Programa de postitulación para docentes de ETP

Hardware libre e IOT (Arduino y ESP8266)

Profesor: Pablo Gaspoz (UTN - FRSF)

Alumnos: Sebastian Rocco, Alejandro Gontero y Pedro Roncoroni

# Proyecto Integrador “Entrenador de alto rendimiento IoT (Internet de las Cosas)”

***Descripción:***

Emular un sistema de entrenamiento que pueda utilizarse en clases de educación física para entrenar el cuerpo y el cerebro, mejorando los tiempos de reacción, la precisión mental, fortalecer el cuerpo, el procesamiento sensorial y otros atributos neurocognitivos. Al entrenar la conexión con lo que ven los ojos (y/o sonido) y lo que el cerebro procesa y cómo reacciona el cuerpo, dichos sistemas permiten mejorar todos los elementos del rendimiento humano.

***Fundamentación:***

Los alumnos en las escuelas realizan múltiples rutinas de ejercicios y juegos que le permiten entrenar mente y cuerpo. Sin embargo en la actualidad, las escuelas no disponen de sistemas de entrenamiento cognitivo y de velocidad que ayude a fortalecer el vínculo entre la mente y el cuerpo, mejorar su reacción y tiempos de respuesta.

Estos sistemas están actualmente disponibles en el mercado y encontramos diferentes propuestas comerciales como fitlightTraining, con un costo aproximado de US$400, BlazePod con un costo aproximado de US$439.

Si bien se plantea una necesidad real, buscaremos en esta etapa diseñar e implementar un prototipo de dispositivo que permite determinar la factibilidad y costo aproximado para una posterior etapa de producción en función de la necesidad de cada institución escolar.

Por lo tanto en esta etapa focalizaremos en los aspectos del conocimiento, la didáctica y las prácticas a ofrecer en un entorno de Taller que se dictará en el segundo ciclo de la ETP.

Se determina adecuada para la Terminalidad de Informática Personal y Profesional, el espacio curricular de Taller en el cuarto año de estudios.

En la terminalidad de Equipos e Instalaciones electromecánicas se considera adecuado el espacio curricular de Electrónica de quinto año.

En cuanto al alcance temporal, se prevé que el taller se desarrollara con una carga de 5horas semanales durante todo el ciclo lectivo.

## Actividades, tareas y metodología: qué tipo de acciones formarán parte del proyecto y cómo se realizarán.

## - Objetivo Generales:

* El objetivo de este proyecto integrador “Entrenador de alto rendimiento IoT” es elaborar y desarrollar un curso de introducción a los conceptos básicos de la Internet de las cosas mediante Arduino y el módulo ESP 8266 de comunicación wifi, orientado a alumnos de escuela de educación técnica profesional.
* El IoT (Internet de las Cosas) se considera la Cuarta Revolución industrial de la humanidad. Esta revolución no tiene que ver con la máquina a vapor, ni con la generación de electricidad, mucho menos con la manufactura de chips electrónicos, sino que es el proceso de capturar datos por una infinidad de sensores para convertirse en información. La generación de valor se basa en servicios o productos que usan plataformas compartidas de múltiples tecnologías de acceso, Cloud Computing, programación y Big Data Analitycs.

## Objetivos Particulares:

* Disponer de un protocolo de comunicación a nivel de aplicación que permita la codificación y transmisión de información a diferentes dispositivos distribuidos en un espacio de al menos 25 mtrs de radio.
* Diseñar un dispositivo que pueda emitir luz y sonido, como así también detectar movimiento entre 5 y 15 cm de distancia.
* Permitir medir el rendimiento logrado por el usuario y el almacenamiento de información para posterior tratamiento. Dicha información deberá ser generada y transmitida por el/los dispositivos para su almacenamiento en formato de texto.

## Alcance:

Desarrollo del Prototipo de un dispositivo que permita contar con:

-Actuadores: Luz Led y Buzzer Activo para sonido.

-Sensor: Proximidad.

