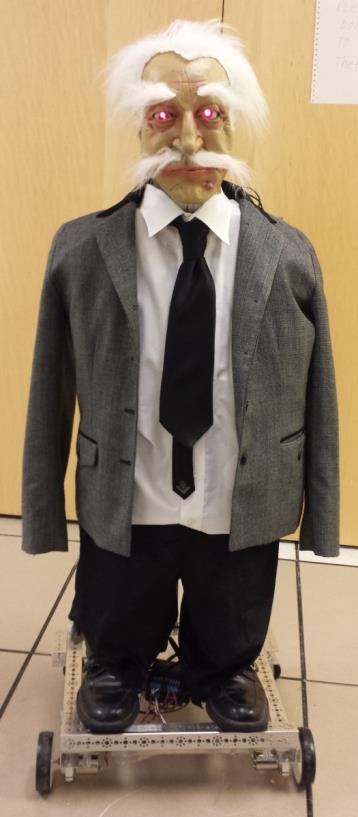
Final Report for Einstein group

ECE478 – Fall 2015



Students

William Harrington

David Hernandez

Waleed Alhaddad

**Table of Contents**

1. Introduction………………………………………………………………………3
2. Homework\_1 Outcomes…………………………………………………………4
   1. First phase Explanation……………………………………………………5
   2. Second Phase Explanation…………………………………………………7
   3. Results……………………………………………………………………9
3. Mask Modification………………………………………………………………10
4. Head/Body Modifications………………………………………………………11
5. Arduino Implementation…………………………………………………………12
6. Suggested Updates………………………………………………………………13
7. Appendix A – Arduino Sensor Shield Layout……………………………………14
8. Appendix B – Arduino Code……………………………………………………15
9. Appendix C – Bill of Materials …………………………………………………19
10. Appendix D – Github Code for HW1 ……………………………………………20
11. Appendix E – Powerpoint code for HW1 Macros ………………………………27
12. Appendix F – Fun ………………………………………………………………29

## Introduction

In this course our job is to integrate Einstein the robot head which has already been built from scratch 6 years before. Since the robot is 6 years old we needed to replace some old components such as main board, change some components connectors and use new powerful power source. Please see video on youtube: <https://www.youtube.com/watch?v=tNiKaAvjkLA>

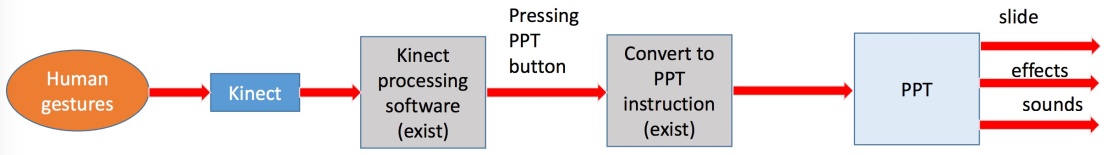
## Homework\_1 Outcomes

Part of Homework 1 which was to create a powerpoint presentation being controlled by Kinect and using fuzzy logic. We were told that our robot was going to be used in a play called “*The Great Quantum Debate*” so we decided to convert Einstein’s lines from the play to PowerPoint and use Kinect with fuzzy logic.

1. Use of Kinect to control a robot, create commands and data.
2. The concept of state machine in robotics
3. The concept and use of fuzzy logic in robotics
4. Using Powerpoint for scenario prototyping
5. Dialogs with robots

## 2.a ) First phase explanation

Figure 1: A high level diagram of the first phase objectives



**The objective for the first phase of this homework was to:**

1. Figure out how to use a Kinect to control the mouse on a computer
2. Figure out how to use Kinect to control a powerpoint presentation
3. Create a powerpoint presentation with info, effects, figures, pictures, and videos about Einstein and the ”Quantum Debate” play
4. Record voice with German accent that is suppose to be Einstein for the powerpoint presentation

In order to meet these objectives, our group did the following:

* We created a powerpoint presentation using Microsoft Powerpoint.
  + The powerpoint presentation contains
    - Famous quotes from Einstein
    - History about Einstein’s life, achievements, and hobbies
    - Einstein’s parts in the ”Great Quantum Debate”, Acts I and II
    - Lots of pictures of Einstein himself and things related to him
    - A voice with a german accent that reads what is on the slide
  + Within the powerpoint presentation, several macros were created using Microsoft Visual Basic for Applications.
    - Macros were used to make buttons that could be clicked on

with the mouse to transition to another slide

* We found software called “*KinectMouse*” for controlling a PC mouse and

powerpoint presentation using the Kinect

* + The software can be located [here](https://kinectmouse.codeplex.com/) [1](#_bookmark0)
  + There are detailed instructions on how to use this software [here](http://futuretechblog.com/?p=26) [2](#_bookmark1)
  + We also found a tutorial on how to use a face to control the mouse with this software [here](http://futuretechblog.com/?p=71) [3](#_bookmark2) but never had time to implement it

