



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA



Escola Tècnica
Superior d'Enginyeria
Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Soluciones basadas en micro:bit para la gestión de residuos sólidos en entornos de interior

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

Autor: María Navarro Jiménez

Tutor: Xavier Molero Prieto

Curso 2018-2019

Especial agradecimiento a Diego Álvarez,
Xavier Molero y Rafael Sánchez, integrantes de
la Cátedra de Tecnologías Cívicas,
por haber hecho posible este proyecto.

Resum

Aquest treball proposa traslladar els progressos que s'estan realitzant en l'àrea de *Smart Waste* —aplicació de la tecnologia per a la monitorització de contenidors amb la fi d'introduir millores en la gestió de residus— a un entorn d'interior, emprant sensors de la targeta microcontroladora *micro:bit* per a recopilar informació i enviar-la a una plataforma per al seu posterior anàlisi. Tot açò constitueix un projecte dins de la Càtedra de Tecnologies Cíviques, concretament en la Escola de Tecnologies Cíviques, facilitant al ciutadà les ferramentes i coneixements necessaris per al disseny de les seues pròpies solucions pel que fa a la gestió de residus, contribuint al compliment dels Objectius de Desenvolupament Sostenible (ODS) promoguts per Nacions Unides.

Paraules clau: *micro:bit*, IoT, residus sòlids, *Smart Waste*, *Smart City*, ciutadania, apoderament, sensorètica, generació de dades, gestió de residus, eficiència, sostenibilitat, reciclatge, Objectius de desenvolupament Sostenible

Resumen

Este trabajo propone trasladar los progresos que se están realizando en el área de *Smart Waste* —aplicación de la tecnología para la monitorización de contenedores con tal de introducir mejoras en la gestión de residuos— a un entorno de interior, empleando sensores de la tarjeta microcontroladora *micro:bit* para recopilar información y enviarla a una plataforma para su posterior análisis. Todo ello constituye un proyecto en el marco de la Cátedra de Tecnologías Cívicas y Empoderamiento de la ETSINF, concretamente en la Escuela de Tecnologías Cívicas, facilitándole al ciudadano las herramientas y conocimientos necesarios para diseñar sus propias soluciones en lo que respecta a la gestión de residuos, contribuyendo al cumplimiento de los Objetivos de Desarrollo Sostenible (ODS) promovidos por Naciones Unidas.

Palabras clave: *micro:bit*, IoT, residuos sólidos, *Smart Waste*, *Smart City*, ciudadanía, empoderamiento, sensorética, generación de datos, gestión de residuos, eficiencia, sostenibilidad, reciclaje, Objetivos de Desarrollo Sostenible

Abstract

This project raises the idea of taking the progresses made so far in the field of "Smart Waste" —application of technology for the monitoring of bins with the aim of making improvements in terms of waste management— and simulate them with indoor environments, using the *micro:bit* device and its external sensors to collect information and send it to a platform for its subsequent analysis. All of this constitutes a project of the *Cátedra de Tecnologías Cívicas y Empoderamiento de la ETSINF*, and more precisely of the Civic Tech School, providing citizens with the tools and the knowledge required for designing their own solutions with regard to waste management, contributing to the achievement of Sustainable Development Goals (SDG), promoted by United Nations.

Key words: *micro:bit*, IoT, solid waste, *Smart Waste*, *Smart City*, citizens, empowering, sensors, data generation, waste management, efficiency, sustainability, recycling, Sustainable Development Goals