-Capacidad para la Recepción y transmisión de datos a una distancia no inferior de 50 mtrs: ESP8266 en Wemos D1 Mini.

## Tiempo aproximado duración: 1 mes (4 clases)

# Disparador

# ANEXO: Entregable Código entrenadorIoT.ino

#include <Adafruit\_NeoPixel.h>

#include <ESP8266WiFi.h>

#include <WiFiUdp.h>

const int TRIGGERPIN = 13;

const int ECHOPIN = 14;

const int BUZZER = 5;

const int LED = 4;

const int NUMLEDS = 3;

const int TAMMAXPAQUDP = 255;

const int DISPOSITIVO = 1;

Adafruit\_NeoPixel pixels(NUMLEDS, LED, NEO\_GRB + NEO\_KHZ800);

#define WIFI\_SSID "FAMILIA CARDOZO"

#define WIFI\_PASSW "colon1492"

const unsigned int UDP\_PORT = 4022;

int tamañoPaquete = 0;

bool errorConexion = false;

WiFiUDP UDP;

char paqueteUDP[TAMMAXPAQUDP];

int dataMaxT = 0;

int dataCRojo = 0;

int dataCVerde = 0;

int dataCAzul = 0;

int dataTono = 0;

int dataDistCM = 0;

bool dataRespuesta = false;

long tactivacion = 0;

String respuestaUDP;

int activar(int maxT, int cRojo, int cVerde, int cAzul, int tono, int maxDistCM){

//Prendemos el LED o los LEDS

unsigned long tiempo = millis();

bool activacion = false;

Serial.println("Entro a la funcion de activacion del dispositivo");

pixels.clear();

for(int i=0; i<NUMLEDS; i++){

pixels.setPixelColor(i, pixels.Color(cRojo, cVerde, cAzul));

}

pixels.show();

Serial.println("Enciende Leds");

//Activamos Buzzer

tone(BUZZER,tono); //Para Buzzer Activo Low Level Trigger es: digitalWrite(BUZZER, LOW);

Serial.println("Activa Buzzer");

//Calibramos sensor distancia

long echoTiempo;

long distCM;

long tiempoActivacion;

//Verificamos proximidad

Serial.print("Tiempo Inicio Activacion: ");

Serial.print(tiempo);

Serial.print(" - Tiempo Max Act: ");

Serial.print(maxT);

Serial.print("- Tiempo Actual: ");

Serial.println(millis());

while (tiempo+maxT > millis()){

digitalWrite(TRIGGERPIN, HIGH);

delayMicroseconds(10);

digitalWrite(TRIGGERPIN, LOW);

echoTiempo= pulseIn(ECHOPIN, HIGH);

Serial.print("EchoTiempo: ");

Serial.println(echoTiempo);

distCM = echoTiempo/59;

Serial.print("Dist CM: ");

Serial.println(distCM);

if(distCM <= maxDistCM){

activacion=true;

tiempoActivacion = millis()-tiempo; //Guarda la cantidad de milisegundos que tardo en activar

//Ahora apagar LED

pixels.clear();

pixels.show();

noTone(BUZZER); //Para Buzzer Activo es: digitalWrite(BUZZER, HIGH);

break;

}

}

if(activacion){

Serial.print("Se activo luego de: ");

Serial.println(tiempoActivacion);

return tiempoActivacion;

}else{

Serial.println("No se logro activar el dispositivo a Tiempo");

pixels.clear();

pixels.show();

noTone(BUZZER); //Para Buzzer Activo es: digitalWrite(BUZZER, HIGH);

return 0;

}

}

int procesarPaqueteUDP(){

String spudp = String(paqueteUDP);

int lastIndex = 0;

int index = spudp.indexOf('#',0);

if(index<0) return -1;

String token = spudp.substring(lastIndex,index);

Serial.print("DISPOSITIVO: ");

