public class
Paint
extends Object

Summary: Nested Classes | Constants | Ctors | Methods | Protected Methods | Inherited Methods | [Expand All]
Added in API level 1

java.lang.Object
↳android.graphics.Paint

▸ Known Direct Subclasses
TextPaint

Class Overview

The Paint class holds the style and color information about how to draw geometries, text and bitmaps.

Summary

Nested Classes

enum Paint.Align	Align specifies how drawText aligns its text relative to the [x,y] coordinates.
enum Paint.Cap	The Cap specifies the treatment for the beginning and ending of stroked lines and paths.
class Paint.FontMetrics	Class that describes the various metrics for a font at a given text size.
class Paint.FontMetricsInt	Convenience method for callers that want to have FontMetrics values as integers.
enum Paint.Join	The Join specifies the treatment where lines and curve segments join on a stroked path.
enum Paint.Style	The Style specifies if the primitive being drawn is filled, stroked, or both (in the same color).

Constants

int ANTI_ALIAS_FLAG	Paint flag that enables antialiasing when drawing.
int DEV_KERN_TEXT_FLAG	Legacy Paint flag, no longer used.
int DITHER_FLAG	Paint flag that enables dithering when blitting.
int EMBEDDED_BITMAP_TEXT_FLAG	Paint flag that enables the use of bitmap fonts when drawing text.
int FAKE_BOLD_TEXT_FLAG	Paint flag that applies a synthetic bolding effect to drawn text.
int FILTER_BITMAP_FLAG	Paint flag that enables bilinear sampling on scaled bitmaps.
int HINTING_OFF	Font hinter option that disables font hinting.
int HINTING_ON	Font hinter option that enables font hinting.
int LINEAR_TEXT_FLAG	Paint flag that enables smooth linear scaling of text.
int STRIKE_THRU_TEXT_FLAG	Paint flag that applies a strike-through decoration to drawn text.
int SUBPIXEL_TEXT_FLAG	Paint flag that enables subpixel positioning of text.
int UNDERLINE_TEXT_FLAG	Paint flag that applies an underline decoration to drawn text.

Public Constructors

Paint ()
Create a new paint with default settings.

Paint (int flags)
Create a new paint with the specified flags.

Paint (Paint paint)
Create a new paint, initialized with the attributes in the specified paint parameter.

Public Methods

float	ascent () Return the distance above (negative) the baseline (ascent) based on the current typeface and text size.
int	breakText (CharSequence text, int start, int end, boolean measureForwards, float maxWidth, float[] measuredWidth) Measure the text, stopping early if the measured width exceeds maxWidth.
int	breakText (String text, boolean measureForwards, float maxWidth, float[] measuredWidth) Measure the text, stopping early if the measured width exceeds maxWidth.
int	breakText (char[] text, int index, int count, float maxWidth, float[] measuredWidth) Measure the text, stopping early if the measured width exceeds maxWidth.
void	clearShadowLayer () Clear the shadow layer.
float	descent () Return the distance below (positive) the baseline (descent) based on the current typeface and text size.

int	getAlpha () Helper to getColor() that just returns the color's alpha value.
int	getColor () Return the paint's color.
ColorFilter	getColorFilter () Get the paint's colorfilter (maybe be null).
boolean	getFillPath (Path src, Path dst) Applies any/all effects (patheffect, stroking) to src, returning the result in dst.
int	getFlags () Return the paint's flags.
float	getFontMetrics (Paint.FontMetrics metrics) Return the font's recommended interline spacing, given the Paint's settings for typeface, textSize, etc.
Paint.FontMetrics	getFontMetrics () Allocates a new FontMetrics object, and then calls getFontMetrics(fm) with it, returning the object.
Paint.FontMetricsInt	getFontMetricsInt ()
int	getFontMetricsInt (Paint.FontMetricsInt fmi) Return the font's interline spacing, given the Paint's settings for typeface, textSize, etc.
float	getFontSpacing () Return the recommend line spacing based on the current typeface and text size.
int	getHinting () Return the paint's hinting mode.
MaskFilter	getMaskFilter () Get the paint's maskfilter object.
PathEffect	getPathEffect () Get the paint's patheffect object.
Rasterizer	getRasterizer () Get the paint's rasterizer (or null).
Shader	getShader () Get the paint's shader object.
Paint.Cap	getStrokeCap () Return the paint's Cap, controlling how the start and end of stroked lines and paths are treated.
Paint.Join	getStrokeJoin () Return the paint's stroke join type.
float	getStrokeMiter () Return the paint's stroke miter value.
float	getStrokeWidth () Return the width for stroking.
Paint.Style	getStyle () Return the paint's style, used for controlling how primitives' geometries are interpreted (except for drawBitmap, which always assumes FILL_STYLE).
Paint.Align	getTextAlign () Return the paint's Align value for drawing text.
void	getTextBounds (char[] text, int index, int count, Rect bounds) Return in bounds (allocated by the caller) the smallest rectangle that encloses all of the characters, with an implied origin at (0,0).
void	getTextBounds (String text, int start, int end, Rect bounds) Return in bounds (allocated by the caller) the smallest rectangle that encloses all of the characters, with an implied origin at (0,0).
Locale	getTextLocale () Get the text Locale.
void	getTextPath (String text, int start, int end, float x, float y, Path path) Return the path (outline) for the specified text.
void	getTextPath (char[] text, int index, int count, float x, float y, Path path) Return the path (outline) for the specified text.
float	getTextScaleX () Return the paint's horizontal scale factor for text.
float	getTextSize () Return the paint's text size.
float	getTextSkewX () Return the paint's horizontal skew factor for text.