Índice general

Índice general	V
Índice de figuras	VII
Índice de tablas	VIII
<hr/>	
1 Introducción	1
1.1 Motivación	1
1.2 Objetivos	1
1.3 Estructura de la memoria	2
1.4 Notas acerca de la bibliografía	2
2 Smart Waste: aplicación de la sensorética a la gestión de residuos	3
2.1 El papel actual de la gestión de residuos en las <i>Smart Cities</i>	3
2.2 La Cátedra de Tecnologías Cívicas y Empoderamiento	5
3 La tarjeta micro:bit	9
3.1 Origen de la tarjeta micro:bit	9
3.2 Arquitectura	10
3.3 Componentes externos	11
3.3.1 Placa Octopus:bit	11
3.3.2 Pantalla OLED	12
3.3.3 Módulo <i>Wi-Fi</i> ESP8266	12
3.3.4 Sensor PIR	12
3.3.5 Sensores emisor y receptor de luz infrarroja	13
3.4 Entornos de desarrollo	14
4 La recolección de datos en la tarjeta micro:bit	17
4.1 Conexión USB y uso de <i>logs</i>	17
4.1.1 Código en MicroPython	17
4.1.2 Recuperar los <i>logs</i> con MicroFS	20
4.2 Conexión <i>Wi-Fi</i>	21
4.2.1 Plataformas de envío de datos	21
4.2.2 Diseño del código	24
4.2.3 Conexión de los componentes	27
4.2.4 Visualización de datos	28
4.3 Conclusiones	29
5 Gestión de residuos con la tarjeta micro:bit	31
5.1 La micro:bit como incentivo al reciclaje	31
5.1.1 Diseño del código	31
5.1.2 Conexión de los componentes	32
5.1.3 Visualización de datos	33
5.2 La tarjeta micro:bit como contador de residuos	34
5.2.1 Diseño del código	34
5.2.2 Conexión de los componentes	35
5.3 La tarjeta micro:bit como herramienta de gestión de residuos en entornos de interior	36

5.3.1	Diseño del código	36
5.3.2	Aplicación de la solución y pruebas	38
5.4	Presupuesto de la solución	40
6	Conclusiones	45
6.1	Cumplimiento de objetivos	45
6.2	Dificultades encontradas	45
6.3	Trabajo a futuro	45
	Bibliografía	47

Índice de figuras

2.1	Proyecto <i>Smart Waste</i> , por Ecoembes y Minsait de Indra.	3
2.2	Proyecto <i>Smart Waste</i> , Citibrain.	4
2.3	Proyecto <i>IBM Intelligent Waste Management Platform</i> , IBM.	5
2.4	Talleres destinados a centros educativos (ESO y Bachillerato).	6
2.5	Taller organizado en Las Naves en colaboración con la ONG Civic Wise.	7
2.6	Encuestas de satisfacción del taller impartido al Colegio San Roque de Valencia.	8
3.1	La tarjeta micro:bit como enseñanza de programación enfocada a niños.	9
3.2	La tarjeta micro:bit.	10
3.3	Elementos de la tarjeta micro:bit.	10
3.4	Kit básico de comercialización de la tarjeta micro:bit.	11
3.5	Placa Octopus:bit.	11
3.6	Pantalla OLED.	12
3.7	Módulo ESP8266 y módulo junto con su adaptador.	12
3.8	Sensor PIR.	13
3.9	Sensor emisor IR compatible con Arduino.	13
3.10	Sensor receptor IR compatible con Arduino	14
3.11	Entorno de programación MakeCode.	14
4.1	Fichero generado por la tarjeta micro:bit con datos de temperatura.	21
4.2	Menú para la descarga de librerías en MakeCode.	22
4.3	Librería iot-lora-node para MakeCode y componente de Lora para micro:bit.	22
4.4	Librería iot-environment-kit para MakeCode.	23
4.5	Plataforma ThingSpeak.	23
4.6	Configuración del canal de ThingSpeak.	24
4.7	Apartado con las claves API del canal de ThingSpeak.	24
4.8	Menú <i>ESP8266_IoT</i> en MakeCode.	25
4.9	Código para el envío de datos de la tarjeta micro:bit vía <i>Wi-Fi</i>	26
4.10	Menú de exploración y código fuente de la función connect thingspeak en JavaScript.	27
4.11	Conexión del módulo ESP8266 a la placa Octopus:bit.	27
4.12	Adaptador de corriente con 5V de salida.	28
4.13	Visualización de datos de temperatura enviados por la tarjeta micro:bit.	28
4.14	Exportar datos de un canal de ThingSpeak.	29
5.1	Código para la solución con el sensor PIR.	32
5.2	Conexión del PIR y el módulo ESP8266 a la placa Octopus:bit.	33
5.3	Datos recibidos por el sensor PIR.	33
5.4	Incentivo al uso del carril bici en la ciudad de Valencia.	34
5.5	Código para la solución con sensores infrarrojos.	35
5.6	Conexión de los sensores infrarrojos.	35
5.7	Código de la tarjeta micro:bit emisora para la solución final.	37
5.8	Código de la tarjeta micro:bit receptora para la solución final.	38

