CGPath Reference

Graphics & Animation: 2D Drawing

2010-08-03

Contents

CGPath Reference

Derived From:	*CFType Reference*
Framework:	ApplicationServices/ApplicationServices.h
Companion guide	Quartz 2D Programming Guide
Declared in	CGContext.h
	CGPath.h

Overview

A **graphics path** is a mathematical description of a series of shapes or lines. `CGPathRef` defines an opaque type that represents an immutable graphics path. `CGMutablePathRef` defines an opaque type that represents a mutable graphics path. Neither `CGPathRef` nor `CGMutablePathRef` define functions to draw a path. To draw a Quartz path to a graphics context, you add the path to the graphics context by calling `CGContextAddPath` and then call one of the context's drawing functions—see *CGContext Reference*.

Each figure in the graphics path is constructed with a connected set of lines and Bézier curves, called a **subpath**. A subpath has an ordered set of **path elements** that represent single steps in the construction of the subpath. (For example, a line segment from one corner of a rectangle to another corner is a path element. Every subpath includes a **starting point**, which is the first point in the subpath. The path also maintains a **current point**, which is the last point in the last subpath.

To append a new subpath onto a mutable path, your application typically calls `CGPathMoveToPoint` (page 22) to set the subpath's starting point and initial current point, followed by a series of `CGPathAdd*` calls to add line segments and curves to the subpath. As segments or curves are added to the subpath, the subpath's current point is updated to point to the end of the last segment or curve to be added. The lines and curves of a subpath are always connected, but they are not required to form a closed set of lines. Your application explicitly closes a subpath by calling `CGPathCloseSubpath` (page 16). Closing the subpath adds a line segment that terminates at the subpath's starting point, and also changes how those lines are rendered—for more information see "Paths" in *Quartz 2D Programming Guide*.

Functions by Task

Creating and Managing Paths

`CGPathCreateMutable` (page 17)
> Creates a mutable graphics path.

Modifying Quartz Paths

Getting Information about Quartz Paths

Functions

CGPathAddArc

Appends an arc to a mutable graphics path, possibly preceded by a straight line segment.

```
void CGPathAddArc (
    CGMutablePathRef path,
    const CGAffineTransform *m,
    CGFloat x,
    CGFloat y,
    CGFloat radius,
    CGFloat startAngle,
    CGFloat endAngle,
    bool clockwise
);
```

Parameters

path

The mutable graphics path to change.

m

A pointer to an affine transformation matrix, or `NULL` if no transformation is needed. If specified, Quartz applies the transformation to the arc before it is added to the path.

x

The x-coordinate of the center point of the arc.

y

The y-coordinate of the center point of the arc.

r

The radius of the arc.

startAngle

The angle (in radians) from the horizontal that determines the starting point of the arc.

endAngle

> The angle (in radians) from the horizontal that determines the ending point of the arc.

clockwise

> A Boolean value that specifies whether or not to draw the arc in the clockwise direction, before applying the transformation matrix.

Discussion

An arc is a segment of a circle with radius r centered at a point (x,y). When you call this function, you provide the center point, radius, and two angles in radians. Quartz uses this information to determine the end points of the arc, and then approximates the new arc using a sequence of cubic Bézier curves. The clockwise parameter determines the direction in which the arc is created. The actual direction may change depending on the coordinate system transformation applied to the path.

A transformation may be applied to the Bézier curves before they are added to the path. If no transform is needed, the second argument should be NULL.

If the specified path already contains a subpath, Quartz implicitly adds a line connecting the subpath's current point to the beginning of the arc. If the path is empty, Quartz creates a new subpath with a starting point set to the starting point of the arc.

The ending point of the arc becomes the new current point of the path.

Availability

Available in Mac OS X v10.2 and later.

