public
abstract class

Added in API level 1

Animation

extends Object
implements Cloneable

java.lang.Object
↳android.view.animation.Animation

Known Direct Subclasses
AlphaAnimation, AnimationSet, RotateAnimation, ScaleAnimation, TranslateAnimation

Class Overview

Abstraction for an Animation that can be applied to Views, Surfaces, or other objects. See the animation package description file (/reference/android/view/animation/package-summary.html).

Summary

Nested Classes

interface Animation.AnimationListener	An animation listener receives notifications from an animation.
class Animation.Description	Utility class to parse a string description of a size.

XML Attributes

Attribute Name	*Related Method*	*Description*
android:detachWallpaper	setDetachWallpaper(boolean)	Special option for window animations: if this window is on top of a wallpaper, don't animate the wallpaper with it.
android:duration	setDuration(long)	Amount of time (in milliseconds) for the animation to run.
android:fillAfter	setFillAfter(boolean)	When set to true, the animation transformation is applied after the animation is over.
		When set to true or when fillEnabled is

android:fillBefore	setFillBefore(boolean)	not set to true, the animation transformation is applied before the animation has started.
android:fillEnabled	setFillEnabled(boolean)	When set to true, the value of fillBefore is taken into account.
android:interpolator	setInterpolator(Interpolator)	Defines the interpolator used to smooth the animation movement in time.
android:repeatCount	setRepeatCount(int)	Defines how many times the animation should repeat.
android:repeatMode	setRepeatMode(int)	Defines the animation behavior when it reaches the end and the repeat count is greater than 0 or infinite.
android:startOffset	setStartOffset(long)	Delay in milliseconds before the animation runs, once start time is reached.
android:zAdjustment	setZAdjustment(int)	Allows for an adjustment of the Z ordering of the content being animated for the duration of the animation.

Constants

int ABSOLUTE	The specified dimension is an absolute number of pixels.
int INFINITE	Repeat the animation indefinitely.
int RELATIVE_TO_PARENT	The specified dimension holds a float and should be multiplied by the height or width of the parent of the object being animated.
int RELATIVE_TO_SELF	The specified dimension holds a float and should be multiplied by the height or width of the object being animated.
int RESTART	When the animation reaches the end and the repeat count is INFINTE_REPEAT or a positive value, the animation restarts from the beginning.
	When the animation reaches the end and the repeat count is INFINTE_REPEAT or a

int REVERSE	positive value, the animation plays backward (and then forward again).
int START_ON_FIRST_FRAME	Can be used as the start time to indicate the start time should be the current time when `getTransformation(long, Transformation)` is invoked for the first animation frame.
int ZORDER_BOTTOM	Requests that the content being animated be forced under all other content for the duration of the animation.
int ZORDER_NORMAL	Requests that the content being animated be kept in its current Z order.
int ZORDER_TOP	Requests that the content being animated be forced on top of all other content for the duration of the animation.

Public Constructors

Animation ()

Creates a new animation with a duration of 0ms, the default interpolator, with fillBefore set to true and fillAfter set to false

Animation (Context context, AttributeSet attrs)

Creates a new animation whose parameters come from the specified context and attributes set.

Public Methods

void	cancel () Cancel the animation.
long	computeDurationHint () Compute a hint at how long the entire animation may last, in milliseconds.
int	getBackgroundColor () Returns the background color behind the animation.
boolean	getDetachWallpaper () Return value of `setDetachWallpaper(boolean)`.
long	getDuration () How long this animation should last
boolean	getFillAfter () If fillAfter is true, this animation will apply its transformation after the end time of the animation.
boolean	getFillBefore () If fillBefore is true, this animation will apply its transformation before the start time of the animation.
Interpolator	getInterpolator () Gets the acceleration curve type for this animation.
int	getRepeatCount () Defines how many times the animation should repeat.
int	getRepeatMode () Defines what this animation should do when it reaches the end.
long	getStartOffset () When this animation should start, relative to StartTime