5.9 Tarjeta micro:bit receptora con sensor PIR.	38
5.10 Tarjeta micro:bit emisora con sensores infrarrojos.	39
5.11 Traza de la solución.	39
5.12 Resultados tras la traza de la solución final.	40
5.13 Simbología tipos de papeleras	40
5.14 Plano de la planta 0 del edificio 1E	41
5.15 Plano de la planta 1 del edificio 1E	41
5.16 Plano de la planta 2 del edificio 1E	42

Índice de tablas

2.1 Talleres impartidos por la Escuela de Tecnologías Cívicas.	8
5.1 Recuento de papeleras del edificio 1E	42
5.2 Recuento de conjuntos de papeleras de reciclaje del edificio 1E	42
5.3 Presupuesto de componentes externos a la tarjeta micro:bit.	43

CAPÍTULO 1

Introducción

Este trabajo presenta tres soluciones que, empleando una tarjeta micro:bit y componentes compatibles con la misma, aportan mejoras tanto en el ámbito de la gestión de residuos en entornos de interior (véase centros educativos, centros sanitarios, etc.) como en la concienciación y motivación al uso de contenedores de reciclaje en contraposición a los convencionales.

1.1 Motivación

La iniciativa de este trabajo surge en el marco de la Cátedra de Tecnologías Cívicas y Empoderamiento, promovida por Diego Álvarez y Xavier Molero y de la cual he sido partícipe como estudiante en prácticas junto a Rafael Sánchez Romero.

Su principal propósito es proveer a los ciudadanos de los conocimientos y herramientas necesarios para ser proactivos en las *Smart Cities* a través de la generación de datos.

Adicionalmente, otro de los motivos por los que surge este trabajo es la contribución para el cumplimiento de los Objetivos de Desarrollo Sostenible ¹, en especial los objetivos 11 y 12, comprometidos con el logro de ciudades más sostenibles y una producción y consumo responsables, respectivamente.

1.2 Objetivos

Los objetivos principales de las soluciones que componen el trabajo son:

- En primer lugar, aportar una mejora en lo que respecta a la gestión de los residuos sólidos en entornos de interior, bien sea recolectando información que ayude a una colocación estratégica de los contenedores según las zonas más frecuentadas por los usuarios, o bien recopilando información del nivel de llenado de las mismas con el fin de mejorar el servicio de recogida.
- En segundo lugar, se buscará la interacción con el usuario que se encuentre próximo al contenedor de reciclaje o haga uso del mismo, con el propósito de promover su uso de manera predominante con respecto a los contenedores convencionales, todo ello enmarcado en la línea de los objetivos de desarrollo sostenible.

¹Detallados en <https://www.un.org/sustainabledevelopment/es/sustainable-consumption-production/>

- Adicionalmente, se buscará en la implementación de las soluciones la sencillez, de manera que puedan ser incorporadas a los talleres impartidos por parte de la Cátedra de Tecnologías Cívicas y Empoderamiento, instruyendo así a los ciudadanos en la implementación de soluciones con aplicación real que aporten mejoras a su vida cotidiana.