Declared In

CGPath.h

CGPathAddArcToPoint

Appends an arc to a mutable graphics path, possibly preceded by a straight line segment.

```
void CGPathAddArcToPoint (
    CGMutablePathRef path,
    const CGAffineTransform *m,
    CGFloat x1,
    CGFloat y1,
    CGFloat x2,
    CGFloat y2,
    CGFloat radius
);
```

Parameters

path

> The mutable path to change. The path must not be empty.

m

> A pointer to an affine transformation matrix, or NULL if no transformation is needed. If specified, Quartz applies the transformation to the arc before it is added to the path.

x1

> The x-coordinate of the user space for the end point of the first tangent line. The first tangent line is drawn from the current point to (x1,y1).

y1

> The y-coordinate of the user space for the end point of the first tangent line. The first tangent line is drawn from the current point to `(x1,y1)`.

x2

> The x-coordinate of the user space for the end point of the second tangent line. The second tangent line is drawn from `(x1,y1)` to `(x2,y2)`.

y2

> The y-coordinate of the user space for the end point of the second tangent line. The second tangent line is drawn from `(x1,y1)` to `(x2,y2)`.

`radius`

> The radius of the arc, in user space coordinates.

Discussion

This function uses a sequence of cubic Bézier curves to create an arc that is tangent to the line from the current point to (x1,y1) and to the line from (x1,y1) to (x2,y2). The start and end points of the arc are located on the first and second tangent lines, respectively. The start and end points of the arc are also the "tangent points" of the lines.

If the current point and the first tangent point of the arc (the starting point) are not equal, Quartz appends a straight line segment from the current point to the first tangent point.

The ending point of the arc becomes the new current point of the path.

For another way to draw an arc in a path, see `CGPathAddArc` (page 7).

Availability

Available in Mac OS X v10.2 and later.

Declared In

`CGPath.h`

CGPathAddCurveToPoint

Appends a cubic Bézier curve to a mutable graphics path.

```
void CGPathAddCurveToPoint (
    CGMutablePathRef path,
    const CGAffineTransform *m,
    CGFloat cp1x,
    CGFloat cp1y,
    CGFloat cp2x,
    CGFloat cp2y,
    CGFloat x,
    CGFloat y
);
```

Parameters

path

> The mutable path to change. The path must not be empty.

m

> A pointer to an affine transformation matrix, or `NULL` if no transformation is needed. If specified, Quartz applies the transformation to the curve before it is added to the path.

cx1

> The x-coordinate of the first control point.

cy1

> The y-coordinate of the first control point.

cx2

> The x-coordinate of the second control point.

cy2

> The y-coordinate of the second control point.

x

> The x-coordinate of the end point of the curve.

y

> The y-coordinate of the end point of the curve.

Discussion

Appends a cubic Bézier curve from the current point in a path to the specified location using two control points, after an optional transformation. Before returning, this function updates the current point to the specified location (x, y).

Availability

Available in Mac OS X v10.2 and later.

Declared In

CGPath.h

CGPathAddEllipseInRect

Adds to a path an ellipse that fits inside a rectangle.

```
void CGPathAddEllipseInRect (
    CGMutablePathRef path,
    const CGAffineTransform *m,
    CGRect rect
);
```

Parameters

path

> The path to modify.

m

> An affine transform to apply to the ellipse, or NULL if you don't want to transform the ellipse.

rect

> A rectangle to enclose the ellipse.

Discussion

The ellipse is approximated by a sequence of Bézier curves. Its center is the midpoint of the rectangle defined by the rect parameter. If the rectangle is square, then the ellipse is circular with a radius equal to one-half the width (or height) of the rectangle. If the rect parameter specifies a rectangular shape, then the major and minor axes of the ellipse are defined by the width and height of the rectangle.

The ellipse forms a complete subpath of the path—that is, the ellipse drawing starts with a move-to operation and ends with a close-subpath operation, with all moves oriented in the clockwise direction. If you supply an affine transform, then the constructed Bézier curves that define the ellipse are transformed before they are added to the path.

