PoloPoints

Versión	Fecha	Autor	Descripción de cambios
1.00	23/04/2023	Pedro Parodi	Versión inicial

Índice

1 Materiales para el armado

- ESP32-DevKitC-V4
- Antena WiFi 2.4GHz (no definido)
- Kit del tablero (LEDs + placas controladoras):
  - 2 números de 2 dígitos para puntajes local y visitante
  - 1 número de 1 dígito para chuker
  - 2 números de 2 dígitos para temporizador (minutos y segundos) con puntos incluidos
  - Placa controladora general del tablero y 1 placa controladora por cada dígito
- Convertidor 12V a 5V (no definido)
- *Transceiver* RS232-UART con cable correspondiente.
- Amplificador de audio (no definido)
- 2 parlantes plásticos 20W c/u (no definido)
- Gabinete del producto
- Materiales y herramientas para armado y soldado.

2 Web Server

El dispositivo ESP32 genera una red WiFi y funciona como access point. Los datos de la red son los siguientes:

- SSID: “PoloPoints”
- Password: 12345678

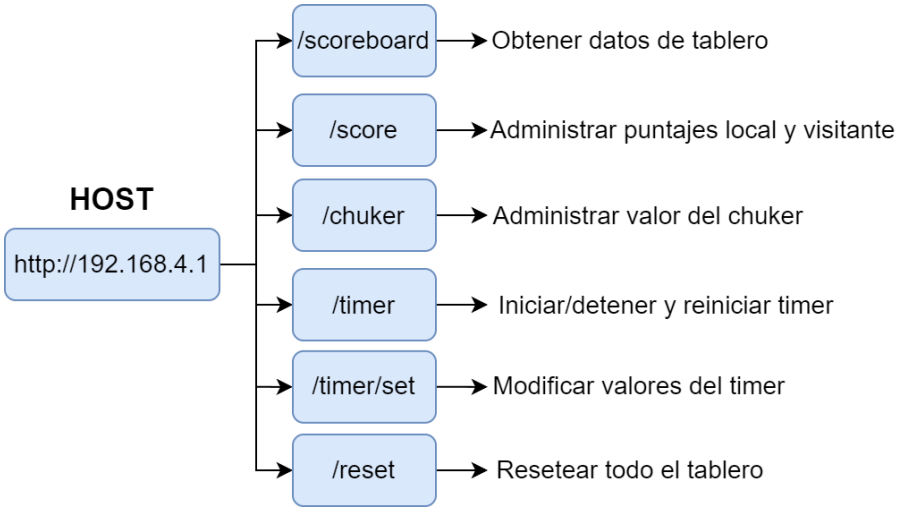
Se implementa un servidor con una serie de *endpoints* para administrar las variables y funcionalidad del tablero. Para ello, se hace uso de los siguientes comandos disponibles, que serán parámetros parte de la URL en las consultas al servidor:

- INC_SCORE_LOCAL = 1 → incrementar puntaje del local
- INC_SCORE_VISITOR = 2 → incrementar puntaje del visitante
- DEC_SCORE_LOCAL = 3 → decrementar puntaje del local
- DEC_SCORE_VISITOR = 4 → decrementar puntaje del visitante
- INC_CHUKER = 5 → incrementar chuker (período)
- DEC_CHUKER = 6 → decrementar chuker (período)
- START_*TIMER* = 7 → iniciar el temporizador
- STOP_*TIMER* = 8 → detener el temporizador
- RESET_*TIMER* = 9 → resetear el temporizador a valor de inicio (*default*)
- SET_CURRENT_*TIMER* = 10 → modificar el valor actual del temporizador
- SET_*DEFAULT_TIMER* = 11 → modificar el valor de inicio (*default*) del temporizador
- RESET_ALL = 12 → resetear el tablero a valores de inicio (*default*)

Todas las solicitudes programadas en el servidor son de tipo GET. Algunas de ellas solo responden con un STATUS_CODE en función de éxito o fracaso de la acción deseada.
Otras, en caso de éxito, responden con los valores actualizados del tablero. Esto lo harán mediante una variable de tipo *string* con todos los datos del tablero concatenados por una coma, con el siguiente formato: “puntaje-local,puntaje-visitante,valor-chuker,valor-*timer*-mm,valor-*timer*-ss,estado-*timer*”. Donde el estado del temporizador (*timer*) puede ser:

- Detenido: 0
- Corriendo: 1
- Finalizado: 2 (temporizador llegó a 00:00)

2.1 *Endpoints*



Resumen de *endpoints* implementados.

