

# KIOSC MANUAL



# Revision History

Revision	Date	Author(s)	Description
1	07.06.2018	ME	Initial version.
2	14.06.2019	BZ	Added Kiosc Touch chapter, added info on elements, expanded the index section
3	01.02.2021	BZ	Added Elements appendix
4	08.01.2023	JL	Added TCP function and details about feedback possibilities
5	23.06.2023	JL	1.16.13 new features update
6	04.07.2023	ME	CE & FCC declaration
7	07.08.2023	JF	1.17.18 new features update
8	10.07.2024	FL	Revised TCP functionality
9	17.12.2024	JL	Added maximum number of pages

# Contents

<b>1</b>	<b>Introduction</b>	<b>7</b>
<b>2</b>	<b>Kiosk App</b>	<b>10</b>
<b>3</b>	<b>Kiosk Touch</b>	<b>30</b>
<b>4</b>	<b>Kiosk Editor</b>	<b>38</b>
<b>5</b>	<b>File Locations</b>	<b>55</b>
	<b>Appendices</b>	<b>58</b>
<b>A</b>	<b>Elements</b>	<b>59</b>
<b>B</b>	<b>API</b>	<b>81</b>

©2024 Visual Productions BV. All rights reserved.

No parts of this work may be reproduced in any form or by any means - graphic, electronic, or mechanical, including photocopying, recording, taping, or information storage and retrieval systems - without the written permission of the publisher.

While every precaution has been taken in the preparation of this document, the publisher and the author assume no responsibility for errors or omissions, or for damages resulting from the use of information contained in this document or from the use of programs and source code that may accompany it. In no event shall the publisher and the author be liable for any loss of profit or any other commercial damage caused or alleged to have been caused directly or indirectly by this document.

Due to the dynamic nature of product design, the information contained in this document is subject to change without notice. Revisions of this information or new editions may be issued to incorporate such changes.

Products that are referred to in this document may be either trademarks and/or registered trademarks of the respective owners. The publisher and the author make no claim to these trademarks.



## Declaration of Conformity

We, manufacturer Visual Productions BV, hereby declare under sole responsibility, that the following device:

### **Kiosc Touch**

Conforms to the following EC Directives, including all amendments:  
EMC Directive 2014/30/EU

And the following harmonized standards have been applied:  
NEN-EN-IEC 61000-6-1:2019

The object of the declaration is in conformity with the relevant Union harmonisation Legislation.

Full name and identification of the person responsible for product quality and accordance with standards on behalf of the manufacturer

Date:  
November 24th, 2022

Place:  
Haarlem, The Netherlands



ing. Maarten Engels  
Managing Director  
Visual Productions BV

## SUPPLIER'S DECLARATION OF CONFORMITY (No. 2023/202-05)

*This Declaration of Conformity is issued under the sole responsibility of the manufacturer*

### MANUFACTURER

Company name	Visual Productions BV
Full address	Izaak Enschedeweg 38A 2031 CR Haarlem
Country	The Netherlands

### RESPONSIBLE PARTY – U.S. CONTACT INFORMATION

Company name	ACT Entertainment
Full address	3581 Larch Lane Jackson, MO 63755
Country	United States of America
Contact details (Phone)	+ 1 800 255 9822

### DESCRIPTION AND IDENTIFICATION OF THE EQUIPMENT

Generic denomination	Lighting control system
Function/Intended use	Remote controller for solid-state lighting controllers
Model	Kiosc Touch

*The object of the Declaration described above is in conformity with all relevant provisions of:*

**FCC (47 CFR Part 15B)**

### *Supplementary information*

This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation.



*Signed for and on behalf of:*

Place of issue: Haarlem  
The Netherlands

Identity:  
Function:

Maarten Engels  
Managing Director

Date of issue: 13-06-2023

Signature:

A handwritten signature in blue ink, appearing to read 'Maarten Engels', with a horizontal line underneath it.

# Chapter 1

## Introduction

Thank you for choosing Kiosc. Kiosc (see figure 1.1) is a simple app for creating custom user-interfaces. These user-interfaces are typically used for controlling lighting and AV equipment. Kiosc communicates with the external equipment via OSC, UDP and TCP; protocols that run over the Ethernet network.

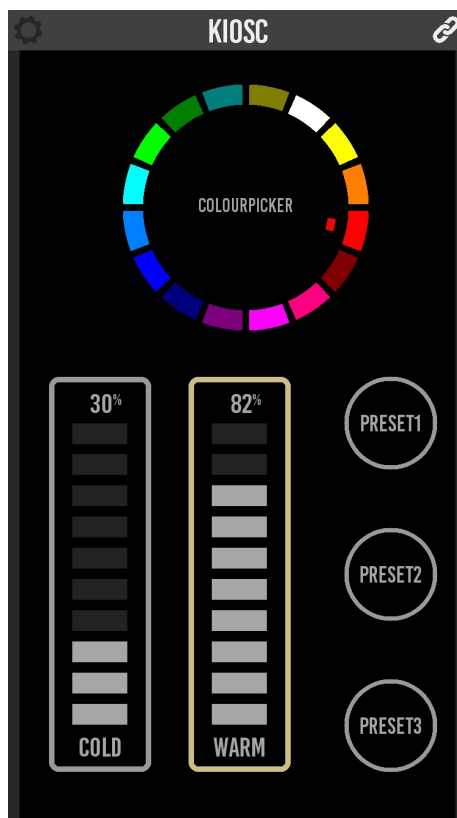


Figure 1.1: Kiosc

Kiosc is accompanied by the Kiosc Editor. This software tool enables you to design a simple control dashboard in a few minutes. While it also carries the features required to create complex multi-page interactive user-interfaces. Importantly, the Kiosc Editor allows you to personalise the dashboard; you are free to add logos, images and change colours.



Figure 1.2: Kiosc Editor

The Kiosc app works well with the Visual Productions range of lighting controllers, in fact, it is compatible with any OSC, UDP and/or TCP capable 3rd party device or software.

Kiosc is available on many operating systems: iOS, Android, Windows, macOS & Ubuntu. The Kiosc Editor is available on Windows, macOS & Ubuntu.



Figure 1.3: Platforms

This manual covers the functionality of the Kiosc app in chapter 2. The Kiosc Touch device is discussed in Chapter 3. Chapter 4 explains how to design custom layouts by using the Kiosc Editor. Finally, chapter 5 discusses the locations where the apps save and load the files on disk.

At the time of writing this manual the Kiosc Touch, App and Editor are at version 1.17.21.



## 1.1 Features

Kiosc has the following features.

- Generic control elements like buttons, sliders and check-boxes
- Lighting control elements like colour wheels and spectrums
- Feedback
- Multiple pages (up to 16)
- Available on 5 operating systems
- OSC, UDP and TCP
- PIN lock

## 1.2 Compatibility

Kiosc is compatible with the following devices.

- Visual Productions DaliCore
- Visual Productions CueCore3
- Visual Productions CueCore2
- Visual Productions CueCore1
- Visual Productions QuadCore
- Visual Productions IoCore2
- Visual Productions IoCore1
- Visual Productions Cuety LPU-2
- Visual Productions B-Station2
- Visual Productions B-Station1
- Visual Productions TimeCore
- Any OSC, UDP OR TCP compatible device

## 1.3 Further Help

If, after reading this manual, you have further questions then please consult the online forum at <https://forum.visualproductions.nl/> for more technical support.

## Chapter 2

# Kiosc App

The Kiosc app will present the customised dashboard to the user, typically running on a touch screen device. The Kiosc app itself has no editing capability in order to prevent the user from making accidental changes.

### 2.1 Modes

The Kiosc app operates in two modes, *Remote* and *File*.

### 2.1.1 Remote

The *Remote* mode serves two purposes. It allows you to quickly connect to a Visual Productions device and control it directly. Secondly, in this mode, Kiosc can connect to a Kiosc Editor and receive live updates as the user works on the design. This way it provides a real-time preview on the actual target device. Please note that due to Microsoft Store security measurements, it is not possible to make a remote connection to Kiosc Editor if Kiosc and Kiosc Editor are running on the same PC. To transfer a showfile, it can be manually copied using the filebrowser. More information can be found in chapter 5.

Although quick to set up, a *Remote* connection to a CueCore2 or similar device does not enable the user to make elaborate designs. The editing capabilities inside the CueCore2 are basic. In order to create more demanding user-interface designs it is recommended to use the *File* mode.

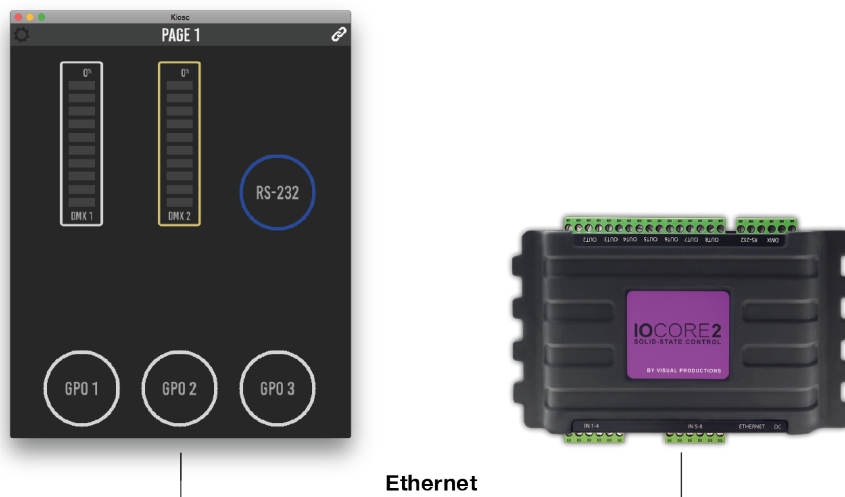


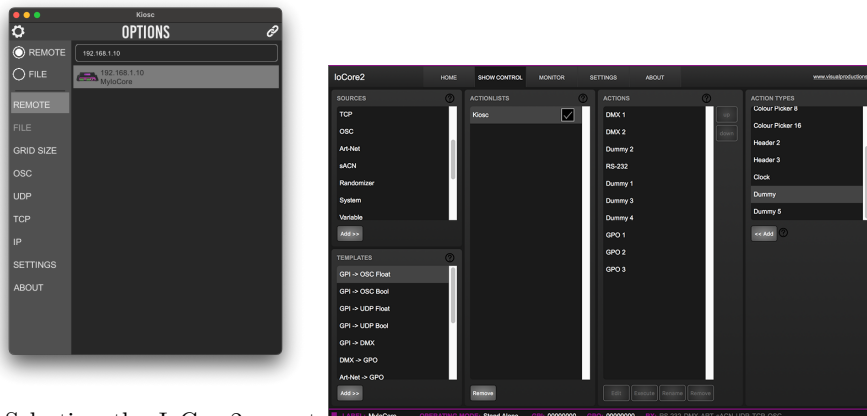
Figure 2.1: Kiosc connected to an IoCore2

Kiosc can only be connected to a single external device when using *Remote* mode. It is using a propriety protocol that is only supported by the Visual Productions range of devices. This protocol is uni-directional and thus provides no means for giving feedback to the Kiosc app. No OSC or UDP is used in this mode.

The *Remote* mode makes Kiosc compatible with VisualTouch, the previous-generation touch screen software from Visual Productions.

Figure 2.1 shows an example of Kiosc being connected to an IoCore2. By using the *Remote* mode, a simple dashboard can be created in a few mouse clicks. In this example Kiosc is used for setting DMX values, transmitting a RS-232

message and toggling GPO relays.



(a) Selecting the IoCore2 remote server

(b) Actions inside the IoCore2

Figure 2.2: Connecting Kiosc to an IoCore2

Figure 2.2 shows how to select the IoCore2 in the remote options. Once connected, the dashboard is created by adding a 'Kiosc' actionlist inside the IoCore's show control page. Buttons, sliders and other controls can be added to this actionlist. The order of the actions determines the position of the controls in the Kiosc screen. Empty positions can be created adding *Dummy* actions. If there are more controls to fit in the screen, then extra pages will be created automatically. How many controls do fit in the screen is dictated by the *Grid Size* setting.

Figure 2.2 displays the actions created for this example. Programming actions is discussed in more detail in the IoCore2's user manual.

### 2.1.2 File

In *File* mode Kiosk renders a layout designed with the Kiosk Editor. The file generated by the Editor needs to be stored locally on the device running Kiosk.

The Editor enables you to make custom control screens and to personalise them with logos and colours.

In *File* mode it is possible to control multiple external devices on the network. By using OSC, UDP or TCP these devices can also send data back to the Kiosk app, providing it with feedback that could update the status of GUI controls.

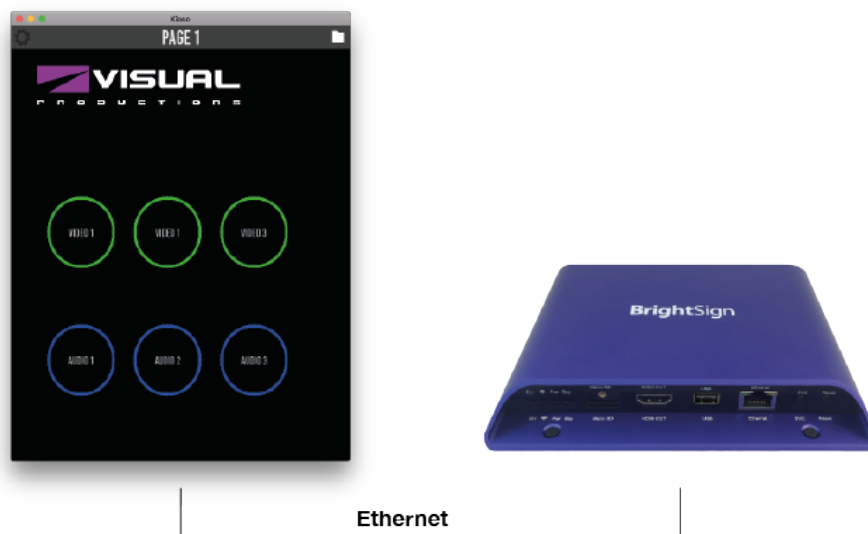


Figure 2.3: Kiosk connected to a media player

Figure 2.3 shows an example of Kiosk controlling a Brightsign, a 3rd party media player. Kiosk is using UDP messages to trigger the playback of different media files inside the Brightsign.