long	getStartTime () When this animation should start.
boolean	getTransformation (long currentTime, Transformation outTransformation, float scale) Gets the transformation to apply at a specified point in time.
boolean	getTransformation (long currentTime, Transformation outTransformation) Gets the transformation to apply at a specified point in time.
int	getZAdjustment () Returns the Z ordering mode to use while running the animation as previously set by `setZAdjustment(int)`.
boolean	hasEnded () Indicates whether this animation has ended or not.
boolean	hasStarted () Indicates whether this animation has started or not.
void	initialize (int width, int height, int parentWidth, int parentHeight) Initialize this animation with the dimensions of the object being animated as well as the objects parents.
boolean	isFillEnabled () If fillEnabled is true, this animation will apply the value of fillBefore.
boolean	isInitialized () Whether or not the animation has been initialized.
void	reset () Reset the initialization state of this animation.
void	restrictDuration (long durationMillis) Ensure that the duration that this animation will run is not longer than *durationMillis*.
void	scaleCurrentDuration (float scale) How much to scale the duration by.
void	setAnimationListener (Animation.AnimationListener listener) Binds an animation listener to this animation.
void	setBackgroundColor (int bg) Set background behind animation.
void	setDetachWallpaper (boolean detachWallpaper) If detachWallpaper is true, and this is a window animation of a window that has a wallpaper background, then the window will be detached from the wallpaper while it runs.
void	setDuration (long durationMillis) How long this animation should last.
void	setFillAfter (boolean fillAfter) If fillAfter is true, the transformation that this animation performed will persist when it is finished.
void	setFillBefore (boolean fillBefore) If fillBefore is true, this animation will apply its transformation before the start time of the animation.
	setFillEnabled (boolean fillEnabled)

void	If fillEnabled is true, the animation will apply the value of fillBefore.
void	setInterpolator (Context context, int resID) Sets the acceleration curve for this animation.
void	setInterpolator (Interpolator i) Sets the acceleration curve for this animation.
void	setRepeatCount (int repeatCount) Sets how many times the animation should be repeated.
void	setRepeatMode (int repeatMode) Defines what this animation should do when it reaches the end.
void	setStartOffset (long startOffset) When this animation should start relative to the start time.
void	setStartTime (long startTimeMillis) When this animation should start.
void	setZAdjustment (int zAdjustment) Set the Z ordering mode to use while running the animation.
void	start () Convenience method to start the animation the first time `getTransformation(long, Transformation)` is invoked.
void	startNow () Convenience method to start the animation at the current time in milliseconds.
boolean	willChangeBounds () Indicates whether or not this animation will affect the bounds of the animated view.
boolean	willChangeTransformationMatrix () Indicates whether or not this animation will affect the transformation matrix.

Protected Methods

void	applyTransformation (float interpolatedTime, Transformation t) Helper for getTransformation.
Animation	clone () Creates and returns a copy of this `Object`.
void	ensureInterpolator () Gurantees that this animation has an interpolator.
void	finalize () Invoked when the garbage collector has detected that this instance is no longer reachable.
float	getScaleFactor () The scale factor is set by the call to `getTransformation`.
float	resolveSize (int type, float value, int size, int parentSize) Convert the information in the description of a size to an actual dimension

Inherited Methods [Expand]

▸ From class java.lang.Object

XML Attributes

android:detachWallpaper

Special option for window animations: if this window is on top of a wallpaper, don't animate the wallpaper with it.

Must be a boolean value, either "`true`" or "`false`".

This may also be a reference to a resource (in the form "`@[`*package*`:]`*type*`:`*name*") or theme attribute (in the form "`?[`*package*`:][`*type*`:]`*name*") containing a value of this type.

This corresponds to the global attribute resource symbol `detachWallpaper` (/reference/android/R.attr.html#detachWallpaper).

Related Methods

setDetachWallpaper(boolean)

android:duration

Amount of time (in milliseconds) for the animation to run.

Must be an integer value, such as "100".

This may also be a reference to a resource (in the form "`@[`*package*`:]`*type*`:`*name*") or theme attribute (in the form "`?[`*package*`:][`*type*`:]`*name*") containing a value of this type.

This corresponds to the global attribute resource symbol `duration` (/reference/android/R.attr.html#duration).

Related Methods

setDuration(long)

android:fillAfter

When set to true, the animation transformation is applied after the animation is over. The default value is false. If fillEnabled is not set to true and the animation is not set on a View, fillAfter is assumed to be true.

Must be a boolean value, either "`true`" or "`false`".

