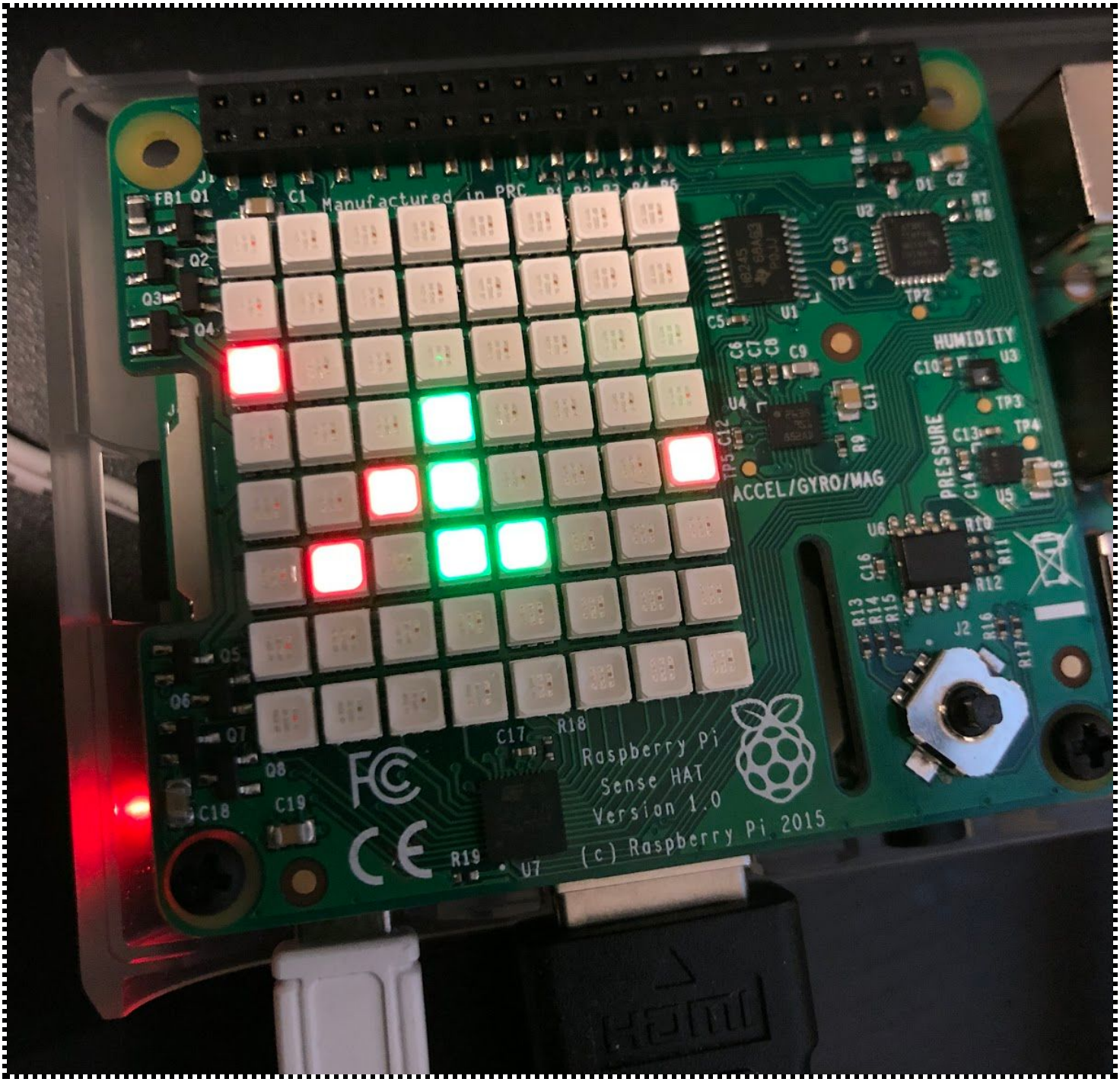
Sense HAT med Raspberry Pi

Rasmus, Lasse og Frederik



Indhold

Opgavebeskrivelse

I skal installere Rasbian og Python på Raspberry Pi'en. I skal så programmere snake som skal køre på Sense HAT interfacet og som skal kunne styres på en anden enhed over internettet eller bluetooth. Styringen skal kunne ske ved hjælp af et joystick på en app eller på nettet.

