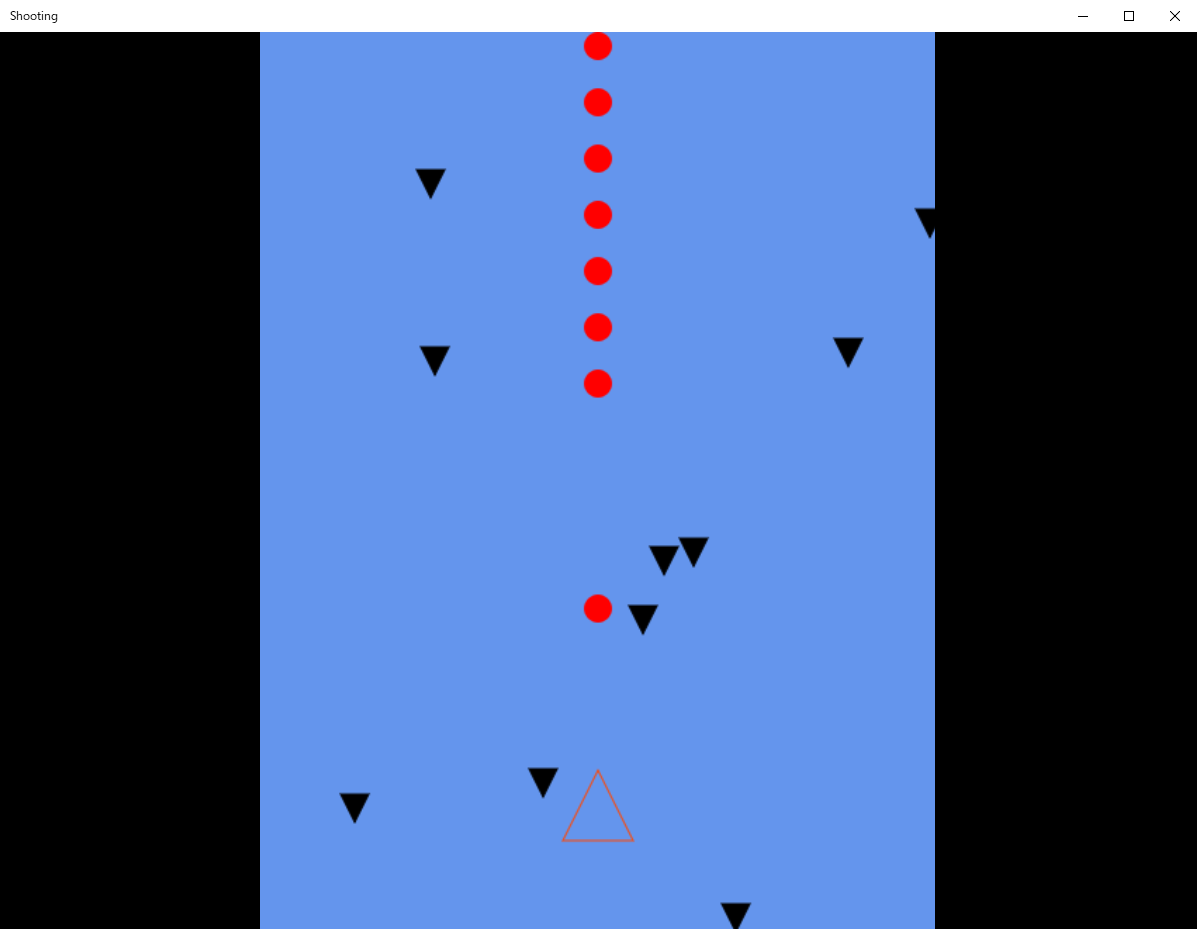
実行結果を11.2に示す。  


図11.2 すべての変更を追加した

**課題**

**レポート課題11.1**

クラス図を報告しなさい

クラス図を図11.3に示す。

地図, テキスト が含まれている画像

自動的に生成された説明

図11.3 クラス図

非常にクラス図が大きく、見にくいので、以下にクラス図のコードを示す。

```graphviz

digraph obj{

node[shape=record]

rankdir="BT"

IHittable[label="{

＜＜interface＞＞\n

IHittable||

IsHitted(c:IRectBounds):bool

}"]

ICrashable[label="{

＜＜interface＞＞\n

ICrashable||

Crash() \n

IsFinished():bool \n

IsCrashing():bool \n

}"]

ITarget[label="{

＜＜interface＞＞\n

ITarget||

}"]

IRectBounds[label="{

＜＜interface＞＞\n

IRectBounds||

GetNorthEastX():int \n

GetNorthEastY():int \n

GetSouthWestX():int \n

GetSouthWestY():int \n

}"]

IMovableRectTarget[label="{

＜＜interface＞＞\n

IMovableRectTarget||

MoveNext()

}"]

ShootingUtils[label="{

ShootingUtils||

+IsIntersected(a:IRectBounds, b:IRectBounds):bool\n

¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯

}"]

SimpleEnemy[label="{

SimpleEnemy|

-d2dDeviceContext:DeviceContext \n

-d2dDevice:Device \n

-y:int \n

-x:int \n

-enemyPath:TransformedGeometry \n

-enemyBrush:SolidColorBrush \n

-firstPoint:Vector2 \n

-secondPoint:Vector2 \n

-thirdPoint:Vector2 \n

-isVisible:bool \n

-MAX\_X:const float = 20f \n

-MAX\_Y:const float = 20f \n

-MOVE\_SPEED:const int = 2 \n|

+SimpleEnemy(ctx:DeviceContext) \n

-Initialize():void \n

+Crash():void \n

+Draw():void \n

+IsHitted(c:IRectBounds):bool \n

+IsMovable():bool \n

+Move(dy:int, dx:int):void \n

+SetPosition(y:int, x:int):void \n

+GetNorthEastX():int \n

+GetNorthEastY():int \n

+GetSouthWestX():int \n

+GetSouthWestY():int \n

+IsFinished():bool \n

+IsCrashing():bool \n

+MoveNext():void

}"]

IUpdatable[label="{

＜＜interface＞＞\n

IUpdatable||

Update():void

}"]

RectTargetManager[label="{

RectTargetManager|

-context:DeviceContext \n

-targetList:List＜IMovableRectTarget＞ \n

-playerShotManager:PlayerShotManager \n

+ENEMY\_MAX\_NUM:const int = 10 \n

+rng:Random \n

-MAX\_WIDTH:const int = 480\n |

+RectTargetManager(ctx:DeviceContext, playerShotManager:PlayerShotManager) \n

-Initialize():void \n

-InitializePosition(e:IMovable):void \n

+Draw():void\n

+Update():void\n

}"]

PlayerShotManager[label="{

PlayerShotManager|

-d2dDeviceContext:DeviceContext \n

-shotList:List＜Shot＞ \n

-drawList:List＜Shot＞ \n

-y:int \n

-x:int \n

-SHOT\_NUM\_MAX:const int = 10 \n

-SHOT\_SPEED:const int = -20 \n|

+PlayerShotManager(ctx:DeviceContext)\n

-Initialize():void\n

+Fire():void\n

+Draw():void\n

+Update():void\n

+Move(dy:int, dx:int):void\n

+SetPosition(y:int, x:int):void\n

+IsMovable():void\n

+IsHitted(c:IRectBounds):bool

}"]

PlayerShot[label="{

PlayerShot|

-shotBrush:Brush \n

-MAX\_X:const float = 10f \n

-MAX\_Y:const float = 10f \n