1https://kinectmouse.codeplex.com/

2<http://futuretechblog.com/?p=26>

3<http://futuretechblog.com/?p=71>

* We found a website that does text to sound in many different accents performed by either a male or a female voice called **IVONA**[4](#_bookmark3)
  + We used the male German accent to suit our robot Einstein
  + We needed to record the sound internally to have a better quality sound, so we used a program called **Audacity**[5](#_bookmark4)
  + We edited the recorded scripts by using a program called **Mixxx**[6](#_bookmark5)
  + All the programs the we used for the sound are available free online

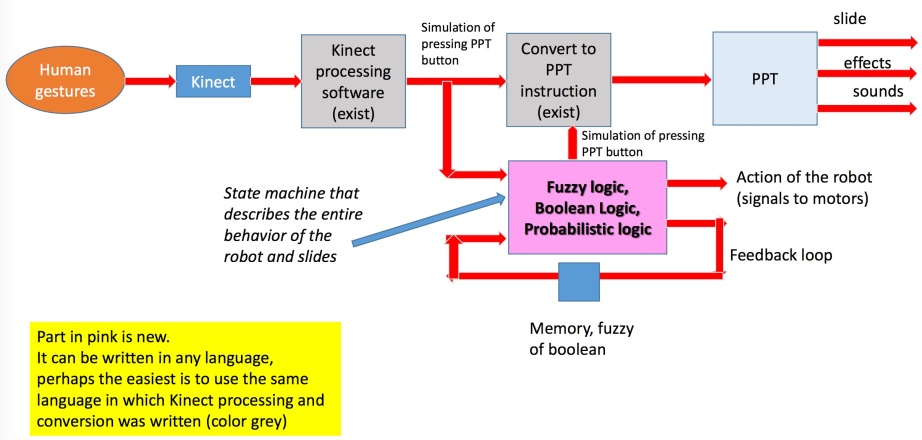
## Group roles for first phase

* Powerpoint: Will, David, Waleed
* KinectMouse: David
* Voice effects: Waleed
* Documentation: Will

4h[ttps://www.ivona.com/](http://www.ivona.com/) 5<http://www.audacityteam.org/> 6<http://mixxx.org/>

## 2.b ) Second phase explanation

Figure 2: A high level diagram of the second phase objectives



**The objective for the second phase of this homework was to:**

1. Create a state machine in software that describes the behavior of the robot, robots, and/or the entire theatre presentation
   * Could be deterministic, probabilistic, or fuzzy, or a mix of these.
   * Can have several machines communicating with one another.
   * Can be programmed in any language.
   * Should use Microsoft Powerpoint and Kinect software
2. Record a video demonstration

In order to meet these objectives, our group did the following:

* We chose to use python for programming the state machine
* A python class object was created to describe the behavior of the Ein- stein robot
  + It is appropriately titled ”Einstein”
  + It can be found [here](https://github.com/wrh2/ECE478/blob/master/Einstein/Einstein.py) [7](#_bookmark6)
  + It contains multiple ways to potentially control behavior of the robot
  + The behavior of the robot is determined by probabilistic logic using random number generators

7https://github.com/wrh2/ECE478/blob/master/Einstein/Einstein.py

* A python program was made to demonstrate the python class object
  + It is called ”main”
  + It can be found [here](https://github.com/wrh2/ECE478/blob/master/Einstein/main.py) [8](#_bookmark7)
  + It takes arguments from the command line, parses them and then calls the appropriate method in the Einstein python class object
    - Code requires the following python libraries to run: argparse,

random, Einstein.py (see footnotes on previous page)

* + - There is also a README located [here](https://github.com/wrh2/ECE478/blob/master/Einstein/README.md) [9](#_bookmark8)
* We introduced more macros within the powerpoint presentation
  + Some of them exhibit probabilistic logic (i.e. randomly choosing slides)
  + Some of them interface with the Python code
  + All buttons that are connected to macros were labelled accordingly
* We recorded a video demonstration of our project
  + Shows the use of the Kinect software to control the mouse and powerpoint
  + Shows effects and voices in powerpoint presentation
  + Shows the interaction between the powerpoint and python soft- ware

## Group roles for second phase

* Powerpoint: Will, David, Waleed
* Powerpoint macros: David
* Python programming: Will
* Video recording/editing: David, Will
* Documentation: Will

8https://github.com/wrh2/ECE478/blob/master/Einstein/main.py

9https://github.com/wrh2/ECE478/blob/master/Einstein/README.md

## 2.c ) Results

* Successfully used Kinect to control mouse and powerpoint presentation
* Successfully demonstrated interaction between powerpoint presenta- tion and python code
* Showcased fun and interesting information about Einstein using voice effects, images, and sounds in powerpoint
* All materials (presentation, video, report, code) can be found [here](https://github.com/wrh2/ECE478/tree/master/Einstein) [10](#_bookmark9)

10https://github.com/wrh2/ECE478/tree/master/Einstein

1. **Mask Modifications**

This project we try to improve the facial gesture and expression for the new plastic mask. Unfortunately, we didn’t have enough good masks that fits or match our character (Einstein), the closes match we found (see figure 1) unfortunately was stiff and hard to control by servos. We needed to modify the mask by making cuts to widen up some face parts such as eyelid and mouth.