Availability
Available in Mac OS X v10.4 and later.

Related Sample Code
Quartz 2D Shadings

Declared In
CGPath.h

CGPathAddLines

Appends an array of new line segments to a mutable graphics path.

```
void CGPathAddLines (
   CGMutablePathRef path,
   const CGAffineTransform *m,
   const CGPoint points[],
   size_t count
);
```

Parameters

path

> The mutable path to change.

m

> A pointer to an affine transformation matrix, or NULL if no transformation is needed. If specified, Quartz applies the transformation to the lines before adding them to the path.

points

> An array of points that specifies the line segments to add.

count

> The number of elements in the array.

Discussion
This is a convenience function that adds a sequence of connected line segments to a path, using the following operation:

```
CGPathMoveToPoint (path, m, points[0].x, points[0].y);
for (k = 1; k < count; k++) {
    CGPathAddLineToPoint (path, m, points[k].x, points[k].y);
}
```

Availability
Available in Mac OS X v10.2 and later.

Related Sample Code
GeekGameBoard

Declared In
CGPath.h

CGPathAddLineToPoint

Appends a line segment to a mutable graphics path.

```
void CGPathAddLineToPoint (
    CGMutablePathRef path,
    const CGAffineTransform *m,
    CGFloat x,
    CGFloat y
);
```

Parameters

path

      The mutable path to change. The path must not be empty.

m

      A pointer to an affine transformation matrix, or `NULL` if no transformation is needed. If specified, Quartz applies the transformation to the line before it is added to the path.

x

      The x-coordinate of the end point of the line.

y

      The y-coordinate of the end point of the line.

Discussion

Before returning, this function updates the current point to the specified location (x,y).

Availability

Available in Mac OS X v10.2 and later.

Related Sample Code

CALayerEssentials

Quartz 2D Shadings

Quartz2DBasics

Declared In

CGPath.h

CGPathAddPath

Appends a path to onto a mutable graphics path.

```
void CGPathAddPath (
    CGMutablePathRef path1,
    const CGAffineTransform *m,
    CGPathRef path2
);
```

Parameters

path1

      The mutable path to change.

m

      A pointer to an affine transformation matrix, or `NULL` if no transformation is needed. If specified, Quartz applies the transformation to `path2` before it is added to `path1`.

path2

      The path to add.

Discussion
If the source path is non-empty, then its path elements are appended in order onto the mutable path. After the call completes, the start point and current point of the path are those of the last subpath in *path2*.

Availability
Available in Mac OS X v10.2 and later.

Declared In
CGPath.h

CGPathAddQuadCurveToPoint

Appends a quadratic Bézier curve to a mutable graphics path.

```
void CGPathAddQuadCurveToPoint (
    CGMutablePathRef path,
    const CGAffineTransform *m,
    CGFloat cpx,
    CGFloat cpy,
    CGFloat x,
    CGFloat y
);
```

Parameters

path

The mutable path to change. The path must not be empty.

m

A pointer to an affine transformation matrix, or NULL if no transformation is needed. If specified, Quartz applies the transformation to the curve before adding it to the path.

cx

The x-coordinate of the control point.

cy

The y-coordinate of the control point.

x

The x-coordinate of the end point of the curve.

y

The y-coordinate of the end point of the curve.

Discussion
Before returning, this function updates the current point to the specified location (x, y).

Availability
Available in Mac OS X v10.2 and later.

Declared In
CGPath.h

CGPathAddRect

Appends a rectangle to a mutable graphics path.

```
void CGPathAddRect (
    CGMutablePathRef path,
    const CGAffineTransform *m,
    CGRect rect
);
```

Parameters

path

> The mutable path to change.

m

> A pointer to an affine transformation matrix, or `NULL` if no transformation is needed. If specified, Quartz applies the transformation to the rectangle before adding it to the path.

rect

> The rectangle to add.