-INNER\_DIFF:const float = 2f \n

-isVisible:bool \n|

+PlayerShot(ctx:DeviceContext):base(ctx) \n

+Draw():void

+SetPosition(y:int, x:int) \n

+GetNorthEastX():int \n

+GetNorthEastY():int \n

+GetSouthWestX():int \n

+GetSouthWestY():int \n

+IsHitted(c:IRectBounds):bool \n

+Crash():void \n

+IsFinished():bool \n

+IsCrashing():bool \n

}"]

Shot[label="{

Shot|

+Crash():void \n

+IsCrashing():bool \n

+IsFinished():bool \n

+IsHitted(IRectBounds c):bool \n

-d2dDeviceContext:DeviceContext \n

-center:Vector2 \n|

+Shot(ctx:DeviceContext)

+Draw():void \n

+IsMovable():bool \n

+Move(dy:int, dx:int) \n

+SetPosition(y:int, x:int)\n

}"]

App [label="{

App||

-Main()\n

¯¯¯¯¯¯¯¯¯

}"]

FrameworkViewSource[label="{

FrameworkViewSource||

+CreateView():IFrameworkView

}"]

FrameworkView[label="{

FrameworkView|

-d2dDeviceContext:SharpDX.Direct2D1.DeviceContext \n

-d2dTarget:Bitmap1 \n

-swapChain:SwapChain1 \n

-mWindow:CoreWindow \n

-tFighterPath:TransformedGeometry \n

-fighterBrush:SolidColorBrush \n

-fighterDisplay:Fighter \n

-displayList:List＜IDrawable＞ \n

-playerShotManager:PlayerShotManager \n

-enemyDisplay:SimpleEnemy \n

-updateList:List＜IUpdatable＞ \n

-targetManager:RectTargetManager \n|

+Initialize(applicationView:CoreApplicationView)\n

+OnActivated(applicationView:CoreApplicationView, args:IActivatedEventArgs)\n

CreateDeviceResources()\n

+SetWindow(window:CoreWindow)\n

+Load(entryPoint:string)\n

+Run()\n

+Uninitialize()

}"]

// クラス継承

edge [arrowhead = "empty"]

PlayerShot -> Shot

// インターフェース

edge [arrowhead = "empty" style="dashed"]

ITarget->IHittable, ICrashable, IDrawable

IMovableRectTarget->ITarget, IRectBounds, IMovable

SimpleEnemy->IMovableRectTarget

RectTargetManager->IUpdatable, IDrawable

PlayerShot -> IRectBounds

Shot -> ITarget, IMovable

FrameworkViewSource -> IFrameworkViewSource

FrameworkView -> IFrameworkView

// 依存

edge [arrowhead = "vee" style="dashed"]

SimpleEnemy->PathGeometry,Vector2,"SharpDX.Mathematics.Interop.RawVector2[]",TransformedGeometry,SolidColorBrush,ShootingUtils

RectTargetManager->List＜IMovableRectTarget＞,Random,SimpleEnemy

PlayerShotManager->IDrawable, IMovable, IFirable, IUpdatable, IHittable

PlayerShotManager->List＜Shot＞

PlayerShot -> SolidColorBrush,ShootingUtils

Shot -> Vector2

App -> FrameworkViewSource

FrameworkViewSource -> FrameworkView

FrameworkView -> "SharpDX.Direct3D11.Device",SwapChainDescription1,SampleDescription,SwapChain1,"SharpDX.Direct2D1.Device","SharpDX.Direct2D1.DeviceContext","Bitmap1",List＜IUpdatable＞,PlayerShotManager,Fighter,List＜IDrawable＞,RectTargetManager,PlayerInputManager

}

```

図11.4 graphvizのコード

**レポート課題11.2**

新しく機能を追加するなら何を追加するか

2種類目の弾丸を実装する。例えば、緑で大きくて遅い弾丸があれば、プレイの幅が広がって楽しくなると思う。