

---

# CGDataProvider Reference

Graphics & Animation: 2D Drawing



2009-01-06



Apple Inc.  
© 2003, 2009 Apple Inc.  
All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, electronic, photocopying, recording, or otherwise, without prior written permission of Apple Inc., with the following exceptions: Any person is hereby authorized to store documentation on a single computer for personal use only and to print copies of documentation for personal use provided that the documentation contains Apple's copyright notice.

The Apple logo is a trademark of Apple Inc.

Use of the "keyboard" Apple logo (Option-Shift-K) for commercial purposes without the prior written consent of Apple may constitute trademark infringement and unfair competition in violation of federal and state laws.

No licenses, express or implied, are granted with respect to any of the technology described in this document. Apple retains all intellectual property rights associated with the technology described in this document. This document is intended to assist application developers to develop applications only for Apple-labeled computers.

Every effort has been made to ensure that the information in this document is accurate. Apple is not responsible for typographical errors.

Apple Inc.  
1 Infinite Loop  
Cupertino, CA 95014  
408-996-1010

Apple, the Apple logo, Mac, Mac OS, and Quartz are trademarks of Apple Inc., registered in the United States and other countries.

Simultaneously published in the United States and Canada.

**Even though Apple has reviewed this document, APPLE MAKES NO WARRANTY OR REPRESENTATION, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT, ITS QUALITY, ACCURACY, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. AS A RESULT, THIS DOCUMENT IS PROVIDED "AS IS," AND YOU, THE READER, ARE ASSUMING THE ENTIRE RISK AS TO ITS QUALITY AND ACCURACY.**

**IN NO EVENT WILL APPLE BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY**

**DEFECT OR INACCURACY IN THIS DOCUMENT, even if advised of the possibility of such damages.**

**THE WARRANTY AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. No Apple dealer, agent, or employee is authorized to make any modification, extension, or addition to this warranty.**

**Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.**

# Contents

## **CGDataProvider Reference 5**

---

Overview	5
Functions	5
CGDataProviderCopyData	5
CGDataProviderCreateDirect	6
CGDataProviderCreateSequential	6
CGDataProviderCreateWithCFData	7
CGDataProviderCreateWithData	7
CGDataProviderCreateWithFilename	8
CGDataProviderCreateWithURL	9
CGDataProviderGetTypeID	9
CGDataProviderRelease	9
CGDataProviderRetain	10
Callbacks by Task	11
Sequential-Access Data Provider Callbacks	11
Direct-Access Data Provider Callbacks	11
Callbacks	11
CGDataProviderGetBytePointerCallback	11
CGDataProviderGetBytesAtOffsetCallback	12
CGDataProviderGetBytesAtPositionCallback	13
CGDataProviderGetBytesCallback	14
CGDataProviderReleaseBytePointerCallback	15
CGDataProviderReleaseDataCallback	15
CGDataProviderReleaseInfoCallback	16
CGDataProviderRewindCallback	17
CGDataProviderSkipBytesCallback	17
CGDataProviderSkipForwardCallback	18
Data Types	19
CGDataProviderRef	19
CGDataProviderCallbacks	19
CGDataProviderDirectAccessCallbacks	20
CGDataProviderDirectCallbacks	21
CGDataProviderSequentialCallbacks	22

## **Appendix A      Deprecated CGDataProvider Functions 25**

---

Deprecated in Mac OS X v10.5	25
CGDataProviderCreate	25
CGDataProviderCreateDirectAccess	25

**Document Revision History 27**

---

**Index 29**

---

# CGDataProvider Reference

---

<b>Derived From:</b>	<i>CType Reference</i>
<b>Framework:</b>	ApplicationServices/ApplicationServices.h
<b>Declared in</b>	CGDataProvider.h

## Overview

The CGDataProvider header file declares a data type that supplies Quartz functions with data. Data provider objects abstract the data-access task and eliminate the need for applications to manage data through a raw memory buffer.

For information on how to use CGDataProvider functions, see *Quartz 2D Programming Guide Programming Guide*.

See also *CGDataConsumer Reference*.

## Functions

### **CGDataProviderCopyData**

Returns a copy of the provider's data.

```
CFDataRef CGDataProviderCopyData(  
    CGDataProviderRef provider  
);
```

#### **Parameters**

*provider*

The data provider whose data you want to copy.

