Smooth Camera Shaker

Support:

Email: firstgeargames@gmail.com
Discord: https://discord.com/invite/4SJbJuk

Video Tutorials:

https://www.youtube.com/playlist?list=PLkx8oFug638qoEsoZAZ9DJv-yazbtSW2R

Documentation Updates (current v2.12 r0):

http://firstgeargames.com/releases/smoothcamerashaker/Documentation.pdf

Changelog:

v2.12:

Changes:

- Custom seed value removed. Unchecking random seed now always results in shakes starting on configured direction.

v2.11:

Fixes:

- Fixed noise options on ShakeData always using default values.

Changes:

- Made editor scripts for all Shakable components.

v2.10:

Changes:

- Renamed several API events. Old names will be removed in a later release.
  * Renamed OnDefaultCameraShakerChanged to OnDefaultShakerChanged.
  * Renamed InstantiatedCameraShakers to InstantiatedShakers.
  * Renamed OnCameraShakerDestroyed to OnShakerDestroyed.
  * Renamed OnCameraShakerInstantiated to OnShakerInstantiated.
- Added ObjectShaker. Now instead of only targeting the camera for shakes you may also shake individual objects.
- Added ShakableTransform/2D.
- Added Localize Shake option to ShakableTransform/Rigidbody/2D. Influences are applied locally based on transform position and rotation.

- ShakeData can now randomly invert influence axes on start, and at runtime using ShakeData API. Allows more random shakes, and can be used to invert shakes, or parts of shakes.
- Added Ignore Self and Include Inactive option to ShakableTransform/Rigidbody/2D. Allows Shakable to ignore self when finding children, and include inactive children.
- Added Randomize Directions to all Shakables. Randomizes direction of influence on start for ShakableCanvas and ShakableTransform, and periodically for ShakeableRigidbody.
- Shakables can now specify if to use CameraShaker or ObjectShaker. When selecting ObjectShaker the first found in same or parent objects is used.

v2.01:

Fixes:

- Fixed RequireInView on ShakeableRigidbodies sometimes not recognizing when an object was in view.
- Fixed camera subscription changes not occurring in ShakableRigidbody components.
- Fixed rare bug where camera view would stick when switching cameras, while using matrix, while copying instances, while not moving new camera.

Changes:

- Camera shaking is now optional, while still being able to affect other shakables.
- ShakableRigidbodies can now include rigidbodies in child objects.

v2.00:

Fixes:

- Fixed occasional jarring at the start of large shakes.
- Fixed camera sometimes not being reset at the end of shakes.
- Fixed camera offsets not resetting on CameraHandler after being disabled, while using LocalSpace as a shake technique.

Changes:

- Removed timing for curves on ShakeData as they can be done by resizing the curves in the editor.
- Removed Affected Elements on ShakeData in favor of checkboxes for each Shakable type.
- Improved overall smoothness at the beginning and ending of shakes.
- Can now optionally specify a shake seed on ShakeData.
- Added CopyShakerInstances API to CameraShakerHandler.
- Can now modify Shakable types at runtime on ShakeData.

v1.10:

- Shakes can now be configured to shake only camera, or canvases and camera.

v1.00:

- First release.

Documentation

Quick Start:

Setup Camera:

Add a ***CameraShaker*** component to your primary camera. Use Matrix as the **Shake Technique**, leave **Make Default OnEnable** checked, leave **Limit Magnitude** unchecked.

Create a ShakeData:

Create a ***ShakeData*** scriptable object by accessing the Create menu (right click in your project tab), FirstGearGames, Smooth Camera Shaker, Shake Data.
Leave this ***ShakeData*** at it's default settings.

Shaking*:*

Create a new script as shown.

```
using FirstGearGames.SmoothCameraShaker;
using UnityEngine;

public class ShakeTester : MonoBehaviour
{
    public ShakeData MyShake;
    private void Start()
    {
        CameraShakerHandler.Shake(MyShake);
    }
}
```

Attach this script to any object in your scene. Drag your newly created ***ShakeData*** scriptable object into the **MyShake** field. After pressing Play in your editor your camera should shake the default time on the ***ShakeData***, then stop. If successful, everything is working! Should you run into problems the video tutorials may help, and please feel free to contact support.