1.3 Estructura de la memoria

La memoria de este trabajo se ha estructurado en los siguientes capítulos:

- El Capítulo 2 de esta memoria se compone del estado del arte en el que surge el trabajo, además de exponer los conocimientos necesarios para la comprensión del mismo.
- Seguidamente, el Capítulo 3 se dedica en exclusiva a la tarjeta micro:bit, tratando el contexto de su lanzamiento, entornos y lenguajes de programación, aspectos técnicos y componentes externos compatibles con ella empleados en este trabajo.
- Una vez abordado todo el conocimiento necesario para entender el funcionamiento de la tarjeta micro:bit, en el Capítulo 4 se aportan dos alternativas para la recogida y almacenamiento de los datos que recolecta este dispositivo.
- A continuación encontramos el Capítulo 5, donde se presentan y analizan las diferentes soluciones que componen el trabajo.
- Finalmente, en el Capítulo 6 se presentan las conclusiones extraídas de este trabajo, analizando tanto el cumplimiento de objetivos como las dificultades encontradas.

1.4 Notas acerca de la bibliografía

En primer lugar, los libros de las referencias [7], [4], [5] y [11] han sido de gran utilidad a la hora de documentarse para la redacción del Capítulo 3 destinado a la micro:bit, así como para el desarrollo de las soluciones.

Por otro lado, el libro de la referencia [2] ha servido de apoyo para hacer hincapié en la importancia de la generación de datos por parte de la ciudadanía, abordado para contextualizar el trabajo en el Capítulo 2.

Para este mismo capítulo también se ha hecho uso de los enlaces y artículos citados en las referencias [10], [3] y [6], que constituyen el estado del arte del cual surge este trabajo.

Por último, las referencias [8], [1] y [9] han sido necesarias a la hora de desarrollar el código presentado en el Capítulo 4.

CAPÍTULO 2

Smart Waste: aplicación de la sensorética a la gestión de residuos

En este capítulo se definen los conceptos *Smart City* y *Smart Waste*, así como las investigaciones que se están llevando a cabo actualmente en estos campos. Además, se relacionan con la labor de la Cátedra de Tecnologías Cívicas y Empoderamiento y, en definitiva, se aborda todo lo necesario para contextualizar la iniciativa de este trabajo.

2.1 El papel actual de la gestión de residuos en las *Smart Cities*

Calificamos con el término *Smart City* a toda área urbana que hace uso de tecnologías para la recolección de datos cuya finalidad es la mejora de la gestión de recursos.

Esta recolección es posible gracias al Internet de las Cosas —*IoT, Internet of Things*—, y es por ello que, en la actualidad tecnológica en la que vivimos con la inminente llegada de las tecnologías 5G, el desarrollo de dispositivos cuyos sensores recogen dichos datos está en auge.

La gestión de residuos inteligente no es más que el desarrollo de tecnologías que nos faciliten y permitan una gestión de residuos más sostenible y eficiente. Dentro de las *Smart Cities* es de las áreas más nuevas, ya que se encuentra actualmente en investigación y desarrollo, planeándose su aplicación en las ciudades en los próximos años.

Un ejemplo de esto es el proyecto *Smart Waste* (véase Figura 2.1), promovido por Ecoembes que a través de su centro de investigación e innovación *The Circular Lab* [3] y en colaboración con *Minsait* de Indra.