We attempted to control the eye brows of the mask using Velcro and the servos that were located in that part of his head but the mask was to stiff and did not look like it was moving, it almost gave him a blinking motion since it was pulling the mask so much so we had to abandon this. We also attempted to resurrect the mouth using Velcro attached to the lips and servos located in the mouth but this also did give very much mouth movement, this was also abandon.

In order to overcome the fit the previous documentation mentioned using Velcro, so we decided to use it to but had some difficulties with it because it would not stay. The previous group mentioned glue to overcome the mounting issues but we thought this would be too messy. In the end we only had Velcro in the ears and overcame the stickiness by using a screw attached to the head to pull the mask over it to hold it in place note there were holes in the ear for this. We did this for both sides of the head; it was kind of like a quick facelift. It was pointed out that the forehead was sunken in, so we stuffed it with a bunched up handkerchief we found in the lab. Figure 2 shows after modifications. Note curly hair was lost during re-arrangement of the lab.



Figure 2 After Mods



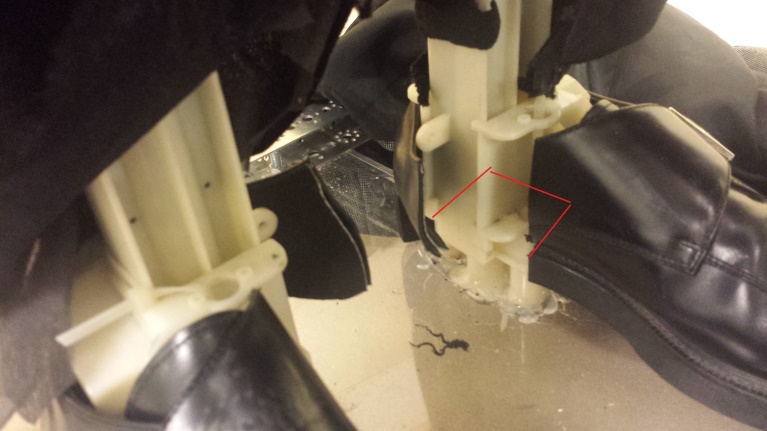
Figure 1 Before Mods

1. **Head/Body Modifications**

The third objective was to fix some of the robot, build parts and remove some requested parts by Prof. Perkawski. We have removed the arm that was attached to the robot base and placed it on the shelf to be reused in the future.

We needed to fix the position of one of the eyeball and how it was held in place. We ended up using a small screw to replace the broken pin to reattach it to its bracket. It was also noticed that one of the eyes is very loose which gave Einstein a lazy eye. We were unable to fix and decided not to since it gave him a little personality.  Also the LEDs for the eyes were falling out so we used hot glue to hold them in place.

We also found one servo that was bad (left ear) so we replaced it with a spare we found in a box full of servos which turned out was also broken too, searched again and found one that worked.

Head base was moving and was not held very well to the body, so we needed to drill through the wooden base into the plastic body base; now it does not move. We also fixed the forehead which was not attached; by using a small screw to secure it in place.

We attached shoes to Einstein with a screw going through the acrylic base into his shoes. His Shoes were modify by using power tools to cut out big sections in the heel in order for them to fit over the pipes that was holding the body to the acrylic wheel base (Red line represents cut in figure 4.1).

Figure 4. Modify Shoes

We have noticed that there was a speaker with bare two wires that was attached to the body that was not being used, so we soldered speaker terminals to it and connected it to an amplifier circuit board (see Appendix B for type used). The amplifier circuit had to be put together by some soldering of the 3.5mm audio jack, the power connector wires and the speaker wires. The small amplifier circuit was able to give us great amplification from our laptop even at low volume. This enabled Einstein to speak which was very useful since he has many speaking parts in the play.

In order to achieve mouth movement we implemented a bent wire hanger to push the bottom part of the mask down and connected to one of the servos in the right ear. A lot of time was spent on designing the mouth to give it an exaggerated movement with a stiff mask. Our first attempt was using a small piece of wire hanger and attaching it to the servos in the mouth (previously used for cheek movement) this gave it very little movement and was not very predictable since the wire moved freely. After deciding to use the servo in the ear more time was spent in custom bending the wire into a jaw shape and giving it enough length to reach the bottom of the mask for an exaggerated movement. This method proved promising, the only issue was that only one side of the wire was attached to the servo and the other side was free in order to give it freedom to move.

