intro to programmable lighting

with CEMMI co-founder Daniel Taub

hosted by the Artisan's Asylum

prerequisites:

- Basic understanding of computers (terminal knowledge a plus)
- Love of all things blinky, flashy, fadey, or pulsey

topics:

1) Color Kinetics
    = Power supply detection and configuration
    = Single fixture control using KiNet
    = DMX512 controllers
2) LED Arrays and NumPy
    = Basic Linear Algebra
    = Light array control using BluewayPx
    = Light Strand Simulator
3) SaikoLED (if time permits)
    = Arduino (1.0) and PWM
    = TouchOSC and liblo (advanced)
    = PureData/MaxMSP (advanced)

resources:

Software:

http://www.colorkinetics.com/ls/controllers/quickplaypro/
https://github.com/vishnubob/kinet
https://github.com/CEMMI-org
http://numpy.scipy.org/

General:

http://www.directionless.org/color-kinetics/Main_Page
http://saikoled.com/
http://en.wikipedia.org/wiki/User:Dcianf (trivia: CK History)
http://cemmi.org/index.php/forum/index

revision 1.0b

© 2012 Daniel M. Taub
http://dmtaub.com
dmtaub@cemmi.org

Part One: Setup and Simple Control

One person in each group will need to install: QuickPlay Pro (QPP)

Configuring a Power Supply

1. Disable other connections and firewalls
2. Plug in Power/Data Supply (PDS)
3. Connect Ethernet Cable
4. Open QPP (If PDS not found, set static IP as listed below*)
5. Set IP for PDS Configuration
6. Set DMX Addresses for Fixture Configuration (serial number?)

Configuring computer to talk to lights*

1. Set static IP and netmask for computer to match PDS
2. Download software listed below

Each person will need the following installed:

- *Python 2.6 or 2.7 (ipython recommended)*
- *git (windows instructions here)*
- *Vishnubob's Kinet (via git as follows:)*

Using git to get source code from github

1. If you want to be able to collaborate online, get a github account
2. If you have git, but aren't on github, use the following command:

   ```
   git clone http://github.com/vishnubob/kinet.git
   ```

3. If you don't have git, you can download code here

Controlling Lights!

1. `> cd kinet`
2. edit examply.py, change line 31 to match your PDS's IP
3. `> python example.py`

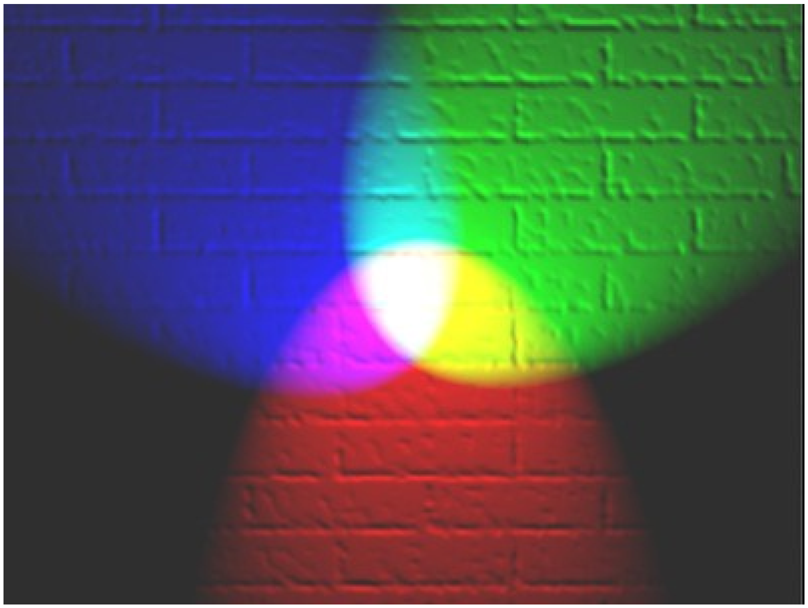
Creating your first Python light control script

```
from kinet import *
pds = PowerSupply("192.168.0.???")
fix = FixtureRGB(0)
pds.append(fix)
fix.set_rgb([0,222,255])
pds.go()
```

revision 1.0b

http://dmtaub.com
dmtaub@cemmi.org

How does it work?



Images:
http://en.wikipedia.org/wiki/File:Et_marquee_2.JPG
http://en.wikipedia.org/wiki/File:RGB_illumination.jpg
http://en.wikipedia.org/wiki/File:Group_of_XLR_connectors_PICT6918.jpg

DMX512 History(from Wikipedia)**:**
-1986: Digital Multiplex with 512 pieces of information
-Revised in 1990, and Entertainment Services and Technology Association (ESTA) worked from 1998-2004 to develop it into an ANSI standard
-Most recent revision was in 2008, birthing DMX-512A was born.

DMX over UDP: ArtNet and KiNet.
-Used from controller to supply. DMX often still used between supplies and from supplies to fixture.
-512 bytes and a Magic Header (Opcode, Protocol, Universe, Sequence Number).
-KiNet Header allows PDS to recognize the packets and respond to them.
-Header to control lights well known through reverse engineering
-Headers for power supply and fixture discovery protocols are still at large.
Get WireShark and try to figure it out yourself !!

Universes:
Maximum RGB lights in a universe:
512 / 3 = 170 ⅔

How many for RGBA lights?

revision 1.0b

http://dmtaub.com
dmtaub@cemmi.org

Part Two: NumPy and Matrix-style control

You will additionally need the following installed:

- *Numpy and/or SciPy (might be easiest via easy_install)*
- *CEMMI's BluewayPx (download or via git as follows:)*

Creating your own branch with git

```
1. > git clone http://github.com/CEMMI-org/Blueway.git
2. > cd Blueway
3. > git checkout class
4. > git checkout -b _your_name_here_
```

Numpy primer: these create equivalent data structures!

```
numpy.array([0,0,0] * 50)
numpy.zeros(150)
numpy.zeros([50,3]).flatten()
numpy.zeros([50,3]).reshape([150])
```

Your first Python Light control application

1. Run the Example: "python example.py 1"
2. Brows To: http://localhost:8000/
3. Investigate the Source Code
    a. Note the __main__ block at the bottom
    b. Note the options parser, try running with "--help"
    c. Note the loop through the matrix (as an array)
4. Follow Instruction to Alter the Program
5. Refer here for more on Numpy

Ideas for projects:

Based on Kinet:

Light that responds to sensors, audio, etc.
Fixture Subclass that.. ..pairs with another fixture!
..knows its location relative to others!
Fixture Collection that allows grouping within PDS

Based on BluewayPx:

Implement Vertical fade! Create a timing engine!
Design Layout Manager to switch between different installations!
Add capabilities to make the web interface a better learning tool!

revision 1.0b

http://dmtaub.com
dmtaub@cemmi.org

Part Three: SaikoLED, Arduino, and Open Sound Control

You will additionally need at least one of following installed:

- *Arduino programming environment*
- *PyLiblo(might be easiest via easy_install)*
- *TouchOSC (free for Android) or Puredata*

Checking out SaikoLED code:

1. > git clone http://github.com/saikoLED/saiko5.git
2. > cd firmware/arduino-sketchbook
3. > git checkout -b _your_name_here_
4. > ln -s * ~username/sketchbook/
5. Puredata only: go to software/puredata and follow README

Modifying code:

1. Open smooth_fade.pde
2. Try to remove the blue section of the fade: only red and green
3. What do you notice about the colors you see?

moar hacking time....

revision 1.0b

© 2012 Daniel M. Taub
http://dmtaub.com
dmtaub@cemmi.org