Discussion

This is a convenience function that adds a rectangle to a path, using the following sequence of operations:

```
// start at origin
CGPathMoveToPoint (path, m, CGRectGetMinX(rect), CGRectGetMinY(rect));

// add bottom edge
CGPathAddLineToPoint (path, m, CGRectGetMaxX(rect), CGRectGetMinY(rect));

// add right edge
CGPathAddLineToPoint (path, m, CGRectGetMaxX(rect), CGRectGetMaxY(rect);

// add top edge
CGPathAddLineToPoint (path, m, CGRectGetMinX(rect), CGRectGetMaxY(rect));

// add left edge and close
CGPathCloseSubpath (path);
```

Availability

Available in Mac OS X v10.2 and later.

Related Sample Code

CoreTextRTF

CoreTextTest

Quartz 2D Shadings

Declared In

CGPath.h

CGPathAddRects

Appends an array of rectangles to a mutable graphics path.

```
void CGPathAddRects (
    CGMutablePathRef path,
    const CGAffineTransform *m,
    const CGRect rects[],
    size_t count
);
```

Parameters

path

The mutable path to change.

m

An affine transformation matrix, or `NULL` if no transformation is needed. If specified, Quartz applies the transformation to the rectangles before adding them to the path.

rects

The array of new rectangles to add.

count

The number of elements in the array.

Discussion

This is a convenience function that adds an array of rectangles to a path, using the following operation:

```
for (k = 0; k < count; k++) {
    CGPathAddRect (path, m, rects[k]);
}
```

Availability

Available in Mac OS X v10.2 and later.

Declared In

CGPath.h

CGPathApply

For each element in a graphics path, calls a custom applier function.

```
void CGPathApply (
    CGPathRef path,
    void *info,
    CGPathApplierFunction function
);
```

Parameters

path

The path to which the function will be applied.

info

A pointer to the user data that Quartz will pass to the function being applied, or `NULL`.

function

A pointer to the function to apply. See `CGPathApplierFunction` (page 23) for more information.

Discussion

For each element in the specified path, Quartz calls the applier function, which can examine (but not modify) the element.

Availability

Available in Mac OS X v10.2 and later.

Declared In

`CGPath.h`

CGPathCloseSubpath

Closes and completes a subpath in a mutable graphics path.

```
void CGPathCloseSubpath (
    CGMutablePathRef path
);
```

Parameters

path

> The path to change.

Discussion

Appends a line from the current point to the starting point of the current subpath and ends the subpath.

After closing the subpath, your application can begin a new subpath without first calling `CGPathMoveToPoint` (page 22). In this case, a new subpath is implicitly created with a starting and current point equal to the previous subpath's starting point.

Availability

Available in Mac OS X v10.2 and later.

Related Sample Code

CALayerEssentials

GeekGameBoard

Quartz 2D Shadings

Quartz2DBasics

Declared In

`CGPath.h`

CGPathContainsPoint

Checks whether a point is contained in a graphics path.

```
bool CGPathContainsPoint (
    CGPathRef path,
    const CGAffineTransform *m,
    CGPoint point,
    bool eoFill
);
```

Parameters

path

> The path to evaluate the point against.

m

An affine transform. If `m` is not `NULL` then the point is transformed by this affine transform prior to determining whether the path contains the point.

point

The point to check.

eoFill

A Boolean value that, if `true`, specifies to use the even-odd fill rule to evaluate the painted region of the path. If `false`, the winding fill rule is used.

Return Value

Returns `true` if the point is contained in the path; `false` otherwise.

Discussion

A point is contained in a path if it would be inside the painted region when the path is filled.

Availability

Available in Mac OS X v10.4 and later.

Related Sample Code

GeekGameBoard

Declared In

CGPath.h

CGPathCreateCopy

Creates an immutable copy of a graphics path.

```
CGPathRef CGPathCreateCopy (
    CGPathRef path
);
```

Parameters

path

The path to copy.

