

# c74 v1.5 manual



## contents

---

- about c74
  - about v1.5
  - c74 desktop application
  - converting v1.4 presets to v1.5 presets
  - connecting c74 to Max/MSP
  - objects
    - creating
    - setting values
    - appearance
    - deleting
  - fetching sensor data
  - messages to the c74 app
    - global settings
    - attributes
    - saving 'patchers'
  - images
    - displaying
    - retrieving
  - sounds
    - playing
    - recording
  - list of objects
-

## about c74

---

c74 is like a remote control for Max/MSP on your iOS device. Since version 1.5 c74 also features a desktop version that acts like a bridge between your iOS device and desktop applications capable of receiving MIDI and OpenSoundControl.

c74 does not run Max/MSP on the iOS device, it does however use a similar syntax. User interface objects that are known within Max/MSP have been ported to similar objects within the c74 app. They can be drawn on the iOS's screen using familiar Max messages. A list of objects can be found at the end of this manual.

c74 is also able to send data from all of the iPhone/iPad's sensors back to Max/MSP, which makes c74 an even more useful controller.

In order to make a connection to Max/MSP you have to download the latest version of the c74 external, which can be found at [www.nr74.org/c74](http://www.nr74.org/c74). Currently the external is OSX only and requires Max 5 or 6. The external is open source so a Windows external might be built in the future by who knows.

---

## about v1.5.2

---

### Fix :

- Compatibility with Max 6.
- Deleting/closing the c74 external now disconnects the iOS app properly.
- iPad now works properly in landscape mode (beyond 720px).
- [read filepath] message now works correctly.
- Fonts & colors are now stored within presets.
- Fixed several small bugs.

### Added :

- gyroscope attribute, including [gyroRate rate] message to set polling frequency
- [accelerometerRate rate] message to set polling frequency for accelerometer data
- flashlight support via the [flashlight state] message (experimental stage...)
- [bringtofront ID] and [sendtoback ID] messages
- rslider object
- swatch object
- panel object
- lcd object (basic implementation)
- button object now returns [ID 1] or [ID 0] instead of [ID bang].
- convert74, this application converts v1.4 presets into v1.5 presets
- d74, this application connects to the c74 iOS app and enables you to create UI's much more intuitive. The application also acts like a bridge between the iOS app and many other apps using MIDI or OpenSoundControl. d74 can be seen as an alternative to the c74 external, though not every function of the external is implemented in d74. d74 is currently in beta.

### Known issues :

- Sounds are recorded in .caf format, which unfortunately can't be read by Max
  - Removed reading/grabbing of images, encountered too much crashes...
-

## **c74 desktop application**

---

There's a new way to interact with c74 using the new desktop version of c74. This application lets you create drag and drop UI's for c74. Simply create a UI on your desktop machine, connect to the iOS version of c74 and it will mirror the document you're working on on your desktop machine. An export feature is also available so UI's can be loaded into the c74 Max external as well.

The c74 desktop application also acts like a bridge between the iOS device and a variety of applications on your desktop machine since it is capable of parsing every iOS output to either MIDI or OpenSoundControl. Output can be scaled to match your needs. Using this application enables you to control every software that accepts MIDI or OpenSoundControl e.g. Ableton, Logic, Isadora but also OpenFrameWorks, Processing etc.

Please note that the c74 desktop application is currently in beta state, therefore not every feature of the c74 Max external is implemented in this desktop application.

---

## **converting presets from v1.4 to v1.5**

---

Since the structure of the presets has dramatically changed it is no longer possible to load your old presets into the new v1.5 external.

If you want to use your old presets convert them using the Convert74 application that is included in the zip containing the v1.5 external at [www.nr74.org/c74](http://www.nr74.org/c74)

---

## connecting c74 to Max/MSP

---

A connection between the iOS device and the OSX device requires both devices to join the same WiFi network. When both devices are pointed to the same network a connection can be established unless some network device blocks the traffic over the port c74 uses.