Problemformulering

Programmet består af disse ting:

- Spil
- Web server (til styring af spil)
    - Flask framework
- Front end
    - Joystick
    - Keyboard input

Spil

Spil logikken skal forgår ved man starter som en prik på led skærmen. Derefter skal man kunne bevæge sig via websiden. Man skal så finde mad på banen og gå ind i det for at spise maden.Når man spiser maden bliver man et hak længere. Hvis man støder ind i kanten af banen eller sig selv starter man forfra.

Web server

Til web server skal der findes et framework som skal kunne sende og modtage http request. Der skal oprettes routes til forsiden og en RESTful api til at bevæge slangen i spillet.

Frontend

Til frontend skal der laves så man kan sende http request til serveren.
Der skal laves et joystick som består af 5 knapper og ved tryk skal der sendes hvilken knap der er blevet trykket på til serveren

Det skal også være muligt og kunne trykke på en tast på tastaturet i stedet for joystick.

Der skal laves en menu så man kan remappe hvilke knapper som gør hvad.

Design

Til opgaven er der blevet brugt:

- Hardware
    - Raspberry pi 3 model B
    - Sense hat
- Software
    - Python
        - Flask
        - Sense Hat Lib
    - JavaScript (Front end)
        - Bootstrap
        - Jquery
        - Popperjs
        - Font Awesome
    - Html
    - Css

Raspberry pi

Til opgaven bruges en raspberry pi model 3 som er en micro computer. En raspberry pi kan bruges til små iot projekter og man har adgang til I/O Pins som man kan koble op med elektroniske enheder og på den måde styre dem.

Sense Hat

En sense hat er en enhed som man kan sætte oven på en raspberry pi ved hjælp af raspberry's I/O pins. En Sense Hat er et 8x8 Led display som man kan styre. Det har også indbyggede sensorer som: Gyro sensor, Temperatur måler og dvs.

Python

Python er et scripting sprog. Python bruges ofte til ML og automation, men bliver også brugt til webservere.

Flask

Flask er et lille libary til python som skal gøre det nemmere at bygge en webserver og forbinde routes.

JavaScript

JavaScript er det mest brugte sprog JavaScript kan køres i browser og på en server. JavaScript er en af de ene sprog som natively køre i en web browser (ud over WebAssembly). JavaScript er det eneste sprog som kan gøre tingene ordentligt i dag.

Bootstrap

Bootstrap er et libary til JavaScript og CSS og er lavet til at lave layout og design i browser noget nemmere. Bootstrap er bygget på CSS flexbox. Man bruger Bootstrap ved hjælp af et 12 points system hvor en side er delt op i 12 felter og man kan så place elementer i felterne og få dem til at skalere.

Jquery

Jquery er et lib til JavaScript og blev originalt lavet til at kunne udføre dom manipulation på samme måde i flere forskellige browser da der ikke var så god support. I dag (da dom manipulation er standardiseret) bruger man Jquery for at gøre dom manipulation nemmere.

Font Awesome

Er en CSS font og man bruger den til iconer til ens browser.

Arbejdsfordeling

Web server, forbindelsen mellem web server og klient og det meste af Snake spillet er lavet af Rasmus. Hjemmesidens design og funktionalitet er lavet af Frederik. Lasse var her ikke de to dage vi lavede alt koden og har derfor kun lavet en lille smule af Snake spillet den første dag. Dokumentationen er en opgave alle har deltaget med at lave.

