

IDEAS FOR INTERACTIVE CD AUDIO

This portion of *The Voyager CD AudioStack* contains ideas that you can use in creating your own HyperCard stacks for controlling a CD audio player.

The audio examples used in this stack are taken from *The Movies Go to the Opera* CD. Of course, you may test the ideas in this stack with any CD, but the sounds will match the descriptions in the examples only if you use *The Movies Go to the Opera*.

In most cases, you will need to look at the script(s) of the button(s) or field(s) to see how the example works. If the *Show Scripts* button at the top left of each card is checked, when you click on a button or field you will be shown its script, along with comments and suggestions. When you press *OK*, the script will be hidden AND will send its instructions to the CD-ROM drive.

AUDIO SYNTAX

The CD AudioStack adds 29 CD commands to HyperTalk, the HyperCard scripting language. These CD commands are divided into XCMDs (HyperCard handlers) and XFCNs (HyperCard functions), and are discussed below in alphabetical order under each division. Each entry describes the command's use, syntax, and parameters, and includes explanatory notes. The "Audio Syntax" section of the AudioStack includes buttons that give examples of the use of each command.

In the commands that follow, optional terms are enclosed with square brackets ([]). If you use an optional term in your scripts, do not include the brackets. An *italicized word* designates a general term that must be replaced with a specific word or number. The "Parameters" section tells you which words or numbers can be used. The commands are not case-sensitive; you can use any combination of uppercase and lowercase letters to type the commands, as you please.

XFCNs always return a value. Many of the CD XCMDs put a value into "the result." You can access and check the result in your HyperCard scripts—using the line of script "Get the result" or "Put the result into xxx"—and can then determine subsequent HyperTalk commands to be executed depending on the result. Here's a simple example of the use of "the result" with a CD command:

```
CDOpen 2
-- Attempts to open a CD in a drive with SCSI ID 2
Get the result
If it = 0 then
    put "There is no CD in a connected drive with SCSI ID 2."
end if
```

If the value returned, or the result, is a negative number, an error has occurred. See "Error Reporting" on page 46.

XCMDs (HYPERCARD HANDLERS)

CDEject

Ejects the disc from the CD drive that is currently open and unmounts the CD.

Command: CDEject

Parameters: None.

Notes: Returns 0 if there is no error, -53 if there is no disc in a mounted drive, or -56 if a CD was ejected from the drive since the last CDOpen was issued. When a CD is unmounted, its icon disappears from the desktop.

CDOpen

Sets up a global variable named "xxxCDRefNum," which is used by all subsequent CD commands until another drive is opened. You must call this XCMD before calling any of the other CD XCMDs or XFCNs.

Command: CDOpen [*SCSI ID*]

Parameters: *SCSI ID* allows you to specify the SCSI ID of a particular drive, if there is more than one CD-ROM drive attached to your Macintosh. (Refer to your CD-ROM drive manual if you don't know the SCSI ID of your drive.)

Notes: If you specify *SCSI ID*, all subsequent CD commands will be sent to that drive until you unmount it or open another drive.

If there is more than one CD-ROM drive, and you do not specify *SCSI ID*, CD commands will be sent to the first drive that was mounted.

If there is no mounted CD-ROM drive with the specified SCSI ID, the result is 0.

A call to CDOpen belongs in an initialization handler such as openStack. For example:


```
on openStack
  CDOpen
end openStack
```

CDPause

Pauses the CD.

Command: CDPause [*PauseMode*]

Parameters: If *PauseMode* = 0, the CD will enter hold state and mute audio (pause). If *PauseMode* = 1, the CD will resume play at the next block. "CDPause" with no parameter toggles between play and pause; it pauses the CD if it is currently playing, and plays the CD if it is currently paused.

Notes: Returns 0 if there is no error, or a negative number if there is an error.

You can also resume play by issuing a CDPlay command with no parameters.

CDPlay

Plays the CD.