Detailed Guide:

CameraShaker Component:

The ***CameraShaker*** component comes with multiple shake techniques. To utilize both, begin by placing your cameras as children of their own parent objects. The camera should have no positional or rotational offset. When you intend to move your camera, move the parent object instead. Note: this step is only mandatory while using Local Space **Shake Technique**.

Add a ***CameraShaker*** component to your camera, and on any additional cameras you wish to shake. Set the **Shake Technique** to Matrix or Local Space. Matrix will shake the camera's view matrix, while Local Space will modify the camera's local space. When designing for VR you will likely use Local Space. Matrix will unconditionally shake Camera Space canvases, and must be used with virtual cameras.

ObjectShaker Component:

ObjectShaker component is used to shake individual objects rather than the camera, and all of it's global listeners. Like ***CameraShaker*** you may limit the magnitude of shakes using **Limit Magnitude**. When ticking **Remove From Manager On Disable** the ***ObjectShaker*** component will remove itself from ***ObjectShakerHandler*** OnDisable, and add itself back OnEnable. When you have many ***ObjectShaker*** components in your scene this may save a small amount of performance. **Shake On Enable** allows you to specify a ***ShakeData*** to run when the ***ObjectShaker*** component becomes enabled.

Add ***ObjectShaker*** to any gameObjects you would like to shake individually. You must reference the added ***ObjectShaker*** when calling shakes.

ShakeData Scriptable Object:

ShakeData are shake presets which may optionally be altered at runtime. Create your first ***ShakeData*** scriptable object by accessing the Create menu (right click in your project tab), FirstGearGames, Smooth Camera Shaker, Shake Data. Select your new ShakeData.

Shakeables To Affect allow the shake to affect targeted elements. Keep in mind world canvases will always shake, and camera canvases will shake unconditionally if using Matrix as a **Shake Technique**. Some elements require you to attach a Shakable script; please see each options tooltip for more information.

Having **Scaled Time** enable will result in your shakes occurring in scaled time. While true, If you were to slow down your game time, your shake would also slow down.

Next you have **Unlimited Duration**, when checked the shake will only end when the shaker is disabled, or when calling Stop or FadeOut on the ***ShakeInstance***. While unticked a **Total Duration** field is shown; this is how long your shake will last. You may also set **Fade In Duration** as well **Fade Out Duration**. **Total Duration** will be whichever is greater of **Total Duration** or both fade values. The **Fade Out Duration** is also the default fade out time when calling FadeOut API.

Magnitude is a multiplier applied towards your offsets, which will be discussed further down. You may also apply **Magnitude Noise** which randomly generates a multiplicative variance of your **Magnitude**. For example: if your **Magnitude** was a value of 2f a variance of 1f would allow your **Magnitude** to range from 0f to 4f. While a variance of 0.5f would allow your **Magnitude** to vary between 1f and 3f.

The **Magnitude Curve** allows you to manually adjust the percentage of your **Magnitude**. A slope starting at 0f and climbing to 1f at the end would result in your shake reaching peek **Magnitude** at the end of it's life.

Roughness is how quickly a shake alternates between offsets. A higher **Roughness** value would make your shakes more violent, bouncing between positive and negative offsets quicker. **Roughness Curve** and **Roughness Noise** work in the same manner as your magnitude counterparts, but applied to **Roughness.**

The **Positional Influence** is the maximum positional offset this shake may move your camera in either direction. **Rotational Influence** is like **Positional Influence** except applied to camera euler angles. Keep in mind both influence types may be exceeded or reduced with **Magnitude** and **Magnitude Noise**. **Invertible Axes** allow the ShakeData to randomly flip axes directions when the shake is executed. This can be called at runtime as well.

With **Random Seed** checked a new starting position and direction is used with every shake. While unchecked you may specify a seed, guaranteeing the shake will start at the same position, and move the same direction with every shake.

ShakableCanvas Component:

Canvases of any type can be shaken with Smooth Camera Shaker. World canvases will automatically be shaken and do not require a ***ShakableCanvas***. Camera space canvases do not need ***ShakableCanvas*** either, when your ***CameraShaker*** is using Matrix as a **Shake Technique**. ***ShakableCanvas*** should only be used on Overlay canvases, or Camera space canvases where the ***CameraShaker*** is using a **Shake Technique** of Local Space.