Teori

Vores ide var at sætte en Raspberry op som brugte Raspbian som OS og bruge Python til at køre vores snake spil. Vores ide var så også at man skulle kunne styre spillet med telefonen over bluetooth eller anden måde såsom http. Python applikationen skulle stå for selve spillet og den skulle virke som server for klienten.

Klient siden af spillet skulle enten laves ved at forbinde en telefon som emulerede et joystick over bluetooth og så mappe knapperne i vores python script, eller også skulle klient siden laves ved at man brugte Flask i python til at hoste en webserver som så havde en grafisk controller på hjemmesiden som skulle være forbundet til snake spillet ved at bruge en api i javascript til at sende beskeder til flask som så ville blive behandlet og udført i spillet.

Dokumentation

Vi startede med at installere Raspian på sd kortet som sidder i Raspberry Pi, vi brugte et program som hed rufus til at formatere og lægge Raspian på sd kortet.

Derefter bootede vi Raspberry Pi'en og satte systemet op med sådan noget som tastatur, klokken, sprog og så installerede vi den dependency som vi skulle bruge i python til at hoste vores webserver som hedder flask, grunden til at vi valgte at lave en webserver i stedet for at bruge en app på en android telefon som emulerede et joystick var fordi at man skulle have en rooted telefon for at man kunne få telefonen til virke med den app vi ville bruge.

Da vi havde installeret alle de nødvendige programmer, startede vi med at lave et simpelt snake spil hvor det eneste man kunne var at styre slangen rundt på banen.
Da det var lavet begyndte vi at lave webserveren så vi kunne styre spillet fra andre steder end på Sense HAT joysticket. Til sidst lavede vi æblerne i spillet og gjorde så man døde hvis man gik ind i sig selv eller ind i væggen.

Routes:
GET: "/" Vil sende html forsiden tilbage.
GET: "/static/filename" returnere en filen med navnet på åarameter filename i public folder
POST: "/api/move" Skal modtage en JSON body med parameter "key" som bestemmer hvilken retning slangen bevæger sig i.

Konklusion

Snake programmet blev lavet rimlig hurtigt uden de stor fejl og vi hoppede derfor hurtigt over til controls.

Vi valgte ikke at bruge bluetooth da vi ikke havde en rooted android telefon hvilket besværliggjorde hvilket data vi kunne sende til raspberry pi'en og vi kunne ikke få noget data sendt overhovedet ind i event loggen.