#### **Return Value**

A new data object containing a copy of the provider's data. You are responsible for releasing this object.

#### **Availability**

Available in Mac OS X v10.5 and later.

#### **Declared In**

CGDataProvider.h

## CGDataProviderCreateDirect

Creates a Quartz direct-access data provider.

```
CGDataProviderRef CGDataProviderCreateDirect (
    void *info,
    off_t size,
    const CGDataProviderDirectCallbacks *callbacks
);
```

### Parameters

*info*

A pointer to data of any type or NULL. When Quartz calls the functions specified in the `callbacks` parameter, it sends each of the functions this pointer.

*size*

The number of bytes of data to provide.

*callbacks*

A pointer to a `CGDataProviderDirectCallbacks` structure that specifies the callback functions you implement to handle the data provider's basic memory management.

### Return Value

A new data provider. You are responsible for releasing this object using [CGDataProviderRelease](#) (page 9).

### Discussion

You use this function to create a direct-access data provider that uses callback functions to read data from your program in a single block.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

`CGDataProvider.h`

## CGDataProviderCreateSequential

Creates a Quartz sequential-access data provider.

```
CGDataProviderRef CGDataProviderCreateSequential (
    void *info,
    const CGDataProviderSequentialCallbacks *callbacks
);
```

### Parameters

*info*

A pointer to data of any type or NULL. When Quartz calls the functions specified in the `callbacks` parameter, it sends each of the functions this pointer.

*callbacks*

A pointer to a `CGDataProviderSequentialCallbacks` structure that specifies the callback functions you implement to handle the data provider's basic memory management.

### Return Value

A new data provider. You are responsible for releasing this object using [CGDataProviderRelease](#) (page 9).

**Discussion**

You use this function to create a sequential-access data provider that uses callback functions to read data from your program in a single block.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

CGDataProvider.h

**CGDataProviderCreateWithCFData**

Creates a Quartz data provider that reads from a CFData object.

```
CGDataProviderRef CGDataProviderCreateWithCFData (
    CFDataRef data
);
```

**Parameters**

*data*

The CFData object to read from.

**Return Value**

A new data provider. You are responsible for releasing this object using [CGDataProviderRelease](#) (page 9).

**Discussion**

You can use this function when you need to represent Quartz data as a CFData type. For example, you might create a CFData object when reading data from the pasteboard.

**Availability**

Available in Mac OS X v10.4 and later.

**Declared In**

CGDataProvider.h

**CGDataProviderCreateWithData**

Creates a Quartz direct-access data provider that uses data your program supplies.

```
CGDataProviderRef CGDataProviderCreateWithData (
    void *info,
    const void *data,
    size_t size,
    CGDataProviderReleaseDataCallback releaseData
);
```

**Parameters**

*info*

A pointer to data of any type, or NULL. When Quartz calls the function specified in the `releaseData` parameter, Quartz sends it this pointer as its first argument.

*data*

A pointer to the array of data that the provider contains.

*size*

A value that specifies the number of bytes that the data provider contains.

*releaseData*

A pointer to a release callback for the data provider, or `NULL`. Your release function is called when Quartz frees the data provider. For more information, see [CGDataProviderReleaseDataCallback](#) (page 15).

#### **Return Value**

A new data provider. You are responsible for releasing this object using [CGDataProviderRelease](#) (page 9).

#### **Discussion**

You use this function to create a direct-access data provider that uses callback functions to read data from your program an entire block at one time.

#### **Availability**

Available in Mac OS X v10.0 and later.

#### **Related Sample Code**

CIVideoDemoGL

GLSL Showpiece Lite

MovieVideoChart

Quartz EB

SampleRaster

#### **Declared In**

`CGDataProvider.h`

## **CGDataProviderCreateWithFilename**

Creates a Quartz direct-access data provider that uses a file to supply data.

```
CGDataProviderRef CGDataProviderCreateWithFilename(
    const char *filename
);
```

#### **Parameters**

*filename*

The full or relative pathname to use for the data provider. When you supply Quartz data via the provider, it reads the data from the specified file.

#### **Return Value**

A new data provider or `NULL` if the file could not be opened. You are responsible for releasing this object using [CGDataProviderRelease](#) (page 9).