With **Use Default CameraShaker** enabled whichever ***CameraShaker*** instance is currently default on the ***CameraShakerHandler*** will determine shake values. If you untick **Use Default CameraShaker** you must serialize your preferred ***CameraShaker*** in the **CameraShaker** field, or at runtime using the SetCameraShaker API.

Encapsulate Children will place all of the immediate children beneath the ***ShakableCanvas*** object in a container, and shake the container rather than the children transforms. This is ideal if you have moving UI elements you wish to shake. When using **Encapsulate Children** you will optionally be allowed to **Monitor Children**; when ticked immediate children added at runtime will also be encapsulated.

At the very bottom exist **Positional Multiplier** and **Rotational Multiplier**. These values are multiplied towards the positional and rotational offsets. Having each at 1f would result in your canvas to shake the same amount as your camera. If you were to use 0.5f the canvas would shake half as much as your camera.

ShakableRigidbody / ShakableRigidbody2D Components:

Used to shake Rigidbodies with your Camera or ***ObjectShaker***. These work automatically without any extra work. **Include Children** will also shake any rigidbodies within the objetcts children, as well any on the parent. Using **Ignore Self** will exclude the parent object while using **Include Children**. **Include Inactive** will include inactive objects in children when using **Include Children**. Optionally you may increase or lower the **Positional Multiplier** and **Rotation Multiplier**, which control how much your rigidbody will shake relative to your camera. If you wish your rigidbodies to only shake when in view tick **Require In View**.

ShakableTransform / ShakableTransform2D Components:

These components behave the same way as ***ShakableRigidbody/2D*** but apply influences to the transform instead.

Usage:

To run shake presets as-is you simply call Shake with your preset reference to a specific ***CameraShaker, ObjectShaker*** or to the ***CameraShakerHandler*** to forward the call to your default ***CameraShaker.*** When Shake is called a ***ShakerInstance*** is returned, which you may use to modify unique values on. ***CameraShaker*** also has additional methods and properties you can access to modify your instances at runtime, such as stopping, pausing, fading out, and more. These methods can also be accessed in the ***CameraShakerHandler***. For more details please view the API section of this file. In addition, every method is fully commented and returns detailed information within your IDE intellisense. In the API you will also find several events which you can subscribe to for receiving status changes on your ***CameraShaker(s)***.

API

ShakeData

ShakeData is a scriptable object camera shake preset. Some values of ShakeData may be modified at runtime but you must create an instance of your preset first.

Methods:

Inverts specified positional axes. Using this in the middle of a shake may create jarring the next frame.
axes: Axes to invert.

```
public void InvertPositionalAxes(InvertibleAxes axes)
```

Inverts specified rotational axes. Using this in the middle of a shake may create jarring the next frame.
axes: Axes to invert.

```
public void InvertRotationalAxes(InvertibleAxes axes)
```

Randomizes inversion for specified positional axes. Using this in the middle of a shake may create jarring the next frame.
axes: Axes to randomly invert.

```
public void RandomlyInvertPositionalAxes(InvertibleAxes axes)
```

Randomizes inversion for specified rotational axes. Using this in the middle of a shake may create jarring the next frame.
axes: Axes to randomly invert.

```
public void RandomlyInvertRotationalAxes(InvertibleAxes axes)
```

Sets a new ShakeCaameras value.
value: New ShakeCameras value.

```
public void SetShakeCameras(bool value)
```

Sets a new ShakeCanvases value.
value: New ShakeCanvases value.

```
public void SetShakeCanvases(bool value)
```

Sets a new ShakeRigidbodies value.
value: New ShakeRigidbodies value.

```
public void SetShakeRigidbodies(bool value)
```

Creates and returns an instance of this data.

```
public ShakeData CreateInstance()
```

Sets a new TotalDuration value. Setting this value to 0 or greater removes unlimited duration; just as setting it to less than 0 sets unlimited duration.
value: New total duration. Using values 0f and lower will make duration unlimited.

```
public void SetTotalDuration(float value)
```

Sets a new fade in duration.
value: New fade in duration.
```
public void SetFadeInDuration(float value)
```

Sets a new fade out duration.
value: New fade out duration.
```
public void SetFadeOutDuration(float value)
```

ShakerInstance

ShakerInstance is generated when performing a shake. Instances contain several methods to change behavior before or during a shake.