Command: CDPlay [*minute, second, block*] [,*minute, second, block*]

Parameters: "CDPlay" with no parameters plays the CD from the current location of the disc. The first set of optional parameters is the address at which to start playing, specified by minute, second, and block. The optional second set of parameters is the address at which to stop playing.

Notes: Returns 0 if no error, and -50 if an invalid minute, second, block was specified.

If no CDStop call was issued before this call, and you do not enter the optional second address, play continues until the end of the CD. If you've set up a Stop condition which is previous to the first address, or previous to the current disc location if you have not entered the option first address, then the CD will not play. (See CDStop on page 36.)

When you issue a CDPlay command in a script, you may want to check first whether or not the CD is playing, and then issue the CDPlay command only if the CD is not playing. (Otherwise, a CDPlay command will cause a brief interruption in the sound.) To see if the CD is already playing, you could use a line like this:

```
if CDStatus() <> 1 then CDPlay
```

Note that the beginning of the disc is not at 00,00,00. You should call CDReadTOC() or CDTocLines() to find the address of the beginning of the first track, or issue "CDPlayTrack 1."

Reading data from a mixed mode audio/data CD-ROM disc will cause any play in progress to halt. (Playing audio is just like reading data. You can do one or the other, but not both simultaneously.)

CDPlay1

Plays a specified track and stops at the end of the track.

Command: CDPlay1 *trackNumber*

Parameters: *TrackNumber* specifies which track will be played.

Notes: Returns 0 if there is no error, -21 if there is no disc in a mounted drive, or -50 if a non-existent track was specified.

This call is the same as a CDStopTrack followed by a CDPlayTrack.

This command sets a stop condition at the start of the next track on the disc. You will need to clear that condition by using CDStop with no parameters or by using a handler like this:

```
on clearCDStops
    get CDDiscTime()
    CDStop item 1 of it, item 2 of it, item 3 of it
    -- resets the stop condition to end of the CD
end clearCDStops
```


CDPlayTrack

Starts playing the CD at the beginning of a specified track.

Command: `CDPlayTrack trackNumber [,stopTrackNumber]`

Parameters: *TrackNumbers* specifies the track at which the CD will begin playing. If the optional *stopTrackNumber* is specified, `CDPlayTrack` will set a stop condition at the end of the specified second track.

Notes: Returns 0 if there is no error, or -21 if a non-existent track was specified.

To find the last track number, get item 2 of `CDTrack()`.

If no *stopTrackNumber* parameter is specified, and no `CDStopTrack` call was issued before this call, the CD will play until the end of the disc. If you previously issued a `CDStopTrack` command with a lower track specified than *trackNumber*, the CD will not play.

If *trackNumber* = *stopTrackNumber*, then `CDPlayTrack trackNumber, stopTrackNumber` is the same as `CDPlay1 trackNumber`.

CDScan

Scans the CD forward or in reverse, starting at the specified location.

Command: `CDScan direction, minute, second, block`

Parameters: If *direction* = 0, the CD scans forward. If *direction* = 1, the CD scans in reverse. *Minute*, *second*, and *block* specify the address from which the CD starts scanning.

Notes: Returns 0 if there is no error, or -1 if the parameters were invalid.

Scanning continues until you issue a `CDPause`, `CDStop`, or `CDPlay` command.

To scan forward from the current location, you can use this handler:


```

on mouseDown
    put CDStatus() into Status
    get CDElapsed() -- find out where the CD is
    do "CDStop " & CDDiscTime()
    CDScan 0, item 2 of it, item 3 of it, item 4 of it
    wait until the mouse is up
    if Status = 1 then CDPlay else CDPause 0
end mouseDown

```

CDSearch

Searches to the specified minute, second, and block.

Command: `CDSearch playFlag, minute, second, block`

Parameters: *Minute*, *second*, and *block* specify the address to search to. If *playFlag* = 0, the CD pauses at the specified minute, second, and block. If *playFlag* = 1, the CD will play from the specified location.

Notes: Returns 0 if there is no error, or -1 if the parameters were invalid.

If you try to search to a location on a mixed mode CD that contains data rather than audio, CDSearch will return -36.