We did a lot of taping to secure the wires to the base to prevent them from breaking at the solder points or being pulled off the board, note it would be a good idea to secure them by staples since the tape will come lose as it gets older. Also the amplifier circuit board was secured to the base by tape since it was so small, so be careful when screwing items to the base that is covered with tape on one side.

**5. Arduino Implementation**

Originally Einstein’s head servos was controlled by board called “*Mini SSC II*” which through some google’ing we found out that the only way to send some commands was by using a phone-jack to serial to USB connector. This by today's standards is obsolete. So we use an up to date way to send commands to the servos an Arduino Uno. We chose Arduino Uno because it was relatively cheap $35, allows up to 14 digital I/O pins (2x more than the mini SSCII), USB connection (not serial) and easy programming. We also bought a Sensor shield (see appendix for board layout) in order to use the servo connections without any modifications. This allows us to easily connect all the servos the Arduino and to add Bluetooth and any other features later. We also used the bottom part of an Arduino case to hold the Arduino in place; this made mounting the Arduino to the back of the robot easier and safer from shorts caused by screws.

We started by first downloading an example servo code from Arduino website and modifying it to test our servos. We tested each servo (10 total) and recorded its min and max range of motion to achieve our desired effect. We then labeled each servo and wires connection to minimize any confusion. Our labeling technique consisted of a letter and a number. The numbering system came from how many servos there were in the head and to make it simpler the numbers matched digital I/O pin connection on the arduino. The lettering system consisted of the first letter of the facial area such as “E” for eyes, “N” for Neck, Mouth, Brows and Unknown.

We then created basic functions for facial/head movements such as eye, neck and mouth (see appendix A). Tested them with only that particular servo connected to the arduino and a final test with all the wires connected to the arduino. The final test did not go very well, the head had very erratic movements due to some random servos activating. We suspected it was because of a power issue since we tested all the servos for an extended period of time we may have reduced our battery capacity to below more than one servo running. We changed batteries and the issue went away.

One issue that kept coming back was when we started using some particular servos we would get some sound feedback coming from the speakers we connected. This could be because we were borrowing the power from a servo to turn on the speaker amplifier. Some shielding is probably needed to prevent this from happening.

We used a 9v battery to power the Arduino when it was not connected to the laptop; luckily we found a small switch connected with a power plug and 9v connector. We also used a 4xAA battery pack to power all the servos; this power pack came with a built in power switch which was very useful. We cut some jumper cables with alligator clips to use as the power connection from the Arduino to the battery pack, just in case if we ever want to change the type of power source. We also cut Einstein shirt to make easier to access the board and power without removing his shirt all time and used Velcro to close it back up.

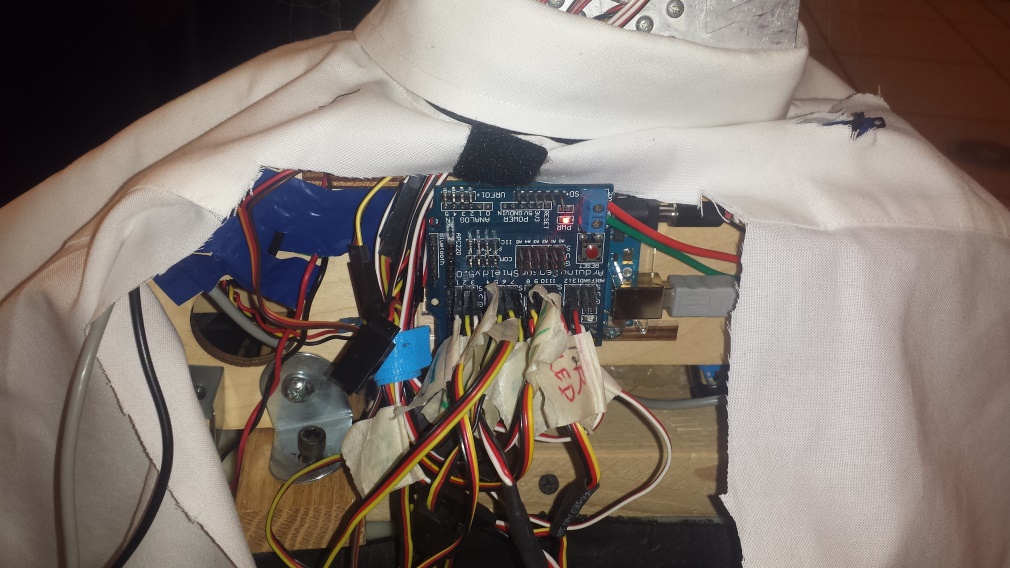
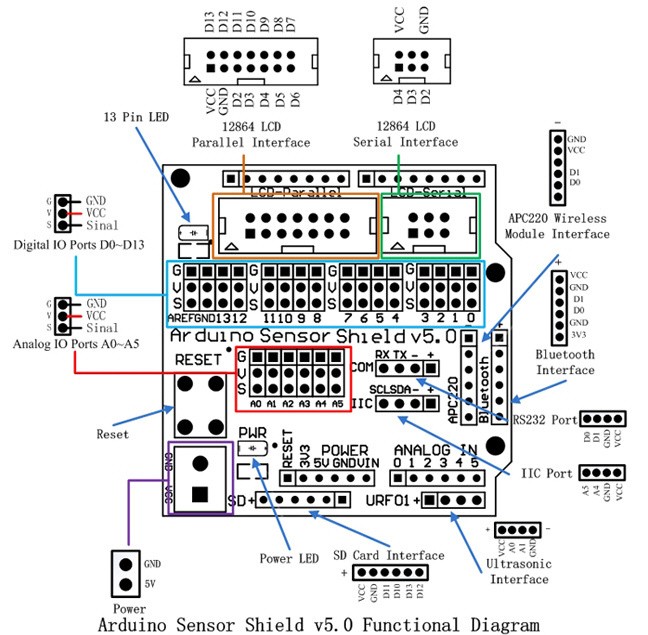