Properties:

Data being used for this instance.
```
public ShakeData Data { get; private set; }
```

True if this shaker instance is paused.
```
public bool Paused;
```

Methods:

Fades out this instance. This operation only works if not already fading out.
durationOverride: Overrides instance Data fade out duration with a new value.
```
public void FadeOut(float? durationOverride = null)
```

Multiplies magnitude values in data by a set amount.
multiplier: Value to multiply by. 1f is standard multiplication, which in result would be default values.
moveRate: How quickly per second to move towards new multiplier. Values 0f and lower are instant.
***rateUsesDistance:** True to modify move rate based on distance from multiplier. False to move towards goal using moveRate unmodified.*
```
public void MultiplyMagnitude(float multiplier, float moveRate,
    bool rateUsesDistance = true)
```

Multiplies roughness values in data by a set amount.
multiplier: Value to multiply by. 1f is standard multiplication, which in result would be default values.
moveRate: How quickly per second to move towards new multiplier. Values 0f and lower are instant.
rateUsesDistance: True to modify move rate based on distance from multiplier. False to move towards goal using moveRate unmodified.
```
public void MultiplyRoughness(float multiplier, float moveRate,
    bool rateUsesDistance = true)
```

Sets the paused state of this shaker. Pausing is independent of timescale.
value: New paused state.
```
public void SetPaused(bool value)
```
Returns if the instance shaker is over.
```
public bool ShakerOver()
```

CameraShaker

CameraShaker exist on cameras. CameraShaker manages and allows manipulation of ShakerInstance occurrences.

Events:

Dispatched when shaking starts after previously being stopped.

```
public event Action<CameraShaker> OnShakingStarted;
```

Dispatched when shaking ends.

```
public event Action<CameraShaker> OnShakingEnded;
```

Dispatched every update a shake occurs.

```
public event Action<CameraShaker, ShakeUpdate> OnShakeUpdate;
```

Dispatched every fixed update a shake occurs. Contains the shake values from last update.

```
public event Action<CameraShaker, ShakeUpdate>
    OnShakeFixedUpdate;
```

Properties:

Active instances which are shaking the camera.

```
public List<ShakerInstance> ShakerInstances { get; private
    set; }
```

Technique used to shake the camera.

```
public ShakeTechniques ShakeTechnique { get; private set; }
```

True if this CameraShaker is currently shaking.

```
public bool Shaking { get; private set; }
```

Current scale applied towards shakes on this CameraShaker. Acts as a multiplier towards ShakerInstances. 1f is normal scale.

```
public float Scale { get; private set; } = 1f;
```

Methods:

Sets Scale value.
value: New scale to use.

```
public void SetScale(float value)
```

Shakes the camera using data.
data: ShakeData to use.
returns: ShakerInstance generated using data.

```
public ShakerInstance Shake(ShakeData data)
```

Sets the paused state of all shaker instances on this CameraShaker. Pausing is independent of timescale.
value: New paused state.

```
public void SetPaused(bool value)
```

Stops all shaker instances on this CameraShaker abruptly.

```
public void Stop()
```

Fades out all ShakerInstances on this CameraShaker.
durationOverride: Overrides each instance Data fade out duration with a new value.

```
public void FadeOut(float? durationOverride = null)
```

Multiplies magnitude of values in data by a set amount for all ShakerInstances on this CameraShaker.
multiplier: Value to multiply by. 1f is standard multiplication, which in result would be default values.
moveRate: How quickly per second to move towards new multiplier. Values 0f and lower are instant.
rateUsesDistance: True to modify move rate based on distance from multiplier. False to move towards goal using moveRate unmodified.

```
public void MultiplyMagnitude(float multiplier, float moveRate,
    bool rateUsesDistance = true)
```

Multiplies roughness values in data by a set amount for all ShakerInstances on this CameraShaker.
multiplier: Value to multiply by. 1f is standard multiplication, which in result would be default values.
moveRate: How quickly per second to move towards new multiplier. Values 0f and lower are instant.
rateUsesDistance: True to modify move rate based on distance from multiplier. False to move towards goal using moveRate unmodified.

```
public void MultiplyRoughness(float multiplier, float moveRate,
    bool rateUsesDistance = true)
```

CameraShakerHandler

Self-activating singleton used to relay actions to and obtain information from instantiated CameraShaker scripts.