int	getTextWidths (String text, float[] widths) Return the advance widths for the characters in the string.
int	getTextWidths (CharSequence text, int start, int end, float[] widths) Return the advance widths for the characters in the string.
int	getTextWidths (String text, int start, int end, float[] widths) Return the advance widths for the characters in the string.
int	getTextWidths (char[] text, int index, int count, float[] widths) Return the advance widths for the characters in the string.
Typeface	getTypeface () Get the paint's typeface object.
Xfermode	getXfermode () Get the paint's xfermode object.
final boolean	isAntiAlias () Helper for getFlags(), returning true if ANTI_ALIAS_FLAG bit is set AntiAliasing smooths out the edges of what is being drawn, but is has no impact on the interior of the shape.
final boolean	isDither () Helper for getFlags(), returning true if DITHER_FLAG bit is set Dithering affects how colors that are higher precision than the device are down-sampled.
final boolean	isFakeBoldText () Helper for getFlags(), returning true if FAKE_BOLD_TEXT_FLAG bit is set
final boolean	isFilterBitmap () Whether or not the bitmap filter is activated.
final boolean	isLinearText () Helper for getFlags(), returning true if LINEAR_TEXT_FLAG bit is set
final boolean	isStrikeThruText () Helper for getFlags(), returning true if STRIKE_THRU_TEXT_FLAG bit is set
final boolean	isSubpixelText () Helper for getFlags(), returning true if SUBPIXEL_TEXT_FLAG bit is set
final boolean	isUnderlineText () Helper for getFlags(), returning true if UNDERLINE_TEXT_FLAG bit is set
float	measureText (String text) Return the width of the text.
float	measureText (CharSequence text, int start, int end) Return the width of the text.
float	measureText (String text, int start, int end) Return the width of the text.
float	measureText (char[] text, int index, int count) Return the width of the text.
void	reset () Restores the paint to its default settings.
void	set (Paint src) Copy the fields from src into this paint.
void	setARGB (int a, int r, int g, int b) Helper to setColor(), that takes a,r,g,b and constructs the color int
void	setAlpha (int a) Helper to setColor(), that only assigns the color's alpha value, leaving its r,g,b values unchanged.
void	setAntiAlias (boolean aa) Helper for setFlags(), setting or clearing the ANTI_ALIAS_FLAG bit AntiAliasing smooths out the edges of what is being drawn, but is has no impact on the interior of the shape.
void	setColor (int color) Set the paint's color.
ColorFilter	setColorFilter (ColorFilter filter) Set or clear the paint's colorfilter, returning the parameter.
void	setDither (boolean dither) Helper for setFlags(), setting or clearing the DITHER_FLAG bit Dithering affects how colors that are higher precision than the device are down-sampled.
void	setFakeBoldText (boolean fakeBoldText) Helper for setFlags(), setting or clearing the FAKE_BOLD_TEXT_FLAG bit
void	setFilterBitmap (boolean filter) Helper for setFlags(), setting or clearing the FILTER_BITMAP_FLAG bit.

void	setFlags (int flags) Set the paint's flags.
void	setHinting (int mode) Set the paint's hinting mode.
void	setLinearText (boolean linearText) Helper for setFlags(), setting or clearing the LINEAR_TEXT_FLAG bit
MaskFilter	setMaskFilter (MaskFilter maskfilter) Set or clear the maskfilter object.
PathEffect	setPathEffect (PathEffect effect) Set or clear the patheffect object.
Rasterizer	setRasterizer (Rasterizer rasterizer) Set or clear the rasterizer object.
Shader	setShader (Shader shader) Set or clear the shader object.
void	setShadowLayer (float radius, float dx, float dy, int color) This draws a shadow layer below the main layer, with the specified offset and color, and blur radius.
void	setStrikeThruText (boolean strikeThruText) Helper for setFlags(), setting or clearing the STRIKE_THRU_TEXT_FLAG bit
void	setStrokeCap (Paint.Cap cap) Set the paint's Cap.
void	setStrokeJoin (Paint.Join join) Set the paint's Join.
void	setStrokeMiter (float miter) Set the paint's stroke miter value.
void	setStrokeWidth (float width) Set the width for stroking.
void	setStyle (Paint.Style style) Set the paint's style, used for controlling how primitives' geometries are interpreted (except for drawBitmap, which always assumes Fill).
void	setSubpixelText (boolean subpixelText) Helper for setFlags(), setting or clearing the SUBPIXEL_TEXT_FLAG bit
void	setTextAlign (Paint.Align align) Set the paint's text alignment.
void	setTextLocale (Locale locale) Set the text locale.
void	setTextScaleX (float scaleX) Set the paint's horizontal scale factor for text.
void	setTextSize (float textSize) Set the paint's text size.
void	setTextSkewX (float skewX) Set the paint's horizontal skew factor for text.
Typeface	setTypeface (Typeface typeface) Set or clear the typeface object.
void	setUnderlineText (boolean underlineText) Helper for setFlags(), setting or clearing the UNDERLINE_TEXT_FLAG bit
Xfermode	setXfermode (Xfermode xfermode) Set or clear the xfermode object.

