

# Examples

## ! Tip

The following examples can be downloaded from the [edurov examples folder](#).

## Minimal working code

This is a bare minimum example so that the image stream and nothing more can be seen in the browser. A great starting point if you want to expand the functionality yourself.

minimal.py

```
from os import path

from edurov import WebMethod

web_method = WebMethod(
    index_file=path.join(path.dirname(__file__), 'index.html')
)
web_method.serve()
```

index.html

```
<!DOCTYPE html>
<html>
<head>
  <title>Minimal</title>
</head>
<body>
  
  <a href="stop">Stop Server</a>
</body>
</html>
```

```
project
├── minimal.py
└── index.html
```

## Features

An example created to explain most of the features in the edurov package. See the *Getting started* page in the [official documentation](#) for a full walkthrough.

features.py

```
import os
import subprocess

import Pyro4

from edurov import WebMethod

def my_response(not_used, path):
    """Will be called by the web server if it not able to process by itself"""
    if path.startswith('/cpu_temp'):
        cmds = ['/opt/vc/bin/vcgencmd', 'measure_temp']
        return subprocess.check_output(cmds).decode()
    else:
        return None

def control_motors():
    """Will be started in parallel by the WebMethod class"""
    with Pyro4.Proxy("PYRONAME:KeyManager") as keys:
        with Pyro4.Proxy("PYRONAME:ROVSyncer") as rovsyncer:
            while rovsyncer.run:
                if keys.state('K_UP'):
                    print('Forward')
                elif keys.state('K_DOWN'):
                    print('Backward')
                elif keys.state('K_RIGHT'):
                    print('Right')
                elif keys.state('K_LEFT'):
                    print('left')

# Create the WebMethod class
web_method = WebMethod(
    index_file=os.path.join(os.path.dirname(__file__), 'index.html'),
    runtime_functions=control_motors,
    custom_response=my_response
)

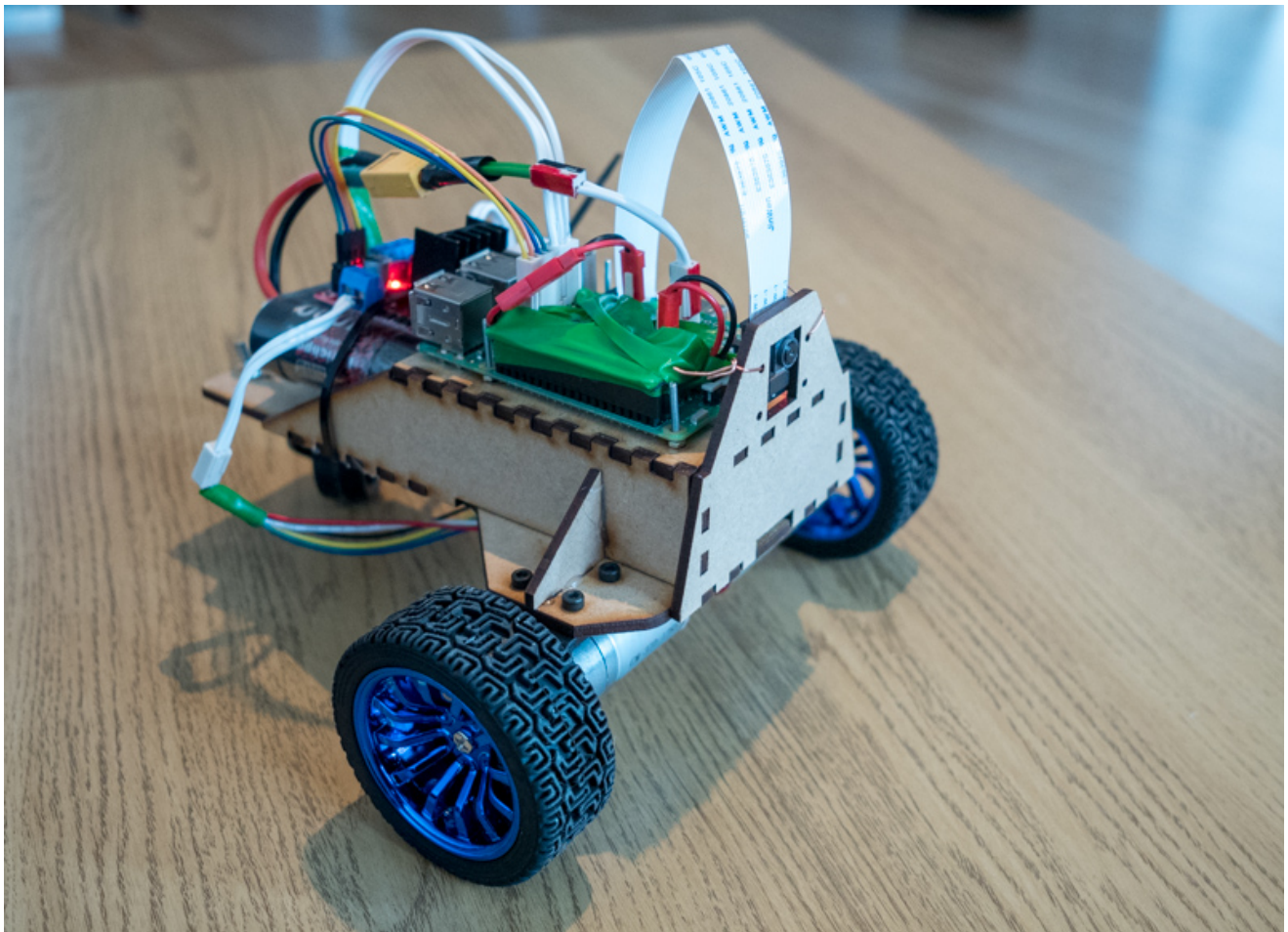
# Start serving the web page, blocks the program after this point
web_method.serve()
```