Events:

Dispatched after the default CameraShaker is changed.

```
public static event Action<CameraShakerChange>
    OnDefaultShakerChanged;
```

Dispatched when shaking starts when previously stopped on all CameraShakers.

```
public static event Action OnAllShakingStarted;
```

Dispatched when shaking ends on all CameraShakers.

```
public static event Action OnAllShakingEnded;

Dispatched every update a shake occurs.
public static event Action<ShakeUpdate> OnAllShakeUpdate;

Dispatched every fixed update a shake occurs. Contains the shake values from
last update.
public static event Action<ShakeUpdate> OnAllShakeFixedUpdate;

Dispatched when shaking starts on any camera shaker.
public static event Action<CameraShaker> OnShakingStarted;

Dispatched when shaking ends on any camera shaker.
public static event Action<CameraShaker> OnShakingEnded;

Dispatched every update a shake occurs on any camera shaker.
public static event Action<CameraShaker, ShakeUpdate>
      OnDefaultShakeUpdate;

Dispatched every fixed updated a shake occurs on any camera shaker.
Contains the shake values from last update.
public static event Action<CameraShaker, ShakeUpdate>
      OnShakeFixedUpdate;



Dispatched after a CameraShaker is added to InstantiatedShakers.
public static event Action<CameraShaker>
      OnShakerInstantiated.

Dispatched after a CameraShaker is removed from InstantiatedShakers.
public static event Action<CameraShaker>
      OnShakerDestroyed;
```

Properties:

```
All instantiated CameraShaker scripts.
public static List<CameraShaker> InstantiatedShakers;


Current default camera shaker.
public static CameraShaker DefaultCameraShaker;

True if any CameraShaker is currently shaking.
public static bool Shaking { get; private set; }
```

Methods:

Copies ShakerInstances from one CameraShaker to another.
from: CameraShaker to copy instances from.
to: CameraShaker to copy instances to.
copyOffset: True to copy the from cameras current offsets as well. Both CameraShakers must have the same ShakeTechnique for this to work.
TIP: perform this before disabling the shaker, as instances are cleared on disable.

```
public static void CopyShakerInstances(CameraShaker from,
      CameraShaker to, bool copyOffset = true)
```

Sets the DefaultCamereaShaker field.
value: New CameraShaker to use as default.

```
public static void SetDefaultCameraShaker(CameraShaker value)
```

Sets Scale value on the default CameraShaker.
value: New scale to use.

```
public static void SetScale(float value)
```

Sets the Scale value of InstantiatedCameraShakers.
Value: New scale to use
includeDisabled: True to issue call on disabled CameraShakers as well.

```
public static void SetScaleAll(float value,
      bool includeDisabled = false)
```

Shakes the default camera shaker using data.
data: ShakeData to use.
returns: ShakerInstance generated using data.

```
public static ShakerInstance Shake(ShakeData data)
```

Shakes all CameraShakers using data.
data: ShakeData to use.
includeDisabled: True to issue call on disabled CameraShakers as well.
returns: Collection of ShakerInstance generated using data.

```
public static List<ShakerInstance> ShakeAll(ShakeData data, bool
      includeDisabled = false)
```

Sets the paused state of all shaker instances on the default camera shaker. Pausing is independent of timescale.
value: New paused state.

```
public static void SetPaused(bool value)
```