This may also be a reference to a resource (in the form "`@[`*package*`:]`*type*`:`*name*") or theme attribute (in the form "`?[`*package*`:][`*type*`:]`*name*") containing a value of this type.

This corresponds to the global attribute resource symbol `fillAfter` (/reference/android/R.attr.html#fillAfter).

Related Methods

setFillAfter(boolean)

android:fillBefore

When set to true or when fillEnabled is not set to true, the animation transformation is applied before the animation has started. The default value is true.

Must be a boolean value, either "`true`" or "`false`".

This may also be a reference to a resource (in the form "`@[`*package*`:]`*type*`:`*name*") or theme attribute (in the form "`?[`*package*`:][`*type*`:]`*name*") containing a value of this type.

This corresponds to the global attribute resource symbol `fillBefore` (/reference/android/R.attr.html#fillBefore).

Related Methods
setFillBefore(boolean)

android:fillEnabled

When set to true, the value of fillBefore is taken into account.

Must be a boolean value, either "`true`" or "`false`".

This may also be a reference to a resource (in the form "`@[`*package*`:]`*type*`:`*name*") or theme attribute (in the form "`?[`*package*`:][`*type*`:]`*name*") containing a value of this type.

This corresponds to the global attribute resource symbol `fillEnabled` (/reference/android/R.attr.html#fillEnabled).

Related Methods
setFillEnabled(boolean)

android:interpolator

Defines the interpolator used to smooth the animation movement in time.

Must be a reference to another resource, in the form "`@[+][`*package*`:]`*type*`:`*name*" or to a theme attribute in the form "`?[`*package*`:][`*type*`:]`*name*".

This corresponds to the global attribute resource symbol `interpolator` (/reference/android/R.attr.html#interpolator).

Related Methods
setInterpolator(Interpolator)

android:repeatCount

Defines how many times the animation should repeat. The default value is 0.

May be an integer value, such as "100".

This may also be a reference to a resource (in the form "`@[`*package*`:]`*type*`:`*name*") or theme attribute (in the form "?

[*package*:][*type*:]*name*") containing a value of this type.

May be one of the following constant values.

Constant	Value	Description
infinite	-1	

This corresponds to the global attribute resource symbol repeatCount (/reference/android/R.attr.html#repeatCount).

Related Methods

setRepeatCount(int)

android:repeatMode

Defines the animation behavior when it reaches the end and the repeat count is greater than 0 or infinite. The default value is restart.

Must be one of the following constant values.

Constant	Value	Description
[illegible]		

This corresponds to the global attribute resource symbol repeatMode (/reference/android/R.attr.html#repeatMode).

Related Methods

setRepeatMode(int)

android:startOffset

Delay in milliseconds before the animation runs, once start time is reached.

Must be an integer value, such as "100".

This may also be a reference to a resource (in the form "@[*package*:]*type*:*name*") or theme attribute (in the form "?[*package*:][*type*:]*name*") containing a value of this type.

This corresponds to the global attribute resource symbol startOffset (/reference/android/R.attr.html#startOffset).

Related Methods

setStartOffset(long)

android:zAdjustment

Allows for an adjustment of the Z ordering of the content being animated for the duration of the animation. The default value is normal.

Must be one of the following constant values.

Constant	Value	Description
		The content being animated be kept in its current Z

`normal`	`0`	order.
`top`	`1`	The content being animated is forced on top of all other content for the duration of the animation.
`bottom`	`-1`	The content being animated is forced under all other content for the duration of the animation.

This corresponds to the global attribute resource symbol `zAdjustment` (`/reference/android/R.attr.html#zAdjustment`).

Related Methods

setZAdjustment(int)

Constants

public static final int **ABSOLUTE**

Added in API level 1

The specified dimension is an absolute number of pixels.

Constant Value: 0 (0x00000000)

public static final int **INFINITE**

Added in API level 1

Repeat the animation indefinitely.

Constant Value: -1 (0xffffffff)

public static final int **RELATIVE_TO_PARENT**

Added in API level 1

The specified dimension holds a float and should be multiplied by the height or width of the parent of the object being animated.

Constant Value: 2 (0x00000002)

public static final int **RELATIVE_TO_SELF**

Added in API level 1

The specified dimension holds a float and should be multiplied by the height or width of the object being animated.