## 2.2 Protocols

Kiosk can communicate with external equipment via three protocols, OSC, UDP and TCP. All protocols run over the Ethernet network. This chapter describes these protocols.

### 2.2.1 OSC

Open Sound Control (OSC) is a protocol for communicating between software and various multi-media type devices. OSC uses the network to send and receive messages.

### 2.2.2 UDP

User Datagram Protocol (UDP) is a simple protocol for sending messages across the network. It is supported by various media devices like video projectors and Show Controllers. It does not incorporate error checking, making it an easy and fast protocol.

### 2.2.3 TCP

Transmission Control Protocol (TCP) is a secured protocol for sending messages across the network. It is supported by various media devices like video projectors and Show Controllers. It creates a secured connection link between the sender and the receiver of the message and incorporates error checking. This makes it slower than UDP.

### 2.2.4 OSC vs UDP

There are many similarities between OSC and UDP. In fact, OSC uses UDP packets under the hood.

There is one difference between these protocols that is significant when choosing the right protocol for the job. OSC has a fixed syntax and UDP does not. Although UDP's technical characteristics are specified in the protocol, the syntax is not. This means that two UDP capable equipment from two different vendors will be able to send each other technically correct UDP messages. What they write into the messages is left to the discretion of the developers. As it turns out, different developers come up with different syntaxes. Even though the two devices say 'UDP' on the box, they might not be able to speak to each other.

This problem can be illustrated by an example. Let's assume a device from vendor X wants to send a message to a device from vendor Y. The message should set a parameter inside device Y to 50%. Device X could send something like:

- `parameter=50%` assuming a range from 0% to 100%
- `parameter=0.5` assuming a range from 0.0 to 1.0
- `parameter=128` assuming a range from 0 to 255
- `parameter=0x80` assuming a single hexadecimal byte

There are even more variations possible. The point being that whatever syntax vendor X has chosen, the chance that vendor Y choose the same is not that great.

OSC solves this problem by incorporating rules on syntax in its protocol specification. Each OSC compatible device uses exactly the same syntax. This is a strong reason to prefer OSC over UDP when possible. Of course, there are many devices out there that do not support OSC, leaving UDP an important asset inside your protocol toolbox.

### **2.2.5 UDP vs TCP**

UDP and TCP are using the same message syntax. Thanks to the error checking TCP is more reliable than UDP and allows the sender to know if the message has really been received by the recipient. This is not possible with UDP, but that makes it faster. As a communication link must be established between the sender and the recipient, it makes broadcast unavailable for TCP compared to OSC and UDP where you can easily broadcast messages.

## **2.3 Options**

The options screen presents a few settings for customising the Kiosk behaviour.

### **2.3.1 Modes**

The modes selector allows you to choose between *Remote* and *File* mode. Modes are discussed on page 10.

### 2.3.2 Remote

This menu (see figure 2.4) is only enabled when in *Remote* mode. It allows you to select the remote server. E.g. a CueCore or Kiosc Editor. Please note that due to Microsoft Store security measurements, it is not possible to make a remote connection to Kiosc Editor if Kiosc and Kiosc Editor are running on the same PC.

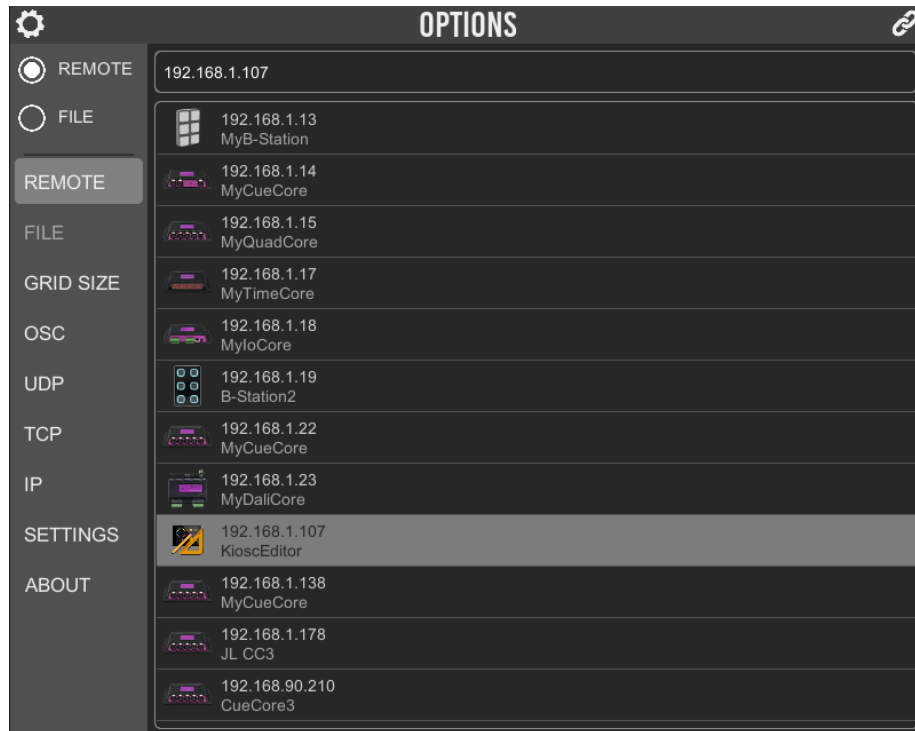


Figure 2.4: Remote options



### 2.3.3 File

This menu is only enabled when in *File* mode. It presents a list of the layout files. It also has buttons for saving, deleting and renaming files.

The files can be locked and unlocked by long pressing them. When locked, they can not be modified.

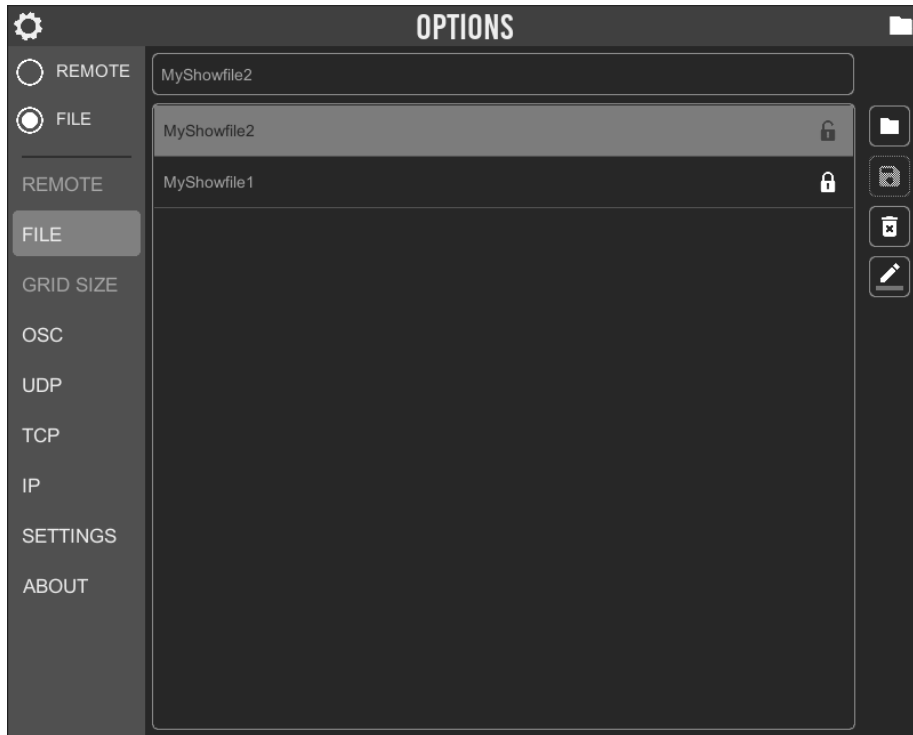


Figure 2.5: File options

The *Folder* button opens a file browser at the path where the files are stored.

### 2.3.4 Grid Size

In *Remote* mode (except when connected to *Kiosc Editor*) the size of the buttons and other controls are determined by the *Grid Size* setting. A smaller *Grid Size* results in larger controls. A larger *Grid Size* will allow more controls to fit on one page.

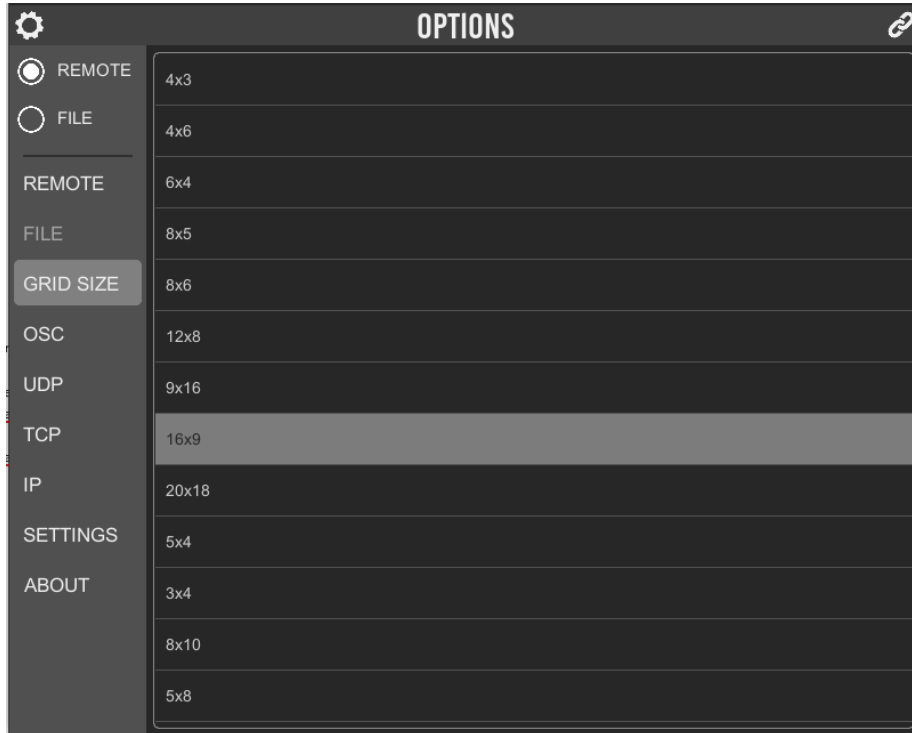


Figure 2.6: Grid Size options

The *Grid Size* menu is only available in *Remote* mode.

### 2.3.5 OSC

The settings in this screen (see figure 2.7) determine to which IP addresses the OSC messages are sent to. Multiple IP addresses can be added to the list. With each IP address there also needs to be specified a port number. Add the port number in the following fashion: `XXX.XXX.XXX.XXX:PORT`. Where the XXXs denote the IP address and PORT is a number between 1 and 65535.

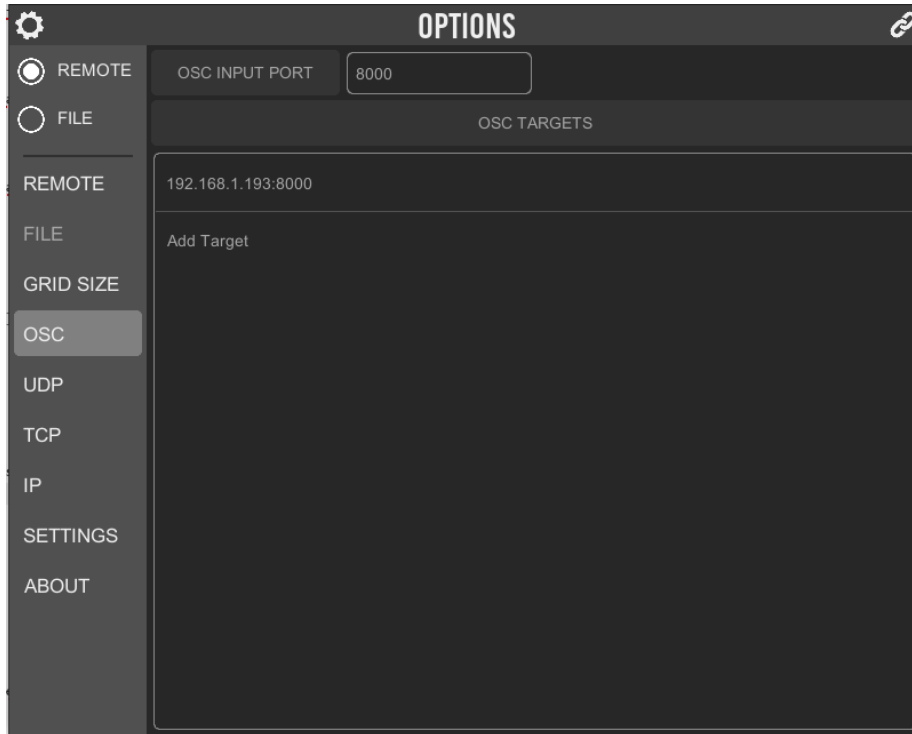


Figure 2.7: OSC options

The incoming port field specifies the port number on which Kiosc will receive OSC messages. External equipment that will send feedback information to Kiosc needs to know this incoming port number. By default this port number is set to 8000.