Return Value

A new, immutable copy of the specified path. You are responsible for releasing this object.

Availability

Available in Mac OS X v10.2 and later.

Declared In

CGPath.h

CGPathCreateMutable

Creates a mutable graphics path.

```
CGMutablePathRef CGPathCreateMutable (
    void
);
```

Return Value

A new mutable path. You are responsible for releasing this object.

Availability

Available in Mac OS X v10.2 and later.

Related Sample Code

CALayerEssentials

CoreTextRTF

CoreTextTest

Quartz 2D Shadings

Quartz2DBasics

Declared In

CGPath.h

CGPathCreateMutableCopy

Creates a mutable copy of an existing graphics path.

```
CGMutablePathRef CGPathCreateMutableCopy (
    CGPathRef path
);
```

Parameters

path

> The path to copy.

Return Value

A new, mutable, copy of the specified path. You are responsible for releasing this object.

Discussion

You can modify a mutable graphics path by calling the various CGPath geometry functions, such as CGPathAddArc (page 7), CGPathAddLineToPoint (page 11), and CGPathMoveToPoint (page 22).

Availability

Available in Mac OS X v10.2 and later.

Declared In

CGPath.h

CGPathEqualToPath

Indicates whether two graphics paths are equivalent.

```
bool CGPathEqualToPath (
    CGPathRef path1,
    CGPathRef path2
);
```

Parameters

path1

The first path being compared.

path2

The second path being compared.

Return Value

A Boolean value that indicates whether or not the two specified paths contain the same sequence of path elements. If the paths are not the same, returns `false`.

Availability

Available in Mac OS X v10.2 and later.

Declared In

`CGPath.h`

CGPathGetBoundingBox

Returns the bounding box containing all points in a graphics path.

```
CGRect CGPathGetBoundingBox (
    CGPathRef path
);
```

Parameters

path

The graphics path to evaluate.

Return Value

A rectangle that represents the bounding box of the specified path. If the path is empty, this function returns `CGRectNull`.

Discussion

The bounding box is the smallest rectangle completely enclosing all points in the path, including control points for Bézier and quadratic curves.

Availability

Available in Mac OS X v10.2 and later.

See Also

Declared In

`CGPath.h`

CGPathGetCurrentPoint

Returns the current point in a graphics path.

```
CGPoint CGPathGetCurrentPoint (
    CGPathRef path
);
```

Parameters

path

> The path to evaluate.

Return Value

The current point in the specified path.

Discussion

If the path is empty—that is, if it has no elements—this function returns `CGPointZero` (see *CGGeometry Reference*). To determine whether a path is empty, use `CGPathIsEmpty` (page 21).

Availability

Available in Mac OS X v10.2 and later.

Declared In

`CGPath.h`

CGPathGetPathBoundingBox

Returns the bounding box of a graphics path.

```
CGRect CGPathGetPathBoundingBox (
    CGPathRef path
);
```

Parameters

path

> The graphics path to evaluate.

Return Value

A rectangle that represents the path bounding box of the specified path. If the path is empty, this function returns `CGRectNull`.

Discussion

The path bounding box is the smallest rectangle completely enclosing all points in the path but not including control points for Bézier and quadratic curves.

Availability

Available in Mac OS X v10.6 and later.

See Also

`CGPathGetBoundingBox` (page 19)

Declared In

`CGPath.h`

CGPathGetTypeID

Returns the Core Foundation type identifier for Quartz graphics paths.

```
CFTypeID CGPathGetTypeID (
    void
);
```

Return Value
The Core Foundation identifier for the opaque type CGPathRef (page 24).

Availability
Available in Mac OS X v10.2 and later.

Declared In
CGPath.h

CGPathIsEmpty

Indicates whether or not a graphics path is empty.

```
bool CGPathIsEmpty (
    CGPathRef path
);
```

Parameters

path

> The path to evaluate.