Protected Methods

void	finalize () Invoked when the garbage collector has detected that this instance is no longer reachable.

Inherited Methods [Expand]

▸ From class java.lang.Object

Constants

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

Paint flag that enables antialiasing when drawing.

Enabling this flag will cause all draw operations that support antialiasing to use it.

See Also
`Paint(int)`
`setFlags(int)`

Constant Value: 1 (0x00000001)

public static final int **DEV_KERN_TEXT_FLAG**

Added in API level 1

Legacy Paint flag, no longer used.

Constant Value: 256 (0x00000100)

public static final int **DITHER_FLAG**

Added in API level 1

Paint flag that enables dithering when blitting.

Enabling this flag applies a dither to any blit operation where the target's colour space is more constrained than the source.

See Also
`Paint(int)`
`setFlags(int)`

Constant Value: 4 (0x00000004)

public static final int **EMBEDDED_BITMAP_TEXT_FLAG**

Added in API level 19

Paint flag that enables the use of bitmap fonts when drawing text.

Disabling this flag will prevent text draw operations from using embedded bitmap strikes in fonts, causing fonts with both scalable outlines and bitmap strikes to draw only the scalable outlines, and fonts with only bitmap strikes to not draw at all.

See Also
`Paint(int)`
`setFlags(int)`

Constant Value: 1024 (0x00000400)

public static final int **FAKE_BOLD_TEXT_FLAG**

Added in API level 1

Paint flag that applies a synthetic bolding effect to drawn text.

Enabling this flag will cause text draw operations to apply a simulated bold effect when drawing a `Typeface (/reference/android/graphics/Typeface.html)` that is not already bold.

See Also
`Paint(int)`
`setFlags(int)`

Constant Value: 32 (0x00000020)

public static final int **FILTER_BITMAP_FLAG**

Added in API level 1

Paint flag that enables bilinear sampling on scaled bitmaps.

If cleared, scaled bitmaps will be drawn with nearest neighbor sampling, likely resulting in artifacts. This should generally be on when drawing bitmaps, unless performance-bound (rendering to software canvas) or preferring pixelation artifacts to blurriness when scaling significantly.

If bitmaps are scaled for device density at creation time (as resource bitmaps often are) the filtering will already have been done.

See Also
`Paint(int)`
`setFlags(int)`

Constant Value: 2 (0x00000002)

public static final int **HINTING_OFF**

Added in API level 14

Font hinter option that disables font hinting.

See Also
`setHinting(int)`

Constant Value: 0 (0x00000000)

public static final int **HINTING_ON**

Added in API level 14

Font hinter option that enables font hinting.

See Also

setHinting(int)

Constant Value: 1 (0x00000001)

public static final int **LINEAR_TEXT_FLAG**

Added in API level 1

Paint flag that enables smooth linear scaling of text.

Enabling this flag does not actually scale text, but rather adjusts text draw operations to deal gracefully with smooth adjustment of scale. When this flag is enabled, font hinting is disabled to prevent shape deformation between scale factors, and glyph caching is disabled due to the large number of glyph images that will be generated.

`SUBPIXEL_TEXT_FLAG` (/reference/android/graphics/Paint.html#SUBPIXEL_TEXT_FLAG) should be used in conjunction with this flag to prevent glyph positions from snapping to whole pixel values as scale factor is adjusted.

See Also

Paint(int)
setFlags(int)

Constant Value: 64 (0x00000040)

public static final int **STRIKE_THRU_TEXT_FLAG**

Added in API level 1

Paint flag that applies a strike-through decoration to drawn text.

See Also

Paint(int)
setFlags(int)

Constant Value: 16 (0x00000010)

public static final int **SUBPIXEL_TEXT_FLAG**

Added in API level 1

Paint flag that enables subpixel positioning of text.

Enabling this flag causes glyph advances to be computed with subpixel accuracy.

This can be used with `LINEAR_TEXT_FLAG` (/reference/android/graphics/Paint.html#LINEAR_TEXT_FLAG) to prevent text from jittering during smooth scale transitions.