Constant Value: 1 (0x00000001)

public static final int **RESTART**

Added in API level 1

When the animation reaches the end and the repeat count is INFINTE_REPEAT or a positive value, the animation restarts from the beginning.

Constant Value: 1 (0x00000001)

public static final int **REVERSE**

Added in API level 1

When the animation reaches the end and the repeat count is INFINTE_REPEAT or a positive value, the animation plays backward (and then forward again).

Constant Value: 2 (0x00000002)

public static final int **START_ON_FIRST_FRAME** Added in API level 1

Can be used as the start time to indicate the start time should be the current time when `getTransformation(long, Transformation)` (/reference/android/view/animation/Animation.html#getTransformation(long, android.view.animation.Transformation)) is invoked for the first animation frame. This can is useful for short animations.

Constant Value: -1 (0xffffffff)

public static final int **ZORDER_BOTTOM** Added in API level 1

Requests that the content being animated be forced under all other content for the duration of the animation.

Constant Value: -1 (0xffffffff)

public static final int **ZORDER_NORMAL** Added in API level 1

Requests that the content being animated be kept in its current Z order.

Constant Value: 0 (0x00000000)

public static final int **ZORDER_TOP** Added in API level 1

Requests that the content being animated be forced on top of all other content for the duration of the animation.

Constant Value: 1 (0x00000001)

Public Constructors

public **Animation** () Added in API level 1

Creates a new animation with a duration of 0ms, the default interpolator, with fillBefore set to true and fillAfter set to false

public **Animation** (Context context, AttributeSet attrs) Added in API level 1

Creates a new animation whose parameters come from the specified context and attributes set.

Parameters

context	the application environment
attrs	the set of attributes holding the animation parameters

Public Methods

public void **cancel** () Added in API level 8

Cancel the animation. Cancelling an animation invokes the animation listener, if set, to notify the end of the animation. If you cancel an animation manually, you must call `reset()` (/reference/android/view/animation/Animation.html#reset()) before starting the animation again.

See Also

`reset()`
`start()`
`startNow()`

public long **computeDurationHint** ()

Added in API level 3

Compute a hint at how long the entire animation may last, in milliseconds. Animations can be written to cause themselves to run for a different duration than what is computed here, but generally this should be accurate.

public int **getBackgroundColor** ()

Added in API level 12

Returns the background color behind the animation.

public boolean **getDetachWallpaper** ()

Added in API level 5

Return value of `setDetachWallpaper(boolean)` (/reference/android/view/animation/Animation.html#setDetachWallpaper(boolean)).

Related XML Attributes

android:detachWallpaper

public long **getDuration** ()

Added in API level 1

How long this animation should last

Related XML Attributes

android:duration

Returns

the duration in milliseconds of the animation

public boolean **getFillAfter** ()

Added in API level 1

If fillAfter is true, this animation will apply its transformation after the end time of the animation.

Related XML Attributes

android:fillAfter

Returns

true if the animation applies its transformation after it ends

public boolean **getFillBefore** ()

Added in API level 1

If fillBefore is true, this animation will apply its transformation before the start time of the animation. If fillBefore is false and `fillEnabled (/reference/android/view/animation/Animation.html#isFillEnabled())` is true, the transformation will not be applied until the start time of the animation.

Related XML Attributes

android:fillBefore

Returns

true if the animation applies its transformation before it starts

public Interpolator **getInterpolator** ()

Added in API level 1

Gets the acceleration curve type for this animation.

Related XML Attributes

android:interpolator

Returns

the `Interpolator` associated to this animation

public int **getRepeatCount** ()

Added in API level 1

Defines how many times the animation should repeat. The default value is 0.

Related XML Attributes

android:repeatCount

Returns

the number of times the animation should repeat, or `INFINITE`

public int **getRepeatMode** ()

Added in API level 1

Defines what this animation should do when it reaches the end.

Related XML Attributes

android:repeatMode

Returns

either one of `REVERSE` or `RESTART`

public long **getStartOffset** ()

Added in API level 1

When this animation should start, relative to StartTime

Related XML Attributes

android:startOffset

Returns

the start offset in milliseconds

public long **getStartTime** ()

Added in API level 1

When this animation should start. If the animation has not startet yet, this

method might return START_ON_FIRST_FRAME (/reference/android/view/animation/Animation.html#START_ON_FIRST_FRAME).