Valores de tablero

Solicitud para obtener el estado actual de las variables del tablero: puntajes, chuker, valor actual del *timer* y estado del *timer*.

Ejemplo de solicitud:

```
GET /scoreboard
Host: http://192.168.4.1
```

Ejemplo de respuesta:

```
Content-type: “text/plain”
Body: “0,0,1,6,30,0”
```

Status code	Description
200 – Ok	Aceptado.

Puntajes

Solicitudes para administrar los puntajes de local y visitante. Cada solicitud permite incrementar o decrementar un determinado puntaje, definido por el parámetro que se incorpore en la URL. La solicitud devuelve el estado actual del tablero. No requiere que el *timer* se encuentre detenido.
El valor del parámetro ‘cmd’ se corresponderá con:

- INC_SCORE_LOCAL = 1
- INC_SCORE_VISITOR = 2
- DEC_SCORE_LOCAL = 3
- DEC_SCORE_VISITOR = 4

Ejemplo de solicitud (**incrementar** puntaje de **local**):

```
GET /score?cmd=1
```

```
Host: http://192.168.4.1
```

Ejemplo de respuesta:

```
Content-type: “text/plain”
Body: “1,0,3,5,22,1”
```

Status code	Description
202 – Accepted	Aceptado.
404 – Bad Request	Falta parámetro o es incorrecto. Solicitud cancelada.

Chuker

Solicitudes para administrar el chuker. El mismo *endpoint* permite incrementarlo o decrementarlo, definido por el parámetro que se incorpore en la URL. La solicitud devuelve el estado actual del tablero. No requiere que el *timer* se encuentre detenido.
El valor del parámetro ‘cmd’ se corresponderá con:

- INC_CHUKER = 5
- DEC_CHUKER = 6

Ejemplo de solicitud (**incrementar** chuker):

```
GET /score?cmd=5
Host: http://192.168.4.1
```

Ejemplo de respuesta:

```
Content-type: “text/plain”
Body: “1,0,2,5,22,1”
```

Status code	Description
202 – Accepted	Aceptado.
404 – Bad Request	Falta parámetro o es incorrecto. Solicitud cancelada.

Timer

Iniciar/detener

Solicitud para iniciar/detener el temporizador que solo devuelve como información un STATUS_CODE. No se considera como error solicitar, por ejemplo, iniciar el *timer* cuando ya se encuentra activo. Tampoco lo será detener el *timer* cuando está detenido.
El valor del parámetro ‘cmd’ se corresponderá con:

- START_*TIMER* = 7
- STOP_*TIMER* = 8

Ejemplo de solicitud (**iniciar** *timer*):

```
GET /timer?cmd=7
Host: http://192.168.4.1
```

Respuesta:

Status code	Description
202 – Accepted	Aceptado.
404 – Bad Request	Falta parámetro o es incorrecto. Solicitud cancelada.

Reiniciar

Solicitud para detener el temporizador y llevarlo al valor *default*. El valor *default* es 6' 30'' salvo que sea modificado a través de otra solicitud. La solicitud, en caso de éxito, devolverá el estado actual del tablero.
El valor del parámetro 'cmd' será siempre RESET_*TIMER* = 9.

Ejemplo de solicitud:

```
GET /timer?cmd=9
Host: http://192.168.4.1
```

Ejemplo de respuesta:

```
Content-type: "text/plain"
Body: "1,3,2,6,30,0"
```

Status code	Description
202 – Accepted	Aceptado.
404 – Bad Request	Falta parámetro o es incorrecto. Solicitud cancelada.

Modificar valores

Solicitud que permite modificar el valor actual o el *default* (de inicio) del temporizador. Se define entre estas dos mediante el parámetro 'cmd' de la solicitud.
En caso de éxito, la solicitud devolverá el estado actual del tablero. Es necesario que el temporizador se encuentre detenido para estar operaciones, caso contrario la solicitud devolverá error. Tampoco será valido querer configurar el valor del *timer* a 00:00.

- SET_CURRENT_*TIMER* = 10
- SET_*DEFAULT_TIMER* = 11

Valor actual

Ejemplo de solicitud (modificar valor **actual** del *timer*):

```
GET /timer/set?mm=2&ss=30&cmd=10
Host: http://192.168.4.1
```

Ejemplo de respuesta:

```
Content-type: "text/plain"
Body: {0,1,1,2,30,0}
```

Status code	Description
202 – Accepted	Aceptado.

404 – Bad Request	Falta parámetro o es incorrecto. Solicitud cancelada.

