Neon Particle Effects

1.0

Generated by Doxygen 1.8.10

Wed Sep 9 2015 21:53:40

Contents

Chapter 1

Namespace Index

1.1 Packages

Here are the packages with brief descriptions (if available):

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 4

Namespace Documentation

4.1 PE2D Namespace Reference

Classes

- class CircularArray

  Simplified version of the circular buffer found at: http://geekswithblogs.net/blackrob/archive/2014/09/01/circu aspx*. Generic storage, used to store particles.*
- class CustomParticle

  Main workhorse for the custom particles. Updates particles state (colour, position, velocity etc), handles interaction with effectors, and applys any screen constraints.
- class CustomParticleEmitter

  Base class for ParticleEmitterInRandomDirection and ParticleEmitterInObjectDirection. Add base classes to Game↩ Objects to easily create particle emitters.
- struct ParticleBuilder

  Holds the particle state. Passed to the ParticleFactory to build particles.
- class ParticleEffector

  Add to a gameobject to effect a particles movement.
- class ParticleEmitterInObjectDirection

  Emits particles based on objects rotation.
- class ParticleEmitterInRandomDirection

  Emits particles from objects position in a random direction.
- class ParticleFactory

  Creates and maintain an object pool of particles.
- class ParticleRenderer

  Simple renderer script for particles that disables the sprite renderer on enable and re-enables the srpite renderer after a time specified by ParticleRenderer::RENDERER_DELAY. Attach to the particle prefab to prevent occasional graphic glitches.
- class Pulsate

  Simple script used to pulse an objects size. Used in the demo scene for the effectors.
- class **StaticExtensions**

  Extensions for static classes. COntains a number of helper methods used throughout project.

Enumerations

- enum WrapAroundType { **None**, **WrapAround**, **Constrain** }

  Screen constraint type.
- enum EffectorType { **Attraction**, **Repel**, **BlackHole** }

  Effector types. Attraction pulls particles towards object, repel pushes particles away from object, and blackhole attracts objects until a certain point and then the particle encircles the object.

4.1.1 Enumeration Type Documentation

4.1.1.1 enum PE2D.EffectorType [strong]

Effector types. Attraction pulls particles towards object, repel pushes particles away from object, and blackhole attracts objects until a certain point and then the particle encircles the object.

4.1.1.2 enum PE2D.WrapAroundType [strong]

Screen constraint type.

Chapter 5

Class Documentation

5.1 PE2D.CircularArray< T > Class Template Reference

Simplified version of the circular buffer found at: http://geekswithblogs.net/blackrob/archive/2014/09/01/cir
aspx. Generic storage, used to store particles.

Public Member Functions

- CircularArray (int capacity)

  Initializes a new instance of the PE2D.CircularArray‘1 class.

Properties

- int Start [get, set]

  Pointer to first entry in array. Note this will not usually be 0.
- int Count [get, set]

  Current object count.
- int Capacity [get]

  Total object count.
- bool reachedCapacity [get]

  Gets a value indicating whether this PE2D.CircularArray‘1 has reached capacity.
- T this[int i] [get, set]

  Gets or sets the PE2D.CircularArray‘1 with the specified i.

5.1.1 Detailed Description

Simplified version of the circular buffer found at: http://geekswithblogs.net/blackrob/archive/2014/09/01/cir
aspx. Generic storage, used to store particles.

5.1.2 Constructor & Destructor Documentation

5.1.2.1 PE2D.CircularArray< T >.CircularArray (int *capacity*)

Initializes a new instance of the PE2D.CircularArray‘1 class.

Parameters

capacity	Capacity.

5.1.3 Property Documentation

5.1.3.1 int PE2D.CircularArray< T >.Capacity [get]

Total object count.

The capacity.

5.1.3.2 int PE2D.CircularArray< T >.Count [get], [set]

Current object count.

The count.

5.1.3.3 bool PE2D.CircularArray< T >.reachedCapacity [get]

Gets a value indicating whether this PE2D.CircularArray‘1 has reached capacity.

`true` if reached capacity; otherwise, `false`.

5.1.3.4 int PE2D.CircularArray< T >.Start [get], [set]

Pointer to first entry in array. Note this will not usually be 0.

The start.

5.1.3.5 T PE2D.CircularArray< T >.this[int i] [get], [set]

Gets or sets the PE2D.CircularArray‘1 with the specified i.