Returns

the time in milliseconds when the animation should start or START_ON_FIRST_FRAME

public boolean **getTransformation** (long currentTime, Transformation outTransformation, float scale)

Added in API level 11

Gets the transformation to apply at a specified point in time. Implementations of this method should always replace the specified Transformation or document they are doing otherwise.

Parameters

currentTime	Where we are in the animation. This is wall clock time.
outTransformation	A transformation object that is provided by the caller and will be filled in by the animation.
scale	Scaling factor to apply to any inputs to the transform operation, such pivot points being rotated or scaled around.

Returns

True if the animation is still running

public boolean **getTransformation** (long currentTime, Transformation outTransformation)

Added in API level 1

Gets the transformation to apply at a specified point in time. Implementations of this method should always replace the specified Transformation or document they are doing otherwise.

Parameters

currentTime	Where we are in the animation. This is wall clock time.
outTransformation	A transformation object that is provided by the caller and will be filled in by the animation.

Returns

True if the animation is still running

public int **getZAdjustment** ()

Added in API level 1

Returns the Z ordering mode to use while running the animation as previously set by setZAdjustment(int) (/reference/android/view/animation/Animation.html#setZAdjustment(int)).

Related XML Attributes

android:zAdjustment

Returns

Returns one of ZORDER_NORMAL, ZORDER_TOP, or ZORDER_BOTTOM.

public boolean **hasEnded** ()

Added in API level 1

Indicates whether this animation has ended or not.

Returns

true if the animation has ended, false otherwise

public boolean **hasStarted** ()

Added in API level 1

Indicates whether this animation has started or not.

Returns

true if the animation has started, false otherwise

public void **initialize** (int width, int height, int parentWidth, int parentHeight)

Added in API level 1

Initialize this animation with the dimensions of the object being animated as well as the objects parents. (This is to support animation sizes being specified relative to these dimensions.)

Objects that interpret Animations should call this method when the sizes of the object being animated and its parent are known, and before calling `getTransformation(long, Transformation)` (/reference/android/view/animation/Animation.html#getTransformation(long, android.view.animation.Transformation)).

Parameters

width	Width of the object being animated
height	Height of the object being animated
parentWidth	Width of the animated object's parent
parentHeight	Height of the animated object's parent

public boolean **isFillEnabled** ()

Added in API level 3

If fillEnabled is true, this animation will apply the value of fillBefore.

Related XML Attributes

android:fillEnabled

Returns

true if the animation will take fillBefore into account

public boolean **isInitialized** ()

Added in API level 1

Whether or not the animation has been initialized.

Returns

Has this animation been initialized.

See Also

`initialize(int, int, int, int)`

public void **reset** ()

Added in API level 1

Reset the initialization state of this animation.

See Also

`initialize(int, int, int, int)`

public void **restrictDuration** (long durationMillis)

Added in API level 1

Ensure that the duration that this animation will run is not longer than *durationMillis*. In addition to adjusting the duration itself, this ensures that the repeat count also will not make it run longer than the given time.

Parameters

durationMillis	The maximum duration the animation is allowed to run.

public void **scaleCurrentDuration** (float scale)

Added in API level 1

How much to scale the duration by.

Parameters

scale	The amount to scale the duration.

public void **setAnimationListener** (Animation.AnimationListener listener)

Added in API level 1

Binds an animation listener to this animation. The animation listener is notified of animation events such as the end of the animation or the repetition of the animation.

Parameters

listener	the animation listener to be notified

public void **setBackgroundColor** (int bg)

Added in API level 12

Set background behind animation.

Parameters

bg	The background color. If 0, no background. Currently must be black, with any desired alpha level.

public void **setDetachWallpaper** (boolean detachWallpaper)

Added in API level 5

If detachWallpaper is true, and this is a window animation of a window that has a wallpaper background, then the window will be detached from the wallpaper while it runs. That is, the animation will only be applied to the window, and the wallpaper behind it will remain static.

Related XML Attributes

android:detachWallpaper

Parameters

detachWallpaper true if the wallpaper should be detached from the animation

public void **setDuration** (long durationMillis) Added in API level 1

How long this animation should last. The duration cannot be negative.