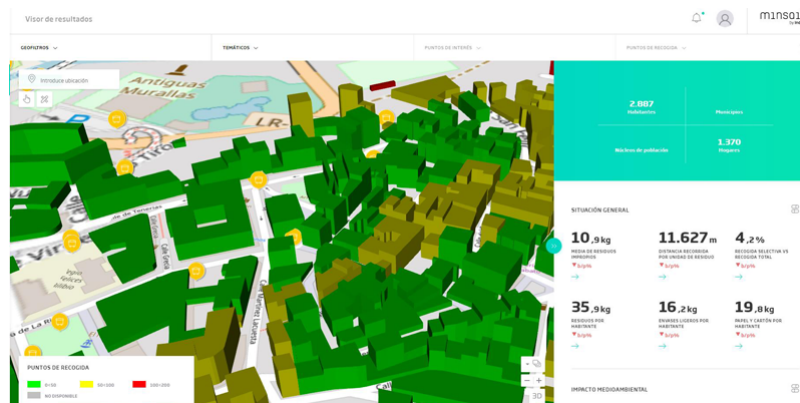


Figura 2.1: Proyecto *Smart Waste*, por Ecoembes y Minsait de Indra.

Smart Waste [10] no es más que un proyecto de recolección de información con el fin de definir rutas eficientes de recogida de basuras de manera que en las zonas en las que el llenado se produzca menos veces se acuda con menor periodicidad, contribuyendo a una menor contaminación innecesaria.

Para ello se precisa, por un lado, que las rutas sean conocidas en tiempo real, de forma que puedan ser corregidas a medida que se reciben los datos; por otro lado, que los contenedores sean inteligentes, es decir, estén sensoretizados, conociendo en todo momento su estado de llenado.

Existen diversos artículos acerca de este proyecto, que todavía se encuentra en estado de pilotaje. Uno de los artículos a destacar es una entrevista¹ con el director de Soluciones Tecnológicas de Minsait, en la que afirma que *en un futuro se llegará a sensorizar el cubo de las comunidades de vecinos, e, incluso, el de los domicilios particulares*.

Además, *Smart Waste* no solo pretende la generación de rutas de recogida eficientes, sino la localización estratégica de basuras, como apunta en otro artículo² el responsable de innovación en Ecoembes: *Si se conoce el perfil de la población que vive en cada zona, se puede, por ejemplo, realizar una ubicación de los contenedores de recogida que favorezca a las personas con mayores problemas de movilidad como los mayores o discapacitados*.

Con todo ello, y con el fin de incorporar la gestión de residuos en las *Smart Cities* promoviendo además el reciclaje y la sostenibilidad, *Smart Waste* es el proyecto español de mayor relevancia en este campo hasta la fecha, pretendiendo ser incluso un referente europeo.

No obstante, existen otros proyectos en relación a la gestión inteligente de residuos. Un ejemplo de ello es la empresa CitiBrain³, con sede en Portugal, que ofrece soluciones de sensoretización y gestión eficiente de residuos, tal y como se muestra en la Figura 2.2.