#### **Discussion**

You use this function to create a direct-access data provider that supplies data from a file. When you supply Quartz with a direct-access data provider, Quartz obtains data from your program in a single block.

#### **Availability**

Available in Mac OS X v10.0 and later.

#### **Related Sample Code**

Fireworks

**Declared In**

CGDataProvider.h

**CGDataProviderCreateWithURL**

Creates a Quartz direct-access data provider that uses a URL to supply data.

```
CGDataProviderRef CGDataProviderCreateWithURL (
    CFURLRef url
);
```

**Parameters***url*

A CFURL object to use for the data provider. When you supply Quartz data via the provider, it reads the data from the URL address.

**Return Value**

A new data provider or NULL if the data from the URL could not be accessed. You are responsible for releasing this object using [CGDataProviderRelease](#) (page 9).

**Discussion**

You use this function to create a direct-access data provider that supplies data from a URL. When you supply Quartz with a direct-access data provider, Quartz obtains data from your program in a single entire block.

**Availability**

Available in Mac OS X v10.0 and later.

**Related Sample Code**

DockBrowser

**Declared In**

CGDataProvider.h

**CGDataProviderGetTypeID**

Returns the Core Foundation type identifier for Quartz data providers.

```
CFTypeID CGDataProviderGetTypeID (
    void
);
```

**Return Value**

The identifier for the opaque type [CGDataProviderRef](#) (page 19).

**Availability**

Available in Mac OS X v10.2 and later.

**Declared In**

CGDataProvider.h

**CGDataProviderRelease**

Decrements the retain count of a data provider.

```
void CGDataProviderRelease (
    CGDataProviderRef provider
);
```

**Parameters**

*provider*

The data provider to release.

**Discussion**

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

**Availability**

Available in Mac OS X v10.0 and later.

**Related Sample Code**

DockBrowser

GLSL Showpiece Lite

MovieVideoChart

Quartz EB

SampleRaster

**Declared In**

CGDataProvider.h

**CGDataProviderRetain**

Increments the retain count of a data provider.

```
CGDataProviderRef CGDataProviderRetain (
    CGDataProviderRef provider
);
```

**Parameters**

*provider*

The data provider to retain.

**Return Value**

The same data provider you passed in as the `provider` parameter.

**Discussion**

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

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

CGDataProvider.h

## Callbacks by Task

### Sequential-Access Data Provider Callbacks

[CGDataProviderGetBytesCallback](#) (page 14)

A callback function that copies from a provider data stream into a Quartz-supplied buffer.

[CGDataProviderReleaseInfoCallback](#) (page 16)

A callback function that releases any private data or resources associated with the data provider.

[CGDataProviderRewindCallback](#) (page 17)

A callback function that moves the current position in the data stream back to the beginning.

[CGDataProviderSkipBytesCallback](#) (page 17)

A callback function that advances the current position in the data stream supplied by the provider.

[CGDataProviderSkipForwardCallback](#) (page 18)

A callback function that advances the current position in the data stream supplied by the provider.

### Direct-Access Data Provider Callbacks

[CGDataProviderGetBytePointerCallback](#) (page 11)

A callback function that returns a generic pointer to the provider data.

[CGDataProviderGetBytesAtOffsetCallback](#) (page 12)

A callback function that copies data from the provider into a Quartz buffer.

[CGDataProviderReleaseBytePointerCallback](#) (page 15)

A callback function that releases the pointer Quartz obtained by calling [CGDataProviderGetBytePointerCallback](#) (page 11).

[CGDataProviderReleaseDataCallback](#) (page 15)

A callback function that releases data you supply to the function [CGDataProviderCreateWithData](#) (page 7).

[CGDataProviderGetBytesAtPositionCallback](#) (page 13)

A callback function that copies data from the provider into a Quartz buffer.

## Callbacks

### CGDataProviderGetBytePointerCallback

A callback function that returns a generic pointer to the provider data.

```
const void * (*CGDataProviderGetBytePointerCallback) (
    void *info
);
```

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

```
void *MyProviderGetBytePointer (
```

```
void *info
);
```

### Parameters

*info*

A generic pointer to private data shared among your callback functions. This is the same pointer you supplied to [CGDataProviderCreateDirectAccess](#) (page 25).