Related XML Attributes

android:duration

Parameters

durationMillis Duration in milliseconds

Throws

IllegalArgumentException if the duration is < 0

public void **setFillAfter** (boolean fillAfter) Added in API level 1

If fillAfter is true, the transformation that this animation performed will persist when it is finished. Defaults to false if not set. Note that this applies to individual animations and when using an `AnimationSet (/reference/android/view/animation/AnimationSet.html)` to chain animations.

Related XML Attributes

android:fillAfter

Parameters

fillAfter true if the animation should apply its transformation after it ends

See Also

`setFillEnabled(boolean)`

public void **setFillBefore** (boolean fillBefore) Added in API level 1

If fillBefore is true, this animation will apply its transformation before the start time of the animation. Defaults to true if `setFillEnabled(boolean) (/reference/android/view/animation/Animation.html#setFillEnabled(boolean))` is not set to true. Note that this applies when using an `AnimationSet (/reference/android/view/animation/AnimationSet.html)` to chain animations. The transformation is not applied before the AnimationSet itself starts.

Related XML Attributes

android:fillBefore

Parameters

fillBefore true if the animation should apply its transformation before it starts

See Also

`setFillEnabled(boolean)`

public void **setFillEnabled** (boolean fillEnabled) Added in API level 3

If fillEnabled is true, the animation will apply the value of fillBefore. Otherwise, fillBefore is ignored and the animation transformation is always applied until the animation ends.

Related XML Attributes

android:fillEnabled

Parameters

fillEnabled	true if the animation should take the value of fillBefore into account

See Also

`setFillBefore(boolean)`
`setFillAfter(boolean)`

public void **setInterpolator** (Context context, int resID) Added in API level 1

Sets the acceleration curve for this animation. The interpolator is loaded as a resource from the specified context.

Related XML Attributes

android:interpolator

Parameters

context	The application environment
resID	The resource identifier of the interpolator to load

public void **setInterpolator** (Interpolator i) Added in API level 1

Sets the acceleration curve for this animation. Defaults to a linear interpolation.

Related XML Attributes

android:interpolator

Parameters

i	The interpolator which defines the acceleration curve

public void **setRepeatCount** (int repeatCount) Added in API level 1

Sets how many times the animation should be repeated. If the repeat count is 0, the animation is never repeated. If the repeat count is greater than 0 or `INFINITE` `(/reference/android/view/animation/Animation.html#INFINITE)`, the repeat mode will be taken into account. The repeat count is 0 by default.

Related XML Attributes

android:repeatCount

Parameters

repeatCount	the number of times the animation should be repeated

public void **setRepeatMode** (int repeatMode)

Added in API level 1

Defines what this animation should do when it reaches the end. This setting is applied only when the repeat count is either greater than 0 or INFINITE (/reference/android/view/animation/Animation.html#INFINITE). Defaults to RESTART (/reference/android/view/animation/Animation.html#RESTART).

Related XML Attributes

android:repeatMode

Parameters

repeatMode	RESTART or REVERSE

public void **setStartOffset** (long startOffset)

Added in API level 1

When this animation should start relative to the start time. This is most useful when composing complex animations using an AnimationSet (/reference/android/view/animation/AnimationSet.html) where some of the animations components start at different times.

Related XML Attributes

android:startOffset

Parameters

startOffset	When this Animation should start, in milliseconds from the start time of the root AnimationSet.

public void **setStartTime** (long startTimeMillis)

Added in API level 1

When this animation should start. When the start time is set to START_ON_FIRST_FRAME (/reference/android/view/animation/Animation.html#START_ON_FIRST_FRAME), the animation will start the first time getTransformation(long, Transformation) (/reference/android/view/animation/Animation.html#getTransformation(long, android.view.animation.Transformation)) is invoked. The time passed to this method should be obtained by calling currentAnimationTimeMillis() (/reference/android/view/animation/AnimationUtils.html#currentAnimationTimeMillis()) instead of currentTimeMillis() (/reference/java/lang/System.html#currentTimeMillis()).

Parameters

startTimeMillis	the start time in milliseconds

public void **setZAdjustment** (int zAdjustment)

Added in API level 1

Set the Z ordering mode to use while running the animation.