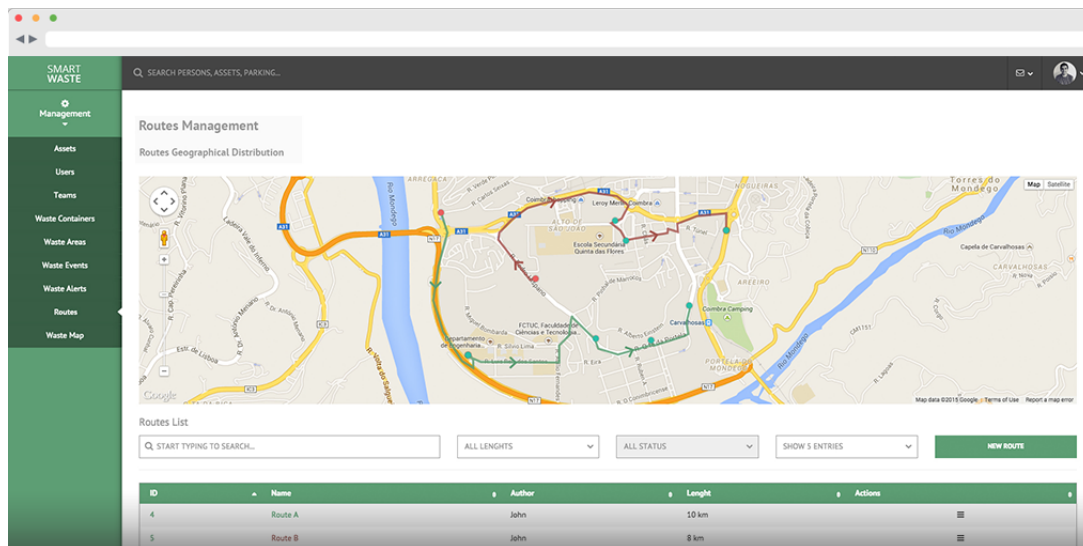


Figura 2.2: Proyecto *Smart Waste*, Citibrain.

¹Disponible en <https://gestoresderesiduos.org/noticias/smart-waste-aplicando-el-internet-de-las-cosas-a-la-gestion-de-residuos>

²Disponible en <https://www.esmartcity.es/2018/03/16/plataforma-smart-waste-lleva-iot-big-data-gestion-recogida-reciclaje-residuos>

³Más información en <http://www.citibrain.com/es/solutions/smart-waste-es/>

Incluso la reconocida empresa IBM presentó en 2015 su proyecto para la gestión inteligente de residuos, *IBM Intelligent Waste Management Platform* [6], en el cual relata la transición de las ciudades a *Smart Cities* y la necesidad del desarrollo de tecnologías —véase Figura 2.3— con el propósito evolucionar hacia una economía circular sostenible en el tiempo.



Figura 2.3: Proyecto *IBM Intelligent Waste Management Platform*, IBM.

En definitiva, están surgiendo diversos proyectos a día de hoy con el fin de incorporar la gestión inteligente de residuos a las *Smart Cities*. Sin embargo, todos lo desarrollado hasta ahora va enfocado a la gestión de rutas inteligentes de recogida de basuras a nivel ciudad. Es precisamente en contraposición a esto último de donde surge la idea para este trabajo cuyo objetivo es la gestión de residuos a pequeña escala, como bien puede ser la gestión de recogida de residuos en centros escolares, hospitales y, en definitiva, entornos de interior lo suficientemente grandes como para requerir una gestión de recogida y verse beneficiados por un sistema que la automatice de la manera más eficiente posible.

2.2 La Cátedra de Tecnologías Cívicas y Empoderamiento

Con el auge de las *Smart Cities*, cobra importancia el hecho de preparar a los ciudadanos para el entendimiento e incluso el desarrollo de soluciones que contribuyan a la generación de datos.

Esto no significa formar programadores o analistas, ya que las soluciones seguirían siendo desarrolladas por profesionales y los datos obtenidos serían analizados por expertos en la materia a la que hagan referencia. Lo que se pretende es que todo ciudadano pueda involucrarse con criterio en las decisiones tomadas por nuestros representantes y no verse engañados con datos cuya veracidad no ha sido probada, o soluciones excesivamente caras que podrían reemplazarse por otras alternativas.

En definitiva, empoderar a la ciudadanía con el conocimiento básico para ser partícipe de las *Smart Cities*, lo que podríamos calificar como *Smart Citizens*, y es este el objetivo que se plantea la Cátedra de Tecnologías Cívicas y Empoderamiento.

La Cátedra de Tecnologías Cívicas y Empoderamiento, dirigida por Diego Álvarez y codirigida por Xavier Molero, doctores en Ingeniería Informática en la Universitat Poli-

técnica de València, es un proyecto de investigación que colabora tanto con la UPV como con la Diputación de Valencia, y del cual he sido partícipe durante este último curso junto con el estudiante Rafael Sánchez Romero, y es durante este periodo y con el propósito de cumplir con el objetivo marcado de empoderar a la ciudadanía hacia las *Smart Cities* cuando nace la Escuela de Tecnologías Cívicas.