See Also

Paint(int)
setFlags(int)

Constant Value: 128 (0x00000080)

public static final int **UNDERLINE_TEXT_FLAG**

Added in API level 1

Paint flag that applies an underline decoration to drawn text.

See Also

Paint(int)
setFlags(int)

Constant Value: 8 (0x00000008)

Public Constructors

public **Paint** ()

Added in API level 1

Create a new paint with default settings.

public **Paint** (int flags)

Added in API level 1

Create a new paint with the specified flags. Use setFlags() to change these after the paint is created.

Parameters

flags	initial flag bits, as if they were passed via setFlags().

public **Paint** (Paint paint) Added in API level 1

Create a new paint, initialized with the attributes in the specified paint parameter.

Parameters

paint	Existing paint used to initialized the attributes of the new paint.

Public Methods

public float **ascent** () Added in API level 1

Return the distance above (negative) the baseline (ascent) based on the current typeface and text size.

Returns

the distance above (negative) the baseline (ascent) based on the current typeface and text size.

public int **breakText** (CharSequence text, int start, int end, boolean measureForwards, float maxWidth, float[] measuredWidth) Added in API level 1

Measure the text, stopping early if the measured width exceeds maxWidth. Return the number of chars that were measured, and if measuredWidth is not null, return in it the actual width measured.

Parameters

text	The text to measure. Cannot be null.
start	The offset into text to begin measuring at
end	The end of the text slice to measure.
measureForwards	If true, measure forwards, starting at start. Otherwise, measure backwards, starting with end.
maxWidth	The maximum width to accumulate.
measuredWidth	Optional. If not null, returns the actual width measured.

Returns

The number of chars that were measured. Will always be <= abs(end - start).

public int **breakText** (String text, boolean measureForwards, float maxWidth, float[] measuredWidth) Added in API level 1

Measure the text, stopping early if the measured width exceeds maxWidth. Return the number of chars that were measured, and if measuredWidth is not null, return in it the actual width measured.

Parameters

text	The text to measure. Cannot be null.
measureForwards	If true, measure forwards, starting with the first character in the string. Otherwise, measure backwards, starting with the last character in the string.
maxWidth	The maximum width to accumulate.
measuredWidth	Optional. If not null, returns the actual width measured.

Returns

The number of chars that were measured. Will always be <= abs(count).

public int **breakText** (char[] text, int index, int count, float maxWidth, float[] measuredWidth) Added in API level 1

Measure the text, stopping early if the measured width exceeds maxWidth. Return the number of chars that were measured, and if measuredWidth is not null, return in it the actual width measured.

Parameters

text	The text to measure. Cannot be null.
index	The offset into text to begin measuring at

count	The number of maximum number of entries to measure. If count is negative, then the characters are measured in reverse order.
maxWidth	The maximum width to accumulate.
measuredWidth	Optional. If not null, returns the actual width measured.

Returns

The number of chars that were measured. Will always be <= abs(count).

public void **clearShadowLayer** ()

Added in API level 1

Clear the shadow layer.

public float **descent** ()

Added in API level 1

Return the distance below (positive) the baseline (descent) based on the current typeface and text size.

Returns

the distance below (positive) the baseline (descent) based on the current typeface and text size.

public int **getAlpha** ()

Added in API level 1

Helper to getColor() that just returns the color's alpha value. This is the same as calling getColor() >>> 24. It always returns a value between 0 (completely transparent) and 255 (completely opaque).

Returns

the alpha component of the paint's color.

public int **getColor** ()

Added in API level 1

Return the paint's color. Note that the color is a 32bit value containing alpha as well as r,g,b. This 32bit value is not premultiplied, meaning that its alpha can be any value, regardless of the values of r,g,b. See the Color class for more details.

Returns

the paint's color (and alpha).

public ColorFilter **getColorFilter** ()

Added in API level 1

Get the paint's colorfilter (maybe be null).

Returns

the paint's colorfilter (maybe be null)

public boolean **getFillPath** (Path src, Path dst)

Added in API level 1

Applies any/all effects (patheffect, stroking) to src, returning the result in dst. The result is that drawing src with this paint will be the same as drawing dst with a default paint (at least from the geometric perspective).

Parameters

src	input path
dst	output path (may be the same as src)

Returns

true if the path should be filled, or false if it should be drawn with a hairline (width == 0)

public int **getFlags** ()

Added in API level 1

Return the paint's flags. Use the Flag enum to test flag values.

Returns

the paint's flags (see enums ending in _Flag for bit masks)

public float **getFontMetrics** (Paint.FontMetrics metrics)

Added in API level 1

Return the font's recommended interline spacing, given the Paint's settings for typeface, textSize, etc. If metrics is not null, return the fontmetric values in it.

Parameters

metrics	If this object is not null, its fields are filled with the appropriate values given the paint's

text attributes.