Figure 5.1 Arduino

1. **Suggested Updates**

* Bluetooth to send commands to Einstein part was bought but were unable to fully integrate it since there was some difficulty with the coding part for Bluetooth.
* Change the use of tape to secure the wires to staples.
* Jaw upgrade for mouth movement instead of wire hanger. Suggest human like jaw maybe with some teeth, this will allow for much greater believable mouth movement.
* Extra DOF in neck for ‘yes’ motion movement, maybe adding another axel to neck.
* Arms It was also suggested that Arms will be attached at some point
* Resurrect Base motors - Note when the arm was removed from Einstein this also removed the components needed to control the base motors.

**Appendix A - Arduino Sensor Shield layout**



**Appendix B - Arduino Code**

// einstein Head Arduino Code ECE478 - Fall15

// Waleed Alhaddad and David Hernandez

// Using arduino Uno R3 with Sensor Shield

#include <Servo.h> // servo library

// Servo variables using wire labeling

Servo E7, E2; // eyes UpDwn=E7, LeftRight=E2

Servo M3, M5; // cheeks/mouth Right=M3, Left=M5

Servo B4, B6; // Brows Right=B4, Left=B6

Servo N9, N10; // Neck sideTilt=N10, rotate"No"=N9

Servo U8, U1; // Unknown-using for Jaw Left=U8, Right=U1

void setup(){

// assigning variable to i/o pins

E2.attach(2); // moves eyes left/right

E7.attach(7); // moves eyes up/down

M3.attach(3); // right cheek

M5.attach(5); // left cheek

B4.attach(4); // right brow

B6.attach(6); // left brow

N9.attach(9); // rotate neck left/right "No"

N10.attach(10); // tilt neck side to side

//U8.attach(8); // Not used

U1.attach(1); // moving jaw up/down

pinMode(11, OUTPUT); // left eye Led

pinMode(12, OUTPUT); // right eye Led

digitalWrite(11, HIGH); // turn left eye LED On

digitalWrite(12,HIGH); // turn right eye LED on

//Serial.begin(9600); // Default connection rate for my BT module

// initial state for head

neckNeutral();

eyesNeutral();

}

void loop(){

//test();

// eyesON();

//eyesOFF();

//eyesBlink();

//eyeWink();

//browRight();

//shiftyEyes();

//UpDownEyes();

//mouthOpenCloseSlow();

mouthOpenCloseFast();

//motionNo();

// NeckTilt();

}

void test(){

// function used for indivudual servo testing purposes

// DOWN:15 -- MIDDLE 165 --- UP:200 E7

// Left:25 -- Middle: 90 -- Right:145 N9

// tiltright:45 -- tiltmiddle: 60 --tiltleft: 100 N10

//E7.write(127);

U1.write(100);

delay(500);

U1.write(150);

delay(1000);

//B6.write(170);

//delay(1U100);

//B4.write(70);

//delay(1000);

//E2.write(30);

//delay(1000);

//N10.write(00);

//delay(2000);

//E2.write(100);

// E7.write(0);

}

void motionNo(){

neckNeutral();

delay(500);

neckTurnRight();

delay(500);

neckNeutral();

delay(500);

neckTurnLeft();

delay(500);

}

void NeckTilt(){

neckTiltLeft();

delay(500);

neckNeutral();

delay(500);

neckTiltRight();

delay(500);

}

void mouthOpenCloseFast(){

U1.write(100);

delay(200);

U1.write(150);

delay(200);

}

void mouthOpenCloseSlow(){

U1.write(100);

delay(500);

U1.write(150);

delay(500);

}

void mouthOpen(){

U1.write(100);

delay(200);

}

void mouthClose(){

U1.write(150);

delay(200);

}

void shiftyEyes(){

eyesNeutral();

delay(500);

eyesRight();

delay(500);

neckNeutral();

delay(500);

eyesLeft();

}

void UpDownEyes(){

eyesUp();

delay(500);

eyesDown();

delay(500);