The most reliable connection can be established by creating a network on the OSX device. Use the following steps to create your own WiFi network :

- Open *System Preferences...*
- Select *Network*
- Select *Airport*
- Select *Create Network...* under *Network Name*
- Give your network any name and optionally use a password

At this point you should point your iOS device to the newly created network.

After you've made sure both devices share the same network open Max/MSP and create an instance of the c74 external. The external can be downloaded from [www.nr74.org/c74](http://www.nr74.org/c74) and should be placed in *Max 5/Cycling '74/Max externals* or elsewhere in Max/MSP's searchpath. Upon creating the external it listens for the c74 app on a given channel. The default channel is 1, but multiple channels can be used to connect to multiple iOS devices.

The port over which data is sent is equal to the channel number used plus 8000. So when you've pointed c74 to channel 15 port 8015 is used. So make sure traffic is allowed over that port when using additional network devices.

Now open the c74 app. After it finishes launching it is waiting for connection and should connect fairly soon when all the above is followed.

---

## objects

---

c74 always opens with a blank screen. It will stay blank till you send the app messages from Max/MSP. The good thing here is that you are in control of the user interface, without having to build it on a tiny screen. Since the external can save these specific user interfaces they can be easily changed.

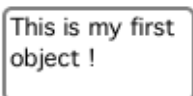
### creating

Objects are always given an ID. This ID is useful when you want to retrieve data back from the object and want to route its specific data. You have to specify each ID, they can however not contain spaces. If you want to create a textedit object for instance use the following syntax :

```
[textedit ID x y w h text]  
(note that the brackets indicate a message box in Max/MSP)
```

The first item indicates that you want to create a textedit object. The second item is the ID. Next are the x and y coordinates of the top/left corner of the given object and the width and height that should be used. In case of the textedit object the rest of the message is the initial text for the textedit object. The message to create the textedit object below is :

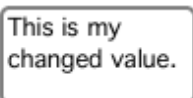
```
[textedit myTextedit 0 0 100 50 This is my first object !]
```



### setting values

You can set the value of certain object by using the set message on its ID.

```
[set myTextedit This is my changed value.]
```



### appearance

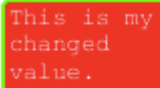
Color messages (colors are specified in rgb and alpha, values range from 0-1) :

```
[bgcolor myTextedit 1. 0. 0. 1.]  
[bordercolor myTextedit 0. 1. 0. 1.]  
[textcolor myTextedit 1. 1. 1. 1.]
```

Font messages :

```
[font myTextedit Courier 12]
```





```
This is my
changed
value.
```

Available fonts in the iOS are : Arial, Courier, AppleGothic, Helvetica, Verdana, TrebuchetMS, AmericanTypewriter, Zapfino and TimesNewRomanPSMT.

### **deleting**

To delete the textedit object above use the following message :

```
[delete myTextedit]
```

In order to clear the entire stage simply send :

```
[clear]
```

---

## fetching sensor data

---

c74 can pass lots of sensor data to Max/MSP. For instance it can send the iOS device's orientation, accelerometer, gyro, average microphone volume etc.

This is a list of sensors that can be queried by Max/MSP. To enable/disable each sensor send [sensor 1] or [sensor 0].

### gps

GPS coordinates are sent back in a list prepended by the string gps. The list outputs : longitude, horizontal accuracy (in meters), latitude, vertical accuracy (in meters), altitude.

### accelerometer

Output : x, y, z values prepended by the string accelerometer. Set the accelerometer's rate in Hz by sending [accelerometerRate 1].

### gyro

Output : x, y, z values prepended by the string gyro. Set the gyro's rate in Hz by sending [gyroRate 1].

### compass

Output : magnetic angle (in degrees), true angle (in degrees), accuracy (in degrees) prepended by the string compass.

### shake

When you enable shake the c74 external outputs a bang prepended by the string shake whenever you shake the iOS device.

### proximity

The proximity sensor will only work on iPhones. The proximity sensor is used to dim the iPhone's screen when in a conversation holding the phone against the ear. The sensor can be read but the dimming of the screen can't be disabled. The sensor is located at the top left corner of the phone and detects objects at about 1 cm.