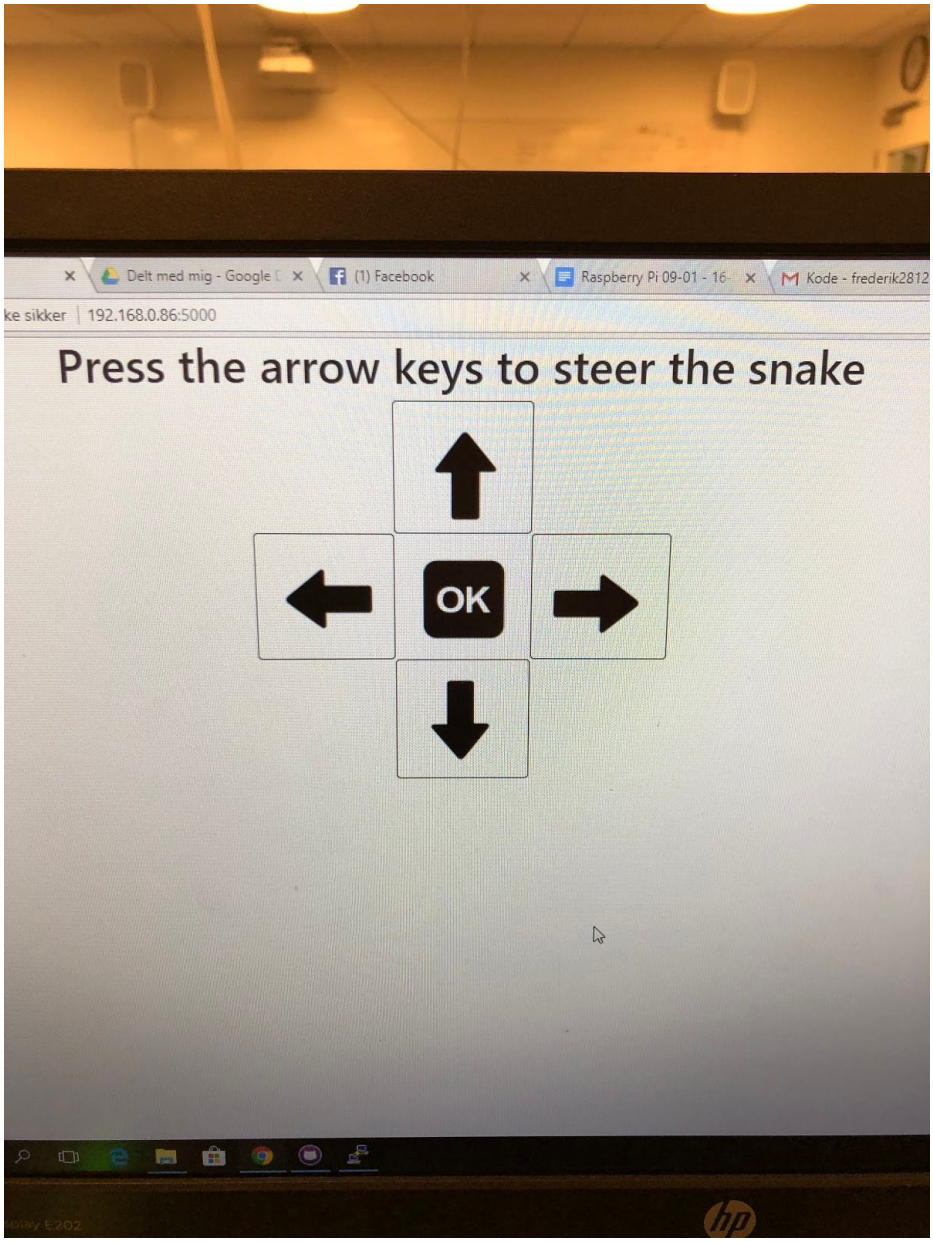
Så vi valgte et alternativ som er en bedre løsning og det var at oprette en webserver med Flask i python. På den måde kunne vi selv bestemme hvordan interface og controls skulle fungere.

Kilder

Sider

Flask Dokumentation: http://flask.pocoo.org/docs/0.12/
Sense HAT Dokumentation: https://pythonhosted.org/sense-hat/Billeder

