Notizen

- Ordner anlegen, catlight module reinkopieren.
- Erst catlight module in bpython zeigen.

```
>>> import sender
>>> s = sender.start_sender()
** Starting SenderThread
** Created pipe to catlight (# 3547)
>>> import color
>>> s.send(color.Color(255, 255, 0))
>>> s.send(color.Color(255, 0, 255))
>>> s.send(color.Color(255, 0, 255, time=10000))
>>> s.send(color.Color(255, 255, 255, time=10000))
>>> s
<SenderThread(Thread-1, started 140615919367936)>
>>> s.list_queue()
[(255, 255, 255, 10000)]
>>> s.send(color.Color(255, 255, 255, time=10000))
>>> s.list_queue()
[(255, 255, 255, 10000), (255,255, 255, 10000)]
>>> import effects
>>> s.send(effects.SimpleFade())
>>> s.send(effects.SimpleFade())
>>> s.send(effects.SimpleFade(speed=0.5, color=color.Color(255, 0, 255)))
>>> s.stop()
```

- `run.sh` ausführen. `mkdir templates`
- Flask Grundgerüst aufbauen.
- Erste View Funktion einführen.

Funktionsreihenfolge:

- `api_rgb()`
- `api_rgb_time()`
- `api_list()`
- `sysinfo()`
- `root()`

rest.py

```
import json
import os
import sys

from flask import Flask, Response, render_template, request, redirect, url_for
import sender
import color

app = Flask(__name__)
queue = sender.start_sender()


def set_rgb_time(r, g, b, time):
    c = color.Color(r, g, b, time)
    queue.send(c)
    return json.dumps([c.red, c.green, c.blue])


@app.route('/api/r/<int:r>/g/<int:g>/b/<int:b>')
def api_rgb(r, g, b):
    return set_rgb_time(r, g, b, 0)


@app.route('/api/r/<int:r>/g/<int:g>/b/<int:b>/time/<int:time>')
def api_rgb_time(r, g, b, time):
    return set_rgb_time(r, g, b, time)


@app.route('/api/list')
def api_list():
    json_list = []
    for col in queue.list_queue():
        json_list.append({
            'rgb': [col.red, col.green, col.blue],
            'time': col.time
            })
    return Response(json.dumps(json_list), mimetype='application/json')


@app.route('/sysinfo')
def sysinfo():
    uname = os.uname()
    sysinfo_all = {
            'OS': uname[0],
            'Host': uname[1],
            'Version': uname[2],
            'Description': uname[3],
            'Arch': uname[4]
    }
    return render_template('sysinfo_all.html', pyversion=sys.version,
            sysinfo=sysinfo_all, show_version=True)


@app.route('/', methods=['POST', 'GET'])
def root():
```

```
    if request.method == 'POST':
        try:
            red = int(request.form['red'])
            green = int(request.form['green'])
            blue = int(request.form['blue'])
            queue.send(color.Color(red, green, blue))
        except:
            print('Some error happened - Redirecting.')
        finally:
            return redirect(url_for('root'))
    else:
        return '''<form action="" method="post">
                    <p>R: <input type=text name=red></p>
                    <p>G: <input type=text name=green></p>
                    <p>B: <input type=text name=blue></p>
                    <p><input type=submit value=submit></p>
                </form>'''

if __name__ == '__main__':
    app.run(debug=True, host='0.0.0.0', port=5000)
```

templates/sysinfo.html

```
<html>
    <body>
        <!-- #2 -->
        {% if show_version %}
        <h2>
            <!-- #1 -->
            This is Pythonz version: {{ pyversion }}
            <!-- #1 -->
        </h2>
        {% endif %}
        <!-- #2 -->

        <!-- #3 -->
        <table border="1">
            {% for key, value in sysinfo.items() %}
            <tr>
                <td>{{ key   }}</td>
                <td>{{ value }}</td>
            </tr>
            {% endfor %}
        </table>
        <!-- #3 -->
    </body>
</html>
```