### 2.3.6 UDP

The settings in this screen (see figure 2.8) determine to which IP addresses the UDP messages are sent to. Multiple IP addresses can be added to the list. With each IP address there also needs to be specified a port number. Add the port number in the following fashion: `XXX.XXX.XXX.XXX:PORT`. Where the XXXs denote the IP address and PORT is a number between 1 and 65535.

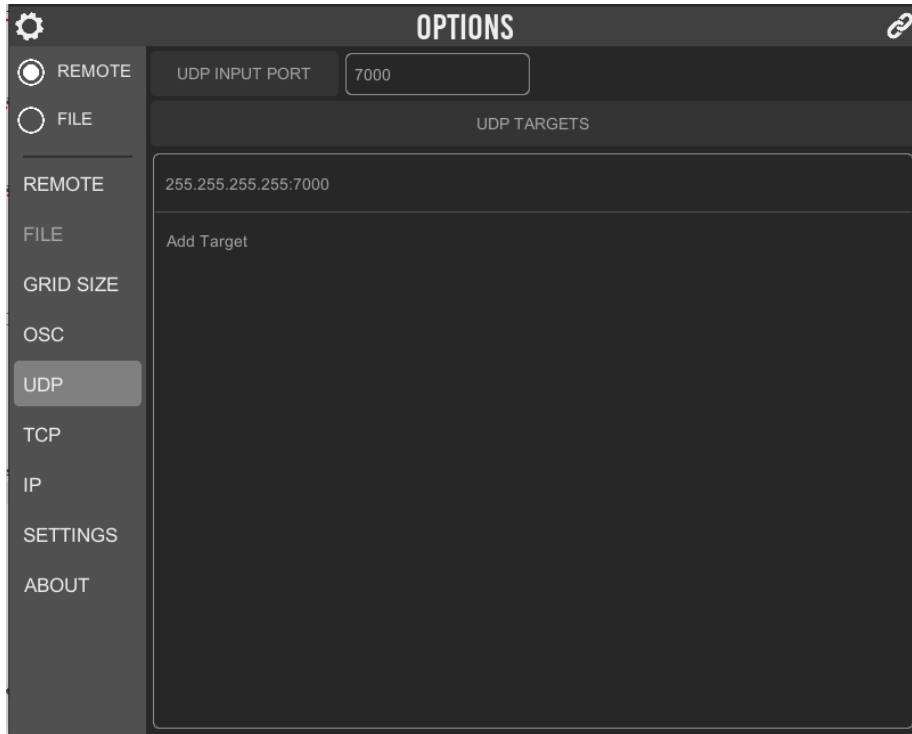


Figure 2.8: UDP options

The incoming port field specifies the port number on which Kiosk will receive UDP messages. External equipment that will send feedback information to Kiosk needs to know this incoming port number. By default this port number is set to 7000.

### 2.3.7 TCP

The settings in this screen (see figure 2.9) determine to which IP addresses the TCP messages are sent to. Multiple IP addresses can be added to the list. With each IP address there also needs to be specified a port number. Add the port number in the following fashion: `XXX.XXX.XXX.XXX:PORT`. Where the XXXs denote the IP address and PORT is a number between 1 and 65535.

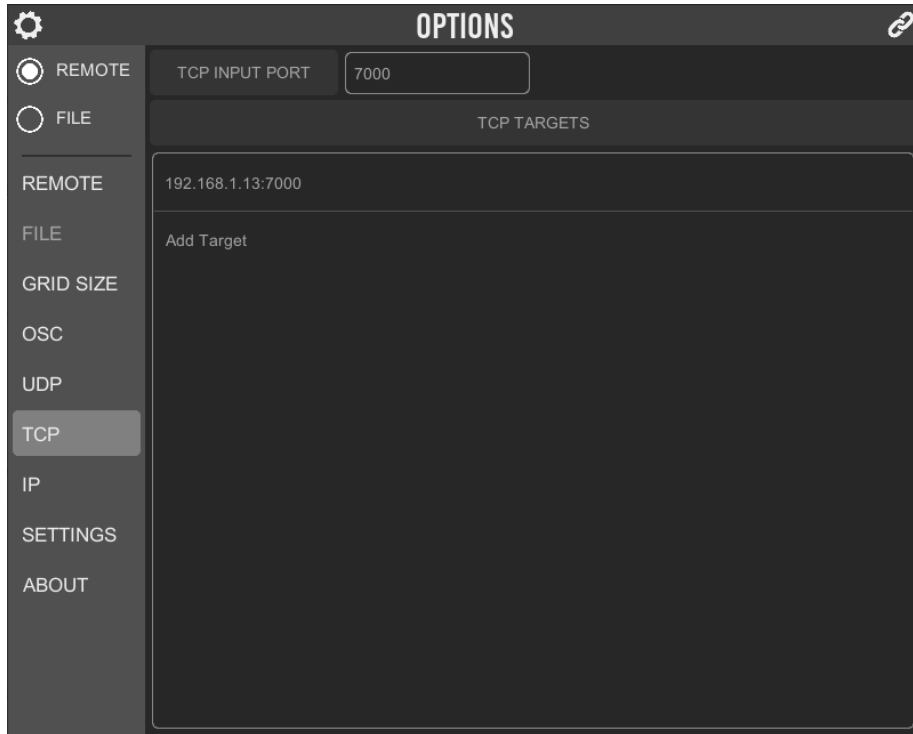


Figure 2.9: TCP options

The incoming port field specifies the port number on which Kiosc will receive TCP messages. External equipment that will send feedback information to Kiosc needs to know this incoming port number. By default this port number is set to 7000.

## 2.3.8 Settings

### Language

You can easily change the language used by the Kiosk choosing one of the languages available in the list. See figure 2.10.

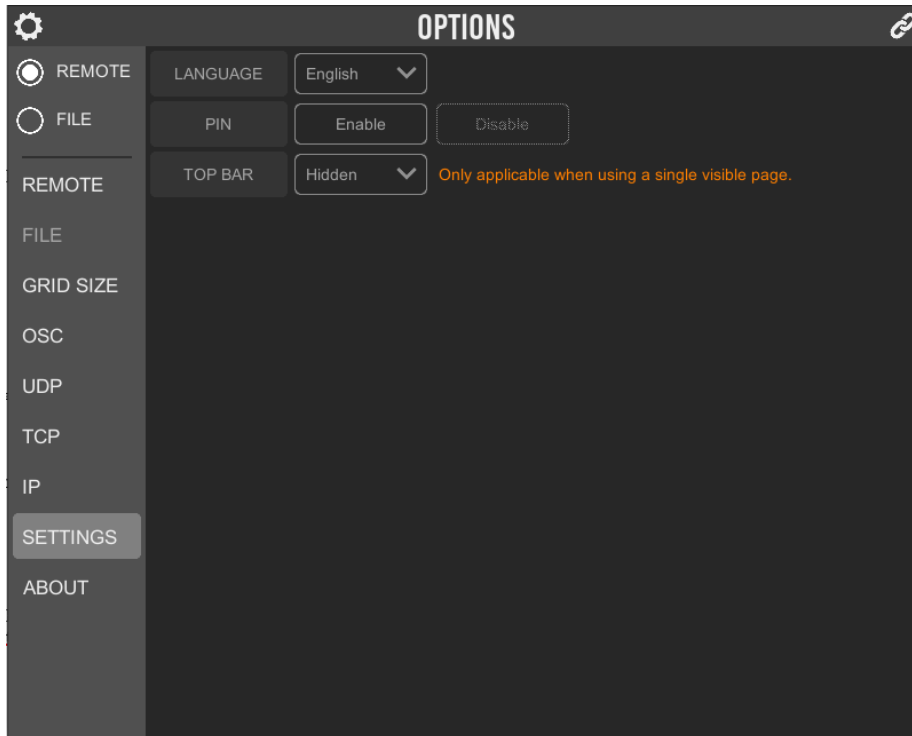


Figure 2.10: Changing settings

### PIN

Although the Kiosk app has no editing capabilities (all editing is done in the Kiosk Editor), it does have several settings that, if erroneously changed, can stop the system from working. To restrict access to the settings a PIN code can be engaged. See figure 2.10.

## Top Bar

The page selection top bar can be set as visible or invisible in the layout. Note that it is only possible to hide it when you have only one visible page in your layout.

When the top bar is hidden, the *gear* icon for accessing the settings will still be available on the top right corner. See figure 2.10.

The *Settings* page of the Kiosk Touch displays some extra information compared to the the Kiosk App, as shown below.

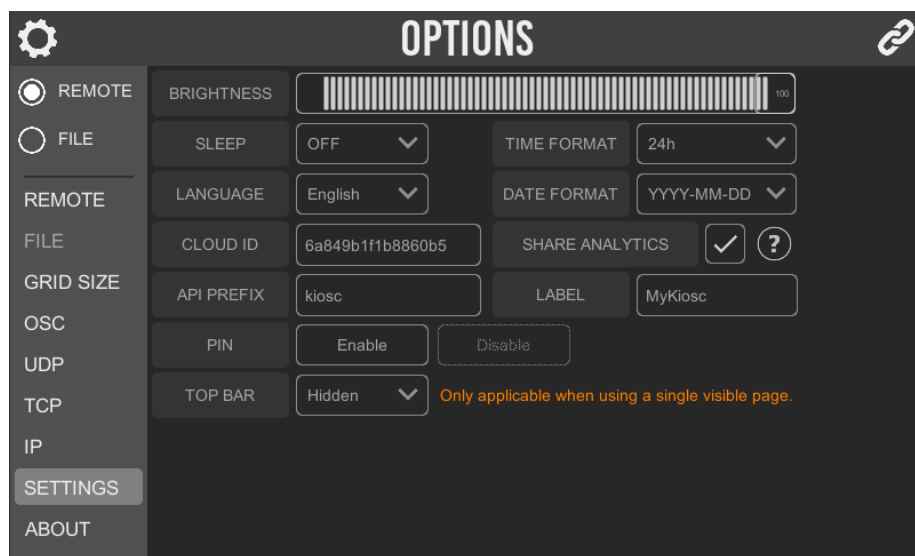


Figure 2.11: Kiosk Touch settings

## Brightness

Set the screen's brightness of the Kiosk Touch. See figure 2.10.

## Sleep

Set the time before the Kiosk Touch's screen goes into sleep mode. In sleep mode, the screen turns dark to avoid useless brightness and save energy. Just touch once the screen to wake up the Kiosk Touch. 2.10.

## Cloud ID

The Kiosk Touch, can be remotely supervised using the Purple Cloud you will find on the website <http://www.visualproductions.nl/>. Once registered in the Purple Cloud, a unique ID that will be issued allows to link Visual Productions devices to the cloud platform. See figure 2.10.

### **API Prefix**

Kiosc Touch, include API commands to directly control some functions via OSC, UDP or TCP. See API commands list in the appendix `refsec:api`. Those commands start with a prefix that is set to *kiosc* by default. When using multiple devices from Visual Productions it can be useful to assign unique labels to these prefix, especially when using broadcasted messages. Read more about feedback loops in paragraph B.3.1. See figure 2.10.

### **Time Format**

Set the format in which Kiosc Touch is going to represent time. There are two options between 24h or AM/PM.

### **Date Format**

Set the format in which Kiosc Touch is going to represent date. There are four different options.

### **Share Analytics**

If enabled, the unit is going to share data to connect to Visual Production's cloud platform: Purple Cloud.

### **Label**

Displays and lets the user change the unit's name.

## **2.3.9 IP**

The IP page differs between the Kiosc App and the Kiosc Touch, as shown below.



## Kiosc App

On the Kiosc App the IP page shows a list of local IP addresses used by the device, as shown in figure 2.12. This information can be useful when trouble shooting network issues. The network settings are handled by the Operating System of the device running the Kiosc App.

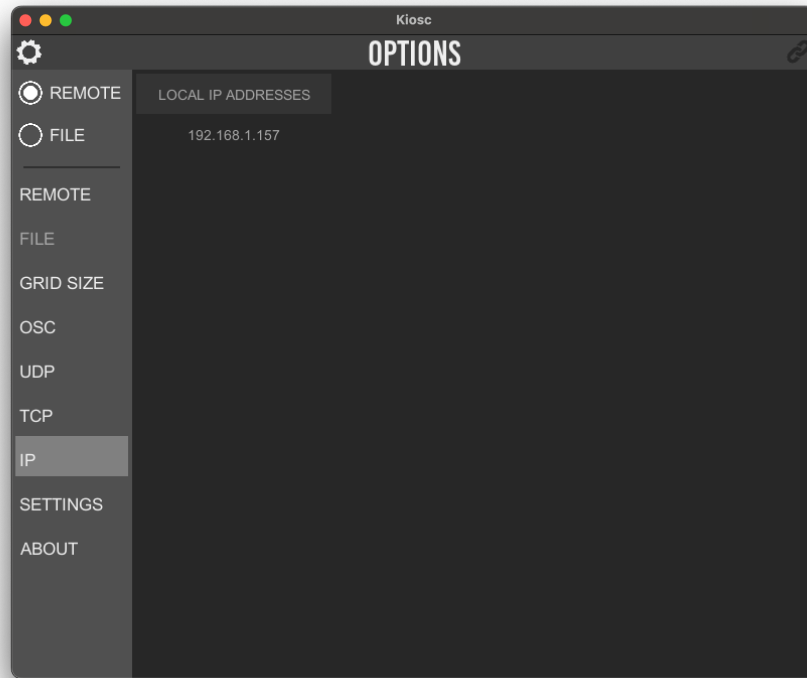


Figure 2.12: IP page on the Kiosc App

## Kiosc Touch

The IP page on the Kiosc Touch is used for setting the desired network settings, see figure 2.14. A static IP can be set, or the DHCP option can be checked for automatic configuration.