Parameters

i	The index.

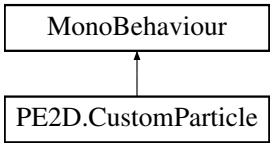
The documentation for this class was generated from the following file:

- /Users/robert/Dropbox/Work/Unity/Particle Effects 2D/Assets/PE2D/Scripts/Helper/CircularArray.cs

5.2 PE2D.CustomParticle Class Reference

Main workhorse for the custom particles. Updates particles state (colour, position, velocity etc), handles interaction with effectors, and applys any screen constraints.

Inheritance diagram for PE2D.CustomParticle:



Static Public Member Functions

- static void UpdateEffectorList ()

  Finds all effectors in scene. Static reference should only be called once for all particles on effector change.

Public Attributes

- bool shouldUpdateAlpha = true

  Update sprites alpha based on velovity.
- bool shouldUpdateScale = true

  Update sprites scale based on velicoty.

Properties

- ParticleBuilder state `[set]`

  Set the state of the particles. Also resets particles properties.
- float duration `[get, set]`

  Maximum duration of particles life. Life may be shorter dependent on velocity.
- float percentLife `[get, set]`

  Range (0, 1). 0 = time to remove from scene, 1 = just spawned.
- SpriteRenderer spriteRenderer `[get]`

  Gets the sprite renderer.

5.2.1 Detailed Description

Main workhorse for the custom particles. Updates particles state (colour, position, velocity etc), handles interaction with effectors, and applys any screen constraints.

5.2.2 Member Function Documentation

5.2.2.1 static void PE2D.CustomParticle.UpdateEffectorList ( ) `[static]`

Finds all effectors in scene. Static reference should only be called once for all particles on effector change.

5.2.3 Member Data Documentation

5.2.3.1 bool PE2D.CustomParticle.shouldUpdateAlpha = true

Update sprites alpha based on velovity.

5.2.3.2 bool PE2D.CustomParticle.shouldUpdateScale = true

Update sprites scale based on velicoty.

5.2.4 Property Documentation

5.2.4.1 float PE2D.CustomParticle.duration `[get]`, `[set]`

Maximum duration of particles life. Life may be shorter dependent on velocity.

The duration.

5.2.4.2 float PE2D.CustomParticle.percentLife [get],[set]

Range (0, 1). 0 = time to remove from scene, 1 = just spawned.

The percent life.

5.2.4.3 SpriteRenderer PE2D.CustomParticle.spriteRenderer [get]

Gets the sprite renderer.

The sprite renderer.

5.2.4.4 ParticleBuilder PE2D.CustomParticle.state [set]

Set the state of the particles. Also resets particles properties.

The state.

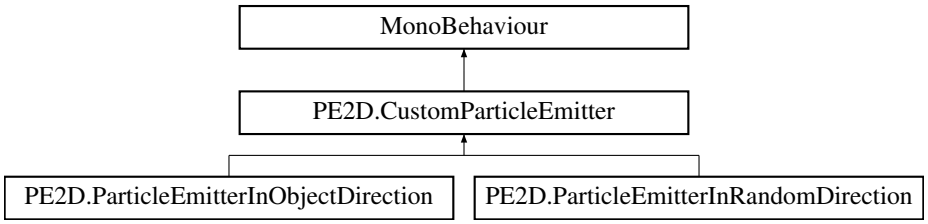
The documentation for this class was generated from the following file:

- /Users/robert/Dropbox/Work/Unity/Particle Effects 2D/Assets/PE2D/Scripts/Particles/CustomParticle.cs

5.3 PE2D.CustomParticleEmitter Class Reference

Base class for ParticleEmitterInRandomDirection and ParticleEmitterInObjectDirection. Add base classes to GameObjects to easily create particle emitters.

Inheritance diagram for PE2D.CustomParticleEmitter:



Public Member Functions

- void TurnOn ()

  Enables particle emission from this object.
- void TurnOff ()

  Disables particle emission from this object.

Public Attributes

- float timeBetweenProjectileRelease = 0f

  The time between projectile release, if equals 0 then particle is released with each call to update.
- Vector2 initialScale = new Vector2 (2f, 1f)

  Initial scale of the particles released. Scale is also dependent on velocity.
- bool particlesEnabled = true

  Turns on/off particle generation from this GameObject.
- float duration = 90f

The maximum duration for each particle. A particles life is also dependent on velocity.