Sets the paused state on all SpawnedCameraShakers. Pausing is independent of timescale.
value: New paused state.
includeDisabled: True to issue call on disabled CameraShakers as well.

```
public static void SetPausedAll(bool value,
      bool includeDisabled = false)
```

```
Abruptly stops all instances on the default camera shaker.
public static void Stop()

Abruptly stops all instances on SpawnedCameraShakers.
includeDisabled: True to issue call on disabled CameraShakers as well.
public static void StopAll(bool includeDisabled = false)

Fades out all ShakerInstances on the default camera shaker.
durationOverride: Overrides each instance Data fade out duration with a new
value.
public static void FadeOut(float? durationOverride = null)

Fades out all ShakerInstances on all SpawnedCameraShakers.
durationOverride: Overrides each instance Data fade out duration with a new
value.
includeDisabled: True to issue call on disabled CameraShakers as well.
public static void FadeOutAll(float? durationOverride = null)

Multiplies magnitude of values in data by a set amount for all
ShakerInstances on this CameraShaker.
multiplier: Value to multiply by. 1f is standard multiplication, which in
result would be default values.
moveRate: How quickly per second to move towards new multiplier. Values 0f
and lower are instant.
rateUsesDistance: True to modify move rate based on distance from multiplier.
False  to move towards goal using moveRate unmodified.
public static void MultiplyMagnitude(float multiplier, float
      moveRate, bool rateUsesDistance  = true)


Multiplies magnitude values for all instances on the default camera shaker.
multiplier: Value to multiply by. 1f is standard multiplication, which in
result would be default values.
moveRate: How quickly per second to move towards new multiplier. Values 0f
and lower are instant.
rateUsesDistance: True to modify move rate based on distance from multiplier.
False  to move towards goal using moveRate unmodified.
includeDisabled: True to issue call on disabled CameraShakers as well.
public static void MultiplyMagnitudeAll(float multiplier, float
      moveRate, bool rateUsesDistance  = true, bool
      includeDisabled = false)

Multiplies roughness values in data by a set amount for all
ShakerInstances on this CameraShaker.
multiplier: Value to multiply by. 1f is standard multiplication, which in
result would be default values.
moveRate: How quickly per second to move towards new multiplier. Values 0f
and lower are instant.
rateUsesDistance: True to modify move rate based on distance from multiplier.
False to move towards goal using moveRate unmodified.
public static void MultiplyRoughness(float multiplier, float
      moveRate, bool rateUsesDistance = true)
```

```
Multiplies roughness values in data by a set amount for all
ShakerInstances on this CameraShaker.
multiplier: Value to multiply by. 1f is standard multiplication, which in
result would be default values.
moveRate: How quickly per second to move towards new multiplier. Values 0f
and lower are instant.
rateUsesDistance: True to modify move rate based on distance from multiplier.
False to move towards goal using moveRate unmodified.
includeDisabled: True to issue call on disabled CameraShakers as well.
public static void MultiplyRoughnessAll(float multiplier, float
    moveRate, bool rateUsesDistance = true,
    bool includeDisabled = false)
```

ObjectShaker

ObjectShaker exist on gameObjects other than the camera. ObjectShaker manages and allows manipulation of ShakerInstance occurrences.

Events:

```
Dispatched when shaking starts after previously being stopped.
public event Action<ObjectShaker> OnShakingStarted;

Dispatched when shaking ends.
public event Action<ObjectShaker> OnShakingEnded;

Dispatched every update a shake occurs.
public event Action<ObjectShaker, ShakeUpdate> OnShakeUpdate;

Dispatched every fixed update a shake occurs. Contains the shake values from
last update.
public event Action<ObjectShaker, ShakeUpdate>
    OnShakeFixedUpdate;
```

Properties:

```
Active instances which are shaking the camera.
public List<ShakerInstance> ShakerInstances { get; private
    set; }

True if this shaker is currently shaking.
public bool Shaking { get; private set; }

Current scale applied towards shakes on this shaker. Acts as a
multiplier towards ShakerInstances. 1f is normal scale.
public float Scale { get; private set; } = 1f;
```