Related XML Attributes

android:zAdjustment

Parameters

zAdjustment	The desired mode, one of ZORDER_NORMAL, ZORDER_TOP, or ZORDER_BOTTOM.

public void **start** ()

Added in API level 1

Convenience method to start the animation the first time getTransformation(long, Transformation) (/reference/android/view/animation/Animation.html#getTransformation(long, android.view.animation.Transformation)) is invoked.

public void **startNow** ()

Added in API level 1

Convenience method to start the animation at the current time in milliseconds.

public boolean **willChangeBounds** ()

Added in API level 1

Indicates whether or not this animation will affect the bounds of the animated view. For instance, a fade animation will not affect the bounds whereas a 200% scale animation will.

Returns

true if this animation will change the view's bounds

public boolean **willChangeTransformationMatrix** ()

Added in API level 1

Indicates whether or not this animation will affect the transformation matrix. For instance, a fade animation will not affect the matrix whereas a scale animation will.

Returns

true if this animation will change the transformation matrix

Protected Methods

protected void **applyTransformation** (float interpolatedTime, Transformation t)

Added in API level 1

Helper for getTransformation. Subclasses should implement this to apply their transforms given an interpolation value. Implementations of this method should always replace the specified Transformation or document they are doing otherwise.

Parameters

interpolatedTime	The value of the normalized time (0.0 to 1.0) after it has been run through the interpolation function.
t	The Transformation object to fill in with the current transforms.

protected Animation **clone** ()

Added in API level 1

Creates and returns a copy of this `Object`. The default implementation returns a so-called "shallow" copy: It creates a new instance of the same class and then copies the field values (including object references) from this instance to the new instance. A "deep" copy, in contrast, would also recursively clone nested objects. A subclass that needs to implement this kind of cloning should call `super.clone()` to create the new instance and then create deep copies of the nested, mutable objects.

Returns

a copy of this object.

Throws

CloneNotSupportedException

protected void **ensureInterpolator** ()

Added in API level 1

Gurantees that this animation has an interpolator. Will use a AccelerateDecelerateInterpolator is nothing else was specified.

protected void **finalize** ()

Added in API level 1

Invoked when the garbage collector has detected that this instance is no longer reachable. The default implementation does nothing, but this method can be overridden to free resources.

Note that objects that override `finalize` are significantly more expensive than objects that don't. Finalizers may be run a long time after the object is no longer reachable, depending on memory pressure, so it's a bad idea to rely on them for cleanup. Note also that finalizers are run on a single VM-wide finalizer thread, so doing blocking work in a finalizer is a bad idea. A finalizer is usually only necessary for a class that has a native peer and needs to call a native method to destroy that peer. Even then, it's better to provide an explicit `close` method (and implement `Closeable (/reference/java/io/Closeable.html)`), and insist that callers manually dispose of instances. This works well for something like files, but less well for something like a `BigInteger` where typical calling code would have to deal with lots of temporaries. Unfortunately, code that creates lots of temporaries is the worst kind of code from the point of view of the single finalizer thread.

If you *must* use finalizers, consider at least providing your own `ReferenceQueue (/reference/java/lang/ref/ReferenceQueue.html)` and having your own thread process that queue.

Unlike constructors, finalizers are not automatically chained. You are responsible for calling `super.finalize()` yourself.

Uncaught exceptions thrown by finalizers are ignored and do not terminate the finalizer thread. See *Effective Java* Item 7, "Avoid finalizers" for more.

Throws

Throwable

protected float **getScaleFactor** ()

Added in API level 11

The scale factor is set by the call to `getTransformation`. Overrides of `getTransformation(long, Transformation, float)` (/reference/android/view/animation/Animation.html#getTransformation(long, android.view.animation.Transformation, float)) will get this value directly. Overrides of `applyTransformation(float, Transformation)` (/reference/android/view/animation/Animation.html#applyTransformation(float, android.view.animation.Transformation)) can call this method to get the value.

Returns

float The scale factor that should be applied to pre-scaled values in an Animation such as the pivot points in `ScaleAnimation` and `RotateAnimation`.

protected float **resolveSize** (int type, float value, int size, int parentSize)

Added in API level 1

Convert the information in the description of a size to an actual dimension

Parameters

type	One of Animation.ABSOLUTE, Animation.RELATIVE_TO_SELF, or Animation.RELATIVE_TO_PARENT.
value	The dimension associated with the type parameter
size	The size of the object being animated
parentSize	The size of the parent of the object being animated

Returns

The dimension to use for the animation