- float velocityDampener = 0.94f

  THe rate at which to reduce particles velocity each time step.

- float lengthMultiplier = 40f

  The length multiplier for the particles.

- WrapAroundType wrapAround = WrapAroundType.None

  The screen constraint type.

- bool randomColour = false

  Particle will spawn as a random colour when enabled.

- Color particleColour

  Set the particles colour.

- bool clampMinLength

  Clamp the minimum length of a particle.

- float minLength

  The minimum length of a particle, only used if clampMinLength = true.

- bool clampMaxLength

  Clamp the maximum length of a particle.

- float maxLength

  The minimum length of a particle, only used if clampMaxLength = true.

- bool removeWhenVelocityReachesThreshold

  Will remove a particle if velocity reaches a threshold.

- float customVelocityThreshold

  The velocity at which a particle will be removed, only used if removeWhenVelocityReachesThreshold = true.

- bool removeWhenAlphaReachesThreshold

  Will remove the particle when its alpha reaches a specified threshold.

- float customAlphaThreshold

  The particles sprites alpha threshold at which a particle will be removed, only used if removeWhenAlphaReaches↩ Threshold = true.

Protected Member Functions

- Color **GetRandomColour** ()
- abstract void **ReleaseParticle** ()

Protected Attributes

- ParticleBuilder **_cachedState**

5.3.1 Detailed Description

Base class for ParticleEmitterInRandomDirection and ParticleEmitterInObjectDirection. Add base classes to GameObjects to easily create particle emitters.

5.3.2 Member Function Documentation

5.3.2.1 void PE2D.CustomParticleEmitter.TurnOff ( )

Disables particle emission from this object.

5.3.2.2 void PE2D.CustomParticleEmitter.TurnOn ( )

Enables particle emission from this object.

5.3.3 Member Data Documentation

5.3.3.1 bool PE2D.CustomParticleEmitter.clampMaxLength

Clamp the maximum length of a particle.

5.3.3.2 bool PE2D.CustomParticleEmitter.clampMinLength

Clamp the minimum length of a particle.

5.3.3.3 float PE2D.CustomParticleEmitter.customAlphaThreshold

The particles sprites alpha threshold at which a particle will be removed, only used if removeWhenAlphaReaches↩Threshold = true.

5.3.3.4 float PE2D.CustomParticleEmitter.customVelocityThreshold

The velocity at which a particle will be removed, only used if removeWhenVelocityReachesThreshold = true.

5.3.3.5 float PE2D.CustomParticleEmitter.duration = 90f

The maximum duration for each particle. A particles life is also dependent on velocity.

5.3.3.6 Vector2 PE2D.CustomParticleEmitter.initialScale = new Vector2 (2f, 1f)

Initial scale of the particles released. Scale is also dependent on velocity.

5.3.3.7 float PE2D.CustomParticleEmitter.lengthMultiplier = 40f

The length multiplier for the particles.

5.3.3.8 float PE2D.CustomParticleEmitter.maxLength

The minimum length of a particle, only used if clampMaxLength = true.

5.3.3.9 float PE2D.CustomParticleEmitter.minLength

The minimum length of a particle, only used if clampMinLength = true.

5.3.3.10 Color PE2D.CustomParticleEmitter.particleColour

Set the particles colour.

5.3.3.11 bool PE2D.CustomParticleEmitter.particlesEnabled = true

Turns on/off particle generation from this GameObject.

5.3.3.12 bool PE2D.CustomParticleEmitter.randomColour = false

Particle will spawn as a random colour when enabled.

5.3.3.13 bool PE2D.CustomParticleEmitter.removeWhenAlphaReachesThreshold

Will remove the particle when its alpha reaches a specified threshold.

5.3.3.14 bool PE2D.CustomParticleEmitter.removeWhenVelocityReachesThreshold

Will remove a particle if velocity reaches a threshold.

5.3.3.15 float PE2D.CustomParticleEmitter.timeBetweenProjectileRelease = 0f

The time between projectile release, if equals 0 then particle is released with each call to update.

5.3.3.16 float PE2D.CustomParticleEmitter.velocityDampener = 0.94f

THe rate at which to reduce particles velocity each time step.