}

void neckNeutral(){

N9.write(90);

delay(500);

N10.write(60);

}

void neckTurnRight(){

N9.write(145);

}

void neckTurnLeft(){

N9.write(25);

}

void neckTiltLeft(){

N10.write(100);

}

void neckTiltRight(){

N10.write(30);

}

void eyesNeutral(){

E2.write(50);

delay(500);

E7.write(165);

}

void eyesLeft(){

E2.write(80);

}

void eyesRight(){

E2.write(10);

}

void eyesUp(){

E7.write(200);

}

void eyesDown(){

E7.write(15);

}

void browRight(){

B4.write(200);

delay(1000);

B4.write(10);

delay(1000);

}

void browLeft(){

B6.write(220);

}

void eyesBlink(){

eyesON();

delay(4000);

eyesOFF();

delay(500);

}

void eyesON(){

digitalWrite(11,HIGH ); // turn left eye LED On

digitalWrite(12,HIGH); // turn right eye LED on

}

void eyesOFF(){

digitalWrite(11, LOW); // turn left eye LED On

digitalWrite(12,LOW); // turn right eye LED on

}

void eyeWink(){

digitalWrite(12,LOW); // turn right eye LED off

delay(1000);

digitalWrite(12,HIGH); // turn right eye LED on

delay(1000);

}

**Appendix C – Bill of Materials**

|  |  |  |
| --- | --- | --- |
| **Item** | **Place Purchased** | **Price** |
| Arduino Uno R3 | RadioShack | 34.99 |
| Sensor Shield V5.0 | ebay | 2.30 |
| Arduino Uno Case | ebay | 2.00 |
| 3W+3W Dual Ch Pwr Digital Amp PAM8403 | ebay | 0.72 |
| KEDSUM® Arduino Wireless Bluetooth Transceiver Module Slave 4Pin Serial + DuPont Cable | Amazon | 9.99 |
| Velcro | HomeDepot | 4.00 |
| 3.5mm audio Jack | Goodwill | 1.00 |
| Double-ended Test Leads Jumper Alligator clips | Ebay | 2.00 |
| 4xAA Battery Box On/OFF switch | ebay | 1.00 |
| 9v battery connector+switch+pwr Plug | Found in lab | 0.00 |
| Batteries AA and 9v | Target | 4.00 |
| Various Screws and tape | Found in lab | 0.00 |
| White Dress Shirt + Black Tie | Goodwill | 1.50 |
| **Total** |  | **63.50** |

**Appendix D – Github Code for HW1**

**https://github.com/wrh2/ECE478/tree/master/Einstein**

# Einstein, the famous theatrical Robot from PSU!

## Table of contents

Einstein.py -- Python class the describes behavior of the imaginary Einstein robot

main.py -- example python program that utilizes Einstein python class

## Example usage(s)

Get einsteins personality traits. Niceness - measure of how nice; Meanness - measure of how mean

python main.py --personality

This should return a tuple of (niceness, meanness)

python main.py -mood

This should return a string that indicates Einstein's "mood"

python main.py -a "integer from 1-10"

This is a action (integer from 1-10) that einstein "reacts" to. Returns a string

python main.py -c "integer from 1-15"

This issues a command to the robot

\*\*\*\*\*\*\*\*\*\*\*\*\* Einstein.py\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

from \_\_future\_\_ import division

import random

# python class for einstein

class Einstein:

def command(self, num):

"""

This function takes in an integer and translates it to a command to the Einstein robot

"""

if(num == 1):

return 'Stop robot'

elif(num == 2):

return 'Robot forward'

elif(num == 3):

return 'Robot backward'

elif(num == 4):

return 'Robot raises right arm'

elif(num == 5):

return 'Robot raises left arm'

elif(num == 6):

return 'Robot moves forward fastly'

elif(num == 7):

return 'Robot moves backward fastly'

elif(num == 8):

return 'Robot turns 45 degrees left'

elif(num == 9):

return 'Robot turns 90 degrees left'

elif(num == 10):

return 'Robot turns 180 degrees left'

elif(num == 11):

return 'Robot turns 360 degrees left'

elif(num == 12):

return 'Robot turns 45 degrees right'

elif(num == 13):

return 'Robot turns 90 degrees right'

elif(num == 14):

return 'Robot turns 180 degrees right'

elif(num == 15):

return 'Robot turns 360 degrees right'

else:

return 'Invalid command'

def slide(self, num):

"""

This function takes in a slide number or interaction number

it then determines what action einstein takes

"""

# generate random int, convert to percentage

chance = random.randint(1,10)/10

# interaction 1.2 or slide 1.6

if(num == 1.2 or num == 1.6):

if(chance > self.niceness):

return 'Einstein attacks, Newton keeps distance'

else:

return 'Newton attacks, Einstein keeps distance'

# interaction 1.3 or slide 1.7

if(num == 1.3 or num == 1.7):

if(chance > self.niceness):

return 'Einstein looks angrily at Bohr'

else:

return 'Einstein calmly smokes his pipe, while looking intently at Bohr'