Serial.println(token);

if(DISPOSITIVO!=token.toInt()) return 0;

lastIndex = index;

index = spudp.indexOf('#',index+1);

if(index<0) return -1;

token = spudp.substring(lastIndex+1,index);

Serial.print("Tiempo Max Encendido: ");

Serial.println(token);

dataMaxT = token.toInt(); //Obtiene la cantidad maxima de tiempo a esperar del dispositivo

lastIndex = index;

index = spudp.indexOf('#',index+1);

if(index<0) return -1;

token = spudp.substring(lastIndex+1,index);

Serial.print("Rojo: ");

Serial.println(token);

dataCRojo = token.toInt(); //Obtiene nivel saturacion color Rojo

lastIndex = index;

index = spudp.indexOf('#',index+1);

if(index<0) return -1;

token = spudp.substring(lastIndex+1,index);

Serial.print("Verde: ");

Serial.println(token);

dataCVerde = token.toInt(); //Obtiene nivel saturacion color Verde

lastIndex = index;

index = spudp.indexOf('#',index+1);

if(index<0) return -1;

token = spudp.substring(lastIndex+1,index);

Serial.print("Azul: ");

Serial.println(token);

dataCAzul = token.toInt(); //Obtiene nivel saturacion color Azul

lastIndex = index;

index = spudp.indexOf('#',index+1);

if(index<0) return -1;

token = spudp.substring(lastIndex+1,index);

Serial.print("Tono: ");

Serial.println(token);

dataTono = token.toInt(); //Obtiene frecuencia activacion Buzzer pasivo

lastIndex = index;

index = spudp.indexOf('#',index+1);

if(index<0) return -1;

token = spudp.substring(lastIndex+1,index);

Serial.print("Distancia Max CM: ");

Serial.println(token);

dataDistCM = token.toInt(); //Obtiene distancia en CM sensor de proximidad

lastIndex = index;

index = spudp.indexOf('#',index+1);

if(index<0) return -1;

token = spudp.substring(lastIndex+1,index);

Serial.print("Solicita Respuesta: ");

Serial.println(token);

if (token.toInt()>0) dataRespuesta =true; //Obtiene si el cliente solicita respuesta o no

else dataRespuesta=false;

return 1;

}

void setup() {

// put your setup code here, to run once:

pinMode(LED, OUTPUT);

pinMode(TRIGGERPIN, OUTPUT);

pinMode(ECHOPIN,INPUT);

pinMode(BUZZER,OUTPUT);

Serial.begin(9600);

Serial.println("Configuro los pines");

pixels.begin();

Serial.println("Inicializa objeto NeoPixels Adafruit");

//Inicializa WiFi

WiFi.mode(WIFI\_STA);

WiFi.begin(WIFI\_SSID,WIFI\_PASSW);

Serial.println("Inicializa WiFi con SSI y PASSW");

//Conecta a la red por WiFi

int intentosConexion = 1;

noTone(BUZZER); //Para Buzzer Activo Low Level Trigger es: digitalWrite(BUZZER, HIGH);

Serial.println("Inicia proceso de intentos de conexion");

while ((WiFi.status() != WL\_CONNECTED) && (intentosConexion<100)){ //intenta por 25 segundos

if(intentosConexion%2 > 0){

//Enciendo la luz blanca

for(int i=0; i<NUMLEDS; i++){

pixels.setPixelColor(i, pixels.Color(255, 255, 255));

Serial.print('.');

}

pixels.show();

}else{ //apaga los Leds

pixels.clear();

pixels.show();

}

//Incrementa intentosConexion en 1

intentosConexion++;

delay(500);

}

//WiFi conectado o error de Conexion

if(intentosConexion<100){

//Se pudo conectar

pixels.clear();

Serial.println();

Serial.println("Conectado a la red");

UDP.begin(UDP\_PORT); //Se pone a escuchar en el puerto UDP

Serial.println("Escucha en puerto 4022");

}else{

//No se pudo conectar - Deja luz blanca encendida como error de conexion

Serial.println("Fallo de conexion WiFi");

for(int i=0; i<NUMLEDS; i++){

pixels.setPixelColor(i, pixels.Color(255, 255, 255));