### Return Value

A generic pointer to your provider data. By supplying this pointer, you are giving Quartz read-only access to both the pointer and the underlying provider data. You must not move or modify the provider data until Quartz calls your [CGDataProviderReleaseBytePointerCallback](#) (page 15) function.

### Discussion

When Quartz needs direct access to your provider data, this function is called.

For information on how to associate your function with a direct-access data provider, see [CGDataProviderCreateDirectAccess](#) (page 25) and [CGDataProviderDirectAccessCallbacks](#) (page 20).

### Availability

Available in Mac OS X v10.3 and later.

### Declared In

CGDataProvider.h

## CGDataProviderGetBytesAtOffsetCallback

A callback function that copies data from the provider into a Quartz buffer.

```
typedef size_t (*CGDataProviderGetBytesAtOffsetCallback) (
    void *info,
    void *buffer,
    size_t offset,
    size_t count
);
```

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

```
size_t MyProviderGetBytesWithOffset (
    void *info,
    void *buffer,
    size_t offset,
    size_t count
);
```

### Parameters

*info*

A generic pointer to private data shared among your callback functions. This is the same pointer you supplied to [CGDataProviderCreateDirectAccess](#) (page 25).

*buffer*

The Quartz-supplied buffer into which you copy the specified number of bytes.

*offset*

Specifies the relative location in the data provider at which to begin copying data.

*count*

The number of bytes to copy.

### Return Value

The number of bytes copied. If no more data can be written to the buffer, you should return 0.

### Discussion

When Quartz is ready to receive data from the provider, your function is called.

For information on how to associate your function with a direct-access data provider, see [CGDataProviderCreateDirectAccess](#) (page 25) and [CGDataProviderDirectAccessCallbacks](#) (page 20).

### Availability

Available in Mac OS X v10.3 and later.

### Declared In

CGDataProvider.h

## CGDataProviderGetBytesAtPositionCallback

A callback function that copies data from the provider into a Quartz buffer.

```
typedef size_t (*CGDataProviderGetBytesAtPositionCallback) (
    void *info,
    void *buffer,
    off_t position,
    size_t count
);
```

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

```
size_t MyProviderGetBytesAtPosition (
    void *info,
    void *buffer,
    off_t position,
    size_t count
);
```

### Parameters

*info*

A generic pointer to private data shared among your callback functions. This is the same pointer you supplied to [CGDataProviderCreateDirect](#) (page 6).

*buffer*

The Quartz-supplied buffer into which you copy the specified number of bytes.

*position*

Specifies the relative location in the data provider at which to begin copying data.

*count*

The number of bytes to copy.

**Return Value**

The number of bytes copied. If no more data can be written to the buffer, you should return 0.

**Discussion**

When Quartz is ready to receive data from the provider, your function is called.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

CGDataProvider.h

**CGDataProviderGetBytesCallback**

A callback function that copies from a provider data stream into a Quartz-supplied buffer.

```
size_t (*CGDataProviderGetBytesCallback) (
    void *info,
    void *buffer,
    size_t count
);
```

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

```
size_t MyProviderGetBytes (
    void *info,
    void *buffer,
    size_t count
);
```

**Parameters**

*info*

A generic pointer to private data shared among your callback functions. This is the same pointer you supplied to [CGDataProviderCreate](#) (page 25).

*buffer*

The Quartz-supplied buffer into which you copy the specified number of bytes.

*count*

The number of bytes to copy.

**Return Value**

The number of bytes copied. If no more data can be written to the buffer, you should return 0.

**Discussion**

When Quartz is ready to receive data from the provider data stream, your function is called. It should copy the specified number of bytes into `buffer`.

For information on how to associate your callback function with a data provider, see [CGDataProviderCreate](#) (page 25) and [CGDataProviderCallbacks](#) (page 19).

**Availability**

Available in Mac OS X v10.3 and later.

**Declared In**

CGDataProvider.h

**CGDataProviderReleaseBytePointerCallback**

A callback function that releases the pointer Quartz obtained by calling [CGDataProviderGetBytePointerCallback](#) (page 11).

```
typedef void (*CGDataProviderReleaseBytePointerCallback) (
    void *info,
    const void *pointer
);
```

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

```
void MyProviderReleaseBytePointer (
    void *info,
    const void *pointer
);
```

**Parameters***info*

A generic pointer to private data shared among your callback functions. This is the same pointer you supplied to [CGDataProviderCreateDirectAccess](#) (page 25).