index.html

```
<!DOCTYPE html>
<html>
<head>
  <title>Features</title>
  <link rel="stylesheet" type="text/css" href="./static/style.css">
  <script src="./static/keys.js"></script>
  <script src="./static/extra.js"></script>
</head>
<body>
  <main>
    <h2>Welcome to the features example</h2>
    
    <p>
      <a href="stop">Stop server</a>
      <button onclick="cpuTemp()">Display CPU temp</button>
    </p>
    <p>
      Use arrow keys to print statements in the terminal window.
    </p>
  </main>
</body>
</html>
```

```
project
├── features.py
├── index.html
├── static
│   ├── keys.js
│   ├── extra.js
│   └── style.css
```

## Wireless RC car with camera feed



Create your very own wireless RC car with camera! The streaming video can be viewed in a browser on any device on the same network, it is controlled by using the arrow keys on the keyboard.

## Bill of materials

Name	Price USD	Comment
Raspberry Pi Zero WH	18	A full size board can also be us
Raspberry Pi Camera Module V2	33	
DC 6V 210RPM Geard Motor Wheel Kit	23	found on eBay
L298N Dual H Bridge Motor Controller Board	1.8	found on eBay
DC-DC 5V 12V Step Down Module Converter 3A	1.6	found on eBay
Total	76	

In addition you will need a swivel wheel, M3/M2.5 bolts and nuts, cables and connectors, 12V battery and a car frame. The car frame used in the picture above was cut from 3mm MDF with a laser cutter and can be found [here](#).