5.3.3.17 WrapAroundType PE2D.CustomParticleEmitter.wrapAround = WrapAroundType.None

The screen constraint type.

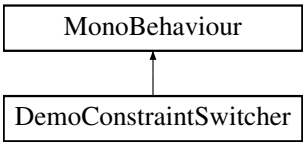
The documentation for this class was generated from the following file:

- /Users/robert/Dropbox/Work/Unity/Particle Effects 2D/Assets/PE2D/Scripts/Particles/Emitters/Custom↩ ParticleEmitter.cs

5.4 DemoConstraintSwitcher Class Reference

Switches between screen constraints in the demo scene.

Inheritance diagram for DemoConstraintSwitcher:



Public Attributes

- DemoMouseController **mouseController**
- Text **constraintText**

5.4.1 Detailed Description

Switches between screen constraints in the demo scene.

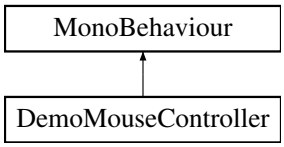
The documentation for this class was generated from the following file:

- /Users/robert/Dropbox/Work/Unity/Particle Effects 2D/Assets/PE2D/Scripts/Demo/DemoConstraint↩ Switcher.cs

5.5 DemoMouseController Class Reference

Spawns a circular explosion of particles on mouse click. Example of how to procedurally create particles.

Inheritance diagram for DemoMouseController:



Public Attributes

- float **speedOffset** = .01f
- float **lengthMultiplier** = 40f
- int **numToSpawn** = 200
- WrapAroundType **wrapAround**

5.5.1 Detailed Description

Spawns a circular explosion of particles on mouse click. Example of how to procedurally create particles.

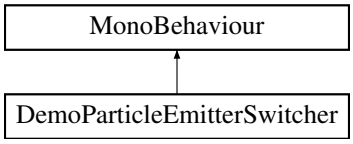
The documentation for this class was generated from the following file:

- /Users/robert/Dropbox/Work/Unity/Particle Effects 2D/Assets/PE2D/Scripts/Demo/DemoMouseController.cs

5.6 DemoParticleEmitterSwitcher Class Reference

Switches between particle emitters in demo scene.

Inheritance diagram for DemoParticleEmitterSwitcher:



Public Attributes

- GameObject[ ] **particleEmitters**
- Text **emitterText**
- string **preEmitterString**
- string **postEmitterString**
- bool **updateEffectorsOnChange** = false

5.6.1 Detailed Description

Switches between particle emitters in demo scene.

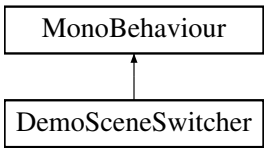
The documentation for this class was generated from the following file:

- /Users/robert/Dropbox/Work/Unity/Particle Effects 2D/Assets/PE2D/Scripts/Demo/DemoParticleEmitter↩ Switcher.cs

5.7 DemoSceneSwitcher Class Reference

Switches between demo scenes when enter key pressed.

Inheritance diagram for DemoSceneSwitcher:



Public Attributes

- int **numberOfScenes** = 3

5.7.1 Detailed Description

Switches between demo scenes when enter key pressed.

The documentation for this class was generated from the following file:

- /Users/robert/Dropbox/Work/Unity/Particle Effects 2D/Assets/PE2D/Scripts/Demo/DemoSceneSwitcher.cs

5.8 PE2D.ParticleBuilder Struct Reference

Holds the particle state. Passed to the ParticleFactory to build particles.

Public Attributes

- Vector2 velocity

  Initial velocity of particle.
- WrapAroundType wrapAroundType

  Screen constraint type.
- float lengthMultiplier

  The particles scale is multipled by this.
- float velocityDampModifier

  The percentage amount that a particles velocity remains each timestep.
- bool ignoreEffectors

  If enables, the particle built with this state will ignore effectors.
- float minLengthClamp

  Clamp the minimum length of a particles sprite.

- float maxLengthClamp

  Clamp the maximum length of a particles sprite.
- bool removeWhenVelocityReachesThreshold

  Will remove a particle if velocity reaches a threshold.
- float customVelocityThreshold

  The velocity at which a particle will be removed, only used if removeWhenVelocityReachesThreshold = true.
- bool removeWhenAlphaReachesThreshold

  Will remove the particle when its alpha reaches a specified threshold.
- float customAlphaThreshold

  The particles sprites alpha threshold at which a particle will be removed, only used if removeWhenAlphaReaches↩ Threshold = true.