# slide 1.11

if(num == 1.11):

return 'Einstein grabs violin, points to Mary,' \

'then looks at Newton and Bohr, and directs them to Dr. Curie salon'

# slide 1.15

if(num == 1.15):

return 'Einstein angrily jumps up and down'

# interaction 2.2 or slide 1.17

if(num == 2.2 or num == 1.17):

if(chance > self.niceness):

return 'Einstein drives always to the person who talks but keeps distance'

else:

return 'Einstein turns always to the person who talks, keeping his distance'

# slide 1.18

if(num == 1.18):

if(chance > self.niceness):

return 'Einstein flicks everybody off'

else:

return 'Einstein sticks his tongue out'

# slide 1.19

if(num == 1.19):

return 'Einstein patronizes Newton then points to Schrodingers cat'

# interaction 2.3 or slide 1.20

if(num == 2.3 or num == 1.20):

return 'Einstein communicates intensely with schrodingers cat'

# slide 1.21

if(num == 1.21):

return 'Einstein goes on long tangent regarding Dark Energy and spacetime'

# slide 1.23

if(num == 1.23):

return 'Einstein turns to schrodingers cat'

# interaction 2.5 or slide 1.27

if(num == 2.5 or num == 1.27):

return 'Einstein moves towards Bohr until he is close, then dance forward and backward'

# interaction 2.5 or slide 1.27

if(num == 2.6 or num == 1.29):

return 'Einstein moves towards Bohr until he is close, then dance forward and backward'

# slide 1.32

if(num == 1.32):

if(chance > self.niceness):

return 'Einstein speaks sarcastically, and acts surprised'

else:

return 'Einstein is genuinley surprised'

# interaction 2.7 or slide 1.33

if(num == 2.7 or num == 1.33):

return 'Einstein loses control and says loudly: SHUT UP!'

# interaction 2.8 or slide 1.35

if(num == 2.8 or num == 1.35):

return 'Einstein dances in circles in the middle'

# more slides from play need to be added

return 'Invalid slide number'

def mood(self):

"""

Function returns string that represents mood

"""

#random chance

chance = random.randint(1,10) / 10

# nicer, and chance > meanness

if(self.niceness > self.meanness and chance > self.meanness):

return 'Einstein is groovy baby'

# nicer, chance < meanness

elif(self.niceness > self.meanness and chance < self.meanness):

return 'Einstein is ok'

# meaner, chance > meanness

elif(self.niceness < self.meanness and chance > self.meanness):

return 'Einstein is meh'

# meaner, chance < meanness

elif(self.niceness < self.meanness and chance < self.meanness):

return 'Einstein says fuck off!'

def personality(self):

"""

Function returns personality traits for Einstein

"""

return (self.niceness, self.meanness)

def reaction(self, Action):

"""

Function takes Action (integer 1 to 10)

and generates a reaction from Einstein

based on some probablistic logic

"""

# keep it within the proper range

if(Action > 10 or Action < 0):

Action = random.randint(1, 10)

# initialize chance

chance = 0

# add up five random integers between 1-10

for i in range(5):

chance += random.randint(1,10)

# divide by max (10, 5 times = 50)

chance /= 50

Action /= 50

# initialize chance2

chance2 = 0

# this weighs the reaction towards a nice one or a mean one

if(random.randint(1,10) <= 5):

chance2 = self.niceness

else:

chance2 = self.meanness

# get reaction

if(chance\*chance2 > Action):

return 'Einstein reacted nicely'

else:

return 'Einstein reacted badly'

def \_\_init\_\_(self, niceness, meanness):

"""

Initialization of Einstein

"""

# need to make sure we get reasonable numbers for niceness and meanness

if(niceness < 0 and meanness < 0 or niceness == meanness):

# niceness and meanness were both less than 0 which doesn't make sense

# or they were equal to each other which also doesn't make sense

# choose random number between 1 & 10 then divide by 10 for both

# this gives us a percentage of this quality

self.niceness = random.randint(1,10) / 10

self.meanness = random.randint(1,10) / 10

# need to make sure those numbers aren't equal to each other

while(self.niceness == self.meanness):

# they were so pick a new one

meanness = random.randint(1,10) / 10

# max is 10, min is 0

# convert into percentage

if(niceness > 10):

self.niceness = 1

elif(niceness < 0):

self.niceness = 0

else:

self.niceness = niceness / 10

if(meanness > 10):

self.meanness = 1

elif(meanness < 0):

self.meanness = 0

else:

self.meanness = meanness / 10

# this is some other stuff for reading text files that we will probably use later

#with open("fname.text", "r") as f:

# content = [x.strip('\n') for x in f.readlines()]

# for stuff in content:

# print stuff

\*\*\*\*\*\*\*\*\*\*\*\*\* main.py\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

"""