## CAD files

Visit <https://grabcad.com/library/772279>

```
project
├── rc_car.py
├── index.html
├── electronics.py
├── static
│   └── keys.js
```

## Engage eduROV

This example is used to control the ROV used in the eduROV project, see [www.edurov.no](http://www.edurov.no).

start.py 

```

import os
import time

import Pyro4

from edurov import WebMethod
from edurov.utils import detect_pi, serial_connection, send_arduino, \
    receive_arduino, free_drive_space, cpu_temperature

if detect_pi():
    from sense_hat import SenseHat

def valid_arduino_string(arduino_string):
    if arduino_string:
        if arduino_string.count(':') == 2:
            try:
                [float(v) for v in arduino_string.split(':')]
                return True
            except:
                return False
    return False

def arduino():
    lastState = '0000'
    ser = serial_connection()
    # 'Letter': [position, value]
    config = {'w': [0, 1],
              's': [0, 2],
              'a': [1, 1],
              'q': [1, 2],
              'd': [2, 1],
              'e': [2, 2]}
    with Pyro4.Proxy("PYRONAME:KeyManager") as keys:
        with Pyro4.Proxy("PYRONAME:ROVSyncer") as rov:
            keys.set_mode(key='l', mode='toggle')
            while rov.run:
                dic = keys.qweasd_dict
                states = [0, 0, 0, 0]
                for key in config:
                    if dic[key]:
                        states[config[key][0]] = config[key][1]
                states[3] = int(keys.state('l'))
                state = ''.join([str(n) for n in states])
                if state != lastState:
                    lastState = state
                    if ser:
                        send_arduino(msg=state, serial_connection=ser)
                    else:
                        print(state)
                if ser:
                    arduino_string = receive_arduino(serial_connection=ser)
                    if valid_arduino_string(arduino_string):
                        v1, v2, v3 = arduino_string.split(':')
                        rov.sensor = {
                            'tempWater': float(v1),
                            'pressureWater': float(v2),
                            'batteryVoltage': float(v3)
                        }

def senser():
    sense = SenseHat()
    with Pyro4.Proxy("PYRONAME:ROVSyncer") as rov:
        while rov.run:
            orientation = sense.get_orientation()
            rov.sensor = {'temp': sense.get_temperature(),

```

```

        'pressure': sense.get_pressure() / 10,
        'humidity': sense.get_humidity(),
        'pitch': orientation['pitch'],
        'roll': orientation['roll'] + 180,
        'yaw': orientation['yaw']}

def system_monitor():
    with Pyro4.Proxy("PYRONAME:ROVSyncer") as rov:
        while rov.run:
            rov.sensor = {'free_space': free_drive_space(),
                          'cpu_temp': cpu_temperature()}
            time.sleep(10)

def main(video_resolution='1024x768', fps=30, server_port=8000, debug=False):
    web_method = WebMethod(
        index_file=os.path.join(os.path.dirname(__file__), 'index.html'),
        video_resolution=video_resolution,
        fps=fps,
        server_port=server_port,
        debug=debug,
        runtime_functions=[arduino, senser, system_monitor]
    )
    web_method.serve()

if __name__ == '__main__':
    main()

```

index.html

```

<html>
<head>
  <title>eduROV</title>
  <script src="./static/dynamic.js"></script>
  <script src="./static/general.js"></script>
  <script src="./static/keys.js"></script>
  <link rel="shortcut icon" href="favicon.ico" type="image/x-icon">
  <link rel="icon" href="favicon.ico" type="image/x-icon">
  <link rel="stylesheet" type="text/css" href="./static/style.css">
  <link rel="stylesheet" type="text/css" href="./static/bootstrap.css">
</head>
<body onload="set_size()">

<div class="grid-container">
  <div class="d-none d-md-block side-panel " style="display:none;">
    <div class="card bg-light cinema">
      <h5 class="card-header">Sensors</h5>
      <div class="card-body">
        <h5>ROV</h5>
        <table class="table table-hover table-sm">
          <tbody>
            <tr>
              <th scope="row">Temperature</th>
              <td id="temp"></td>
              <td>&#8451</td>
            </tr>
            <tr>
              <th scope="row">Pressure</th>
              <td id="pressure"></td>
              <td>kPa</td>
            </tr>
            <tr>
              <th scope="row">Humidity</th>
              <td id="humidity"></td>
              <td>%</td>
            </tr>
            <tr>
              <th scope="row">Pitch</th>
              <td id="pitch"></td>
              <td>&#176</td>
            </tr>
            <tr>
              <th scope="row">Roll</th>
              <td id="roll"></td>
              <td>&#176</td>
            </tr>
            <tr>
              <th scope="row">Yaw</th>
              <td id="yaw"></td>
              <td>&#176</td>
            </tr>
          </tbody>
        </table>
        <h5>Water</h5>
        <table class="table table-sm">
          <tbody>
            <tr>
              <th scope="row">Temperature</th>
              <td id="tempWater"></td>
              <td>&#8451</td>
            </tr>
            <tr>
              <th scope="row">Pressure</th>
              <td id="pressureWater"></td>
              <td>kPa</td>
            </tr>
          </tbody>
        </table>
      </div>
    </div>
  </div>