Returns
the font's recommended interline spacing.

public Paint.FontMetrics **getFontMetrics** () Added in API level 1

Allocates a new FontMetrics object, and then calls getFontMetrics(fm) with it, returning the object.

public Paint.FontMetricsInt **getFontMetricsInt** () Added in API level 1

public int **getFontMetricsInt** (Paint.FontMetricsInt fmi) Added in API level 1

Return the font's interline spacing, given the Paint's settings for typeface, textSize, etc. If metrics is not null, return the fontmetric values in it. Note: all values have been converted to integers from floats, in such a way has to make the answers useful for both spacing and clipping. If you want more control over the rounding, call getFontMetrics().

Returns
the font's interline spacing.

public float **getFontSpacing** () Added in API level 1

Return the recommend line spacing based on the current typeface and text size.

Returns
recommend line spacing based on the current typeface and text size.

public int **getHinting** () Added in API level 14

Return the paint's hinting mode. Returns either `HINTING_OFF` `(/reference/android/graphics/Paint.html#HINTING_OFF)` or `HINTING_ON` `(/reference/android/graphics/Paint.html#HINTING_ON)`.

public MaskFilter **getMaskFilter** () Added in API level 1

Get the paint's maskfilter object.

Returns
the paint's maskfilter (or null)

public PathEffect **getPathEffect** () Added in API level 1

Get the paint's patheffect object.

Returns
the paint's patheffect (or null)

public Rasterizer **getRasterizer** () Added in API level 1

Get the paint's rasterizer (or null).

The raster controls/modifies how paths/text are turned into alpha masks.

Returns
the paint's rasterizer (or null)

public Shader **getShader** () Added in API level 1

Get the paint's shader object.

Returns
the paint's shader (or null)

public Paint.Cap **getStrokeCap** () Added in API level 1

Return the paint's Cap, controlling how the start and end of stroked lines and paths are treated.

Returns
the line cap style for the paint, used whenever the paint's style is Stroke or StrokeAndFill.

public Paint.Join **getStrokeJoin** ()

Added in API level 1

Return the paint's stroke join type.

Returns
the paint's Join.

public float **getStrokeMiter** ()

Added in API level 1

Return the paint's stroke miter value. Used to control the behavior of miter joins when the joins angle is sharp.

Returns
the paint's miter limit, used whenever the paint's style is Stroke or StrokeAndFill.

public float **getStrokeWidth** ()

Added in API level 1

Return the width for stroking.

A value of 0 strokes in hairline mode. Hairlines always draws a single pixel independent of the canva's matrix.

Returns
the paint's stroke width, used whenever the paint's style is Stroke or StrokeAndFill.

public Paint.Style **getStyle** ()

Added in API level 1

Return the paint's style, used for controlling how primitives' geometries are interpreted (except for drawBitmap, which always assumes FILL_STYLE).

Returns
the paint's style setting (Fill, Stroke, StrokeAndFill)

public Paint.Align **getTextAlign** ()

Added in API level 1

Return the paint's Align value for drawing text. This controls how the text is positioned relative to its origin. LEFT align means that all of the text will be drawn to the right of its origin (i.e. the origin specifieds the LEFT edge of the text) and so on.

Returns
the paint's Align value for drawing text.

public void **getTextBounds** (char[] text, int index, int count, Rect bounds)

Added in API level 1

Return in bounds (allocated by the caller) the smallest rectangle that encloses all of the characters, with an implied origin at (0,0).

Parameters

text	Array of chars to measure and return their unioned bounds
index	Index of the first char in the array to measure
count	The number of chars, beginning at index, to measure
bounds	Returns the unioned bounds of all the text. Must be allocated by the caller.

public void **getTextBounds** (String text, int start, int end, Rect bounds)

Added in API level 1

Return in bounds (allocated by the caller) the smallest rectangle that encloses all of the characters, with an implied origin at (0,0).

Parameters

text	String to measure and return its bounds
start	Index of the first char in the string to measure
end	1 past the last char in the string measure
bounds	Returns the unioned bounds of all the text. Must be allocated by the caller.

public Locale **getTextLocale** ()

Added in API level 17

Get the text Locale.

Returns

the paint's Locale used for drawing text, never null.

public void **getTextPath** (String text, int start, int end, float x, float y, Path path) Added in API level 1

Return the path (outline) for the specified text. Note: just like Canvas.drawText, this will respect the Align setting in the paint.

Parameters

text	The text to retrieve the path from
start	The first character in the text
end	1 past the last charcter in the text
x	The x coordinate of the text's origin
y	The y coordinate of the text's origin
path	The path to receive the data describing the text. Must be allocated by the caller.

public void **getTextPath** (char[] text, int index, int count, float x, float y, Path path) Added in API level 1

Return the path (outline) for the specified text. Note: just like Canvas.drawText, this will respect the Align setting in the paint.

Parameters

text	The text to retrieve the path from
index	The index of the first character in text
count	The number of characterss starting with index
x	The x coordinate of the text's origin
y	The y coordinate of the text's origin
path	The path to receive the data describing the text. Must be allocated by the caller.

public float **getTextScaleX** () Added in API level 1

Return the paint's horizontal scale factor for text. The default value is 1.0.