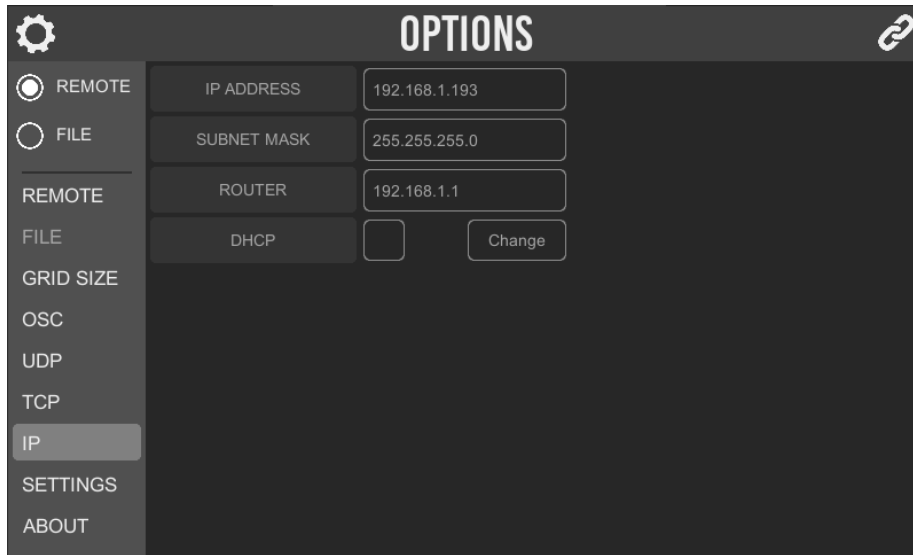


Figure 2.13: IP page on the Kiosc Touch

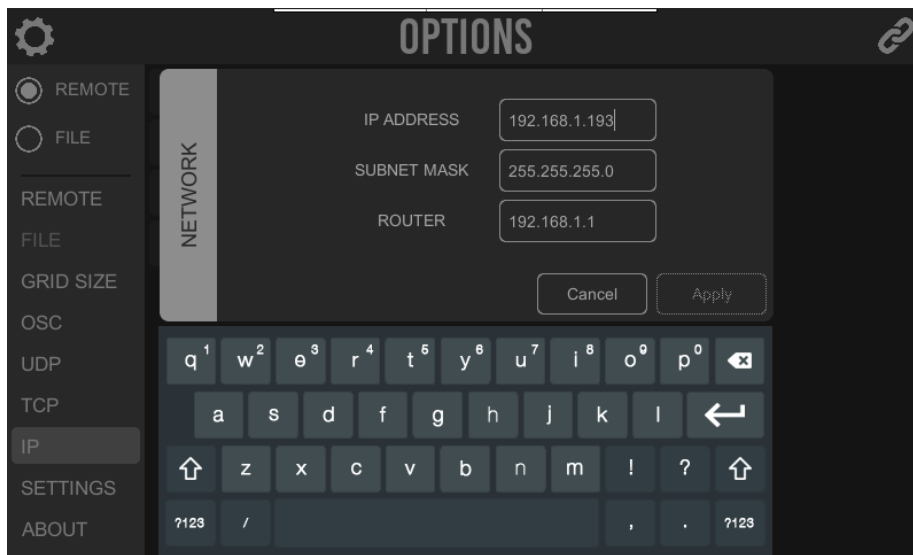


Figure 2.14: IP dialog on the Kiosc Touch

### 2.3.10 About

Apart from the ubiquitous copyright notice, the about screen in figure 2.16 indicates the current software version.

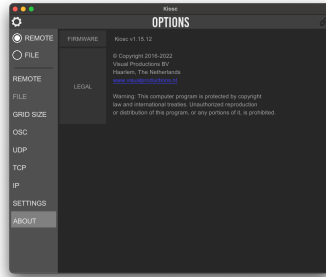


Figure 2.15: KIOSC app About screen

The *About* page of Kiosc Touch shows some complementary information about your device compare to the Kiosc app version. You will be able to find:

- Serial number
- PCB version
- Label (you can change it via vManager or in Settings page)
- Uptime since last power cycle.

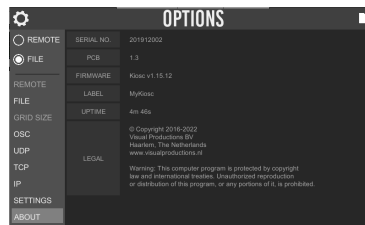


Figure 2.16: Kiosc Touch About screen

## 2.4 Installation

The Kiosc app is available on a wide range of operating systems, both mobile and desktop.



Figure 2.17: Kiosc on macOS, Windows and Ubuntu

The software is distributed through app-stores; there are no stand-alone installation packages (e.g. EXE, APK, DMG or DEB files) available. One of the major advantages of installing an app via an app store is to receive future software updates automatically.

- **iOS**

Kiosc can be downloaded from the Apple iOS app-store at:  
<https://itunes.apple.com/us/app/kiosc/id1131605366>

- **Android**

Kiosc can be found on the Google Play store at  
<https://play.google.com/store/apps/details?id=nl.visualproductions.kiosc>

Android 6.0 or higher is required.

- **Windows**

Kiosc can be downloaded from the Microsoft store at:  
<https://www.microsoft.com/store/apps/9nblggh4s74f>

Windows 10 is required.

- **macOS**

Visit the Apple macOS app store at:  
<https://apps.apple.com/us/app/kiosc/id1131605497>

macOS 11+ is recommended.

- **Ubuntu**

You can acquire the Kiosk app from the snap store:

<https://snapcraft.io/kiosc>

Alternatively, it can be installed by using the command-line:

```
snap find kiosc  
snap install kiosc
```

To update the apps later on via the command-line type:

```
snap refresh kiosc
```

Ubuntu 22.04 LTS is recommended. The app is only available for the amd64 architecture.

## Chapter 3

# Kiosc Touch

The Kiosc Touch is a Touch screen solution created by Visual Productions. It is a standalone PoE-powered device running the Kiosc app.



Figure 3.1: Kiosc Touch

### 3.1 Compliance

This device is in compliance with CE, UKCA and FCC regulations.

### 3.2 Specifications

The Kiosc Touch has the following specifications.

- PoE (Power-over-Ethernet) Class III.

- 7 Inch full colour display
- Viewable screen size: 155mm x 86mm
- Capacitive multitouch
- Resolution of 800 x 480 pixels
- Wall mountable
- VESA 75
- 460 gram (1 lbs)
- Operating temperature -20°C to +50°C (-4°F to 122°F)
- Compliance EN55103-1 EN55103-2

### 3.3 Mounting

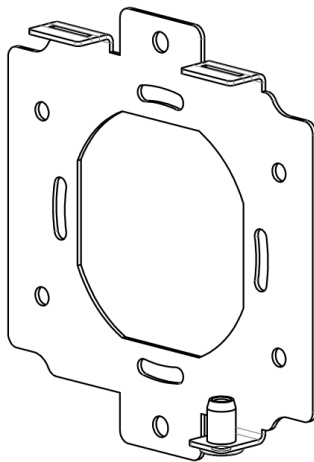


Figure 3.2: Kiosk Touch mounting bracket

The Kiosk Touch is wall-mount, it is compatible with European and American gang boxes. If no gang box is available then the Kiosk Touch can also be surface mounted. The mounting bracket of the Kiosk Touch suggests where you can create a hole for the cable to enter.

### 3.4 vManager

A free-of-charge software tool called vManager has been developed to manage the devices. vManager allows for:

- Discovering any Kiosk on the Network
- Perform firmware upgrades

- Set the date and time of the internal clock
- Identify the unit thanks to the blink function
- Set the unit to its default settings
- Backup/restore the unit

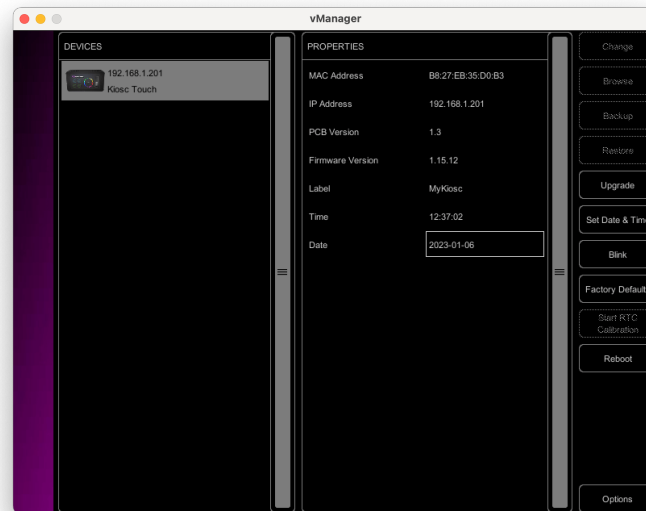


Figure 3.3: vManager

## 3.5 Network

The Kiosc Touch is a network device. A network connection between a computer running Kiosc Editor and the unit is required to program the Kiosc Touch. More information on this can be found in chapter 4.

There are multiple possible arrangements for connecting the computer and the Kiosc Touch. They can be connected peer-to-peer, via a network switch or via a Wi-Fi bridge device. Figure 3.4 illustrates these different arrangements.



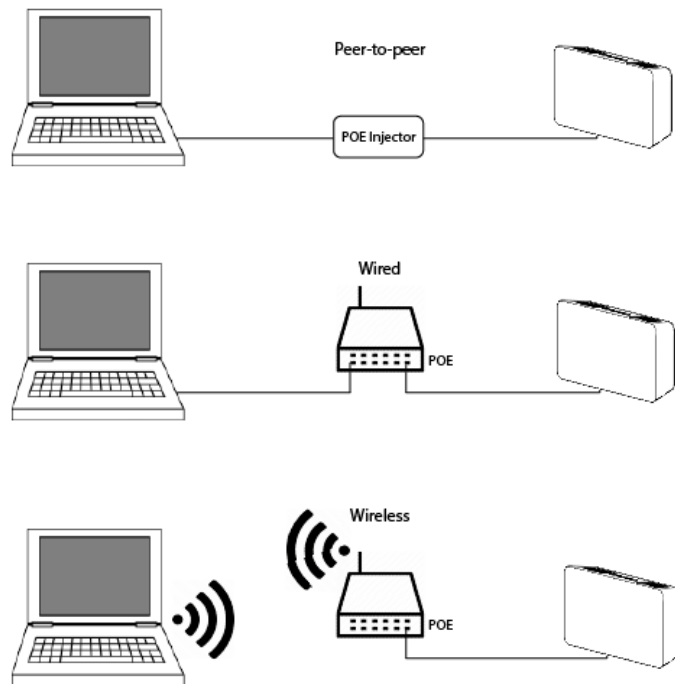


Figure 3.4: Network arrangements

The Ethernet port on the Kiosk Touch is auto-sensing; it does not matter whether a cross or straight network-cable is being used.

### 3.6 IP Address

The Kiosk Touch supports both static IP addresses and automatic IP addresses. By default, the Kiosk Touch is set to DHCP in which it will automatically be assigned an IP address by the DHCP server in the network. The 'DHCP server' is typically part of the router's functionality.

Static IP addresses are useful when there is no DHCP server in the network, for instance when there is a direct peer-to-peer connection between a Kiosk Touch and a computer. It is also useful in permanent installations where the IP address of the Kiosk Touch is known by other equipment and therefore should not change. When using DHCP there is always the risk of automatically being given a new IP address in the event that the DHCP server is replaced. When using static IP addresses make sure that all equipment on the network have unique IP addresses.

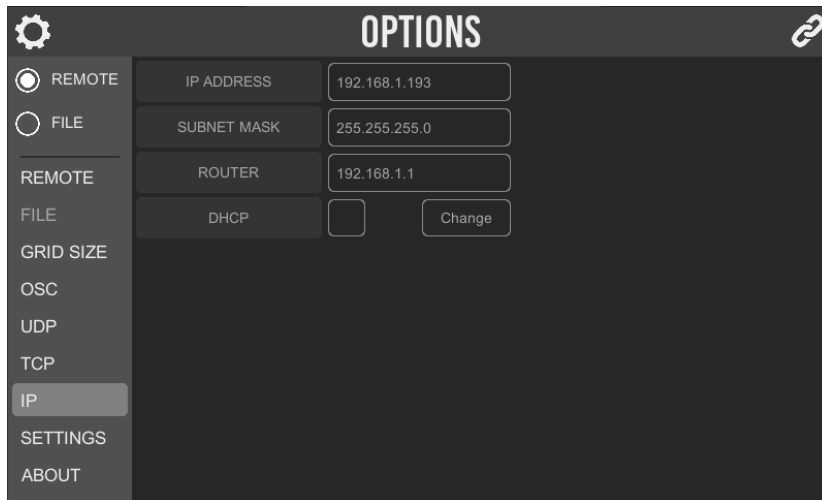


Figure 3.5: IP Settings

The IP address or DHCP mode of the Kiosk Touch can be set in the 'IP' tab in the settings menu, as shown in figure 3.5. The 'IP Address' field is also where the current IP is shown.

