PixelController

PixelController - a matrix control project by Michael Vogt , (c) 2010-2013. The main goal of this application is to create an easy to use matrix controller software which creates stunning visuals!

Primary Website: http://www.pixelinvaders.ch

My Blog: http://www.neophob.com

Facebook: https://www.facebook.com/PixelInvaders

HOWTO USE PIXELCONTROLLER

Prerequisite: * Java Runtime, v1.6+

Run `PixelController.jar` to start the application. Make sure your led matrix connected to you computer before the application is started and you **configured** your output hardware in the data/config.properties file.

You can **download** PixelController on Google Code: http://code.google.com/p/pixelcontroller/downloads/

DEMO

Check out http://vimeo.com/27453711 and http://vimeo.com/32580251 to see PixelController in action on two PixelInvaders panels.

SUPPORTED HARDWARE

PixelController supports different (LED) matrix hardware devices:

- PixelInvaders 3d Panels (see Readme.PixelInvaders, http://www.pixelinvaders.ch)
- Seeedstudios Rainbowduino V2 (see Readme.rainbowduinoV2)
- Seeedstudios Rainbowduino V3
- ArtNet Devices, multiple universe are supported,510 Channels (170 RGB Pixels) per universe
- MiniDmx Devices (like the SEDU board of http://www.led-studien.de)
- Adavision (http://www.adafruit.com/products/611)
- Element Labs Stealth LED panel. No longer in production (http://cled.barcousa.com/support/STEALTH/STEALTH_Users_Guide.pdf)
- Generic UDP Devices (for example Raspberry Pi, check out the PixelPi Software)
- TPM2 Serial devices (see http://ledstyles.de for more information)
- TPM2 Net devices (see http://ledstyles.de for more information)

Check out the `data/ArduinoFW` directory, all Arduino based firmware files are stored there.

FRONTENDS

There are different frontends for PixelController:

- Native Java: the default frontend is started when PixelController starts.
- PixConCli: Command Line Interface for PixelController, works also remote. The CLI tool is called `PixConCli.cmd` on Windows and `PixConCli.sh` on Linux/OSX.
- PureData: PureData frontend (http://puredata.info/ download the extended Version), very flexible, extensible (OSC, MIDI). The PureData file is called `PixelController.pd`.

INTERFACES

- OSC interface, default listening port 9876. Processing examples included how to communicate with PixelController via OSC protocol
- FUDI interface, default listening port 3448, used to communicate with PureData

Valid commands:

```
CHANGE_GENERATOR_A          # of parameters: 1     <INT> change first generator for current visual
CHANGE_GENERATOR_B          # of parameters: 1     <INT> change first generator for current visual
CHANGE_EFFECT_A             # of parameters: 1     <INT> change first effect for current visual
CHANGE_EFFECT_B             # of parameters: 1     <INT> change second effect for current visual
CHANGE_MIXER                # of parameters: 1     <INT> change mixer for current visual
```

```
CURRENT_VISUAL              # of parameters: 1     <INT> select actual visual
CURRENT_COLORSET            # of parameters: 1     <INT> select actual ColorSet

CHANGE_OUTPUT_VISUAL        # of parameters: 1     <INT> change visual for current output
CHANGE_OUTPUT_FADER         # of parameters: 1     <INT> change fader for current output
CHANGE_ALL_OUTPUT_VISUAL    # of parameters: 1     <INT> change visual for all outputs
CHANGE_ALL_OUTPUT_FADER     # of parameters: 1     <INT> change fader for all outputs
CURRENT_OUTPUT              # of parameters: 1     <INT> select current output

BLINKEN                     # of parameters: 1     <STRING> file to load for the blinkenlights generator
IMAGE                       # of parameters: 1     <STRING> image to load for the simple image generator
TEXTDEF                     # of parameters: 1     <INT> select texture deformation option, 1-11
COLOR_SCROLL_OPT            # of parameters: 1     <INT> select color scroll fading direction, 1-14
COLOR_FADE_LENGTH           # of parameters: 1     <INT> fading length for the color fade generator
COLOR_SCROLL_LENGTH         # of parameters: 1     <INT> fading distance for the color scroll generator
TEXTWR                      # of parameters: 1     <STRING> update text for textwriter generator
CHANGE_BRIGHTNESS           # of parameters: 1     <int> output brightness 0 .. 100

CHANGE_THRESHOLD_VALUE      # of parameters: 1     <INT> select current threshold for the threshold effect, 0-255
CHANGE_ROTOZOOM             # of parameters: 1     <INT> select angle for the rotozoom effect, -127-127

STATUS                      # of parameters: 0     refresh whole gui
STATUS_MINI                 # of parameters: 0     just refresh parts of the gui
CHANGE_PRESENT              # of parameters: 1     <INT> select current present id
CHANGE_SHUFFLER_SELECT      # of parameters: 14    14 times <INT>, 14 parameter to enable or disable the shuffler option (gets
SAVE_PRESENT                # of parameters: 0     save current present settings
LOAD_PRESENT                # of parameters: 0     load current present settings
RANDOM                      # of parameters: 1     <ON|OFF> enable/disable random mode
RANDOMIZE                   # of parameters: 0     one shot randomizer
PRESET_RANDOM               # of parameters: 0     one shot randomizer, use a pre-stored present
JMX_STAT                    # of parameters: 0     show JMX runtime statistic, default port: 1337 (use the -p switch)
SCREENSHOT                  # of parameters: 0     save screenhot
FREEZE                      # of parameters: 0     toggle pause mode
```

IT DOES NOT WORK!