*pointer*

A pointer to your provider data. This is the same pointer you returned in [CGDataProviderGetBytePointerCallback](#) (page 11).

**Discussion**

When Quartz no longer needs direct access to your provider data, your function is called. You may safely modify, move, or release your provider data at this time.

For information on how to associate your function with a direct-access data provider, see [CGDataProviderCreateDirectAccess](#) (page 25) and [CGDataProviderDirectAccessCallbacks](#) (page 20).

**Availability**

Available in Mac OS X v10.3 and later.

**Declared In**

CGDataProvider.h

**CGDataProviderReleaseDataCallback**

A callback function that releases data you supply to the function [CGDataProviderCreateWithData](#) (page 7).

```
typedef void (*CGDataProviderReleaseDataCallback) (
    void *info,
    const void *data,
    size_t size
);
```

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

```
void MyProviderReleaseData (
    void *info,
    const void *data,
    size_t size
);
```

### Parameters

*info*

A generic pointer to private data shared among your callback functions. This is the same pointer you supplied to [CGDataProviderCreateWithData](#) (page 7).

*data*

A pointer to your provider data.

*size*

The size of the data.

### Discussion

When Quartz no longer needs direct access to your provider data, your function is called. You may safely modify, move, or release your provider data at this time.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

`CGDataProvider.h`

## CGDataProviderReleaseInfoCallback

A callback function that releases any private data or resources associated with the data provider.

```
void (*CGDataProviderReleaseInfoCallback) (
    void *info
);
```

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

```
void MyProviderReleaseInfo (
    void *info
);
```

**Parameters***info*

A generic pointer to private information shared among your callback functions. This is the same pointer you supplied to [CGDataProviderCreate](#) (page 25).

**Discussion**

When Quartz frees a data provider that has an associated release function, the release function is called.

For information on how to associate your callback function with a data provider, see [CGDataProviderCreate](#) (page 25) and [CGDataProviderCallbacks](#) (page 19).

**Availability**

Available in Mac OS X v10.3 and later.

**Declared In**

CGDataProvider.h

**CGDataProviderRewindCallback**

A callback function that moves the current position in the data stream back to the beginning.

```
void (*CGDataProviderRewindCallback) (
    void *info
);
```

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

```
void MyProviderRewind (
    void *info
);
```

**Parameters***info*

A generic pointer to private data shared among your callback functions. This is the same pointer you supplied to [CGDataProviderCreate](#) (page 25).

**Discussion**

When Quartz needs to read from the beginning of the provider's data stream, your function is called.

For information on how to associate your callback function with a data provider, see [CGDataProviderCreate](#) (page 25) and [CGDataProviderCallbacks](#) (page 19).

**Availability**

Available in Mac OS X v10.3 and later.

**Declared In**

CGDataProvider.h

**CGDataProviderSkipBytesCallback**

A callback function that advances the current position in the data stream supplied by the provider.

```
void (*CGDataProviderSkipBytesCallback) (
    void *info,
    size_t count
);
```

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

```
void MyProviderSkipBytes (
    void *info,
    size_t count
);
```

### Parameters

*info*

A generic pointer to private data shared among your callback functions. This is the same pointer you supplied to [CGDataProviderCreate](#) (page 25).

*count*

The number of bytes to skip.

### Discussion

When Quartz needs to advance forward in the provider's data stream, your function is called.

For information on how to associate your callback function with a data provider, see [CGDataProviderCreate](#) (page 25) and [CGDataProviderCallbacks](#) (page 19).

### Availability

Available in Mac OS X v10.3 and later.

### Declared In

`CGDataProvider.h`

## CGDataProviderSkipForwardCallback

A callback function that advances the current position in the data stream supplied by the provider.

```
off_t (*CGDataProviderSkipForwardCallback) (
    void *info,
    off_t count
);
```

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

```
off_t MyProviderSkipForwardBytes (
    void *info,
    off_t count
);
```

### Parameters

*info*

A generic pointer to private data shared among your callback functions. This is the same pointer you supplied to [CGDataProviderCreate](#) (page 25).

*count*

The number of bytes to skip.

**Return Value**

The number of bytes that were actually skipped.