If you set *playFlag* to 0 in order to pause, you can resume play at the specified location by issuing the command CDPause, CDPause 1, or CDPlay with no parameters.

CDSearchTrack

Searches to the beginning of the specified track number on the CD.

Command: `CDSearchTrack playFlag, track`

Parameters: *Track* specifies the track to search to. If *playFlag* = 0, the CD pauses at the beginning of the track identified by the *track* parameter. If *playFlag* = 1, the CD plays from the beginning of the specified track.

Notes: Returns 0 if there is no error, or -1 if the parameters were invalid.

If you try to search to a location on a mixed mode CD that contains data rather than audio, CDSearchTrack

will return -36.

If you set *playflag* to 0 in order to pause, you can resume play at the beginning of the specified track by issuing the command `CDPause`, `CDPause 1`, or `CDPlay` with no parameters.

CDSetDiscTitle

Sets the disc title in the file "CD Remote Programs" to the title specified by the user.

Command: `CDSetDiscTitle` *title*

Parameters: *Title* specifies the title you want to assign to the CD.

Notes: As with any HyperCard command, if the title contains spaces or special characters you should enclose it in quotation marks like this:

`CDSetDiscTitle "Movies Go to the Opera"`

If `CDSetDiscTitle` can't find an entry that already exists for the current disc, it will create a new disc entry with default values for the track titles.

CDSetTrackTitle

Sets the track title in the file "CD Remote Programs" to the given title.

Command: `CDSetTrackTitle` *title* [,*trackNumber*]

Parameters: *Title* specifies the title you want to assign to the track. If the optional parameter *trackNumber* is specified, `CDSetTrackTitle` sets the title for the specified track. Otherwise, it sets the title for the current track.

Notes: As with any HyperCard command, if your title contains spaces or special characters you should enclose it in quotation marks like this:

`CDSetTrackTitle "Ride of the Valkyries", 8`

If `CDSetTrackTitle` can't find an entry that already exists for the current disc, it will create a new disc entry with default values for the other track titles and the disc title.

CDStop

CDStop stops playing at the specified minute, second, and block.

Command: CDStop [*minute,second,block*]

Parameters: *Minute*, *second*, and *block* specify the address at which you want the CD to stop playing. A CDStop command without parameters sets the stop value to the end of the disc and searches to the beginning of the disc. If you are using a mixed mode audio/data CD, it cannot search to the beginning of the disc and will return an error of -36, because the first track on a mixed CD is the data track.

Notes: Returns 0 if there is no error, or -50 if an invalid minute, second or block was specified.

When issued with parameters, the CDStop command sets a stop condition, which must be cleared before play can go on past that point. The drive continues to play until the stop condition is satisfied; therefore, if you set a stop before the current play, you stop immediately.

To play a specific segment of a CD, you can issue a CDStop with the location of the end point, followed by a CDPlay with the location of the start point. For example, the following series of commands would play 2 seconds of sound beginning at minute 12, exactly:

```
CDStop 12, 2, 0
-- Stop at 12 minutes, 2 secs, 0 blocks.
CDPlay 12, 0, 0
-- Play from minute 12 exactly.
```

Issuing a CDStop before a CDPlay or CDPlayTrack command ensures that you don't have a previous CDStop or CDStopTrack command in place that will kill your CDPlay command.

The following handler for mixed mode discs sets the stop condition to the end of the disc and searches to the first audio track (track 2):


```
on mouseUp
    global playFlag
    -- CDStop with no parameters will
    -- move to the start of track 1
    -- and reset the stop condition
    -- to the end of the disc
    CDStop-- returns -36 if this is a mixed mode CD
    If the result <> 0 then
        -- avoid the start of track 1, which is data,
        -- and find track 2
        get CDReadTOC()
        -- get the table of contents
        CDSearch 0,item 4 of it,item 5 of it,item 6 of it
        -- search to start of track 2, where audio begins
    end if
end mouseUp
```

CDStopTrack

Stops playing at the end of the specified track.