Billede af hjemmeside



Python

```
#importing libs
import math
import time
import threading
from random import randint
from sense_hat import SenseHat
from flask import Flask, request, send_from_directory
#setting up web server
app = Flask(__name__, static_folder="public")

sense = SenseHat()

#initalize snake variables
food = [[0,2]]
snake = [[0,0], [1,1], [2,2]]
snake_speed = [1,0]
game_size = [8, 8]
paused = True

#will start the game loop
def start():
    update();

#will call it self on a new thread and up the game
def update():
    threading.Timer(0.20, update).start()
    if not paused:
        place_food(10)
        move()
        check_collision();
        sense.clear([0,0,0])
        draw_array(food, [255,0,0])
        draw_array(snake, [0,255,0])

#a small chance og spawning food at a random location
def place_food(chance):
    if (randint(0, 100) < chance):
```

```
        food.append([randint(0,game_size[0] - 1), randint(0,game_size[1] -
1)])

#will set the speed if the input is a new way
def set_speed(x, y):
    if abs(x) == abs(snake_speed[1]) and abs(y) == abs(snake_speed[0]):
        snake_speed[0] = x
        snake_speed[1] = y

#called each frame and will move the head and all the parts of the snake
def move():
    for body_part in range(len(snake) - 1, 0, -1):
        snake[body_part][0] = snake[body_part - 1][0]
        snake[body_part][1] = snake[body_part - 1][1]
    snake[0][0] += snake_speed[0]
    snake[0][1] += snake_speed[1]

#checks if the snake goes outside the map, hits itself or finds food.
def check_collision():
    if check_border(snake[0]) or collision(snake[0], snake[1:]):
        reset()
    else:
        item = collision(snake[0], food)
        if item:
            snake.append([snake[0][0], snake[0][1]])
            food.remove(item)

#resets the snake and food variables
def reset():
    snake.clear()
    snake.append([1,1])
    food.clear()
    snake_speed[0] = 0
    snake_speed[1] = 1

#take a vector and an array of vectors and checks if the single vector
overlaps any in the array.
def collision(obj, arr):
    for item in arr:
        if obj[0] == item[0] and obj[1] == item[1]:
            return item
```

```
    return False

#checks if the parameter head is outside the game.
def check_border(head):
    if head[0] >= game_size[0] or head[1] >= game_size[1] or head[0] < 0 or
head[1] < 0:
        return True
    return False

#Draws all vectors in an array to the sense with the color parameter
def draw_array(arr, color):
    for pos in arr:
        sense.set_pixel(pos[0], pos[1], color)

#sets up a callback to call set_speed according to the input
sense.stick.direction_up = lambda event: set_speed(event, 0, -1)
sense.stick.direction_down = lambda event: set_speed(event, 0, 1)
sense.stick.direction_left = lambda event: set_speed(event, -1, 0)
sense.stick.direction_right = lambda event: set_speed(event, 1, 0)

#starts the game
start()

#Routes

#default route
@app.route("/")
def main():
    return send_from_directory(app.static_folder ,"index.html")

#static route for files
@app.route("/static/<path:filename>")
def send_public(filename):
    print(filename)
    return send_from_directory(app.static_folder, filename)

#api route for posting new keypress
@app.route('/api/move', methods=['POST'])
def set_move():
    global paused
    content = request.get_json(silent=True)
```

```
    key = content["key"]
    print(key)

    if key == "ArrowUp": set_speed(0, -1)
    elif key == "ArrowRight": set_speed(1, 0)
    elif key == "ArrowDown": set_speed(0, 1)
    elif key == "ArrowLeft": set_speed(-1, 0)
    elif key == "Enter": paused = not paused

    return "Succes"

#starts web server
if __name__ == "__main__":
    app.run(port="5000", host='0.0.0.0')
```

HTML Kode

```
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Snake</title>
    <link rel="stylesheet" type="text/css" href="/static/style.css">
</head>
<body>
      <div class="show">
    <h1>Press the arrow keys to steer the snake</h1>
      </div>
    <div class="joystick_container">
        <div class="flexbox_container">
            <button id="up" class="btn btn-dark"></button></div>
        <div class="flexbox_container">
            <button id="left" class="btn btn-dark"></button>
            <button id="center" class="btn btn-light"></button>
            <button id="right" class="btn btn-dark"></button>
        </div>
        <div class="flexbox_container">
```

```
            <button id="down" class="btn btn-dark"></button>
        </div>
    </div>
    <button class="btn btn-light btn-lg pull-right collapsed"
data-toggle="collapse" data-target="#dropdown-menu" onclick="settings()">
        <i class="fa fa-cog fa-lg"></i>
    </button><div class="pull-right collapse" id="dropdown-menu" style="">
        <li class="dropdown-item">
            <ul><input id="up_setting" class="input_settings pull-left"
type="text" maxlength="1"></input>  up</ul>
            <ul><input id="down_setting" class="input_settings pull-left"
type="text" maxlength="1"></input> down</ul>
            <ul><input id="left_setting" class="input_settings pull-left"
type="text" maxlength="1"></input>  left</ul>
            <ul><input id="right_setting" class="input_settings pull-left"
type="text" maxlength="1"></input>  right</ul>
            <ul><input id="enter_setting" class="input_settings pull-left"
type="text" maxlength="5"></input>  pause</ul>
            </li>
      </div><script src="/static/script.js"></script>
      <link rel="stylesheet"
href="/static/lib/bootstrap-4.0.0-beta-dist/css/bootstrap.min.css">
    <link rel="stylesheet"
href="/static/lib/font-awesome-4.7.0/css/font-awesome.min.css">
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.11.0/umd/popper.min
.js"></script>
    <script src="/static/lib/jquery-3.2.1/jquery-3.2.1.min.js"
charset="utf-8"></script>
    <script src="/static/lib/bootstrap-4.0.0-beta-dist/js/bootstrap.min.js"
charset="utf-8"></script>
</body>
</html>
```