Valor *default* (de inicio)

Ejemplo de solicitud (modificar valor **de inicio** del *timer*):

```
GET /timer/set?mm=9&ss=0&cmd=11
Host: http://192.168.4.1
```

Ejemplo de respuesta:

```
Content-type: “text/plain”
Body: {0,1,1,9,0,0}
```

Nota: este comando también configura el timer actual.

Status code	Description
202 – Accepted	Aceptado.
404 – Bad Request	Falta parámetro o es incorrecto. Solicitud cancelada.

Resetear tablero

La solicitud frena el *timer*, en caso de que se encuentre activo, y lleva todos los valores del tablero a su valor de inicio. Salvo modificaciones, estos valores son:

- Puntaje local: 0
- Puntaje visitante: 0
- Chuker: 1
- *Timer*: 6’ 30’’

La solicitud devuelve los datos actualizados del tablero.

Ejemplo de solicitud:

```
GET /reset
Host: http://192.168.4.1
```

Ejemplo de respuesta:

```
Content-type: “text/plain”
Body: “0,0,1,6,30,0”
```

Status code	Description
202 – Accepted	Aceptado.

3 Diagrama de flujo

El programa en el dispositivo ESP32 puede dividirse en dos bloques sencillos: servidor web y máquina de estados que administra el tablero.

El servidor web es de tipo asincrónico, por lo que corre de fondo como un hilo independiente y administra y ejecuta todas las solicitudes que recibe el *host* (ESP32). Las solicitudes son las descriptas anteriormente y permiten actuar sobre los valores y estado del tablero. Cada vez que hay un cambio en algún dato (por ejemplo, un puntaje) a través de una solicitud, la misma solicitud informa mediante un *flag* a la máquina de estados principal para denotar que los LEDs del tablero deben actualizarse.
Por otro lado, la máquina de estados alterna entre tres estados:

- INIT: inicialización de tablero con valores iniciales. Se ejecuta por única vez al comienzo del programa.
- IDLE: estado de reposo. Queda a la espera de que pase 1 segundo (con *timer* activo) o que se reciba alguna solicitud HTTP que modifique algún valor del tablero. Luego invoca al siguiente estado para actualizar el tablero.
- REFRESH_SCOREBOARD: actualizar tablero (LEDs). Luego, retorna a estado de reposo IDLE.

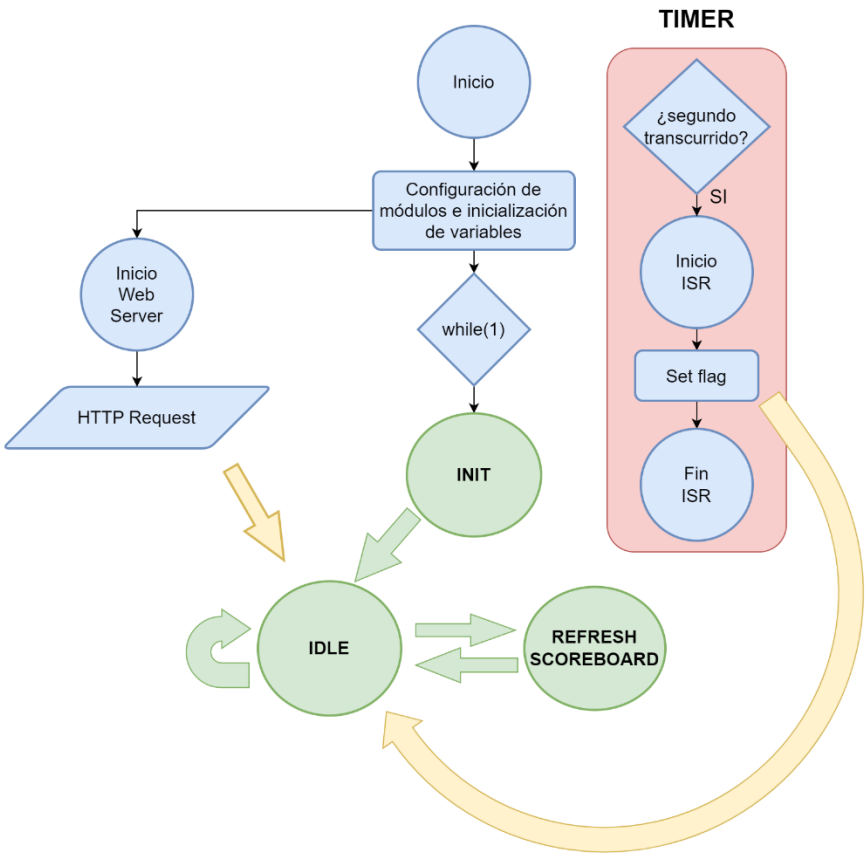


Diagrama de flujo del programa.