The sensor outputs 1 or 0 prepended by the string proximity.

### microphone

Sends the microphone's detected volume as a float between 0 and 1 prepended by the string microphone.

---

## messages to the c74 app

---

There are a few messages c74 listens to :

[vibrate]	will vibrate the iOS device
[flashlight 1/0]	toggles flashlight
[batteryLevel]	get the iOS device's battery level
[batteryStatus]	gets the battery status (full, charging, unplugged)
[url http://url]	retrieves the content of the given url
[appstore]	link to the iTunes store or spam

### global settings

The triangle in the lower right corner in the c74 app will display a menu in which all of the sensors can be enabled/disabled and where the channel can be set.

If you use c74 on a device that is offered to other users (e.g. in a museum setting) you might want to get rid of the triangle to make sure no one can access the menu. This can be achieved by sending [menu 0] to disable the menu and [menu 1] to reenale the menu.

If you don't want the interface of the app to turn according to the orientation of the device use [rotate 0] and [rotate 1]. Note that the orientation of the iOS device is also sent to Max/MSP.

### attributes

Attributes may set the external's initial state. All the settings defined in the main menu can also be set using attributes. So for instance you can create the external as followed :

```
[c74 @channel 15 @gps 1 @shake 1]
```

### saving 'patchers'

The [write] message will open a dialog that allows you to save the current state of the c74 app to a file. This enables you to easily switch between 'patchers'. You can also use [write filepath] or [writeagain] messages. Files can be read using [read] or [read filepath].

---

## images

---

Images can be sent to the iOS device. Images that are sent to c74 are kept in memory until the application quits or upon sending the [delete imageID] message.

### displaying

Displaying a single image therefor requires you to first send the image using :

```
[image imageID test.jpg]
```

The image message is accompanied by an imageID, which you can set. Supported image formats are jpg, png, pdf and gif. The c74 external responds with imageID 1 when the image has been received by the iOS device. In order to display test.jpg you need to create an instance of the fpic object by sending :

```
[fpic myFpic imageID 10 10 100 100]
```

The above example will display test.jpg at location 10,10 with dimensions 100x100 pixels. Note that myFpic is the ID of the fpic object, use this variable to delete or change the image of the fpic object. The images are stored using their imageID's so they don't have to be resent each time you need an image more than once. Also you could send a bunch of images prior to displaying them at a later point.

## **sounds**

---

Playing and recording sounds on the iOS device within c74 is in an experimental stage. The global idea of using sounds is much like how images are treated. Sounds are stored according to an soundID you specify.

### **playing**

In order to play a sound first send the file to c74 using :

```
[sound soundID test.mp3]
```

The c74 external outputs soundID 1 when the sound has been received. Playing the sound file is done by :

```
[sfplay soundID]
```

Note that the sfplay message is not appended by an ~. Supported formats are aif, wav, caf, and mp3.

### **recording**

```
[sfrecord soundID 10 /Users/leovanderveen/Desktop]
```

Records sound on the iOS device for a specified duration on the desktop. For now c74 only records in the (unfortunately for Max/MSP unreadable) .caf format. The soundID can be set to whatever you like (no spaces allowed), the recording will be named after its soundID. In this case you'll find soundID.caf at my desktop.

Recorded sounds can be directly played back on the iOS device using the [sfplay soundID] message.

---

## objects

---

### button

syntax	[button ID x y size]
example	[button b 10 10 25]
commands	-
appearance	[bgcolor ID red green blue alpha]
returns	ID bang
notes	-

### toggle

syntax	[toggle ID x y size]
example	[toggle t 10 10 25]
commands	[set t state(int)]
appearance	[bgcolor ID red green blue alpha]
returns	ID state(int)
notes	-

### number

syntax	[number ID x y]
example	[number n 10 10]
commands	[set n value(int)]
appearance	[bgcolor ID red green blue alpha]
returns	ID value(int)
notes	-