Css kode

```
.flexbox_container{
       display: flex;
```

```
}
.joystick_container{

}
.joystick_container .flexbox_container{
	justify-content: center;
}
.joystick_container button{
	width: 12%;
	text: auto;
}
.joystick_container button:after{
	content: "";
	display: block;
	padding-bottom: 100%;
}
h1{
	text-align: center;
}
.input_settings {
	width: 50px;
	margin: 0 auto;
}
#up{
	background-image: url("../public/images/up-arrow.jpg");
	background-repeat: no-repeat;
	background-size: 100%;
}
#right{
	background-image: url("../public/images/right-arrow.jpg");
	background-repeat: no-repeat;
	background-size: 100%;
}
#left{
	background-image: url("../public/images/left-arrow.jpg");
	background-repeat: no-repeat;
	background-size: 100%;
}
#down{
	background-image: url("../public/images/down-arrow.jpg");
	background-repeat: no-repeat;
	background-size: 100%;
}
#center{
	background-image: url("../public/images/ok.jpg");
	background-repeat: no-repeat;
	background-size: 100%;
}
```

Javascript kode

```
const upButton = document.querySelector("#up");
const downButton = document.querySelector("#down");
const leftButton = document.querySelector("#left");
const rightButton = document.querySelector("#right");
const centerButton = document.querySelector("#center");
var buttons =
[["w","ArrowUp"],["s","ArrowDown"],["a","ArrowLeft"],["d","ArrowRight"],["Enter","Enter"]];
const sendMove = (key) => {
   console.log(key);
    fetch("/api/move",{
        headers: {
          'Accept': 'application/json',
          'Content-Type': 'application/json'
        },
        method: "POST",
        body: JSON.stringify({key})
    })
};

const addClick = (element, post) => element.addEventListener("click",() => sendMove(post));

addClick(upButton, "ArrowUp");
addClick(downButton, "ArrowDown");
addClick(leftButton, "ArrowLeft");
addClick(rightButton, "ArrowRight");
addClick(centerButton, "Enter");
document.addEventListener("keydown", (key) =>
 buttons.forEach(function(item,count){
               if(item[0] == key.key) sendMove(item[1]);
               else if(item[1] == key.key) sendMove(key.key);
 }));

function settings(){
       if (document.getElementById("dropdown-menu").className.indexOf(' show') > -1){
              var obj = document.getElementsByClassName("input_settings")
              for(let i = 0; i < obj.length; i++)  buttons[i][0] = obj[i].value;
       }
       else{
       document.getElementById("up_setting").value = buttons[0][0];
```

```
        document.getElementById("down_setting").value = buttons[1][0];
        document.getElementById("left_setting").value = buttons[2][0];
        document.getElementById("right_setting").value = buttons[3][0];
        document.getElementById("enter_setting").value = buttons[4][0];
        }
}
```