Sacred Heart University
School of Computer Science and Engineering

Mira

Interactive Smart Mirror

Ryan Nguyen, Peter Flemister, Tram Nguyen
D. Cenk Erdil, Kevin Bowlyn, Michael Altis

Abstract

As time goes on and generations pass by each other, both society and its needs develop. Therefore, we need to be constantly adapting to the change happening around us. The mirror is an object that we use in everyday life that could use some adapting. It performs a very simple function, one with no flair, yet it is one of the most commonly used objects in the world. It is designed with only one purpose in mind: let the user see their reflection. Despite the amount of time we spend looking in a mirror, it has not evolved or developed past this singular function.

Problem Statement

Considering that people use mirrors daily, it can be said that mirrors have become an integral tool to modern society (how else can people do their makeup?). However, as a tool, mirrors have not evolved past its original purpose: to allow someone to see their reflection. When the very first telephone was invented, it was considered an incredibly useful tool, even though the only function it was meant to perform was to send and receive audio calls. Looking at any “telephone” now, anyone can see that the tool has evolved immensely to become what is known as the “smartphone”. There are millions of apps that users can download on their smartphone that enhances their device’s functionality and efficiency.

Likewise, it is time for the mirror to evolve. There needs to be a “smart mirror” that still performs the same function of reflecting people’s appearances, but has additional features and functionality that can allow its users to be more efficient.

Background

There have been many versions of a “smart mirror” that have been made, so research consisted of studying and examining other smart mirrors while comparing the pros and cons of each product.

One common component across all the versions of smart mirrors was that there was always some sort of microprocessor/microcontroller, such as a Raspberry Pi or an Arduino, controlling the mirror. Arduinos are meant to run one program at at time, over and over again. Raspberry Pis are more of a general-purpose computer. The smart mirrors that were built around an Arduino were very simple and straightforward, while the smart mirrors that had a Raspberry Pi had a lot more features.

Implementation

Our project utilizes HTML, CSS, JavaScript, a passive infrared sensor, and a Raspberry Pi 3. HTML is the support code, the code used to design the general layout of the webpage. CSS is used to display the style of the site, including formatting and colors. JavaScript is utilized to implement the widget display and functionality, also allowing widgets to be moved within the webpage.

Widgets

We designed the webpage in basic HTML and CSS, creating a grid that displays spaces for widgets to fit into. JavaScript is utilized to implement these widgets, pulling them from various websites. We created movable containers in HTML, and placed each widget inside a container. Each container can fit inside a space within the grid, which allows for the customization of the display to fit the user’s liking. The grid becomes invisible once the user has finished customizing their display, projecting the user’s choice of widgets on top of a black background. This black background is invisible against the mirror, creating the illusion that the mirror is a screen.

Figure 1. The web app running on the monitor without the mirror

Motion Sensor

In an attempt to limit energy consumption, we also added motion sensing capabilities to the mirror. This would allow the mirror to turn off after a set time of no detected motion, and automatically turn back on when the user walks into the room. To accomplish this, we connected a passive infrared sensor to the Raspberry Pi, and utilized few python scripts. The passive infrared (PIR) sensor is used to detect motion in a given space. The sensor is split up into different sectors, each one projecting a different IR signal. It detects movement by tracking the interruption in the IR signals through different sectors.

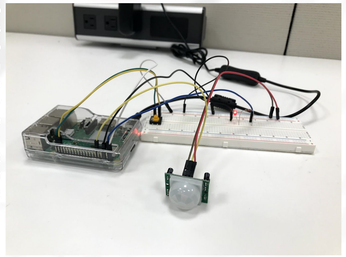

Figure 2. The circuitry for the PIR motion sensor along with the Raspberry Pi

Discussion

Our team had to learn and demonstrate best practices for GitHub, HTML, JavaScript, CSS, and Raspberry Pi configuration. Before attempting the project, our team had moderate to little experience with the aforementioned tools. Building Mira was a very educational experience for us in terms of learning about different tools and systems.

Our team ran through and implemented the waterfall model for the project. However, we found that by iterating through the different stages and going back to visit the previous steps, we were able to develop the project in a more efficient manner than just by simply going through the different stages completely before going to the next step. For example, our team had to go back and forth from focusing on requirements, analysis, and the system design many times in order to progress through the project. So, we learned that an iterative software development life cycle would have suited us better than the waterfall model that we tried to use.

A problem that users may encounter is the display functionality. There is a monitor behind the one-way mirror that projects an image on to the mirror through a light. That light is crucial when it comes to viewing the application. The only way that it may be activated is if the monitor is constantly running along with the program. This may be an issue when trying to save power and electricity. Even though the mirror may appear to be off, it works in a similar fashion as a screensaver. The screensaver is a black image that displays no light so when the motion sensor detects a person that is near, the black screen will automatically switch to the application.

The next step with this project would be to implement a way for users to be able to customize their personal Mira through their smart devices. They would be able to gain access into a Mira app by logging in; this would give users the capability to rearrange the widgets at their fingertips. It would then simulate the appearance of the mirror layout from their devices onto Mira. This innovation will hopefully spark an interest in a wide range of people.

References

“13. API - Input Devices.” 13. API - Input Devices - Gpiozero 1.5.0 Documentation, gpiozero.readthedocs.io/en/stable/api_input.html.

Ada, Lady. “PIR Motion Sensor.” How PIRs Work | PIR Motion Sensor | Adafruit Learning System, Adafruit, 28 Jan. 2014, learn.adafruit.com/pir-passive-infrared-proximity-motion-sensor/how-pirs-work.

Clinton. “How to Make a Safe Shutdown Button for Raspberry Pi - Tutorial.” Core Electronics, 18 Jan. 2019, core-electronics.com.au/tutorials/how-to-make-a-safe-shutdown-button-for-raspberry-pi.html.

“LSBInitScripts.” LSBInitScripts - Debian Wiki, 15 July 2014, wiki.debian.org/LSBInitScripts.

Sanjeev, Arvind. “How to Interface a PIR Motion Sensor With Raspberry Pi GPIO | Raspberry Pi.” Maker Pro, Maker Pro, 9 Apr. 2019, maker.pro/raspberry-pi/tutorial/how-to-interface-a-pir-motion-sensor-with-raspberry-pi-gpio.

Published by DigitalCommons@SHU, 2019