}

pixels.show();

errorConexion = true;

}

}

void loop() {

//Recepción de paquete UDP

tamañoPaquete = UDP.parsePacket();

if(tamañoPaquete){

Serial.println("Recibe paquete UDP");

//Se recibió un paquete

UDP.read(paqueteUDP,TAMMAXPAQUDP);

Serial.println("Paquete: ");

Serial.println(paqueteUDP);

//Se procesa el paquete para obtener los campos

if(procesarPaqueteUDP()>0){

Serial.println("Proceso el Paquete");

//El procesamiento tuvo exito, se obtuvieron los campos

tactivacion = activar(dataMaxT, dataCRojo, dataCVerde, dataCAzul, dataTono, dataDistCM);

if(tactivacion>0){

//El usuario activo el dispositivo

if(dataRespuesta>0){

//Enviar respuesta al clliente

respuestaUDP= (String)DISPOSITIVO+"#"+(String)tactivacion+"##";

UDP.beginPacket(UDP.remoteIP(),UDP.remotePort());

UDP.print(respuestaUDP);

UDP.endPacket();

Serial.print("Trama Respuesta Enviada:");

Serial.println(respuestaUDP);

} //Sino se solicita respuesta no hace nada

}else{ //No se pudo activar el dispositivo en el tiempo maximo

if(dataRespuesta>0){

//Enviar respuesta al clliente

UDP.beginPacket(UDP.remoteIP(),UDP.remotePort());

respuestaUDP= (String)DISPOSITIVO+"#0##";

UDP.beginPacket(UDP.remoteIP(),UDP.remotePort());

UDP.print(respuestaUDP);

UDP.endPacket();

Serial.print("Trama Respuesta Enviada:");

Serial.println(respuestaUDP);

} //Sino se solicita respuesta no hace nada

}

}else{

Serial.println("Error en el procesamiento del paquete");