Command: **CDStopTrack** *trackNumber*

Parameters: *TrackNumber* specifies the number of the track at the end of which the CD will stop playing.

Notes: Returns 0 if there is no error or -21 if a non-existent track is specified.

To play only one specific track of a CD, you can issue a **CDStopTrack** followed by a **CDPlayTrack** for that track. Issuing a **CDStopTrack** before a **CDPlayTrack** command ensures that you don't have a previous **CDStop** or **CDStopTrack** command in place that will kill your **CDPlay** command:

```
on mouseUp
    CDStopTrack 2
    -- stop playing at the end of track 2
    CDPlayTrack 2
    -- start playing at the beginning of track 2
end mouseUp
```


XFCNS

(HYPERCARD FUNCTIONS)

CDDiscTime()

Returns the total time on the disc in a comma separated list. Items 1, 2, 3 of the list are the total time on the disc in absolute minutes, seconds, and blocks.

Command: CDDiscTime()

Parameters: None

Notes: The returned value is actually one block less than the real total time. If the real total time were returned, you couldn't pass the returned value to CDStop.

CDDiscTitle()

Returns the title of the disc as it is stored in the file "CD Remote Programs."

Command: CDDiscTitle()

Parameters: None

Notes: Returns an empty string if the disc isn't stored in the database, or if an error occurred.

The "CD Remote Programs" file is stored in the System Folder of the startup disk.

CDElapsed()

Returns the current track number and elapsed time from the beginning of the disc in a comma separated list. Item 1 is the current track number. Items 2, 3, and 4 are the elapsed time from the beginning of the disc in minutes, seconds, and blocks.

Command: CDElapsed()

Parameters: None

CDElapsedTrack()

Returns the current track number and elapsed time from the beginning of the track in a comma separated list. Item 1 is the current track number. Items 2, 3, and 4 are the elapsed time from the beginning of the track in minutes, seconds, and blocks.

Command: CDElapsedTrack()

Parameters: None

CDIdent()

Returns a unique id number for the CD in the currently open drive.

Command: CDIdent()

Parameters: None

Notes: Returns -21 if there is no CD drive, or if there is no CD in the drive.

The ID number is the total number of blocks on the CD. It's very unlikely that two CDs are the same length down to the 75th of a second. This same identification scheme is used by the CD Remote desk accessory.

If you write a stack that uses a specific CD, you might use a script like this to determine whether the user has inserted the correct CD:

```
on mouseUp
  if CDIdent() <> "7406896" then
    beep
    answer "Works with Movies Go to the Opera"
    with Eject or Continue
      if it is "Eject" then
        CDEject
        exit mouseUp
      end if
    end if
    -- If correct disc is in the drive, or user chooses
    -- to continue anyway, rest of the button works
    CDPlay1 2
  end mouseUp
```


CDProgram()

Returns a list specifying the order in which the tracks on the CD will be played by the CD Remote desk accessory.

Command: CDProgram()

Parameters: None

Notes: Returns a comma separated list with two entries for each track. The first item is the track number in the sequence specified in the file "CD Remote Programs." The second item is 1 if the track should be played, and 0 if the track should not be played.

If the currently mounted disc has no program file (because you've never entered any information about the tracks) CDProgram() returns the default set, as follows:

01,01,02,01,03,01,04,01,05,01... (and so on)

CDReadTOC()

Reads the table of contents of the current CD and returns a comma separated list of beginning track times.

Command: CDReadTOC()

Parameters: None

Notes: Items 1, 2, and 3 are the start of the first track (minute, second, block). Items 4, 5, and 6 are the start of the second track (minute, second, block), and so on.

CDRemaining()

Returns the current track number and the remaining time to the end of the disc in a comma separated list.

Command: CDRemaining()

Parameters: None

Notes: Item 1 is the track number. Items 2, 3, and 4 are the remaining time to the end of the disc in minutes, seconds, and blocks.

CDRemainingTrack()

Returns the current track number and the remaining time to

the end of the track in a comma separated list.