Returns
the paint's scale factor in X for drawing/measuring text

public float **getTextSize** () Added in API level 1

Return the paint's text size.

Returns
the paint's text size.

public float **getTextSkewX** () Added in API level 1

Return the paint's horizontal skew factor for text. The default value is 0.

Returns
the paint's skew factor in X for drawing text.

public int **getTextWidths** (String text, float[] widths) Added in API level 1

Return the advance widths for the characters in the string.

Parameters

text	The text to measure
widths	array to receive the advance widths of the characters. Must be at least a large as the text.

Returns
the number of unichars in the specified text.

public int **getTextWidths** (CharSequence text, int start, int end, float[] widths) Added in API level 1

Return the advance widths for the characters in the string.

Parameters

text	The text to measure. Cannot be null.

start	The index of the first char to to measure
end	The end of the text slice to measure
widths	array to receive the advance widths of the characters. Must be at least a large as (end - start).

Returns

the actual number of widths returned.

public int **getTextWidths** (String text, int start, int end, float[] widths)

Added in API level 1

Return the advance widths for the characters in the string.

Parameters

text	The text to measure. Cannot be null.
start	The index of the first char to to measure
end	The end of the text slice to measure
widths	array to receive the advance widths of the characters. Must be at least a large as the text.

Returns

the number of unichars in the specified text.

public int **getTextWidths** (char[] text, int index, int count, float[] widths)

Added in API level 1

Return the advance widths for the characters in the string.

Parameters

text	The text to measure. Cannot be null.
index	The index of the first char to to measure
count	The number of chars starting with index to measure
widths	array to receive the advance widths of the characters. Must be at least a large as count.

Returns

the actual number of widths returned.

public Typeface **getTypeface** ()

Added in API level 1

Get the paint's typeface object.

The typeface object identifies which font to use when drawing or measuring text.

Returns

the paint's typeface (or null)

public Xfermode **getXfermode** ()

Added in API level 1

Get the paint's xfermode object.

Returns

the paint's xfermode (or null)

public final boolean **isAntiAlias** ()

Added in API level 1

Helper for getFlags(), returning true if ANTI_ALIAS_FLAG bit is set AntiAliasing smooths out the edges of what is being drawn, but is has no impact on the interior of the shape. See setDither() and setFilterBitmap() to affect how colors are treated.

Returns

true if the antialias bit is set in the paint's flags.

public final boolean **isDither** ()

Added in API level 1

Helper for getFlags(), returning true if DITHER_FLAG bit is set Dithering affects how colors that are higher precision than the device are down-sampled. No dithering is generally faster, but higher precision colors are just truncated down (e.g. 8888 -> 565). Dithering tries to distribute the error inherent in this process, to reduce the visual artifacts.

Returns

true if the dithering bit is set in the paint's flags.

public final boolean **isFakeBoldText** ()

Added in API level 1

Helper for getFlags(), returning true if FAKE_BOLD_TEXT_FLAG bit is set

Returns

true if the fakeBoldText bit is set in the paint's flags.

public final boolean **isFilterBitmap** ()

Added in API level 1

Whether or not the bitmap filter is activated. Filtering affects the sampling of bitmaps when they are transformed. Filtering does not affect how the colors in the bitmap are converted into device pixels. That is dependent on dithering and xfermodes.

See Also

`setFilterBitmap()`

public final boolean **isLinearText** ()

Added in API level 1

Helper for getFlags(), returning true if LINEAR_TEXT_FLAG bit is set

Returns

true if the lineartext bit is set in the paint's flags

public final boolean **isStrikeThruText** ()

Added in API level 1

Helper for getFlags(), returning true if STRIKE_THRU_TEXT_FLAG bit is set

Returns

true if the strikeThruText bit is set in the paint's flags.

public final boolean **isSubpixelText** ()

Added in API level 1

Helper for getFlags(), returning true if SUBPIXEL_TEXT_FLAG bit is set

Returns

true if the subpixel bit is set in the paint's flags

public final boolean **isUnderlineText** ()

Added in API level 1

Helper for getFlags(), returning true if UNDERLINE_TEXT_FLAG bit is set

Returns

true if the underlineText bit is set in the paint's flags.

public float **measureText** (String text)

Added in API level 1

Return the width of the text.

Parameters

text	The text to measure. Cannot be null.

Returns

The width of the text

public float **measureText** (CharSequence text, int start, int end)

Added in API level 1

Return the width of the text.

Parameters

text	The text to measure
start	The index of the first character to start measuring
end	1 beyond the index of the last character to measure

Returns

The width of the text

public float **measureText** (String text, int start, int end)

Added in API level 1

Return the width of the text.

Parameters

text	The text to measure. Cannot be null.
start	The index of the first character to start measuring
end	1 beyond the index of the last character to measure

Returns

The width of the text

public float **measureText** (char[] text, int index, int count) Added in API level 1

Return the width of the text.