5.8.1 Detailed Description

Holds the particle state. Passed to the ParticleFactory to build particles.

5.8.2 Member Data Documentation

5.8.2.1 float PE2D.ParticleBuilder.customAlphaThreshold

The particles sprites alpha threshold at which a particle will be removed, only used if removeWhenAlphaReaches↩ Threshold = true.

5.8.2.2 float PE2D.ParticleBuilder.customVelocityThreshold

The velocity at which a particle will be removed, only used if removeWhenVelocityReachesThreshold = true.

5.8.2.3 bool PE2D.ParticleBuilder.ignoreEffectors

If enables, the particle built with this state will ignore effectors.

5.8.2.4 float PE2D.ParticleBuilder.lengthMultiplier

The particles scale is multipled by this.

5.8.2.5 float PE2D.ParticleBuilder.maxLengthClamp

Clamp the maximum length of a particles sprite.

5.8.2.6 float PE2D.ParticleBuilder.minLengthClamp

Clamp the minimum length of a particles sprite.

5.8.2.7 bool PE2D.ParticleBuilder.removeWhenAlphaReachesThreshold

Will remove the particle when its alpha reaches a specified threshold.

5.8.2.8 bool PE2D.ParticleBuilder.removeWhenVelocityReachesThreshold

Will remove a particle if velocity reaches a threshold.

5.8.2.9 Vector2 PE2D.ParticleBuilder.velocity

Initial velocity of particle.

5.8.2.10 float PE2D.ParticleBuilder.velocityDampModifier

The percentage amount that a particles velocity remains each timestep.

5.8.2.11 WrapAroundType PE2D.ParticleBuilder.wrapAroundType

Screen constraint type.

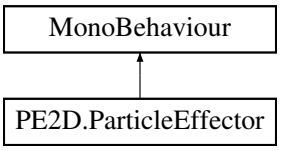
The documentation for this struct was generated from the following file:

- /Users/robert/Dropbox/Work/Unity/Particle Effects 2D/Assets/PE2D/Scripts/Particles/ParticleBuilder.cs

5.9 PE2D.ParticleEffector Class Reference

Add to a gameobject to effect a particles movement.

Inheritance diagram for PE2D.ParticleEffector:



Public Attributes

- EffectorType **effectorType**
- float **distance**
- float **rotateDistance**
- float **force**

5.9.1 Detailed Description

Add to a gameobject to effect a particles movement.

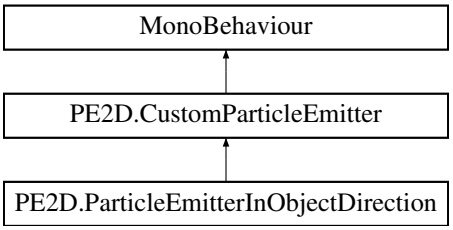
The documentation for this class was generated from the following file:

- /Users/robert/Dropbox/Work/Unity/Particle Effects 2D/Assets/PE2D/Scripts/Particles/ParticleEffector.cs

5.10 PE2D.ParticleEmitterInObjectDirection Class Reference

Emits particles based on objects rotation.

Inheritance diagram for PE2D.ParticleEmitterInObjectDirection:



Protected Member Functions

- override void **ReleaseParticle** ()

Additional Inherited Members

5.10.1 Detailed Description

Emits particles based on objects rotation.

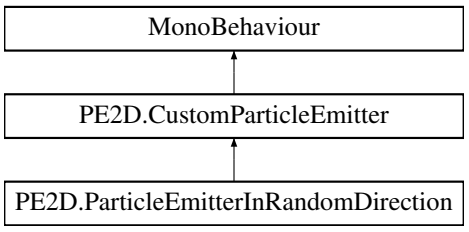
The documentation for this class was generated from the following file:

- /Users/robert/Dropbox/Work/Unity/Particle Effects 2D/Assets/PE2D/Scripts/Particles/Emitters/Particle↩ EmitterInObjectDirection.cs

5.11 PE2D.ParticleEmitterInRandomDirection Class Reference

Emits particles from objects position in a random direction.