```



```

    </div>
</div>
<div class="card bg-light cinema">
  <h5 class="card-header">System</h5>
  <div class="card-body">
    <table class="table table-sm">
      <tbody class="table-borderless">
        <tr id="voltageTr">
          <th scope="row">Battery</th>
          <td id="batteryVoltage"></td>
          <td>V</td>
        </tr>
        <tr id="diskTr">
          <th scope="row">Disk space</th>
          <td id="free_space"></td>
          <td>MB</td>
        </tr>
        <tr id="cpuTr">
          <th scope="row">CPU temp</th>
          <td id="cpu_temp"></td>
          <td>&#8451</td>
        </tr>
      </tbody>
    </table>
  </div>
</div>
<div class="center-panel">
  
  
</div>
<div class="d-none d-md-block side-panel">
  <div class="card bg-light cinema">
    <h5 class="card-header">Options</h5>
    <div class="card-body">
      <button type="button" onclick="toggle_armed()" id="armBtn"
        class="btn btn-outline-success btn-sm btn-block"
        title="Use this to arm the robot">
        Arm
      </button>
      <button type="button" onclick="rotate_image()"
        class="btn btn-outline-primary btn-sm btn-block"
        title="Will rotate the video 180 degrees">
        Flip video
      </button>
      <button type="button" onclick="toggle_roll()" id="rollBtn"
        class="btn btn-outline-primary btn-sm btn-block active"
        title="Toggle the roll indicator on/off">
        Roll
      </button>
      <button type="button" onclick="toggle_cinema()"
        class="btn btn-outline-primary btn-sm btn-block"
        title="Toggle cinema mode which hides everything except video">
        Cinema
      </button>
      <button type="button" onclick="set_update_frequency()"
        class="btn btn-outline-primary btn-sm btn-block"
        title="Changes the sensor update frequency to desired value">
        Sensor frequency
      </button>
      <button type="button" onclick="toggle_light()" id="lightBtn"
        class="btn btn-outline-warning btn-sm btn-block"
        title="Toggle the light on the ROV on/off">Light
      </button>
      <button type="button" onclick="stop_rov()"
        class="btn btn-outline-danger btn-sm btn-block"
        title="Stops the ROV, this page will stop working">
        Shutdown
      </button>
    </div>
  </div>
</div>

```

```

        </div>
    </div>
    <div class="card bg-light cinema">
        <h5 class="card-header">Hotkeys</h5>
        <div class="card-body">
            <table class="table table-sm">
                <tbody>
                    <tr>
                        <td><b>F11</b></td>
                        <td>Fullscreen</td>
                    </tr>
                    <tr>
                        <td><b>L</b></td>
                        <td>Lights</td>
                    </tr>
                    <tr>
                        <td><b>C</b></td>
                        <td>Cinema</td>
                    </tr>
                    <tr>
                        <td><b>ENTER</b></td>
                        <td>Arm</td>
                    </tr>
                </tbody>
            </table>
        </div>
    </div>
</div>
</div>
</body>
</html>

```

```

project
├── entry.py
├── start.py
├── index.html
├── static
│   ├── keys.js
│   ├── general.js
│   ├── dynamic.js
│   ├── roll.png
│   ├── bootstrap.css
│   └── style.css

```