Parameters

text	The text to measure. Cannot be null.
index	The index of the first character to start measuring
count	THe number of characters to measure, beginning with start

Returns

The width of the text

public void **reset** () Added in API level 1

Restores the paint to its default settings.

public void **set** (Paint src) Added in API level 1

Copy the fields from src into this paint. This is equivalent to calling get() on all of the src fields, and calling the corresponding set() methods on this.

public void **setARGB** (int a, int r, int g, int b) Added in API level 1

Helper to setColor(), that takes a,r,g,b and constructs the color int

Parameters

a	The new alpha component (0..255) of the paint's color.
r	The new red component (0..255) of the paint's color.
g	The new green component (0..255) of the paint's color.
b	The new blue component (0..255) of the paint's color.

public void **setAlpha** (int a) Added in API level 1

Helper to setColor(), that only assigns the color's alpha value, leaving its r,g,b values unchanged. Results are undefined if the alpha value is outside of the range [0..255]

Parameters

a	set the alpha component [0..255] of the paint's color.

public void **setAntiAlias** (boolean aa) Added in API level 1

Helper for setFlags(), setting or clearing the ANTI_ALIAS_FLAG bit AntiAliasing smooths out the edges of what is being drawn, but is has no impact on the interior of the shape. See setDither() and setFilterBitmap() to affect how colors are treated.

Parameters

aa	true to set the antialias bit in the flags, false to clear it

public void **setColor** (int color) Added in API level 1

Set the paint's color. Note that the color is an int containing alpha as well as r,g,b. This 32bit value is not premultiplied, meaning that its alpha can be any value, regardless of the values of r,g,b. See the Color class for more details.

Parameters

color	The new color (including alpha) to set in the paint.

public ColorFilter **setColorFilter** (ColorFilter filter) Added in API level 1

Set or clear the paint's colorfilter, returning the parameter.

Parameters

filter	May be null. The new filter to be installed in the paint

Returns
filter

public void **setDither** (boolean dither)

Added in API level 1

Helper for setFlags(), setting or clearing the DITHER_FLAG bit Dithering affects how colors that are higher precision than the device are down-sampled. No dithering is generally faster, but higher precision colors are just truncated down (e.g. 8888 -> 565). Dithering tries to distribute the error inherent in this process, to reduce the visual artifacts.

Parameters

dither	true to set the dithering bit in flags, false to clear it

public void **setFakeBoldText** (boolean fakeBoldText)

Added in API level 1

Helper for setFlags(), setting or clearing the FAKE_BOLD_TEXT_FLAG bit

Parameters

fakeBoldText	true to set the fakeBoldText bit in the paint's flags, false to clear it.

public void **setFilterBitmap** (boolean filter)

Added in API level 1

Helper for setFlags(), setting or clearing the FILTER_BITMAP_FLAG bit. Filtering affects the sampling of bitmaps when they are transformed. Filtering does not affect how the colors in the bitmap are converted into device pixels. That is dependent on dithering and xfermodes.

Parameters

filter	true to set the FILTER_BITMAP_FLAG bit in the paint's flags, false to clear it.

public void **setFlags** (int flags)

Added in API level 1

Set the paint's flags. Use the Flag enum to specific flag values.

Parameters

flags	The new flag bits for the paint

public void **setHinting** (int mode)

Added in API level 14

Set the paint's hinting mode. May be either `HINTING_OFF` (/reference/android/graphics /Paint.html#HINTING_OFF) or `HINTING_ON` (/reference/android/graphics/Paint.html#HINTING_ON).

public void **setLinearText** (boolean linearText)

Added in API level 1

Helper for setFlags(), setting or clearing the LINEAR_TEXT_FLAG bit

Parameters

linearText	true to set the linearText bit in the paint's flags, false to clear it.

public MaskFilter **setMaskFilter** (MaskFilter maskfilter)

Added in API level 1

Set or clear the maskfilter object.

Pass null to clear any previous maskfilter. As a convenience, the parameter passed is also returned.

Parameters

maskfilter	May be null. The maskfilter to be installed in the paint

Returns
maskfilter

public PathEffect **setPathEffect** (PathEffect effect)

Added in API level 1

Set or clear the patheffect object.

Pass null to clear any previous patheffect. As a convenience, the parameter passed is also returned.

Parameters

effect	May be null. The patheffect to be installed in the paint

Returns
effect

public Rasterizer **setRasterizer** (Rasterizer rasterizer) Added in API level 1

Set or clear the rasterizer object.

Pass null to clear any previous rasterizer. As a convenience, the parameter passed is also returned.

Parameters

rasterizer	May be null. The new rasterizer to be installed in the paint.

Returns
rasterizer

public Shader **setShader** (Shader shader) Added in API level 1

Set or clear the shader object.

Pass null to clear any previous shader. As a convenience, the parameter passed is also returned.