Note: This only applies to the Kiosk Touch, on the Kiosk App the network settings are configured using the operating system of the device running Kiosk.

### 3.7 Web Interface

The Kiosk Touch has a web interface that can be accessed via vManager, or via browser if typing the unit's IP.

### 3.7.1 Home

All the unit's specific information.

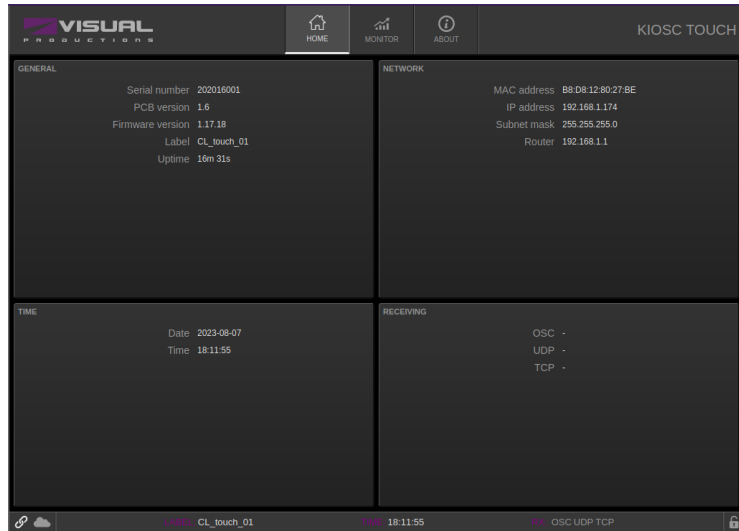


Figure 3.6: Web Interface home page

### 3.7.2 Monitor

Displays all the incoming and outgoing messages.

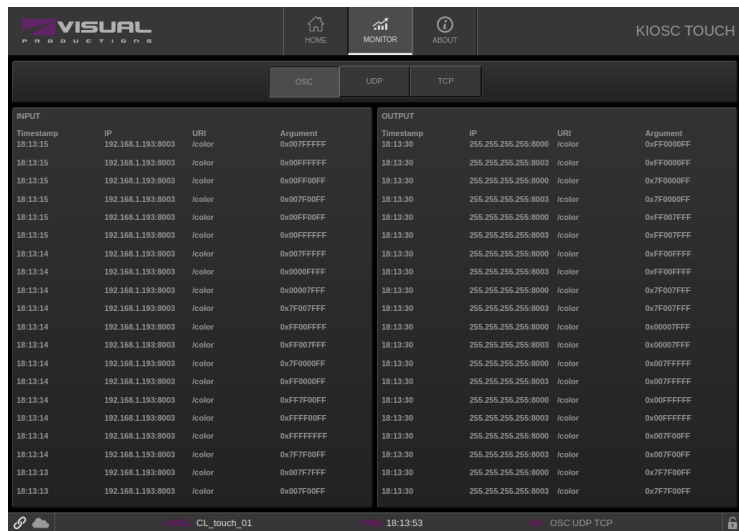


Figure 3.7: Web Interface monitor page

### 3.7.3 About

About information.

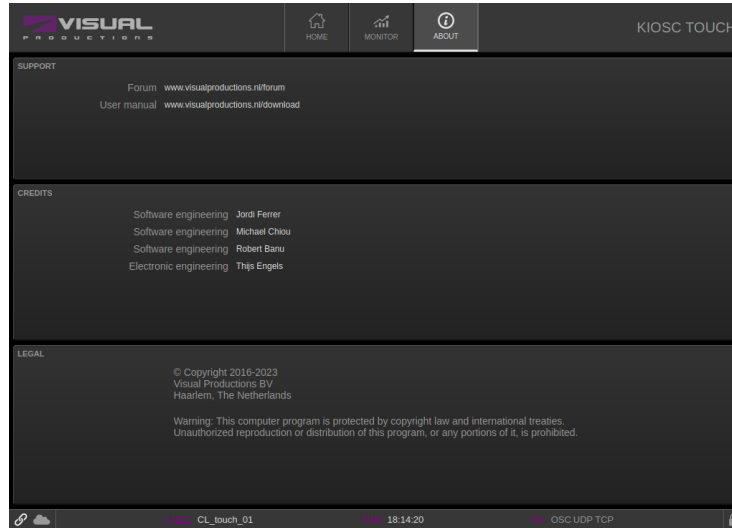


Figure 3.8: Web Interface about page

## 3.8 Cloud Interface

The Kiosc Touch can be linked with a *Purple Cloud* account. From there, the same information displayed in a local web interface can be remotely accessed.

To link the unit, copy the Cloud ID found on the Account section and paste it into Settings page. See figure 3.9

Communication between the device and *Purple Cloud* is AES 256 CBC encrypted to ensure data transmission security.

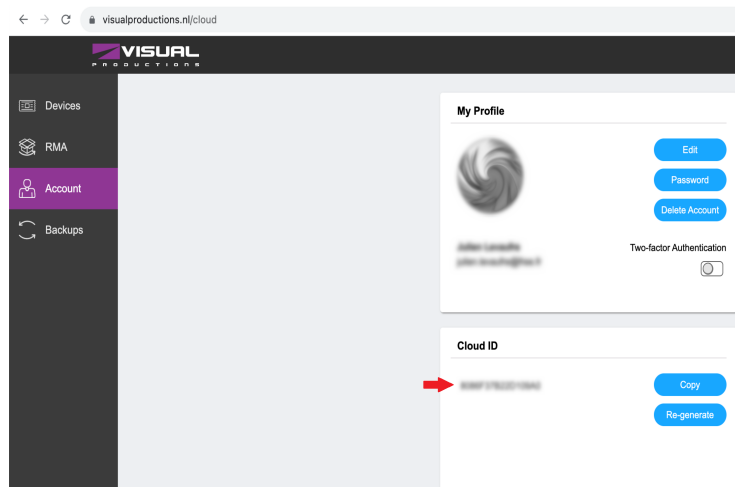


Figure 3.9: Cloud ID

# Chapter 4

## Kiosc Editor

The Kiosc editor is a software tool for designing Kiosc layouts.

### 4.1 Installation

The Editor is available on Microsoft Windows, Apple macOS and Ubuntu Linux. The software is distributed through app-stores; there are no stand-alone installation packages (e.g. EXE, APK, DMG or DEB files) available. One of the major advantages of installing an app via an app store is to receive future software updates automatically.

- **Windows**

Kiosc Editor can be downloaded from the Microsoft store at:  
<https://www.microsoft.com/store/apps/9n4ph6cs6tth>.

Windows 10 is required.

- **macOS**

Visit the Apple macOS app store at:  
<https://apps.apple.com/us/app/kiosc-editor/id1257068566>

macOS 11+ is recommended.

- **Ubuntu**

You can acquire Kiosc Editor from the Snapcraft store:  
<https://snapcraft.io/kiosceditor>

Alternatively, it can be installed by using the command-line:  
`snap find kiosceditor`  
`snap install kiosceditor`

To update the apps later on via the command-line type:  
`snap refresh kiosceditor`

Ubuntu 20.04 LTS is recommended. The app is only available for the amd64 architecture.

## 4.2 The Editor

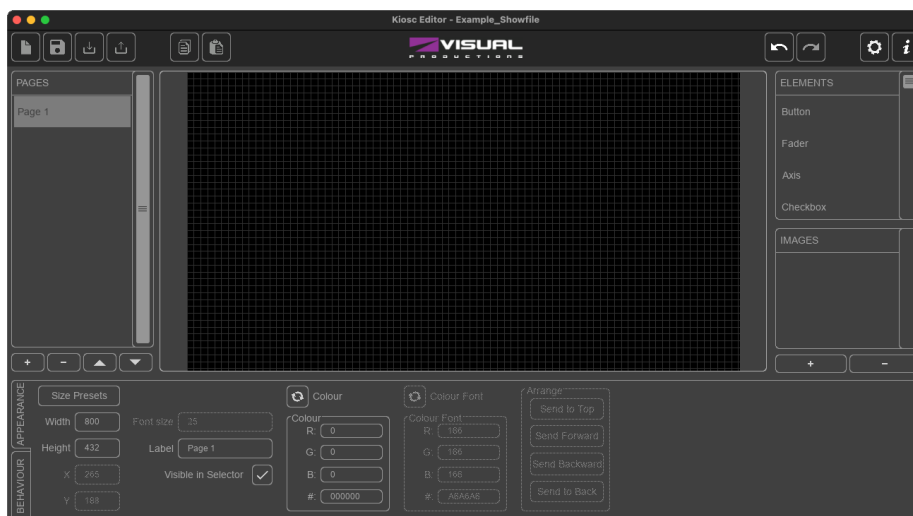


Figure 4.1: Kiosc Editor user-interface

Figure 4.1 shows the Editor’s user-interface. Each section is discussed in detail below.

### 4.2.1 Menu Bar

At the top of the Editor screen there is a menu bar. Reading from left to right, it contains buttons for New File, Save, Save As and Open. There are clipboard buttons for Copy and Paste. Next buttons are Undo and Redo. Finally, there are buttons to open the Options and About dialogs.



Figure 4.2: Menu bar

### 4.2.2 Pages

Your design needs at least one page and can include up to 16 pages. Add more pages by pressing the + button in figure 4.3. The *Up* and *Down* buttons allow you to change the order of the pages.



Figure 4.3: Pages

You can rename the page by selecting it and then changing the *Label*.



### 4.2.3 Elements

Controls like buttons, faders, encoders, and others are called *Elements* (see figure 4.4). A layout is basically created by dragging several *Elements* into the *Pages*.

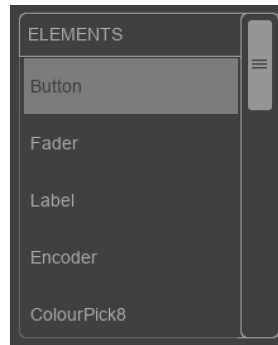


Figure 4.4: Elements

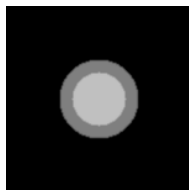
More information on each element can be found in appendix A. The available Elements include:

#### Button



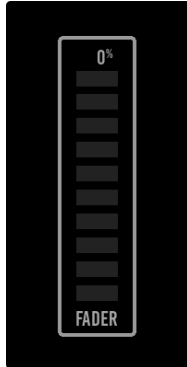
The button can be used to send a bool value (ON/OFF) or integer value (0 or 1). When set to Control, the Button sends ON or 1 when pressed, and OFF or 0 when released. Using Set the Button can send any integer value when pressed.

#### Indicator



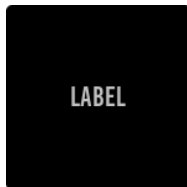
The indicator can be used as a virtual LED. It does not send anything, but using feedback the inner circle can light up and change colour.

### Fader



The fader can send a range of values. The range can be set using the Min and Max values. The type of value can be set to a float or an integer. A fader can simply be resized to get a horizontal fader. Displayed value can be changed between numeric value and percentage.

### Label



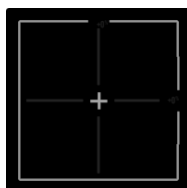
The label element is for creating a clear, easy to use layout. It has no function when pressed.

### Encoder



The Encoder sends a float or integer value. Minimum, maximum and default value can be set.

### Axis



The Axis element is for controlling XY values, like a moving head's pan and tilt.

### Colourpicker 8



The Colourpicker 8 has eight colours to choose from. These are sent as OSC colour value, or via UDP/TCP using Hex #RRGGBB.

### Colourpicker 16



The Colourpicker 16 has sixteen colours to choose from. These are sent as OSC colour value, or via UDP/TCP using Hex #RRGGBB.

### ColourSpectrum



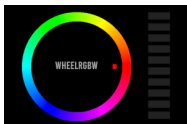
The Colourspectrum has many colours to choose from. These are sent as OSC colour value, or via UDP/TCP using Hex #RRGGBB.

### Wheelcolour



The Wheelcolour has many colours to choose from. These are sent as OSC colour value, or via UDP/TCP using Hex #RRGGBB.

### WheelRGBW



The WheelRGBW has many colours to choose from, which can be mixed with the fader that controls the white value. The colour is sent as OSC colour value (abusing the alpha channel for white) or via UDP/TCP using Hex #RRGGBBWW.

### WheelColTemp



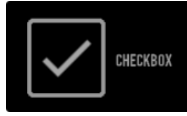
The WheelColTemp is ideal to mix between warm and cold white. It sends a float value between 0.00 and 1.00

### Text



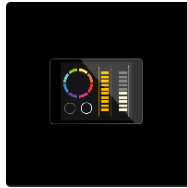
The text element is a simple text box. You can type text here and send it as a OSC or UDP/TCP message. It can also receive and display text.

### Checkbox



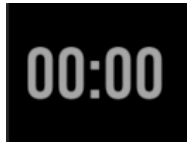
A checkbox element can be used as a button. It will send ON or 1 when checked, and OFF or 0 when unchecked.

### Image



The image element is perfect for adding your own logo to your Kiosk layout. It behaves similar to the button element, so it can be used to create custom buttons.

### Clock



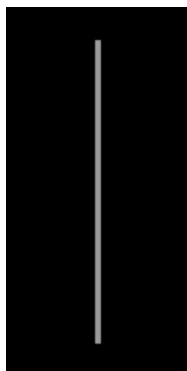
The clock element displays the current time of the system. It has no other functions when pressed.

### Date



The date element displays the current date of the system. It has no other functions when pressed.