Command: `CDRemainingTrack()`

Parameters: None

Notes: Item 1 is the current track number. Items 2, 3, and 4 are the remaining time to the end of the track in minutes, seconds, and blocks.

CDScsiID()

Returns a number from 0 to 7, indicating the SCSI ID of the CD-ROM drive that is currently open, to which all CD commands are being sent.

Command: `CDScsiID()`

Parameters: None

Notes: Returns a -1 if you don't have a CD-ROM drive open, or if an error occurs while trying to find the SCSI ID.

CDStatus()

Returns a status value.

Command: `CDStatus()`

Parameters: None

Notes: If the value is negative, an error occurred. If the value is 0 or positive, it has the following meaning:

- 0 audio pause in operation
- 1 audio play in progress
- 2 audio muting on
- 3 audio play operation complete
- 4 error occurred during audio play operation
- 5 CD is not currently playing

You can use the `CDStatus()` call to figure out when to terminate a loop, like this:

```
CDPlay1 6
repeat until CDStatus() = 3
-- Waits until first passage has finished playing
-- before second passage can begin
end repeat
CDPlay1 2
```


To let the user know whether a CD is playing or not, you could use a script like this:

```
CDPlay 31,32,11,34,53,55
repeat while CDStatus() = 1
    put "Playing" into field "Status"
end repeat
put "Stopped" into field "Status"
```

CDTime()

Returns a comma separated list with information on the disc time.

Command: CDTime()

Parameters: None

Notes: Items 1, 2, and 3 are the elapsed time on the current track (minute, second, block).

Items 4, 5, and 6 are the remaining time on the current track (minute, second, block).

Items 7, 8, and 9 are the elapsed time on the disc (minute, second, block).

Items 10, 11, and 12 are the remaining time on the disc (minute, second, block).

Items 13, 14, and 15 are the total run time on the disc (minute, second, block).

CDTOCLines()

Reads the CD's table of contents and returns text in a number of lines equal to the number of tracks on the CD.

Command: CDTOCLines()

Parameters: None

Notes: Each line contains a comma separated list of three items. Items 1, 2, and 3 in each line are the start of the track (minute, second, block) corresponding to that line.

For a CD containing three tracks, CDTOCLines() could return something like this:


```
00,02,30 -- Line 1 is the start of track 1.  
23,24,61 -- Line 2 is the start of track 2.  
41,19,39 -- Line 3 is the start of track 3.
```

This could be useful in creating a handler that plays a specified portion of a track, as follows:

```
on playFiveSecondsOfTrack2  
  put line 2 of CDTOCLines() into startPoint  
  put startPoint into endPoint  
  add 5 to item 2 of endPoint  
  -- add 5 seconds to where play begins  
  do "CDPlay" && startPoint & "," & endPoint  
end playFiveSecondsOfTrack2
```

CDTrack()

Returns the current track and the total number of tracks on the disc.

Command: CDTrack()

Parameters: None

Notes: The first item in the comma separated list is the number of the current track; the second item is the total number of tracks on the disc.

CDTrackTime()

Returns the length of the specified track in a comma separated list.

Command: CDTrackTime([*trackNumber*])

Parameters: *TrackNumber* specifies the number of the track whose length you want to find. With no parameters, CDTrackTime() returns the length of the current audio track.

Notes: Items 1, 2, and 3 are the length of the specified track (minutes, seconds, blocks).

CDTrackTitle()

Returns the title of the specified track as it is stored in the file "CD Remote Programs."

Command: CDTrackTitle([*trackNumber*])

Parameters: With no parameters, `CDTrackTitle()` returns the title of the current audio track. If you specify an optional track number, the title for that track is returned.

Notes: Returns an empty string if the disc isn't stored in the database, or if an error occurred.

The file "CD Remote Programs" is maintained by the CD Remote desk accessory and the XCMDs `CDSetDiscTitle` and `CDSetTrackTitle`. This database file is stored in the System Folder of the startup disk.