Los principales propósitos de dicho proyecto es la divulgación del conocimiento por todos los medios posibles:

- Impartiendo talleres de sensorética en la UPV, destinados a alumnos de instituto desde ESO hasta Bachillerato (véase en la Figura 2.4), como alumnos de la propia universidad (para el Máster Universitario de Gestión de la Información).



Figura 2.4: Talleres destinados a centros educativos (ESO y Bachillerato).

- Impartiendo talleres solicitados por ayuntamientos o en colaboración con ONGs, como muestra la Figura 2.5. Además, en estos talleres se concienciaba de los problemas cívicos como son la contaminación acústica o las altas temperaturas en las aulas, aportando soluciones mediante las cuales registrar datos que permitieran la iniciación de una queja con motivo final de solventar dichos problemas.



Figura 2.5: Taller organizado en Las Naves en colaboración con la ONG Civic Wise.

- Impartiendo talleres solicitados por ayuntamientos o en colaboración con ONGs, como muestra la Figura 2.5.
- Adicionalmente, se ha creado una página web⁴ en la que almacenar todo el material didáctico empleado en los talleres, además de aportar una explicación audiovisual por medio de tutoriales subidos a YouTube y relacionar cada proyecto con su correspondiente solución subida a un repositorio de GitHub⁵. De esta forma se ponen los recursos en manos de la ciudadanía, permitiendo cualquier persona que no pueda atender los cursos puede aprender de manera autónoma.

Existen diversas alternativas para el desarrollo de dispositivos que, mediante el uso de sensores —integrados o externos— nos permiten la recolección de datos, siendo Arduino y Raspberry las más conocidas.

Sin embargo, con motivo de cumplir con la finalidad instructiva de la Cátedra de Tecnologías Cívicas y Empoderamiento, el dispositivo elegido para los talleres de enseñanza de sensorética sin requerir conocimientos previos de informática es la tarjeta micro:bit, que se explicará con mayor detalle en el Capítulo 3. Esta elección se debe a que el dispositivo surge como un proyecto para enseñar programación a los niños de Reino Unido, y es por ello que se programa de manera sencilla e intuitiva.

En conclusión, el proyecto de la Escuela de Tecnologías Cívicas ha cumplido de manera satisfactoria los objetivos marcados, logrando en dos meses un total de 191 asistentes —véase en la Tabla 2.1—, y consiguiendo numerosas valoraciones positivas a través de las encuestas de satisfacción de los talleres impartidos, de las cuales se muestra un ejemplo en la Figura 2.6.

⁴Disponible en <https://etc.webs.upv.es>

⁵Disponible en <https://github.com/etc-catedratce>

Fecha	Centro educativo / Organización	Asistentes
5/3/19-6/3/19	Civic Wise	5
10/4/2019	IES Sorolla	16
10/4/2019	IES Sant Jordi	16
12/4/2019	Colegio Pío XII	24
16/4/2019	IES Albal	20
17/4/2019	Colegio San Roque	23
8/5/2019	IES Veles e Vents	27
10/5/2019	Ayuntamiento de Sueca	20
14/5/2019	IES Albal	30
21/5/2019	MUGI - ETSINF UPV	10
Total		191

Tabla 2.1: Talleres impartidos por la Escuela de Tecnologías Cívicas.

Taller Colegio San Roque (17/04/19)