**Discussion**

When Quartz needs to advance forward in the provider's data stream, your function is called.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

CGDataProvider.h

## Data Types

### CGDataProviderRef

Defines an opaque type that supplies Quartz with data.

```
typedef struct CGDataProvider *CGDataProviderRef;
```

**Discussion**

Some Quartz routines supply blocks of data to your program. Rather than reading through a raw memory buffer, data provider objects of type `CGDataProviderRef` allow you to supply Quartz functions with data.

In Mac OS X version 10.2 and later, `CGDataProviderRef` is derived from `CTypeRef` and inherits the properties that all Core Foundation types have in common. For more information, see *CType Reference*.

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

CGDataProvider.h

### CGDataProviderCallbacks

Defines a structure containing pointers to client-defined callback functions that manage the sending of data for a sequential-access data provider.

```

struct CGDataProviderCallbacks {
    CGDataProviderGetBytesCallback getBytes;
    CGDataProviderSkipBytesCallback skipBytes;
    CGDataProviderRewindCallback rewind;
    CGDataProviderReleaseInfoCallback releaseProvider;
};
typedef struct CGDataProviderCallbacks CGDataProviderCallbacks;

```

**Fields**

getBytes

A pointer to a function that copies data from the provider. For more information, see [CGDataProviderGetBytesCallback](#) (page 14).

skipBytes

A pointer to a function that Quartz calls to advance the stream of data supplied by the provider. For more information, see [CGDataProviderSkipBytesCallback](#) (page 17).

rewind

A pointer to a function Quartz calls to return the provider to the beginning of the data stream. For more information, see [CGDataProviderRewindCallback](#) (page 17).

releaseProvider

A pointer to a function that handles clean-up for the data provider, or NULL. For more information, see [CGDataProviderReleaseInfoCallback](#) (page 16).

**Discussion**

The functions specified by the `CGDataProviderCallbacks` structure are responsible for sequentially copying data to a memory buffer for Quartz to use. The functions are also responsible for handling the data provider's basic memory management. You supply a `CGDataProviderCallbacks` structure to the function [CGDataProviderCreate](#) (page 25) to create a sequential-access data provider.

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

`CGDataProvider.h`

**CGDataProviderDirectAccessCallbacks**

Defines pointers to client-defined callback functions that manage the sending of data for a direct-access data provider.

```

struct CGDataProviderDirectAccessCallbacks {
    CGDataProviderGetBytePointerCallback getBytePointer;
    CGDataProviderReleaseBytePointerCallback releaseBytePointer;
    CGDataProviderGetBytesAtOffsetCallback getBytes;
    CGDataProviderReleaseInfoCallback releaseProvider;
};
typedef struct CGDataProviderDirectAccessCallbacks
CGDataProviderDirectAccessCallbacks;

```

**Fields**

getBytePointer

A pointer to a function that returns a pointer to the provider's data. For more information, see [CGDataProviderGetBytePointerCallback](#) (page 11).

`releaseBytePointer`

A pointer to a function that Quartz calls to release a pointer to the provider's data. For more information, see [CGDataProviderReleaseBytePointerCallback](#) (page 15).

`getBytes`

A pointer to a function that copies data from the provider. For more information, see [CGDataProviderGetBytesAtOffsetCallback](#) (page 12).

`releaseProvider`

A pointer to a function that handles clean-up for the data provider, or NULL. For more information, see [CGDataProviderReleaseInfoCallback](#) (page 16).

### Discussion

You supply a `CGDataProviderDirectAccessCallbacks` structure to the function [CGDataProviderCreateDirectAccess](#) (page 25) to create a data provider for direct access. The functions specified by the `CGDataProviderDirectAccessCallbacks` structure are responsible for copying data a block at a time to a memory buffer for Quartz to use. The functions are also responsible for handling the data provider's basic memory management. For the callback to work, one of the `getBytePointer` and `getBytes` parameters must be non-NULL. If both are non-NULL, then `getBytePointer` is used to access the data.

### Availability

Available in Mac OS X v10.0 and later.

### Declared In

`CGDataProvider.h`

## CGDataProviderDirectCallbacks

Defines pointers to client-defined callback functions that manage the sending of data for a direct-access data provider.

```
struct CGDataProviderDirectCallbacks {
    unsigned int version;
    CGDataProviderGetBytePointerCallback getBytePointer;
    CGDataProviderReleaseBytePointerCallback releaseBytePointer;
    CGDataProviderGetBytesAtPositionCallback getBytesAtPosition;
    CGDataProviderReleaseInfoCallback releaseInfo;
};
typedef struct CGDataProviderDirectCallbacks CGDataProviderDirectCallbacks;
```

### Fields

`version`

The version of this structure. It should be set to 0.