AUDIO CHANNEL ROUTING

A global variable named "`xxxCDPlayMode`" is used to determine which source channels are played through which speakers. By default, the `CDOpen` command sets the value of `xxxCDPlayMode` to 9, which means stereo. At any time after issuing `CDOpen`, you can set this variable to any value from 0 to 15, with the following meanings:

<i>xxxCDPlayMode</i>	<i>Effect</i>
0	muting on (no audio)
1	right channel through right channel only
2	left channel through right channel only
3	left and right channel through right channel only
4	right channel through left channel only
5	right channel through left and right channel
6	right channel through left channel, left channel through right channel
7	right channel through left channel, left and right channel through right channel
8	left channel through left channel only

- | | |
|----|--|
| 9 | left channel through left channel, right channel through right channel (stereo) |
| 10 | left channel through left and right channel |
| 11 | left channel through left channel, left and right channel through right channel |
| 12 | left and right channel through left channel |
| 13 | left and right channel through left channel, right channel through right channel |
| 14 | left and right channel through left channel, left channel through right channel |
| 15 | left and right channel through left channel, left and right channel through right channel (monaural) |

To change the audio output play mode, use these Hypertalk commands:

```
global xxxCDPlayMode
put n into xxxCDPlayMode
```

where *n* is an integer ranging from 0 through 15 (the setting for the desired play mode, as listed above). You can then issue any of the commands CDPlay, CDPlay1, CDPlayTrack, CDSearch, CDSearchTrack, or CDStop, and the disc will play in the chosen audio mode.

Since a CDOpen command automatically sets the value of xxxCDPlayMode to 9 (stereo), you will need to add scripting following any CDOpen commands in your stack if you want to maintain a previously set audio channel routing other than stereo. For example:

```
global xxxCDPlayMode
put xxxCDPlayMode into Holder
CDOpen
if (Holder >= 0) and (Holder <= 15) then
    put Holder into xxxCDPlayMode
    -- Holder contains the previous value
    -- of xxxCDPlayMode.
    -- The audio play mode is changed to Holder only if
    -- Holder contained a useable value to begin with.
end if
```


ERROR REPORTING

If you issue one of the CD XCMDs or XFCNs and an error occurs, the driver will return an error and the XCMD or XFCN will then return an error value—always a small negative number. Error values may be examined as The Result for XCMDs, and as the variable It for XFCNs.

If you don't pass in the proper number of parameters, you'll get the error code "-1." Notice that -21 (bad unit error) and -50 (parameter error) can mean different things depending upon which call was made.

Here are some typical error numbers, what they might mean for the AppleCD SC, and the definitions from *Inside Macintosh*:

- 1 Wrong number of parameters
- 17 Driver can't respond to the Control call. (control error)
- 21 Illegal track. (bad unit error)
- 21 The disc isn't ready to receive commands during mount. (bad unit error)
- 21 You issued a command with no disk in the drive. (bad unit error)
- 36 The track you want to search to is a data track. (I/O error)
- 50 This is a data track, not an audio track. (parameter error)
- 50 You specified something wrong. (parameter error)
- 53 You issued CDEject with no disk in the drive. (volume not on line error)
- 56 You issued a command with no disk in the drive. (no such drive)
- 65 No CD in drive. (read requested for an off-line drive)
- 108 Ran out of memory. (not enough room in heap zone)
- 194 Unable to create new CD information in "CD Remote Programs." (AddResource failed)

LICENSING INFORMATION

The 29 CD XCMDs and XFCNs, Str# ID 8078, and the four ResCopy resources which are included in The Voyager CD AudioStack are under license from Apple Computer.

Developers who want to incorporate any or all of these resources into products intended for sale must apply for a royalty-free license from Apple Computer and pay an annual fee. Inquiries should be directed to:

Apple Computer, Inc.
Software Licensing, mail stop 38-I
20525 Mariani Avenue
Cupertino, CA 95014 U.S.A.



The Voyager Company

1351 Pacific Coast Highway

Santa Monica, CA 90401

213 451-1383

a joint venture of Janus Films and Voyager Press

ISBN 1-55940-050-1