Parameters

shader	May be null. the new shader to be installed in the paint

Returns
shader

public void **setShadowLayer** (float radius, float dx, float dy, int color) Added in API level 1

This draws a shadow layer below the main layer, with the specified offset and color, and blur radius. If radius is 0, then the shadow layer is removed.

public void **setStrikeThruText** (boolean strikeThruText) Added in API level 1

Helper for setFlags(), setting or clearing the STRIKE_THRU_TEXT_FLAG bit

Parameters

strikeThruText	true to set the strikeThruText bit in the paint's flags, false to clear it.

public void **setStrokeCap** (Paint.Cap cap) Added in API level 1

Set the paint's Cap.

Parameters

cap	set the paint's line cap style, used whenever the paint's style is Stroke or StrokeAndFill.

public void **setStrokeJoin** (Paint.Join join) Added in API level 1

Set the paint's Join.

Parameters

join	set the paint's Join, used whenever the paint's style is Stroke or StrokeAndFill.

public void **setStrokeMiter** (float miter) Added in API level 1

Set the paint's stroke miter value. This is used to control the behavior of miter joins when the joins angle is sharp. This value must be >= 0.

Parameters

miter	set the miter limit on the paint, used whenever the paint's style is Stroke or StrokeAndFill.

public void **setStrokeWidth** (float width) Added in API level 1

Set the width for stroking. Pass 0 to stroke in hairline mode. Hairlines always draws a single pixel independent of the canva's matrix.

Parameters

width set the paint's stroke width, used whenever the paint's style is Stroke or StrokeAndFill.

public void **setStyle** (Paint.Style style) Added in API level 1

Set the paint's style, used for controlling how primitives' geometries are interpreted (except for drawBitmap, which always assumes Fill).

Parameters

style The new style to set in the paint

public void **setSubpixelText** (boolean subpixelText) Added in API level 1

Helper for setFlags(), setting or clearing the SUBPIXEL_TEXT_FLAG bit

Parameters

subpixelText true to set the subpixelText bit in the paint's flags, false to clear it.

public void **setTextAlign** (Paint.Align align) Added in API level 1

Set the paint's text alignment. This controls how the text is positioned relative to its origin. LEFT align means that all of the text will be drawn to the right of its origin (i.e. the origin specifieds the LEFT edge of the text) and so on.

Parameters

align set the paint's Align value for drawing text.

public void **setTextLocale** (Locale locale) Added in API level 17

Set the text locale. The text locale affects how the text is drawn for some languages. For example, if the locale is `CHINESE` (/reference/java/util/Locale.html#CHINESE) or `CHINA` (/reference/java/util/Locale.html#CHINA), then the text renderer will prefer to draw text using a Chinese font. Likewise, if the locale is `JAPANESE` (/reference/java/util/Locale.html#JAPANESE) or `JAPAN` (/reference/java/util/Locale.html#JAPAN), then the text renderer will prefer to draw text using a Japanese font. This distinction is important because Chinese and Japanese text both use many of the same Unicode code points but their appearance is subtly different for each language. By default, the text locale is initialized to the system locale (as returned by `getDefault()` (/reference/java/util/Locale.html#getDefault())). This assumes that the text to be rendered will most likely be in the user's preferred language. If the actual language of the text is known, then it can be provided to the text renderer using this method. The text renderer may attempt to guess the language script based on the contents of the text to be drawn independent of the text locale here. Specifying the text locale just helps it do a better job in certain ambiguous cases

Parameters

locale the paint's locale value for drawing text, must not be null.

public void **setTextScaleX** (float scaleX) Added in API level 1

Set the paint's horizontal scale factor for text. The default value is 1.0. Values > 1.0 will stretch the text wider. Values < 1.0 will stretch the text narrower.

Parameters

scaleX set the paint's scale in X for drawing/measuring text.

public void **setTextSize** (float textSize) Added in API level 1

Set the paint's text size. This value must be > 0

Parameters

textSize set the paint's text size.

public void **setTextSkewX** (float skewX) Added in API level 1

Set the paint's horizontal skew factor for text. The default value is 0. For approximating oblique text, use values around -0.25.

Parameters

skewX set the paint's skew factor in X for drawing text.

public Typeface **setTypeface** (Typeface typeface)

Added in API level 1

Set or clear the typeface object.

Pass null to clear any previous typeface. As a convenience, the parameter passed is also returned.

Parameters

typeface	May be null. The typeface to be installed in the paint

Returns

typeface

public void **setUnderlineText** (boolean underlineText)

Added in API level 1

Helper for setFlags(), setting or clearing the UNDERLINE_TEXT_FLAG bit

Parameters

underlineText	true to set the underlineText bit in the paint's flags, false to clear it.

public Xfermode **setXfermode** (Xfermode xfermode)

Added in API level 1

Set or clear the xfermode object.

Pass null to clear any previous xfermode. As a convenience, the parameter passed is also returned.

Parameters

xfermode	May be null. The xfermode to be installed in the paint

Returns

xfermode

Protected Methods

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