### Line



The line element is used purely for design, it does not send any commands but is ideal as a separator between controls. It can be turned into an horizontal line by changing the size ratio. It can change colour using feedback.

## 4.2.4 Images

A *Page* can have a background image. Also, *Image Element* can display an image. To use images in your design, you first have to import them into the images list (see figure 4.5). Add images by pressing the + button. Images are read from a specific folder. The import dialog allows you to browse this folder by clicking on the *Folder* button.

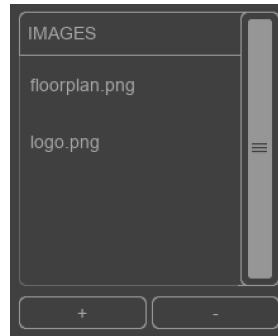


Figure 4.5: Images

To set a background image for a *Page*, select the *Page* and click on one of the images from the list. You can clear the background image clicking on an empty position in the list or clicking on the same image again.

Setting an *Image Element* is done in a similar way. Add the *Image Element*, make sure it is selected, then click on one of the images from the list.

## 4.2.5 Appearance Properties

This section allows you to change the (default) appearance of the page or selected *Elements*.



Figure 4.6: Appearance properties

## 4.3 Screen Size



Figure 4.7: Setting the Screen Size

Before starting to design your dashboard it is recommended to set the screen size for your layout. To change to screen size of your layout, make sure no Elements are selected, then set the desired width and height in the fields as shown in 4.7.

For your convenience, there is a popup dialog with size presets for common mobile devices. Press the *Size Presets* button to open this dialog.

### Size Presets

When no element is selected, the size affects the size of the canvas. Presets can be chosen by pressing the *Size Presets* button, and selecting the target device. From this windows you will also be able to select if you want your layout to be landscape or portrait, and also if your layout will include the top bar or not as this will impact the actual size of it.

If your layout already include some elements and you change the size of it, then a popup message will ask you if you want to automatically scale the elements.

### Size and position

When an element is selected, Size Presets is disabled. The size of the element is decided by setting a desired *Width* and *Height*. The placing of the element can be done either via drag&drop, or by setting the *X* and *Y* values.

### Fontsize and Label

This decides the label and the font size for the selected element. When a page is selected, the label of the page can be set here.

### Visible in selector

Currently only applies to pages and buttons. When selected, the page or button will not be shown on the device running Kiosk. Useful for hiding work-in-progress pages or special buttons.

### Colours

The colour of the *Element* can be changed by setting the RGB fields in figure 4.8. It also features RGB fields for changing the *Element's* font colour. A different colourpicker for selecting the colour is possible by pressing the button next to *RGB Fields*.

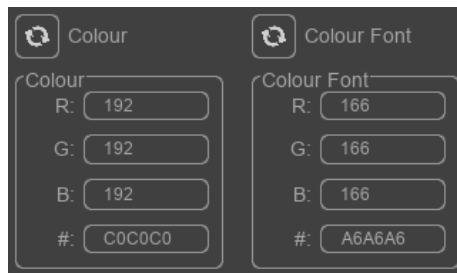


Figure 4.8: Colour properties

### Arrange

When *Elements* overlap, you can use the *Arrange* buttons to control which *Element* lies at the top (see figure 4.9).



Figure 4.9: Arrange buttons

## Properties

Some elements can have specific properties that can be set in the properties section.

**Visible** Page and buttons have a "Visible" property. Not visible pages can be displayed via the *automation* section.

**Pin Protect** The access to some pages can be locked via a requested password. To use it, the *Pin Protect* option must be activated and a *Pin* must be set in the *Kiosc app* or *Kiosc Touch*.

**Percentage** Select this option if you want your fader to display a percentage value instead of the numeric value.

**Flip** For the *Colour temperature wheel* you can select it if you want the warm temperature to be on the left or on the right of the wheel.  
For the *Fader* you can select it if you want the range to be flipped.

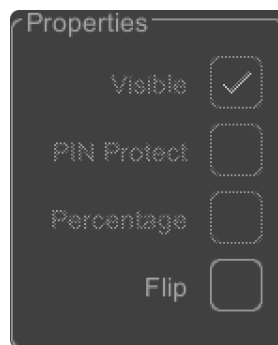


Figure 4.10: Properties options

### 4.3.1 Behaviour Properties



Figure 4.11: Behaviour properties



## OSC/UDP/TCP

When the user clicks on a button in Kiosc, or controls any other element, it typically sends an OSC, UDP or TCP message to an external device. Which message is send out is determined by the *OSC*, *UDP* or *TCP* settings (see figure 4.13).

An OSC tag should always start with a '/' prefix (if you forget it typing your tag *Kiosc Editor* will automatically add it). For some *Elements* it is possible the specify the data type that will be sent in the OSC, UDP or TCP message. For instance, for a button *Bool* (true/false) or *Unsigned* (1/0) can be specified. The Function and Value fields can further specify the value that will be sent. More information on each specific element can be found in appendix A.

The image shows three side-by-side configuration panels for OSC, UDP, and TCP tags. Each panel has a 'Tag' input field, a 'Type' dropdown menu, a 'Function' dropdown menu, and a 'Value' input field. The OSC panel has 'Tag' set to '/page1' and 'Type' set to 'Integer'. The UDP panel has 'Tag' set to 'page1' and 'Type' set to 'Float'. The TCP panel has 'Tag' empty and 'Type' set to 'Float'.

Figure 4.12: Tags

## UDP

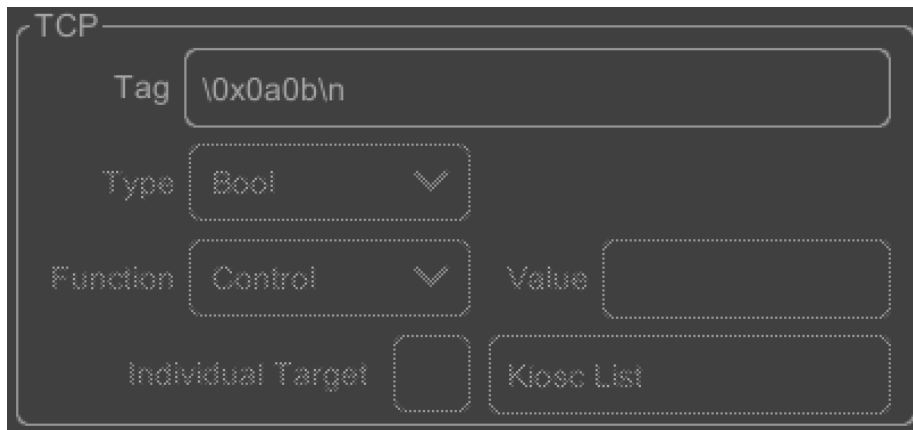
You can use the UDP tag to send hex values. To do that just start the UDP tag by "\0x". You can indicate the end of the TCP message by adding "\n" at the end of your message or just "\" if you want to add a normal text part to your tag.

## TCP

The TCP tag gets some extra features.

You can use the TCP tag to send hex values. To do that just start the TCP tag by "\0x". You can indicate the end of the TCP message by adding "\n" at the end of your message or just "\" if you want to add a normal text part to your tag. Make sure to NOT place any spaces between your hex values. This will make the line be read as text instead.

The valid special characters are: "\n" (line feed), "\r" (carriage return), "\t" (tab), "\0" (number 0) and "\\" (backslash).



TCP

Tag

Type

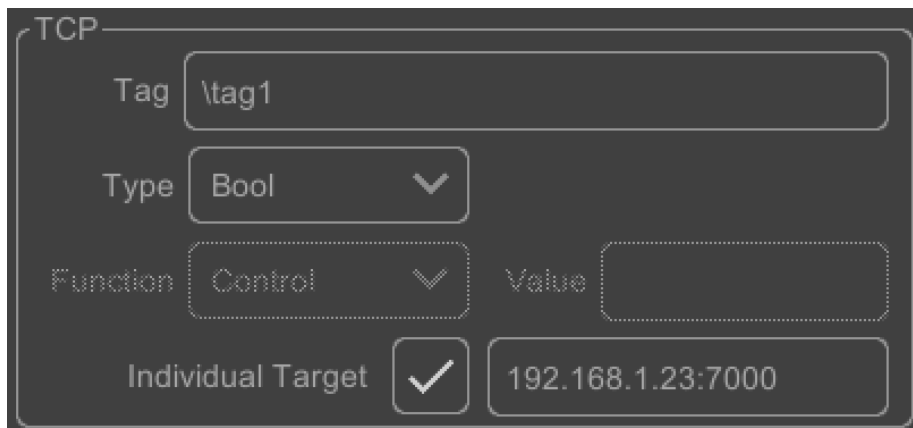
Function  Value

Individual Target  Kiosc List

Figure 4.13: TCP tags

The TCP tag can also be set to be sent to a unique IP address. Just check the *individual target* check box to activate it and set the target IP and port in the beside input.

Note that when the *individual target* is activated the TCP tag will not be sent to the IP addresses set in the TCP target list of KIOSC.



TCP

Tag

Type

Function  Value

Individual Target  Kiosc List

Figure 4.14: TCP individual target

## Values

Some *Elements* (e.g. faders) allow to specify the minimum, maximum and default value.



Figure 4.15: Values

## Automation

**Toggle** When a button is selected, it is optional to convert it into a toggling button by selecting *Toggle Button*. When function is set to control, the button will send its on-value when toggled on, and off-value when toggled off. When using the set-function, it will send the same value in both cases.

**Open Page** In *Kiosc* you can use a button or an image to switch between layout pages. A page doesn't have to be set as *Visible* to be selectable and displayed via a button. This is really handy when you want some pages to be only available from a specific page.

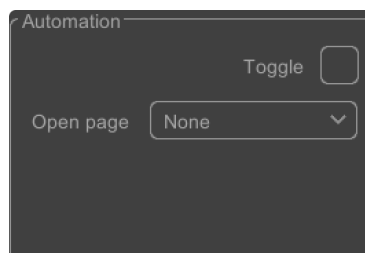


Figure 4.16: Automation Properties

### 4.3.2 Layout

This is where you can view and edit the currently selected page. After dragging in an element from the Elements menu on the right, it can be resized by selecting the element, and dragging the small square located at the corner of the element. It can be removed by dragging it back to the Elements menu or pressing Delete on your keyboard.

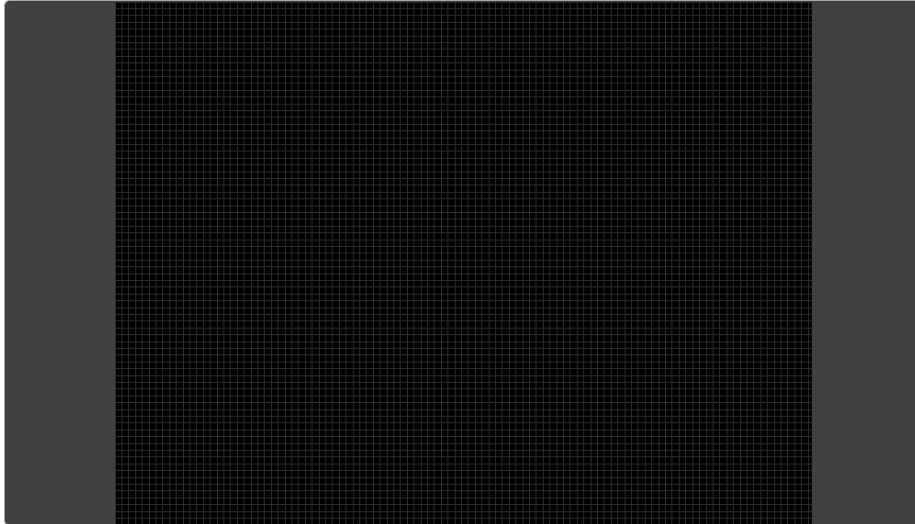


Figure 4.17: Layout

### 4.3.3 Options

The Editor only has a few options, see figure 4.18.

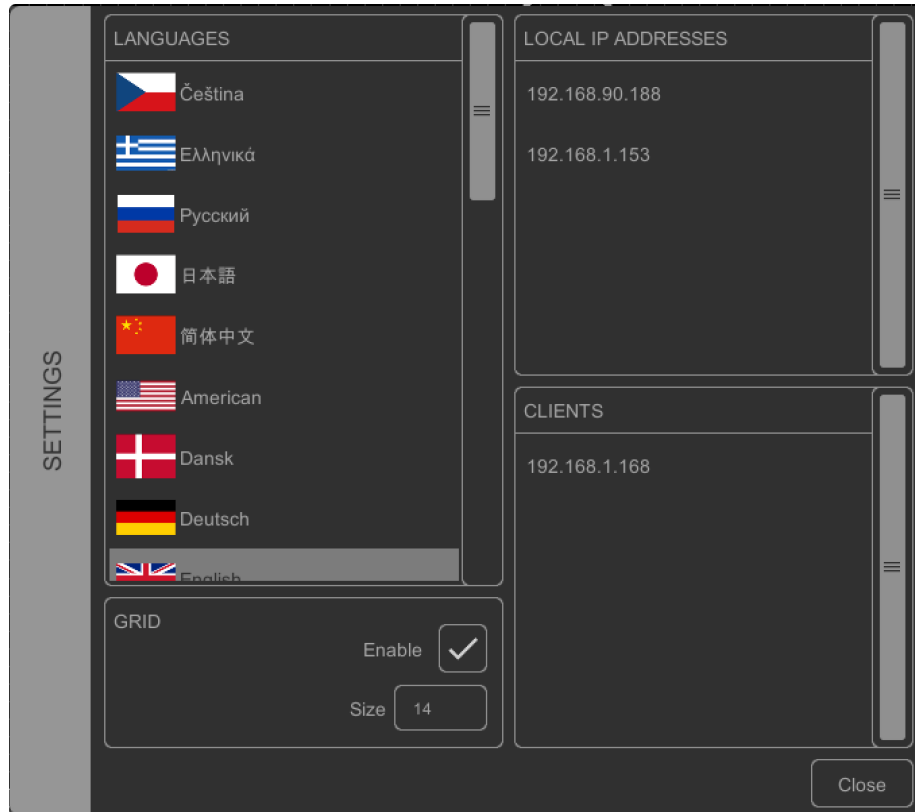


Figure 4.18: Options

#### Language

There are various languages available.