`getBytePointer`

A pointer to a function that returns a pointer to the provider's data. For more information, see [CGDataProviderGetBytePointerCallback](#) (page 11).

`releaseBytePointer`

A pointer to a function that Quartz calls to release a pointer to the provider's data. For more information, see [CGDataProviderReleaseBytePointerCallback](#) (page 15).

`getBytesAtPosition`

A pointer to a function that copies data from the provider.

`releaseInfo`

A pointer to a function that handles clean-up for the data provider, or `NULL`. For more information, see [CGDataProviderReleaseInfoCallback](#) (page 16).

#### Discussion

You supply a `CGDataProviderDirectCallbacks` structure to the function [CGDataProviderCreateDirect](#) (page 6) to create a data provider for direct access. The functions specified by the `CGDataProviderDirectCallbacks` structure are responsible for copying data a block at a time to a memory buffer for Quartz to use. The functions are also responsible for handling the data provider's basic memory management. For the callback to work, one of the `getBytesPointer` and `getBytesAtPosition` parameters must be non-`NULL`. If both are non-`NULL`, then `getBytesPointer` is used to access the data.

#### Availability

Available in Mac OS X v10.5 and later.

#### Declared In

`CGDataProvider.h`

## CGDataProviderSequentialCallbacks

Defines a structure containing pointers to client-defined callback functions that manage the sending of data for a sequential-access data provider.

```
struct CGDataProviderSequentialCallbacks {
    unsigned int version;
    CGDataProviderGetBytesCallback getBytes;
    CGDataProviderSkipForwardCallback skipForward;
    CGDataProviderRewindCallback rewind;
    CGDataProviderReleaseInfoCallback releaseInfo;
};
typedef struct CGDataProviderSequentialCallbacks CGDataProviderSequentialCallbacks;
```

#### Fields

`version`

The version of this structure. It should be set to 0.

`getBytes`

A pointer to a function that copies data from the provider. For more information, see [CGDataProviderGetBytesCallback](#) (page 14).

`skipForward`

A pointer to a function that Quartz calls to advance the stream of data supplied by the provider.

`rewind`

A pointer to a function Quartz calls to return the provider to the beginning of the data stream. For more information, see [CGDataProviderRewindCallback](#) (page 17).

`releaseInfo`

A pointer to a function that handles clean-up for the data provider, or `NULL`. For more information, see [CGDataProviderReleaseInfoCallback](#) (page 16).

#### Discussion

The functions specified by the `CGDataProviderSequentialCallbacks` structure are responsible for sequentially copying data to a memory buffer for Quartz to use. The functions are also responsible for handling the data provider's basic memory management. You supply a `CGDataProviderCallbacks` structure to the function [CGDataProviderCreateSequential](#) (page 6) to create a sequential-access data provider.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

CGDataProvider.h



# Deprecated CGDataProvider Functions

---

A function identified as deprecated has been superseded and may become unsupported in the future.

## Deprecated in Mac OS X v10.5

### CGDataProviderCreate

Creates a Quartz sequential-access data provider. (Deprecated in Mac OS X v10.5.)

```
CGDataProviderRef CGDataProviderCreate (
    void *info,
    const CGDataProviderCallbacks *callbacks
);
```

#### Parameters

*info*

A pointer to data of any type or NULL. When Quartz calls the functions specified in the `callbacks` parameter, it sends each of the functions this data.

*callbacks*

A pointer to a `CGDataProviderCallbacks` structure that specifies the callback functions you implement to handle the data provider's basic memory management. For a complete description, see [CGDataProviderCallbacks](#) (page 19).

#### Return Value

A new data provider. You are responsible for releasing this object using [CGDataProviderRelease](#) (page 9).

#### Discussion

You use this function to create a sequential-access data provider that uses callback functions to read data from your program in a stream.

#### Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

#### Declared In

`CGDataProvider.h`

### CGDataProviderCreateDirectAccess

Creates a Quartz direct-access data provider. (Deprecated in Mac OS X v10.5.)