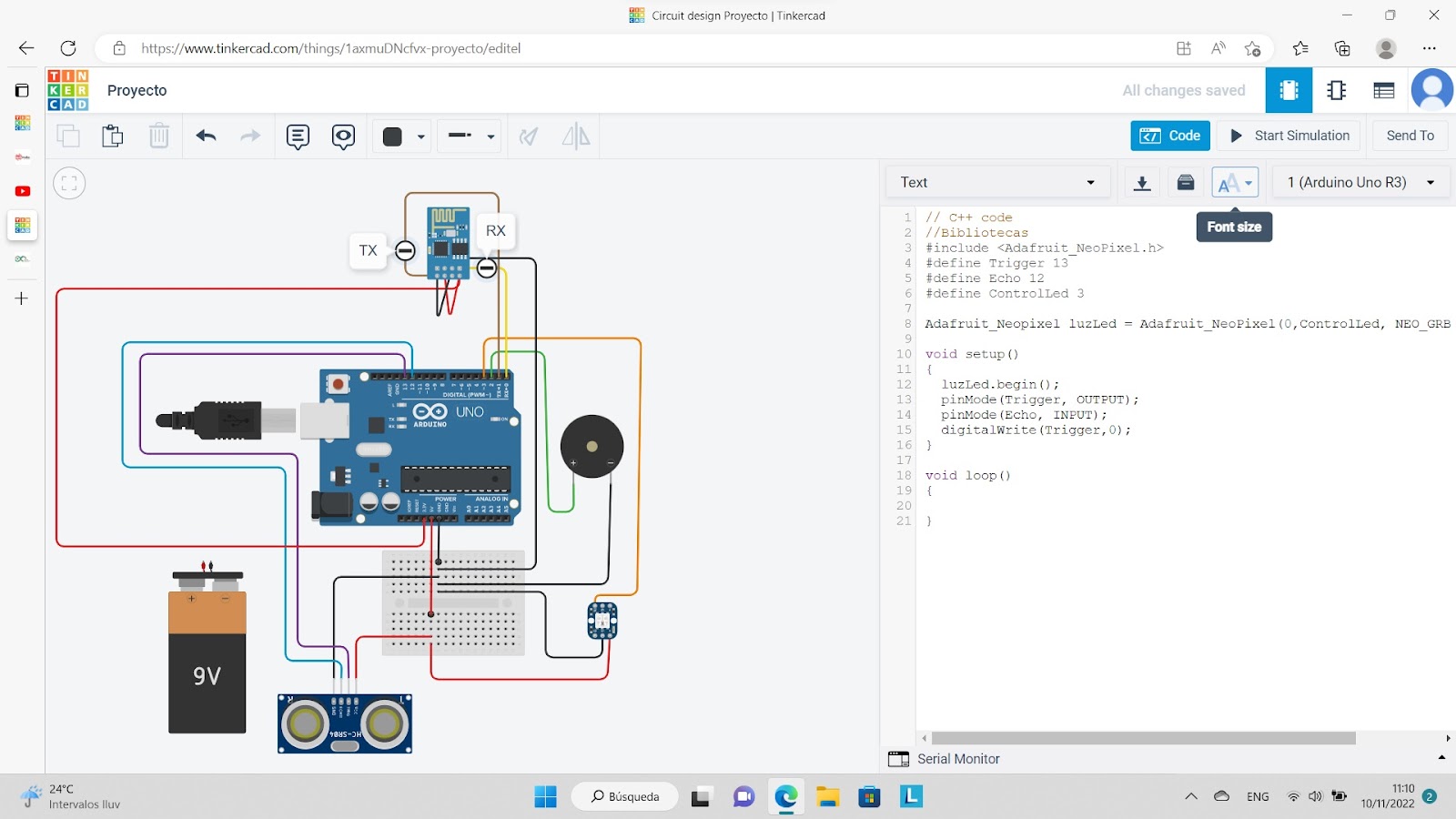
}

}

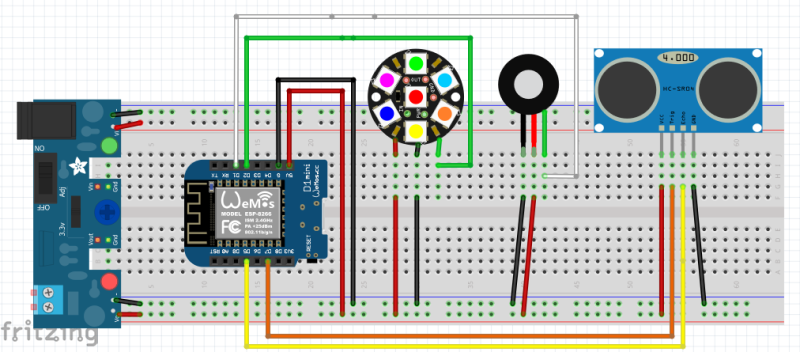
}

# Anexo: Montaje Entrenador IoT

## Diseño preliminar con ARDUINO y ESP8266:



## Diseño mejorado con Wemos D1 mini:



# Elementos Montaje:

|  |  |
| --- | --- |
| Wemos D1 Mini | Fuente Protoboard |
| HC SR04 | Buzzer Activo/Pasivo |
| Led RGB WS2812 | Protoboard |
| Cables Macho-Macho | Cable USB a Mini USB |

# Recursos:

**Actualmente el taller cuenta con sensores, buzzers, Arduino, cables Dupont M-M, ESP8266, Leds WS2812, protoboards y un par de Wemos D1 Mini. Estos recursos se consiguen con planes de mejora Institucionales y partidas de insumos plan 39 destinadas a los talleres.**