#### Grid Enable

Enable the *Grid* to have an aid in aligning Elements that will automatically snap to the closest grid connection. The size of the grid can be specified in the *size* field.

#### Local IP Addresses

A list with local IP addresses used by the computer running the Kiosk Editor. This information can be useful when trouble shooting network issues.

#### Clients

A list of all Kiosk clients currently connected to this Editor.

## 4.4 Feedback

Feedback in the Kiosk app is created by having the external equipment sending messages back to Kiosk. The feedback message should match the *Tag* of the *Element*. More information on which element supports which kind of feedback can be found in appendix A.

## Chapter 5

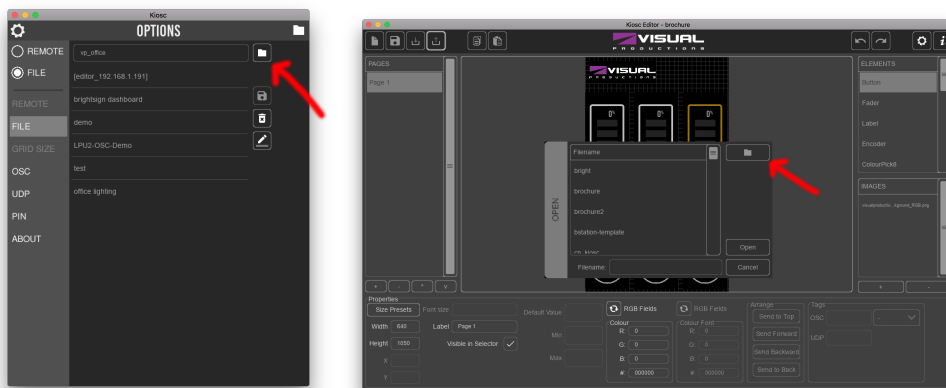
# File Locations

When Kiosc is running in *Remote* mode there are no files involved, however, when running in *File* mode Kiosc must open a file that has been created by the Editor.

Apps distributed by app stores are not allowed to access files outside this designated location. Even apps made by the same developer like Kiosc and Kiosc Editor are not allowed to reach each other files.

Therefore, it is important to know where Kiosc and Kiosc Editor are storing their files, as you will need to transfer the Editor's file to the Kiosc app's location.

The designated file location differs per operating system and is likely to be a long and obscure path. For this reason, Kiosc and Kiosc Editor provide you with a shortcut to the correct file location. A *Folder* button can be found in the file related dialogs in both softwares. Clicking this button will open a file browser at the appropriate folder.



(a) Kiosc

(b) Kiosc Editor

Figure 5.1: Folder shortcut button

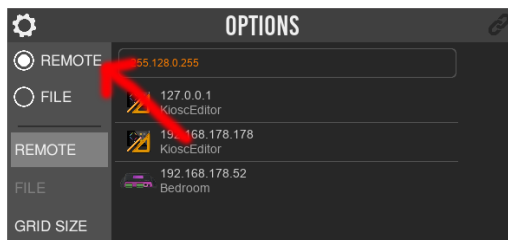
The layout file created by the Editor will have the `.kiosc` file extension.

Transferring files from the Kiosc Editor running on a desktop OS to the Kiosc app running on a mobile OS can be done with a tool for copying files to the mobile device. E.g. Apple iTunes can copy files to an iOS device.

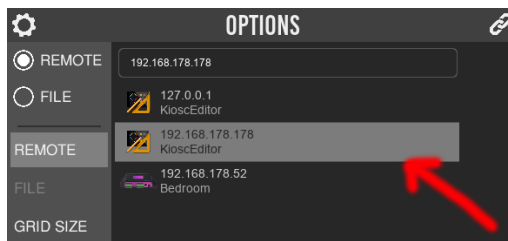
To make copying files to Kiosc on a mobile device more convenient, Kiosc is capable to connect to a Kiosc Editor under *Remote* mode and subsequently save it as a local file in *File* mode. Please note that due to Microsoft Store security measurements, it is not possible to make a remote connection between Kiosc and Kiosc Editor if Kiosc and Kiosc Editor are running on the same PC.

To store the showfile locally on the device running Kiosc, follow these steps:

1. Activate *Remote* mode.

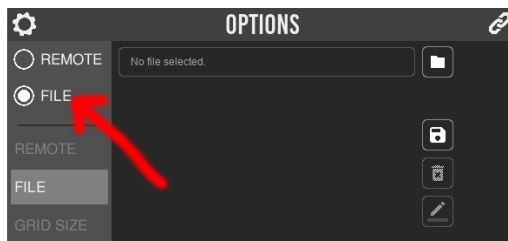


2. Connect to the Editor.



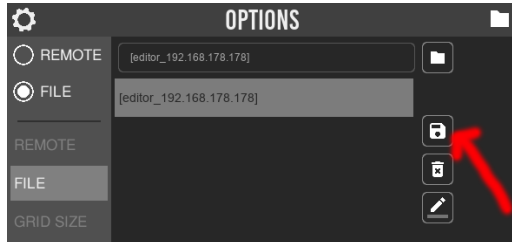
The layout is now loaded into Kiosc's memory.

3. Activate *File* mode.





4. Save the file by pressing the *Save* button.



# Appendices

# Appendix A

## Elements

This chapter will give a more detailed explanation of each element's ability to send and receive various protocols.

## A.1 Page

The Page is not a real element but can be used to send messages when displayed. It can also be displayed using feedback messages. The following sections will give a detailed explanation about the send and receive capabilities of a Page.

### A.1.1 Send

#### OSC

Type	Function	Displayed
-	Active	true

#### UDP and TCP

Type	Function	Displayed	Example
-	Active	true	my-page=true

### A.1.2 Feedback

Feedback can be sent to a page using its tag to directly display a specific page.

#### OSC

Type	Value	Controls
Bool	On	→ Display

#### UDP and TCP

Type	Value	Controls	Example
Bool	On	→ Display	page1=on

## A.2 Button

The button element can be used to send various messages. It can be controlled using feedback messages. The following sections will give a detailed explanation about the send and receive capabilities of the Button element.

### A.2.1 Send

## OSC

Type	Function	Property	Value Down	Value Up
Bool	Control	-	On	Off
Bool	Control	Toggle	On	Off
Integer	Control	-	1	0
Integer	Control	Toggle	1	0
Integer	Set	-	<Number>	-
Integer	Set	Toggle	<Number>	<Number>

## UDP and TCP

Type	Function	Property	Value Down	Value Up	Example
None	-	-	Tag only	-	my-button
None	-	Toggle	Tag only	Tag only	my-button
Bool	Control	-	On	Off	my-button=On
Bool	Control	Toggle	On	Off	my-button=On
Integer	Control	-	1	0	my-button=1
Integer	Control	Toggle	1	0	my-button=1
Integer	Set	-	<Number>	-	my-button=23
Integer	Set	Toggle	<Number>	<Number>	my-button=23

## A.2.2 Feedback

Feedback can be sent to a button using its tag. The type of message determines which attribute of the button it changes. The following attributes of a button can be set using feedback:

- **Visual state**

The visual state of a button can be used as feedback to the user, indicating for example if a light or cue is active. It is much like having an LED light inside a physical button. When enabled, either via a feedback message or by pressing a button with the special property "Toggle", the inside of the button will be filled.

- **Label**

The label is a string of text displayed inside the button. The default value is set in the showfile, but it can be temporarily changed using a feedback message. Reloading the showfile will return the button to its default label.

- **Colour**

The default colour of the button is set in the showfile, but it can be temporarily changed using a feedback message. Reloading the showfile will return the button to its default colour.

## OSC

Type	Value	Controls
Bool	On/Off	→ State
Integer	1/0	→ State
String	<text>	→ Label
Colour	<colour>	→ Colour

## UDP and TCP

Type	Value	Controls	Example
Bool	On/Off	→ State	my-button=on
Integer	1/0	→ State	my-button=1
String	<text>	→ Label	my-button=text
Colour	<HEX RGB colour>	→ Colour	my-button=#FF00FF

## A.3 CheckBox

The check box can be used to send various messages. It can be controlled using feedback messages. The following sections will give a detailed explanation about the send and receive capabilities of a check box.

### A.3.1 Send

#### OSC

Type	Function	Value Checked	Value Unchecked
Bool	Control	On	Off
Integer	Control	1	0

#### UDP and TCP

Type	Function	Value Checked	Value Unchecked	Example
Bool	Control	On	Off	my-check=On
Integer	Control	1	0	my-check=1

### A.3.2 Feedback

Feedback can be send to a check box using its tag. The type of message determines which attribute of the check box it changes. The following attributes of a check box can be set using feedback:

- **State**  
The state of a check box can be used as visual feedback to the user, indicating for example if a light or cue is active. When enabled, either via a feedback message or by pressing check box area, the inside of the box will be filled.
- **Label**  
The label is a string of text displayed close to the check box. The default value is set in the showfile, but it can be temporarily changed using a feedback message. Reloading the showfile will return the check box to its default label.
- **Colour**  
The default colour of the check box is set in the showfile, but it can be temporarily changed using a feedback message. Reloading the showfile will return the check box to its default colour.

## OSC

Type	Value	Controls
Bool	On/Off	→ State
Integer	1/0	→ State
String	text	→ Label
Colour	<colour>	→ Colour

## UDP and TCP

Type	Value	Controls	Example
Bool	On/Off	→ State	my-check=on
Integer	1/0	→ State	my-check=1
String	text	→ Label	my-check=awesome
Colour	<HEX RGB colour>	→ Colour	my-check=#FF00FF

## A.4 Fader

The fader element can be used to sent various control values. It can be controlled using feedback messages. The following sections will give a detailed explanation about the send and receive capabilities of the fader element.

### A.4.1 Send

#### OSC

Type	Function	Value	Range
Float	Control	<decimal number>	Min-Max
Integer	Control	<whole number>	Min-Max

#### UDP and TCP

Type	Function	Value	Range	Example
Float	Control	<decimal number>	Min-Max	my-fader=0.50
Integer	Control	<whole number>	Min-Max	my-fader=127



## A.4.2 Feedback

Feedback can be send to a fader using its tag. The type of message determines which attribute of the fader it changes. The following attributes of a fader can be set using feedback:

- **Value**

The current value of a fader can be set by a feedback message, for example to indicate the current intensity of a playback. The fader responds to feedback matching its type, and it takes into account the min and max range set in the showfile.

- **Label**

The label is a string of text displayed inside the fader. The default value is set in the showfile, but it can be temporarily changed using a feedback message. Reloading the showfile will return the fader to its default label.

- **Colour**

The default colour of the fader is set in the showfile, but it can be temporarily changed using a feedback message. Reloading the showfile will return the fader to its default colour.

## OSC

Type	Value	Controls
Float	Min/Max	→ Value
Integer	Min/Max	→ Value
String	<text>	→ Label
Colour	<colour>	→ Colour

## UDP and TCP

Type	Value	Controls	Example
Float	Min/Max	→ Value	my-fader=0.75
Integer	Min/Max	→ Value	my-fader=190
String	<text>	→ Label	my-fader=text
Colour	<HEX RGB colour>	→ Colour	my-fader=#FF00FF

## A.5 Encoder

The encoder element can be used to send various control values. It can be controlled using feedback messages. The following sections will give a detailed explanation about the send and receive capabilities of the encoder element.

### A.5.1 Send

#### OSC

Type	Function	Value	Range
Float	Control	<decimal number>	Min-Max
Integer	Control	<whole number>	Min-Max

#### UDP and TCP

Type	Function	Value	Range	Example
Float	Control	<decimal number>	Min-Max	my-encoder=0.50
Integer	Control	<whole number>	Min-Max	my-encoder=127

### A.5.2 Feedback

Feedback can be send to an encoder using its tag. The type of message determines which attribute of the fader it changes. The following attributes of an encoder can be set using feedback:

- **Value**  
The current value of an encoder can be set by a feedback message, for example to indicate the current intensity of a playback. The encoder responds to feedback matching its type, and it takes into account the min and max range set in the showfile.
- **Label**  
The label is a string of text displayed inside the encoder. The default value is set in the showfile, but it can be temporarily changed using a feedback message. Reloading the showfile will return the encoder to its default label.
- **Colour**  
The default colour of the encoder is set in the showfile, but it can be temporarily changed using a feedback message. Reloading the showfile will return the encoder to its default colour.

## OSC

Type	Value	Controls
Float	Min/Max	→ Value
Integer	Min/Max	→ Value
String	<text>	→ Label
Colour	<colour>	→ Colour

## UDP and TCP

Type	Value	Controls	Example
Float	Min/Max	→ Value	my-encoder=0.75
Integer	Min/Max	→ Value	my-encoder=190
String	<text>	→ Label	my-encoder=text
Colour	<HEX RGB colour>	→ Colour	my-encoder=#FF00FF

## A.6 Axis

The axis element can be used to sent position values in a 2 float number between -1.00 and 1.00 separated by a coma (,) format. It can be controlled using feedback messages. The following sections will give a detailed explanation about the send and receive capabilities of the axis element.