### flonum

syntax	[flonum ID x y]
example	[flonum f 10 10]

commands	[set f value(float)]
appearance	[bgcolor ID red green blue alpha]
returns	ID value(float)
notes	-

#### comment

syntax	[comment ID x y value]
example	[comment c 10 10 I am comment.]
commands	[set c And you???)
appearance	[textcolor ID red green blue alpha] [font ID fontname fontsize] [font ID fontname]
returns	-
notes	-

#### message

syntax	[message ID x y value(list)]
example	[message m 10 10 I am message.]
commands	[set m And you???)
appearance	[bgcolor ID red green blue alpha] [textcolor ID red green blue alpha] [font ID fontname fontsize] [font ID fontname]
returns	ID value(list)
notes	-

#### slider

syntax	[slider ID x y height]
example	[slider s 130 210 60]
commands	[set ID value(int)]
appearance	[bgcolor ID red green blue alpha]
returns	ID value(int)

notes The value of the slider depends on the slider's height.

## hslider

syntax [hslider ID x y width]

example [hslider hd 130 210 60]

commands [set ID value(int)]

appearance [bgcolor ID red green blue alpha]

returns ID value(int)

notes The value of the slider depends on the slider's width.

## multislider

syntax [multislider ID x y width height numberofsliders]

example [multislider ms 10 10 300 300 10]

commands -

appearance [border ID state(int)]  
[bordercolor ID red green blue alpha]  
[slidercolor ID red green blue alpha]  
[bgcolor ID red green blue alpha]

returns ID index(int) value(int)

notes Multitouch enabled. Values depend on the multislider's height. Set function is not yet implemented in this version.

## kslider

syntax [kslider ID x y heightofkey numberofkeys]

example [kslider ks 10 10 42 12]

commands -

appearance -

returns ID note(int) velocity(int)

notes Multitouch enabled. Velocity increases when hitting the key at a higher point. Sliding from one key to another is currently not implemented.



## textedit

syntax	[textedit ID x y width height value(list)]
example	[textedit te 10 10 300 300 Click here to edit.]
commands	[set ID value(list)]
appearance	[border ID state(int)] [bordercolor ID red green blue alpha] [bgcolor ID red green blue alpha] [textcolor ID red green blue alpha] [font ID fontname fontsize] [font ID fontname]
returns	ID value(list)
notes	-

## umenu

syntax	[umenu ID x y width items(list)]
example	[umenu um 10 10 300 This\,that\,the other]
commands	[set ID items(list)]
appearance	[bordercolor ID red green blue alpha] [bgcolor ID red green blue alpha] [textcolor ID red green blue alpha] [font ID fontname fontsize] [font ID fontname]
returns	ID index(int) value(string)
notes	-

## xy

syntax	[xy ID x y width height]
example	[xy x 10 10 300 300]
commands	-
appearance	[border ID state(int)] [bordercolor ID red green blue alpha] [bgcolor ID red green blue alpha]
returns	ID finger(int) x(float) y(float) state(int)

notes                   Multitouch enabled. Sends values according to it's width and height values. The state value returns whether a finger is moved (1) or released (0).

## web

syntax                   [web ID x y width height]  
example                  [web w 10 10 300 300]  
commands                [set ID url(string)]  
appearance               -  
returns                   ID loadstate(int)  
notes                    Loads a webpage within a given rectangle on the iOS device.

## rslider

syntax                   [rslider ID x y width height]  
example                  [rslider myRangeSlider 10 10 300 25]  
commands  
appearance               -  
returns                   ID min(float) max(float)  
notes                    Range slider

## lcd

syntax                   [lcd ID x y width height]  
example                  [lcd mylcd 10 10 300 300]  
commands                [lcd mylcd clear]  
appearance               -  
returns                   ID x(float) y(float) state(int)  
notes                    Basic implementation of the lcd object, can be used for basic drawing on the iOS device.

## swatch

syntax                   [swatch ID x y width height]

example	[swatch myswatch 10 10 300 300]
commands	
appearance	-
returns	ID red(float) green(float) blue(float) alpha(blue)
notes	Swatch object, does not return properly yet...