Inheritance diagram for PE2D.ParticleEmitterInRandomDirection:



Protected Member Functions

- override void **ReleaseParticle** ()

Additional Inherited Members

5.11.1 Detailed Description

Emits particles from objects position in a random direction.

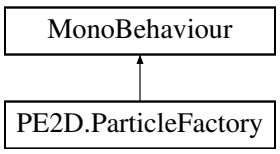
The documentation for this class was generated from the following file:

- /Users/robert/Dropbox/Work/Unity/Particle Effects 2D/Assets/PE2D/Scripts/Particles/Emitters/Particle↩ EmitterInRandomDirection.cs

5.12 PE2D.ParticleFactory Class Reference

Creates and maintain an object pool of particles.

Inheritance diagram for PE2D.ParticleFactory:



Public Member Functions

- void CreateParticle (Vector2 position, Color colour, float duration, Vector2 initialScale, ParticleBuilder state)

  Creates a particle at position with the specified state.
- void RemoveAllActiveParticles ()

  Sets all enabled particles to be removed in the next time step.

Public Attributes

- GameObject particlePrefab

  Particle prefab.
- int maxParticleCount

  The max particle count. This number of particles is created at runtime and placed in a finite pool.

Properties

- static ParticleFactory instance `[get]`

  Gets the instance of this class. Can be called from any script. Only one instance of a particle factory can exist in one scene.

5.12.1 Detailed Description

Creates and maintain an object pool of particles.

5.12.2 Member Function Documentation

5.12.2.1 void PE2D.ParticleFactory.CreateParticle (Vector2 *position*, Color *colour*, float *duration*, Vector2 *initialScale*, ParticleBuilder *state*)

Creates a particle at position with the specified state.

Parameters

position	Initial position of particle.
tint	The initial colour of particle.
duration	The maximum duration of particle.

scale	Initial scale of particle.
state	THe particle state.

5.12.2.2 void PE2D.ParticleFactory.RemoveAllActiveParticles ( )

Sets all enabled particles to be removed in the next time step.

5.12.3 Member Data Documentation

5.12.3.1 int PE2D.ParticleFactory.maxParticleCount

The max particle count. This number of particles is created at runtime and placed in a finite pool.

5.12.3.2 GameObject PE2D.ParticleFactory.particlePrefab

Particle prefab.

5.12.4 Property Documentation

5.12.4.1 ParticleFactory PE2D.ParticleFactory.instance [static],[get]

Gets the instance of this class. Can be called from any script. Only one instance of a particle factory can exist in one scene.

The instance.

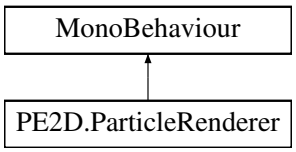
The documentation for this class was generated from the following file:

- /Users/robert/Dropbox/Work/Unity/Particle Effects 2D/Assets/PE2D/Scripts/Particles/ParticleFactory.cs

5.13 PE2D.ParticleRenderer Class Reference

Simple renderer script for particles that disables the sprite renderer on enable and re-enables the srpite renderer after a time specified by ParticleRenderer::RENDERER_DELAY. Attach to the particle prefab to prevent occasional graphic glitches.

Inheritance diagram for PE2D.ParticleRenderer:



5.13.1 Detailed Description

Simple renderer script for particles that disables the sprite renderer on enable and re-enables the srpite renderer after a time specified by ParticleRenderer::RENDERER_DELAY. Attach to the particle prefab to prevent occasional graphic glitches.

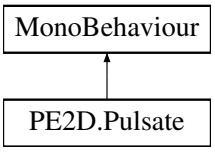
The documentation for this class was generated from the following file:

- /Users/robert/Dropbox/Work/Unity/Particle Effects 2D/Assets/PE2D/Scripts/Particles/ParticleRenderer.cs

5.14 PE2D.Pulsate Class Reference

Simple script used to pulse an objects size. Used in the demo scene for the effectors.

Inheritance diagram for PE2D.Pulsate:



5.14.1 Detailed Description

Simple script used to pulse an objects size. Used in the demo scene for the effectors.

The documentation for this class was generated from the following file:

- /Users/robert/Dropbox/Work/Unity/Particle Effects 2D/Assets/PE2D/Scripts/Pulsate.cs

Index