Try to understand **WHAT** does not work, which component? is it the frontend? PixelController itself? or no output?

Here are some common errors:

- Did you forgot to **edit the configuration file** `config.properties`. Take a look at the config.examples subdirectory!
- Did you flash the **correct firmware** to you Arduino/Teensy?
- **PixelInvaders panels**: Make sure that the Panel shows an **animated rainbow pattern** when the panels are powered on (make sure that you also power the Arduino/Teensy board). If you don't see a animated rainbow, make sure the directon of the modules is correct and that the Arduino/Teensy, LED modules and PSU share common ground. Verify the Arduino IDE don't spit out errors when you upload the firmware to the teensy
- **PixelInvaders panels**: A User reported that the PixelInvader firmware did not work on a new Arduino UNO r3 board. I think the reason for this is the big serial latency. However using a Arduino UNO r1 worked flawlessly. Technically this is not a big deal, as the timeout value cold be adjusted in the firmware.
- Make sure you're using an up-to date Java Runtime (JRE), this usually helps if the JVM crashes.
- If you use an extra long USB Cable (more than 5 meter) you might discover strange issues, try to use a short cable especially if you're uploading a firmware.

HOWTO BUILD PIXELCONTROLLER

Prerequisite:

- Maven v2.x (if you use Maven 3, make sure to read http://neophob.com/2011/11/maven-3-is-evil/ first!)
- JDK 1.6+

Then run

```
# mvn initialize
to install the needed packages in your local repo and
# mvn clean package
to build PixelController, the distribution directory is "target/assembly/PixelController-VERISON/".
```

Hint: if you're using eclipse and you see an error like this
`java.lang.NoClassDefFoundError: Could not initialize class gnu.io.RXTXVersionjava.lang.NoClassDefFoundError: Could not initialize`
make sure you add the lib/serial directory as "Native library location"

ADD NEW HARDWARE SUPPORT

It should be pretty simple to add support for new hardware. All Output code should go into the com.neophob.sematrix.output package (`src/main/java/com/neophob/sematrix/output` directory). All you need to do in the Output class is, take an array of int's (one int is used to store the 24 bpp) and send this buffer to your output device (via serial port, ethernet, bluetooth...). Maybe you need to reduce the color depth, flip each second scanline due hardware wiring, such helper methods should go into the `OutputHelper.java` class.

As a string point, add your hardware in the `OutputDeviceEnum.java` class and have a look where the other entries are referenced. **Take a look at the existing Output classes**, this should help you!

PERFORMANCE

With the JMX interface you can monitor the status of your PixelController instance in real time. This will provide you with useful data such as required time for each layer (generator, effect, mixer…), the frame rate of your instance, allowing you to diagnose problems or performance issues. To read the JMX data, you will need to use a JMX client or the PixConCli util.

Example how to use PixConCli:

```
localhost:PixelController-1.3-SNAPSHOT michu$ ./PixConCli.sh -c JMX_STAT -p 1337
Create an RMI connector client and connect it to the RMI connector server 127.0.0.1:1337
Get an MBeanServerConnection...

Generic:
server version           : 1.1
current fps              : 20,036 (100% of configured fps: 20)
frame count              : 1771
running since            : 0:01:28.980

The following average times have been collected during the last 10.007 seconds:
   generator             : 0,310ms
   effect                : 0,000ms
   output schedule       : 0,140ms
   fader                 : 0,000ms
   debug window          : 15,210ms
   output prepare wait   : 0,005ms
   output update wait    : 0,005ms
   matrix emulator window: 0,440ms

Ouput-specific average times for output #1: NULL (NullDevice)
   prepare               : 1,550ms
   update                : 0,000ms

Close the connection to the server
```

CREDITS

- **Michael Vogt**: Project Lead, Main Developer
- **Markus Lang**: Maven enhancements, Output enhancements, Performance enhancements, Rainbowduino V3 support
- **McGyver666**: Contributor
- **Rainer Ostendorf**: Artnet Output
- **Pesi**: miniDMX Output, Tester
- **Scott Wilson**: Arduino/Rainbowduino Howto
- **Noxx6**: Bugfixes
- **Psykon**: Example Visuals
- **okyeron**: Stealth output device
- **Dr. Stahl**: Documentation, Tester