Class GameEngine

Namespace: Spicy_Invaders

Assembly: Spicy_Invaders.dll

GameLogic class the handles all game behaviours/calculations.

```
public class GameEngine
```

Inheritance

object ← GameEngine

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

GameEngine()

```
public GameEngine()
```

GameEngine(List<ConsoleKey>)

```
public GameEngine(List<ConsoleKey> consoleKeys)
```

Parameters

`consoleKeys` List<ConsoleKey>

Properties

ControlKeys

Keys for moving player

```
public List<ConsoleKey> ControlKeys { get; set; }
```

Property Value

List<ConsoleKey>

Enemies

enemies

```
public List<Enemy> Enemies { get; }
```

Property Value

List<Enemy>

PlayerShip

The player controlled ship

```
public PlayerShip PlayerShip { get; }
```

Property Value

PlayerShip

Projectiles

all projectiles

```
public List<Projectile> Projectiles { get; }
```

Property Value

List<Projectile>

Methods

CheckPlayerBounderies(Direction)

Method responsible for making sure the player is within gameboard bounderies

```
public bool CheckPlayerBounderies(Direction direction)
```

Parameters

`direction` Direction

The direction the player is trying to move in

Returns

bool

Returns true player has reached a limit, otherwise false

CheckProjectileBounderies()

Method responsible for checking projectile positions and comparing them with the gameboard limits, removing projectiles if they reached a certain boundery

```
public void CheckProjectileBounderies()
```

MoveEnemy()

Method responsible for moving enemy objects from Enemies list, and removing them from said list, based on enemy move direction and gameboard limits.

```
public void MoveEnemy()
```

MoveProjectile()

Method responsible for moving projectile objects from Projectiles list, and removing them from said list. Based on projectile move direction and gameboard limits

```
public void MoveProjectile()
```

PlayerControls()

Method responsible translating key inputs into player actions.

```
public void PlayerControls()
```

ProjectileCollisionDetection()

Method responsible for checking projectile positions and comparing them with the enemy/player positions to see if there is a collision.

```
public void ProjectileCollisionDetection()
```

RemoveDeadEnemey()

Removes dead enemies from the enemy list

```
public int RemoveDeadEnemey()
```

Returns

int

ResetHitAnimations()

resets hit animations for player and/or enemies

```
public void ResetHitAnimations()
```

SpawnEnemy(bool, int)

Method responsible for creating new enemy objects and adding them to the Enemies list.

```
public void SpawnEnemy(bool isMelon, int wave = 0)
```

Parameters

`isMelon` bool

bool if is melon enemy is to be spawned

`wave` int

UpdateExplosionLevel()

updates explosion levels based on the current explosion level.

```
public void UpdateExplosionLevel()
```