## Deprecated CGDataProvider Functions

```
CGDataProviderRef CGDataProviderCreateDirectAccess (  
    void *info,  
    size_t size,  
    const CGDataProviderDirectAccessCallbacks *callbacks  
);
```

**Parameters**

*info*

A pointer to data of any type or NULL. When Quartz calls the functions specified in the `callbacks` parameter, it sends each of the functions this pointer.

*size*

A value that specifies the number of bytes that the data provider contains.

*callbacks*

A pointer to a `CGDataProviderDirectAccessCallbacks` structure that specifies the callback functions you implement to handle the data provider's basic memory management. For a complete description, see [CGDataProviderDirectAccessCallbacks](#) (page 20).

**Return Value**

A new data provider. You are responsible for releasing this object using [CGDataProviderRelease](#) (page 9).

**Discussion**

You use this function to create a direct-access data provider that uses callback functions to read data from your program in a single block.

**Availability**

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

**Related Sample Code**

MassiveImage

**Declared In**

CGDataProvider.h

# Document Revision History

This table describes the changes to *CGDataProvider Reference*.

Date	Notes
2009-01-06	Added entries for the <code>CGDataProviderCopyData</code> and <code>CGDataProviderCreateWithFilename</code> functions.
2008-04-08	Added the version field to two callback data structures.
2007-10-31	Corrected typographical errors.
2007-06-26	Updated for Mac OS X v10.5.
	Added the function <code>CGDataProviderCreateSequential</code> (page 6) and <code>CGDataProviderCreateDirect</code> (page 6).
	Added the callbacks <code>CGDataProviderSkipForwardCallback</code> (page 18) and <code>CGDataProviderGetBytesAtPositionCallback</code> (page 13).
	Added the data structures <code>CGDataProviderDirectCallbacks</code> (page 21) and <code>CGDataProviderSequentialCallbacks</code> (page 22).
2005-04-29	Updated for Mac OS X v10.4.
	Changed the name of the <code>releaseInfo</code> field to <code>releaseProvider</code> and then modified the fields in the data structures <code>CGDataProviderCallbacks</code> (page 19) and <code>CGDataProviderDirectAccessCallbacks</code> (page 20) to match the field descriptions. Formerly, they showed a callback prototype in the field instead of the callback data type. The usage of these data structures remains the same.
	Added the function <code>CGDataProviderCreateWithCFData</code> (page 7).
	Added the callback <code>CGDataProviderReleaseDataCallback</code> (page 15).
2004-08-31	Added introductory material.
2004-02-26	First version of this document. An earlier version of this information appeared in <i>Quartz 2D Reference</i> .

## REVISION HISTORY

### Document Revision History

# Index

---

## C

---

CGDataProviderCallbacks **structure** 19  
CGDataProviderCopyData **function** 5  
CGDataProviderCreate **function** (Deprecated in Mac OS X v10.5) 25  
CGDataProviderCreateDirect **function** 6  
CGDataProviderCreateDirectAccess **function** (Deprecated in Mac OS X v10.5) 25  
CGDataProviderCreateSequential **function** 6  
CGDataProviderCreateWithCFData **function** 7  
CGDataProviderCreateWithData **function** 7  
CGDataProviderCreateWithFilename **function** 8  
CGDataProviderCreateWithURL **function** 9  
CGDataProviderDirectAccessCallbacks **structure** 20  
CGDataProviderDirectCallbacks **structure** 21  
CGDataProviderGetBytePointerCallback **callback** 11  
CGDataProviderGetBytesAtOffsetCallback **callback** 12  
CGDataProviderGetBytesAtPositionCallback **callback** 13  
CGDataProviderGetBytesCallback **callback** 14  
CGDataProviderGetTypeID **function** 9  
CGDataProviderRef **data type** 19  
CGDataProviderRelease **function** 9  
CGDataProviderReleaseBytePointerCallback **callback** 15  
CGDataProviderReleaseDataCallback **callback** 15  
CGDataProviderReleaseInfoCallback **callback** 16  
CGDataProviderRetain **function** 10  
CGDataProviderRewindCallback **callback** 17  
CGDataProviderSequentialCallbacks **structure** 22  
CGDataProviderSkipBytesCallback **callback** 17  
CGDataProviderSkipForwardCallback **callback** 18