### A.6.1 Send

#### OSC

Type	Function	Value	Range
Float	Control	<decimal number, decimal number>	(-1.00,-1.00) - (1.00,1.00)

#### UDP and TCP

Type	Function	Value	Range	Example
Float	Control	<dec. num., dec. num.>	(-1.00,-1.00) - (1.00,1.00)	axis=0.50,-0.46

### A.6.2 Feedback

Feedback can be send to an axis using its tag. The type of message determines which attribute of the fader it changes. The following attributes of an encoder can be set using feedback:

- **Value**  
The current value of an axis can be set by a feedback message, for example to indicate the current position of a moving head. The axis responds to feedback matching its type.
- **Colour**  
The default colour of the axis is set in the showfile, but it can be temporarily changed using a feedback message. Reloading the showfile will return the axis to its default colour.

#### OSC

Type	Value	Controls
Float	0.00,0.00	→ Value
Colour	<colour>	→ Colour

## UDP and TCP

Type	Value	Controls	Example
Float	0.00,0.00	→ Value	my-axis=0.75,-0,27
Colour	<HEX RGB colour>	→ Colour	my-axis=#FF00FF

## A.7 Text

The Text element can be used to send text messages. It can be controlled using feedback messages. The following sections will give a detailed explanation about the send and receive capabilities of the Text element.

### A.7.1 Send

#### OSC

Type	Function	Value
String	set	<text>

#### UDP and TCP

Type	Function	Value	Example
String	Set	<text>	my-text=awesome

### A.7.2 Feedback

Feedback can be sent to a text element using its tag. The type of message determines which attribute of the text element it changes. The following attributes of a text element can be set using feedback:

- **Text**  
The displayed text of a Text element can be set by a feedback message, for example to indicate the name of the current light scene running.
- **Colour**  
The default colour of the Text element is set in the showfile, but it can be temporarily changed using a feedback message. Reloading the showfile will return the text element to its default colour.

#### OSC

Type	Value	Controls
String	<text>	→ text
Colour	<colour>	→ Colour

#### UDP and TCP

Type	Value	Controls	Example
String	<text>	→ Label	my-text=awesome
Colour	<HEX RGB colour>	→ Colour	my-text=#FF00FF

## A.8 Indicator

The indicator element can be seen as a virtual LED. It can only be used for visual feedback, it does not send any message. The indicator can be changed using feedback messages. The following sections will give a detailed explanation about the receive capabilities of the indicator element.

### A.8.1 Feedback

Feedback can be sent to an indicator using its tag. The type of message determines which attribute of the indicator it changes. The following attributes of an indicator can be set using feedback:

- **State**

The state of an indicator can be used as visual feedback to the user, indicating for example if a light or cue is active. It is much like having an LED light. By default the indicator state is off and the colour of the indicator is dimmed. When enabled via a feedback message the colour inside of the indicator will be brightened.

- **Colour**

The default colour of the indicator is set in the showfile, but it can be temporarily changed using a feedback message. Reloading the showfile will return the indicator to its default colour.

## OSC

Type	Value	Controls
Bool	On/Off	→ State
Integer	0 or lower	→ State (Off)
Integer	1 or higher	→ State (On)
Float	0 or lower	→ State (Off)
Float	0.001 or higher	→ State (On)
Colour	<colour>	→ Colour

## UDP and TCP

Type	Value	Controls	Example
Bool	On/Off	→ State	my-indicator=on
Integer	0 or lower	→ State (Off)	my-indicator=-10
Integer	1 or higher	→ State (On)	my-indicator=1
Float	0 or lower	→ State (Off)	my-indicator=0
Float	0.001 or higher	→ State (On)	my-indicator=0.5
Colour	<HEX RGB colour>	→ Colour	my-indicator=#FF00FF



## A.9 Image

The image element can be used to send various messages. It can be controlled using feedback messages. The following sections will give a detailed explanation about the send and receive capabilities of the image element.

### A.9.1 Send

#### OSC

Type	Function	Property	Value Down	Value Up
Bool	Control	-	On	Off
Bool	Control	Toggle	On	Off
Integer	Control	-	1	0
Integer	Control	Toggle	1	0
Integer	Set	-	<Number>	-
Integer	Set	Toggle	<Number>	<Number>

#### UDP and TCP

Type	Function	Property	Value Down	Value Up	Example
None	-	-	Tag only	-	my-image
None	-	Toggle	Tag only	Tag only	my-image
Bool	Control	-	On	Off	my-image=On
Bool	Control	Toggle	On	Off	my-image=On
Integer	Control	-	1	0	my-image=1
Integer	Control	Toggle	1	0	my-image=1
Integer	Set	-	<Number>	-	my-image=23
Integer	Set	Toggle	<Number>	<Number>	my-image=23

### A.9.2 Feedback

Feedback can be sent to an image using its tag. The type of message determines which attribute of the button it changes. The following attributes of an image can be set using feedback:

- **Alpha**

The default alpha of the image is set to 100%, but it can be temporarily changed using a feedback message. Reloading the showfile will return the image to its default alpha.

## OSC

Type	Value	Controls
Bool	On/Off	→ Alpha
Float	1.00/0.00	→ Alpha

## UDP and TCP

Type	Value	Controls	Example
Bool	On/Off	→ Alpha	my-image=on
Float	1.00/0.00	→ Alpha	my-image=0.5

## A.10 Label

The label element can only be used for feedback, it does not send any message. The label can be changed using feedback messages. The following sections will give a detailed explanation about the receive capabilities of the label element.

### A.10.1 Feedback

Feedback can be sent to a label using its tag. The type of message determines which attribute of the label it changes. The following attributes of a label can be set using feedback:

- **Label**

The label is a string of text. The default value is set in the showfile, but it can be temporarily changed using a feedback message. Reloading the showfile will return the default text to the label element.

- **Colour**

The default font colour of the label is set in the showfile, but it can be temporarily changed using a feedback message. Reloading the showfile will return the label to its default colour.

## OSC

Type	Value	Controls
String	<text>	→ Label
Colour	<colour>	→ Colour

## UDP and TCP

Type	Value	Controls	Example
String	<text>	→ Label	my-label=awesome
Colour	<HEX RGB colour>	→ Colour	my-label=#800080

## A.11 ColorSpectrum

The spectrum elements can send color value on an hex format. The following sections will give a detailed explanation about the receive capabilities of the Color Spectrum element.

### A.11.1 Send

## OSC

Type	Function	Value	Range
Colour	Control	<HEX RGB colour>	#000000-#FFFFFF

## UDP and TCP

Type	Function	Value	Range	Example
String	Control	<HEX RGB colour>	#000000-#FFFFFF	my-color=#800080

### A.11.2 Feedback

Feedback can be sent to a ColorSpectrum using its tag.

- **Colour**

The color value can be set via feedback if the ColorSpectrum element has such value. Check available colors at A.16.

## OSC

Type	Value	Controls
Colour	Control	<HEX RGBA colour> #00000000-#FFFFFFFF

## UDP and TCP

Type	Value	Controls	Example
Colour	<HEX RGB colour>	→ Colour	my-spectrum=#800080

## A.12 Color Pickers: ColorPicker8, ColorPicker16, WheelColor

The color picker elements can send color value on an hex format. The label can be changed using feedback messages. The following sections will give a detailed explanation about the receive capabilities of the color picker elements.

### A.12.1 Send

## OSC

Type	Function	Value	Range
Colour	Control	<HEX RGBA colour>	#000000FF-#FFFFFFFF

## UDP and TCP

Type	Function	Value	Range	Example
String	Control	<HEX RGB colour>	#000000-#FFFFFF	my-color=#800080

### A.12.2 Feedback

Feedback can be sent to a color picker using its tag.

- **Label**

The label is a string of text. The default value is set in the showfile, but it can be temporarily changed using a feedback message. Reloading the showfile will return the default text to the color picker element.

- **Colour**

The color value can be set via feedback if the Color Picker element has such value. Check available colors at A.16.

## OSC

Type	Value	Controls
String	<text>	→ Label
Colour	Control	<HEX RGBA colour> #00000000-#FFFFFFFF

## UDP and TCP

Type	Value	Controls	Example
String	<text>	→ Label	my-colorpicker=awesome
Colour	<HEX RGB colour>	→ Colour	my-colorpicker=#800080

## A.13 WheelRGBW

The WheelRGBW elements can send color value on an hex format. The label can be changed using feedback messages. The following sections will give a detailed explanation about the receive capabilities of the WheelRGBW element.

### A.13.1 Send

## OSC

Type	Function	Value	Range
Colour	Control	<HEX RGBA colour>	#00000000-#FFFFFFFF

## UDP and TCP

Type	Function	Value	Range	Example
Colour	<HEX RGB colour>	→	Colour	my-wheel=#800080

### A.13.2 Feedback

Feedback can be sent to a WheelRGBW element using its tag.

- **Label**  
The label is a string of text. The default value is set in the showfile, but it can be temporarily changed using a feedback message. Reloading the showfile will return the default text to the color picker element.
- **Colour**  
The color value can be set via feedback if the WheelRGBW element has such value. Check available colors at A.16.

## OSC

Type	Value	Controls	
String	<text>	→	Label
Colour	Control	<HEX RGBA colour>	#00000000-#FFFFFFFF

## UDP and TCP

Type	Value	Controls	Example
String	<text>	→ Label	my-wheelrgbw=awesome

## A.14 WheelColTemp

The WheelColTemp elements can send color temperature ratio between warm white and cold white on an float format. It can be controlled using feedback messages. The following sections will give a detailed explanation about the send and receive capabilities of the WheelColTemp element.

### A.14.1 Send

#### OSC

Type	Function	Value	Range
Float	Control	<decimal number>	0.00-1.00

## UDP and TCP

Type	Function	Value	Range	Example
Float	Control	<decimal number>	0.0-1.0	my-temp=0,82

### A.14.2 Feedback

Feedback can be sent to a WheelRGBW element using its tag. The label of the element can be set using feedback:

- **Label**

The label is a string of text. The default value is set in the showfile, but it can be temporarily changed using a feedback message. Reloading the showfile will return the default text to the color picker element.

## OSC

Type	Value	Controls
String	<text>	→ Label

## UDP and TCP

Type	Value	Controls	Example
String	<text>	→ Label	my-wheelrgbw=awesome

## A.15 Date, Clock and Line

The Date, Clock and Line elements can only be used for feedback, it does not send any message. The color can be changed using feedback messages. The following sections will give a detailed explanation about the receive capabilities of these elements.

### A.15.1 Feedback

Feedback can be sent to a Date, Clock or Line element using its tag.

- **Colour**

The default font colour of a Date, Clock or Line element is set in the showfile, but it can be temporarily changed using a feedback message. Reloading the showfile will return the element to its default colour.

## OSC

Type	Value	Controls
Colour	<colour>	→ Colour

## UDP and TCP

Type	Value	Controls	Example
Colour	<HEX RGB colour>	→ Colour	my-date=#800080

## A.16 Color elements values

### A.16.1 ColorSpectrum

Any color with RGB channels is available:

- #XXXXXX (with X from 0 to F).

### A.16.2 Color Pickers

#### ColorPicker8

The following colors are available:



#### ColorPicker16

The following colors are available:



#### WheelColor

Any color with one of the channels at 0 is available:

- #00XXXX (with X from 0 to F).
- #XX00XX (with X from 0 to F).
- #XXXX00 (with X from 0 to F).

### A.16.3 WheelRGBW

Any color with RGBW channels is available:

- #XXXXXXXX (with X from 0 to F).



# Appendix B

## API

This chapter will give the list of all the API commands you can send to KIOSC and what will be its feedback. The API commands are only compatible with *KIOSC Touch*.

## B.1 OSC

---

URI	Parameter	Description
/kiosc/hello	-	The unit will reply with the same Hello message
/kiosc/goodbye	-	The unit will erase the IP from the feedback list
/kiosc/blink	-	Momentarily flashes the Kiosk's header
/kiosc/page/select	<integer>	Switch to page index of the layout

---

## B.2 UDP and TCP

---

URI	Parameter	Description
kiosc-hello	-	The unit will reply with the same Hello message
kiosc-goodbye	-	The unit will erase the IP from the feedback list
kiosc-blink	-	Momentarily flashes the Kiosk's header
kiosc-page-select=	<integer>	Switch to page index of the layout

---

## B.3 Feedback

The Kiosk is able to send feedback to external equipment using its API, so called 'clients'. The Kiosk keeps a memory of the last eight OSC clients, the last eight UDP clients and last eight TCP clients.

The API's feedback message will only be sent to the client list of the protocol used to send the command.

For example, if you send an API command via OSC, only the clients registered in the OSC clients list will receive the feedback message. Below is a table listing the messages the Kiosk will send back to its clients.

---

OSC	UDP/TCP
/core/hello	core-hello

---

The `hello` command is ideal for polling the device; it allows you to verify that the Kiosk is online at the IP address and port that you expect.

A power-cycle will clear the internal client lists. Send `/kiosc/goodbye` or

`kiosc-goodbye` to explicitly be removed from the client list.

### **B.3.1 Preventing a feedback loop**

Feedback is automatically sent to a device which uses the OSC, UDP or TCP API. If the external device is also a Visual Productions unit then the feedback message could be interpreted by the external unit as a new command. This can result in another feedback message being generated. An endless stream of feedback messages can stall the units involved.

This feedback loop can be prevented by assigning a unique label to the device's API prefix. This setting is discussed on page 24.