Methods:

```
Sets Scale value.
value: New scale to use.
public void SetScale(float value)
```

```
Shakes the camera using data.
data: ShakeData to use.
returns: ShakerInstance generated using data.
public ShakerInstance Shake(ShakeData data)

Sets the paused state of all shaker instances on this CameraShaker. Pausing
is independent of timescale.
value: New paused state.
public void SetPaused(bool value)

Stops all shaker instances on this CameraShaker abruptly.
public void Stop()

 Fades out all ShakerInstances on this CameraShaker.
durationOverride: Overrides each instance Data fade out duration with a new
value.
public void FadeOut(float? durationOverride = null)

Multiplies magnitude of values in data by a set amount for all
ShakerInstances on this CameraShaker.
multiplier: Value to multiply by. 1f is standard multiplication, which in
result would be default values.
moveRate: How quickly per second to move towards new multiplier. Values 0f
and lower are instant.
rateUsesDistance: True to modify move rate based on distance from multiplier.
False  to move towards goal using moveRate unmodified.
public void MultiplyMagnitude(float multiplier, float moveRate,
      bool rateUsesDistance = true)


Multiplies roughness values in data by a set amount for all
ShakerInstances on this CameraShaker.
multiplier: Value to multiply by. 1f is standard multiplication, which in
result would be default values.
moveRate: How quickly per second to move towards new multiplier. Values 0f
and lower are instant.
rateUsesDistance: True to modify move rate based on distance from multiplier.
False to move towards goal using moveRate unmodified.
public void MultiplyRoughness(float multiplier, float moveRate,
      bool rateUsesDistance = true)
```

ObjectShakerHandler

Self-activating singleton used to relay actions to and obtain information from instantiated ObjectShaker scripts.

Properties:

```
All instantiated CameraShaker scripts.
public static List<CameraShaker> InstantiatedShakers;
```

Methods:

Copies ShakerInstances from one ObjectShaker to another.
from: CameraShaker to copy instances from.
to: CameraShaker to copy instances to.
TIP: perform this before disabling the shaker, as instances are cleared on disable.

```
public static void CopyShakerInstances(CameraShaker from,
      CameraShaker to)
```

Sets the Scale value of InstantiatedCameraShakers.
Value: New scale to use
includeDisabled: True to issue call on disabled CameraShakers as well.

```
public static void SetScaleAll(float value,
      bool includeDisabled = false)
```

Shakes all CameraShakers using data.
data: ShakeData to use.
includeDisabled: True to issue call on disabled CameraShakers as well.
returns: Collection of ShakerInstance generated using data.

```
public static List<ShakerInstance> ShakeAll(ShakeData data, bool
      includeDisabled = false)
```

Sets the paused state on all SpawnedCameraShakers. Pausing is independent of timescale.
value: New paused state.
includeDisabled: True to issue call on disabled CameraShakers as well.

```
public static void SetPausedAll(bool value,
      bool includeDisabled = false)
```

Abruptly stops all instances on SpawnedCameraShakers.
includeDisabled: True to issue call on disabled CameraShakers as well.

```
public static void StopAll(bool includeDisabled = false)
```

Fades out all ShakerInstances on all SpawnedCameraShakers.
durationOverride: Overrides each instance Data fade out duration with a new value.
includeDisabled: True to issue call on disabled CameraShakers as well.

```
public static void FadeOutAll(float? durationOverride = null)
```

Multiplies magnitude values for all instances on the default camera shaker.
multiplier: Value to multiply by. 1f is standard multiplication, which in result would be default values.
moveRate: How quickly per second to move towards new multiplier. Values 0f and lower are instant.
rateUsesDistance: True to modify move rate based on distance from multiplier. False to move towards goal using moveRate unmodified.
includeDisabled: True to issue call on disabled CameraShakers as well.

```
public static void MultiplyMagnitudeAll(float multiplier, float
      moveRate, bool rateUsesDistance  = true, bool
      includeDisabled = false)
```

```
Multiplies roughness values in data by a set amount for all
ShakerInstances on this CameraShaker.
multiplier: Value to multiply by. 1f is standard multiplication, which in
result would be default values.
moveRate: How quickly per second to move towards new multiplier. Values 0f
and lower are instant.
rateUsesDistance: True to modify move rate based on distance from multiplier.
False to move towards goal using moveRate unmodified.
includeDisabled: True to issue call on disabled CameraShakers as well.
public static void MultiplyRoughnessAll(float multiplier, float
      moveRate, bool rateUsesDistance = true,
      bool includeDisabled = false)
```