**Presupuesto aproximado de los materiales para el proyecto a diciembre de 2022:**

|  |  |
| --- | --- |
| **Wemos d1 Mini** | **$2074** |
| **Protoboard 830 puntos** | **$1000** |
| **Cables Dupont Macho-Macho x 40** | **$650** |
| **Fuente Protoboard 5V 3.3V Mb102** | **$700** |
| **Cable USB a USB mini** | **$350** |
| **Circulo 3 Leds RGB 5050 Ws2812 Neopixel** | **$642** |
| **Modulo Buzzer Activo/Pasivo** | **$490** |
| **HC SR04** | **$750** |
| **TOTAL ELEMENTOS:** | **$6656** |

# Programa Taller:

Total Horas Cátedra Semanales: 5

Dictado: Anual

## Destinatarios:

Alumnos del segundo ciclo de la Educación Técnica Profesional.

Terminalidad Informática Personal y Profesional: Taller - 4to año.

Terminalidad Equipos e Instalaciones Electromecánicas: Electrónica – 5to año.

## Programa

Arduino: Introducción

* Microcontroladores y microprocesadores.
* Interfaces Analógico – Digital: DAC, ADC.
* Modelos de Arduino: UNO, MEGA, PRO MINI, MICRO, NANO…
* Hardware: Arduino(PINOUT), Shields, Sensores, Actuadores, periféricos, Entradas analógicas y digitales …
* Instalación de IDE (Entorno de Desarrollo Integrado) Arduino.
* Instalación de drivers (CH340)

Tiempo aproximado duración: 1 mes (4 clases)

Arduino: Lenguaje.

* Boceto (sketch): bloques setup(), loop().
* Nociones básicas del lenguaje C++.
* Análisis de código (sketch)
* Carga del boceto a Arduino, compilación y subida del sketch (bootloader).
* Funciones básicas: pinMode, delay, digitalWrite, DigitalRead, analogRead…
* Pines PWM y analogWrite.

Tiempo aproximado duración: 1 mes (4 clases)

Arduino: Prácticas

* Montaje de Circuitos e implementación de sketchs.
  + Prácticas con Leds (Blink, RGB).
  + Monitor Serial (Serial Object). Comunicar Arduino con la PC, Debugging)
  + Sensores: fotoresistencias, sensor de humedad y temperatura, ultrasonido, infrarrojo, detección de gases…
  + Interruptores (resistencias Pull Up y Pull Down), codificadores
  + Transistores, optoacopladores, relé.
  + Motores de Corriente Continua, drivers L298N…
  + Motores Paso a Paso, drivers A4988, driver DRV8825
  + Displays, pantallas.
  + Comunicación por bus SPI, ISP, I2C
  + Comunicación Bluethoot
  + ESP8266 – Wemos D1 Mini – Nodemcu para Comunicación WiFi (TCP-UDP)
  + Domótica - Zigbee

Tiempo aproximado duración: 5 meses (20 clases)

# Evaluación:

La evaluación de la asignatura se realiza mediante la corrección de los ejercicios prácticos, como así también de las presentaciones realizadas por los alumnos en forma grupal producto de la investigación realizada a partir de los distintos disparadores propuestos por la cátedra.

Se considera el taller como un espacio para que el alumno pueda “hacer…” y ese hacer involucra también la producción del conocimiento a través de la prueba y el error, pero siempre partiendo de la investigación, el debate y el análisis como metodologías inherentes al proceso de resolución de las diferentes situaciones problemáticas que se plantean como puntos de partida para los distintos proyectos del taller.

Nota: Como material audiovisual se proveerá de un video en formato mp4 donde se muestra el prototipo funcionando en tiempo real. Para su accionamiento se utilizará un software (Packet Sender) para el envío y recepción de paquetes UDP al dispositivo prototipo.

En una etapa posterior se busca desarrollar una aplicación Web que permite leer archivos (formato Excel) con rutinas de ejercicios que puedan reproducirse enviando los paquetes UDP a los dispositivos y recibiendo sus tramas de respuesta.

Finalmente se buscará perfeccionar el prototipo incorporando todos los componentes en un estructura resistente y ajustable, como así también montar los componentes en una placa.