Return Value
A Boolean value that indicates whether the specified path is empty.

Discussion
An empty path contains no elements.

Availability
Available in Mac OS X v10.2 and later.

Declared In
CGPath.h

CGPathIsRect

Indicates whether or not a graphics path represents a rectangle.

```
bool CGPathIsRect (
    CGPathRef path,
    CGRect *rect
);
```

Parameters

path

> The path to evaluate.

rect

> On input, a pointer to an uninitialized rectangle. If the specified path represents a rectangle, on return contains a copy of the rectangle.

Return Value
A Boolean value that indicates whether the specified path represents a rectangle. If the path represents a rectangle, returns `true`.

Availability
Available in Mac OS X v10.2 and later.

Declared In
`CGPath.h`

CGPathMoveToPoint

Starts a new subpath at a specified location in a mutable graphics path.

```
void CGPathMoveToPoint (
    CGMutablePathRef path,
    const CGAffineTransform *m,
    CGFloat x,
    CGFloat y
);
```

Parameters
path

> The mutable path to change.

m

> A pointer to an affine transformation matrix, or `NULL` if no transformation is needed. If specified, Quartz applies the transformation to the point before changing the path.

x

> The x-coordinate of the new location.

y

> The y-coordinate of the new location.

Discussion
This function ends the subpath already in progress (if any) and starts a new subpath, initializing the starting point and the current point to the specified location (x,y) after an optional transformation.

Availability
Available in Mac OS X v10.2 and later.

Related Sample Code
CALayerEssentials

Quartz 2D Shadings

Quartz2DBasics

Declared In
`CGPath.h`

CGPathRelease

Decrements the retain count of a graphics path.

```
void CGPathRelease (
    CGPathRef path
);
```

Parameters

path

The graphics path to release.

Discussion

This function is equivalent to `CFRelease`, except that it does not cause an error if the `path` parameter is `NULL`.

Availability

Available in Mac OS X v10.2 and later.

Related Sample Code

CALayerEssentials

Quartz2DBasics

Declared In

`CGPath.h`

CGPathRetain

Increments the retain count of a graphics path.

```
CGPathRef CGPathRetain (
    CGPathRef path
);
```

Parameters

path

The graphics path to retain.

Return Value

The same path you passed in as the `path` parameter.

Discussion

This function is equivalent to `CFRetain`, except that it does not cause an error if the `path` parameter is `NULL`.

Availability

Available in Mac OS X v10.2 and later.

Declared In

`CGPath.h`

Callbacks

CGPathApplierFunction

Defines a callback function that can view an element in a graphics path.

```
typedef void (*CGPathApplierFunction) (
    void *info,
    const CGPathElement *element
);
```

If you name your function `MyCGPathApplierFunc`, you would declare it like this:

```
void MyCGPathApplierFunc (
    void *info,
  const CGPathElement *element
);
```

Discussion
See also `CGPathApply` (page 15).

Availability
Available in Mac OS X v10.2 and later.

Declared In
CGPath.h

Data Types

CGPathRef

An opaque type that represents an immutable graphics path.

```
typedef const struct CGPath *CGPathRef;
```

Availability
Available in Mac OS X v10.2 and later.

Declared In
CGPath.h

CGMutablePathRef

An opaque type that represents a mutable graphics path.

```
typedef struct CGPath *CGMutablePathRef;
```

Availability
Available in Mac OS X v10.2 and later.

Declared In
CGPath.h

CGPathElement

A data structure that provides information about a path element.

```
struct CGPathElement {
    CGPathElementType type;
    CGPoint * points;
};
typedef struct CGPathElement CGPathElement;
```

Fields

`type`

>   An element type (or operation).

`points`

>   An array of one or more points that serve as arguments.

Availability
Available in Mac OS X v10.2 and later.