Example program of using command line arguments with Einstein class

Written by William Harrington for ECE478 HW1

"""

from Einstein import Einstein # grab the einstein class

import argparse # library for parsing command line arguments

import random # library for random numbers

# initialize argument parser

parser = argparse.ArgumentParser()

# these lines are here to show that we can make his personality traits customizable on the command line

#parser.add\_argument('-m', action = 'store', dest = 'meanness', required = True, help = 'Meanness is a personality quality of Einstein, required for init')

#parser.add\_argument('-n', action = 'store', dest = 'niceness', required = True, help = 'Niceness is a personality quality of Einstein, required for init')

# argument for getting mood of Einstein

parser.add\_argument('-mood', action = 'store\_true', help = 'gets einsteins mood')

# argument for getting personality traits (niceness, meanness) of Einstein

parser.add\_argument('--personality', action = 'store\_true', help = 'gets personality traits of Einstein')

# argument for asserting action that Einstein will react to

# should be integer between 1-10

parser.add\_argument('-a', action = 'store', dest = 'Action', required = False, help = 'optional argument for action to einstein to react to')

# argument for getting robot to perform command

parser.add\_argument('-c', type = int, action = 'store', dest = 'Command', required = False, help = 'optional argument for getting robot to perform command')

# parse the arguments

arguments = parser.parse\_args()

# again this line just shows we can customize the personality traits from the command line

#Einstein = Einstein(arguments.niceness, arguments.meanness)

# for example sake and testing purposes, I'm just gonna make it random for now

Einstein = Einstein(random.randint(1,10), random.randint(1,10))

if(arguments.Command):

print arguments.Command

print Einstein.command(arguments.Command)

# mood argument present

if(arguments.mood):

# show his mood

print Einstein.mood()

# personality argument present

if(arguments.personality):

# show personality traits

print Einstein.personality()

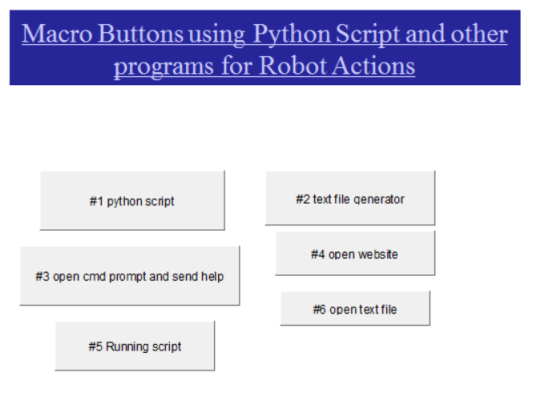
# action argument present

if(arguments.Action):

# show reaction of Einstein

print Einstein.reaction(arguments.Action)

**Appendix E – Powerpoint code for macros**



Private Sub CommandButton1\_Click() ' #1 python script

args = "C:\Users\Ultra\_Dav\Downloads\einstein\main.py" & "-mood"

Debug.Print args

Call Shell("C:\Python27\python.exe " & args, vbNormalFocus)

End Sub

Private Sub CommandButton2\_Click() ' #2 text file generator

Dim sPath As String

Dim sName As String

sName = "Move" ' name given to file

sPath = "C:\Users\Ultra\_Dav\Downloads\einstein\" & sName & ".txt" ' save location+filename+extension

Open sPath For Output As 1

Print #1, "Sad" ' text saved into file

Close #1

End Sub

Private Sub CommandButton3\_Click() ' #3 open cmd prompt and send help

Call Shell("cmd.exe /S /K" & "help", vbNormalFocus) ' /S modifies treatment of string after /K leaves window open, /C closes window

End Sub

Private Sub CommandButton4\_Click() ' #4 open website

Call Shell("C:\Program Files (x86)\Google\Chrome\Application\chrome.exe" & " -url" & " " & "stackoverflow.com", vbMaximizedFocus)

End Sub

Private Sub CommandButton5\_Click() ' #5 running script

Dim retval2

'retval2 = Shell("cmd.exe /S /K" & "C:\Users\Ultra\_Dav\Downloads\einstein\main.py -c 11", vbNormalFocus)

retval2 = Shell("cmd.exe /S /K" & "C:\Users\Ultra\_Dav\Downloads\einstein\main.py -mood", vbNormalFocus)

End Sub

Private Sub CommandButton6\_Click() ' #6 open text file

Dim retval

retval = Shell("notepad.exe C:\Users\Ultra\_Dav\Downloads\einstein\Move.txt", vbNormalFocus)

End Sub

Private Sub CommandButton1\_Click() ' fuzzy logic

Dim chosenNum As Integer

Randomize

chosenNum = Int(10 \* Rnd) + 1

If chosenNum < 5 Then

ActivePresentation.SlideShowWindow.View.GotoSlide 11

Else

ActivePresentation.SlideShowWindow.View.GotoSlide 12

End If

End Sub

**Appendix F – Fun**



David Einstein Waleed

