Lesson Six:
Pressing Keys

Myo™

Lesson Six:
Pressing Keys

Overview

Last lesson, we looked at detecting gestures and outputting them to the debug console. We then looked at how to detect a specific gesture, and assign that gesture a function.

This lesson goes deeper into assigning functions for each pose, so each pose can perform an action. In this case, we will be producing button presses that can easily be tested on any window. We will also take a quick glance at how the application works if the user switches arms.

Objective

Once this lesson is finished, you should be able to make a simple connector for any program you want to implement the Myo armband with. While the implementation will be basic, the same concept will be used for any future projects.

Goals

- Learn how to use keyboard output
- Assign different outputs to each Myo pose

Lesson Length (Time)

60 Minutes

PART 1: USING ONPOSEEDGE

Using the same implementation as the previous lesson, our `onPoseEdge` function looks something like this:

```
function onPoseEdge(pose, edge)
   myo.debug("onPoseEdge: " .. pose .. ": " .. edge)

   if (edge == "on") then
          if (pose == "waveOut") then
                 onWaveOut()
          elseif (pose == "waveIn") then
                 onWaveIn()
          elseif (pose == "fist") then
                 onFist()
          elseif (pose == "fingersSpread") then
                 onFingersSpread()
          end
   end
end
```

We are only concerned with when a `pose` starts, so `edge` must be `on` every time. There is no `else` condition inside the `if` statement checking for poses either, since there are other poses we are not handling at this time.

If you save and load this code, you will notice an error message every time you try a gesture. This is because none of the inner functions (onWaveOut, onWaveIn, ...etc) have been implemented.

PART 2: KEYBOARD OUTPUT

Myo Scripts focus on using the Myo armband to trigger key presses. Once you detect the user is using the right application and they have done the right gesture, you can trigger a key or mouse press.

To press a key you will use the `myo.keyboard(keyname, edge, modifiers..)` function. This takes the name of any key, an `edge` (one of `down`, `up` or `press`), and any number of modifiers (`shift`, `control`, `alt`, `command`, or `win`). For `edge`, `down` and `up` correspond to whether the key is being pressed down or released. `press` is down then up right away. You'll have to figure it out within the application you are scripting for whether `down`, `up`, or `press` is more appropriate.

For example, if you wanted to press tab you would type the following code:

```
myo.keyboard(“tab”, “press”)
```

PART 3: CREATING FUNCTIONS

Our first build will use generic keyboard commands. "Tab" usually takes you to the next thing you can focus on. "Shift-tab" will take you backwards. "Return" (the same as pressing "Enter") will select something. "Esc" will back up or take you to a menu,

Map `waveOut` to tab, `waveIn` to tab with shift pressed, `fist` to return, and `fingersSpread` to escape. Try implementing each of those functions we defined to trigger those key presses, and print some debug output so you can verify that it is the command you are inputting. You can find out more information on what keys are called in the scripting API reference[1].

Your functions may look something like this:

```
function onWaveOut()
   myo.debug(“Next”)
   myo.keyboard(“tab”, “press”)
end

function onWaveIn()
   myo.debug(“Previous”)
   myo.keyboard(“tab”, “press”, “shift”)
end

function onFist()
   myo.debug(“Enter”)
   myo.keyboard(“return”,”press”)
end

function onFingersSpread()
   myo.debug(“Escape”)
   myo.keyboard(“escape”, “press”)
end
```

You could modify things like using the `onPoseEdge`’s `edge` value for the `myo.keyboard edge` parameter to let users hold down buttons if they wanted, but we’re going to leave it as is for now. There is another interesting thing here that you may or may not have noticed: Try switching your myo to your other arm and tab around (after syncing Myo).

Notice that Wave out is still away from your body. If you started on your left arm, the Wave out to go forward might feel more intuitive. The Myo UX guidelines[2] recommend waving to the “right” to be forward, and “left” to be back. We may want to switch Wave out and Wave in ourselves to support left handed use, but still allow right handed use. This will be covered in the **Challenge Activities** section, as this implementation is not necessary.

Challenge Activities

1. Write the code that correlates with each action using the `myo.keyboard` function. Note: You do not have to write a working script for these questions.

a) Hold down the “A” key.

b) Spell “Myo”.

2. `myo.getArm()` allows us to determine which arm the Myo armband is on. This will return either `left`, `right`, or `unknown`. Modify `onPoseEdge` so it outputs to the debug console the current arm Myo is being worn.

3. Write a helper function that switches `waveOut` for `waveIn` and `waveIn` for `waveOut` if the user is wearing their Myo on their left arm. For example, If the function receives `waveOut` as an input, it should return `waveIn` as it’s output. This is to allow the script to work if the user switches arms.

4. Implement the function from Question 1 into your `onPoseEdge` function. The function should be called after a pose is received but before performing the gesture specific functions.

Footnotes

[1] https://developer.thalmic.com/docs/api_reference/platform/script-reference.html

[2] https://developer.thalmic.com/ux

Solutions

1. a)

```
myo.keyboard("a", "down")
```

1. b)

```
myo.keyboard("m", "press", "shift")
myo.keyboard("y", "press")
myo.keyboard("o", "press")
```

2.

```
scriptId = 'com.thalmic.examples.lessonsix'
scriptTitle = "Outputs Arm"
scriptDetailsUrl = ""

function onForegroundWindowChange()
        return true
end

function onPoseEdge()
        myo.debug("Arm: " .. myo.getArm())
end
```

3.

```
function conditionallySwapWave(pose)
        if myo.getArm() == "left" then
                if pose == "waveIn" then
                        pose = "waveOut"
                elseif pose == "waveOut" then
                        pose = "waveIn"
                end
        end
        return pose
end
```

4.

```
function onPoseEdge(pose, edge)
        myo.debug("onPoseEdge: " .. pose .. ": " .. edge)

        pose = conditionallySwapWave(pose)
        if (edge == “on”) then
                if (pose == “waveOut”) then
                        onWaveOut()
                elseif (pose == “waveIn”) then
                        onWaveIn()
                elseif (pose == “fist”) then
                        onFist()
                elseif (pose == “fingersSpread”) then
                        onFingersSpread()
                end
        end
end
```