Declared In
`CGPath.h`

Constants

CGPathDrawingMode

Options for rendering a path.

```
enum CGPathDrawingMode {
    kCGPathFill,
    kCGPathEOFill,
    kCGPathStroke,
    kCGPathFillStroke,
    kCGPathEOFillStroke
};
typedef enum CGPathDrawingMode CGPathDrawingMode;
```

Constants

`kCGPathFill`

>   Render the area contained within the path using the non-zero winding number rule.
>
>   Available in Mac OS X v10.0 and later.
>
>   Declared in `CGContext.h`.

`kCGPathEOFill`

>   Render the area within the path using the even-odd rule.
>
>   Available in Mac OS X v10.0 and later.
>
>   Declared in `CGContext.h`.

`kCGPathStroke`

>   Render a line along the path.
>
>   Available in Mac OS X v10.0 and later.
>
>   Declared in `CGContext.h`.

```
kCGPathFillStroke
```
      First fill and then stroke the path, using the nonzero winding number rule.

      Available in Mac OS X v10.0 and later.

      Declared in `CGContext.h`.

```
kCGPathEOFillStroke
```
      First fill and then stroke the path, using the even-odd rule.

      Available in Mac OS X v10.0 and later.

      Declared in `CGContext.h`.

Discussion

You can pass a path drawing mode constant to the function `CGContextDrawPath` to specify how Quartz should paint a graphics context's current path.

CGPathElementType

The type of element found in a path.

```
enum CGPathElementType {
    kCGPathElementMoveToPoint,
    kCGPathElementAddLineToPoint,
    kCGPathElementAddQuadCurveToPoint,
    kCGPathElementAddCurveToPoint,
    kCGPathElementCloseSubpath
};
typedef enum CGPathElementType CGPathElementType;
```

Constants

```
kCGPathElementMoveToPoint
```
      The path element that starts a new subpath. See the function `CGPathMoveToPoint` (page 22).

      Available in Mac OS X v10.2 and later.

      Declared in `CGPath.h`.

```
kCGPathElementAddLineToPoint
```
      The path element that adds a line from the current point to the specified point. See the function `CGPathAddLineToPoint` (page 11).

      Available in Mac OS X v10.2 and later.

      Declared in `CGPath.h`.

```
kCGPathElementAddQuadCurveToPoint
```
      The path element that adds a quadratic curve from the current point to the specified point. See the function `CGPathAddQuadCurveToPoint` (page 13).

      Available in Mac OS X v10.2 and later.

      Declared in `CGPath.h`.

```
kCGPathElementAddCurveToPoint
```
      The path element that adds a cubic curve from the current point to the specified point. See the function `CGPathAddCurveToPoint` (page 9).

      Available in Mac OS X v10.2 and later.

      Declared in `CGPath.h`.

`kCGPathElementCloseSubpath`

The path element that closes and completes a subpath. See the function `CGPathCloseSubpath` (page 16).

Available in Mac OS X v10.2 and later.

Declared in `CGPath.h`.

Discussion

For more information about paths, see `CGPathRef` (page 24).

Document Revision History

This table describes the changes to *CGPath Reference*.

Date	Notes
2010-08-03	Updated CGPathGetPathBoundingBox.
2010-07-01	Clarified many function descriptions. Expanded the introduction to provide better definitions of common path terminology.
2010-04-26	Added the CGPathGetPathBoundingBox function.
2009-05-26	Updated the CGPathAddArc function description to account for iPhone coordinate systems.
2006-12-22	Updated for Mac OS X v10.5.
	All instances of the `float` data type were changed to the `CGFloat` data type.
2006-01-10	Added clarification on using the function CGPathContainsPoint.
2005-04-29	Updated for Mac OS X v10.4.
	Added the functions `CGPathAddEllipseInRect` (page 10) and `CGPathContainsPoint` (page 16).
2004-08-31	Added introductory material.
2004-02-26	First version of this document. An earlier version of this information appeared in *Quartz 2D Reference*.