	Espacio	Instalaciones	Personal	Iniciativa	Materiales	Duración (1-Muy largo, 3-Adecuado, 5-Muy corto)	Dificultad (1-Muy difícil, 3-Adecuado, 5-Muy fácil)	Recomendaría (1-No, 5-Sí)	Sugerencias y comentarios
Encuesta 1	5	5	5	5	5	5	3	5	Muy bien llevado y entretenido a la par que educativo.
Encuesta 2	5	5	5	5	5	3	3	5	
Encuesta 3	5	5	5	5	5	3	3	5	
Encuesta 4	5	5	5	5	5	3	3	5	
Encuesta 5	4	5	5	5	5	1	3	5	
Encuesta 6	4	5	5	5	5	4	3	5	
Encuesta 7	5	5	5	5	5	4	3	5	
Encuesta 8	5	5	5	5	4	4	3	5	
Encuesta 9	5	5	5	5	4	5	3	5	
Encuesta 10	5	5	4	4	4	3	3	5	
Encuesta 11	5	5	5	5	4	4	3	5	
Encuesta 12	5	5	5	5	4	3	3	5	
Encuesta 13	5	5	5	5	5	5	3	1	
Encuesta 14	5	5	5	5	5	5	3	5	
Encuesta 15	5	5	5	5	5	5	3	5	
Encuesta 16	5	5	5	5	5	5	3	5	
Encuesta 17	5	5	5	5	5	4	5	2	Me ha gustado muchísimo este taller y me lo he pasado bien.
Encuesta 18	5	5	5	5	5	5	3	2	5
Encuesta 19	5	5	5	5	5	5	3	3	5
Encuesta 20	5	5	5	5	5	5	3	3	5
NUM_VOTOS	20	20	20	20	20	20	20	20	2
TOTAL	98	100	99	95	91	62	64	96	
MEDIA	4,9	5	4,95	4,75	4,55	3,1	3,2	4,8	

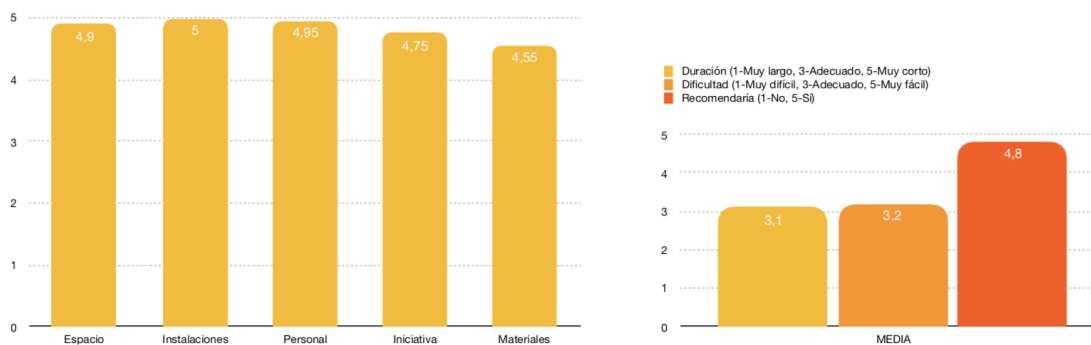


Figura 2.6: Encuestas de satisfacción del taller impartido al Colegio San Roque de Valencia.

CAPÍTULO 3

La tarjeta micro:bit

En esta sección introduciremos qué es la tarjeta micro:bit y cuál es su principal motivación, relacionándola con propuestas similares a la misma. Además, se detallan tanto aspectos de software —lenguajes y entornos de programación asociados a la misma— como aspectos físicos, es decir, arquitectura y sensores que la componen, así como los sensores externos compatibles necesarios para este trabajo.

3.1 Origen de la tarjeta micro:bit

Con la velocidad a la que está evolucionando la tecnología y la importancia que ha cobrado en nuestras vidas en las últimas décadas, no se trata de una idea desorbitada el considerar la programación a nivel básico como un conocimiento tan necesario hoy en día como lo es aprender a sumar o escribir. Es por ello que están surgiendo soluciones para su enseñanza en colegios, cuyo denominador común son las interfaces intuitivas que hagan de la programación una tarea lo más sencilla posible.

Un ejemplo de esto es Scratch, plataforma que permite la programación *online*, principalmente enfocado a la creación de juegos con el fin de captar el interés de los más pequeños, iniciándoles a través de juegos la habilidad del pensamiento computacional. La tarjeta micro:bit no es más que otra propuesta educativa (véase la Figura 3.1) con este fin, impulsada por la BBC (cadena pública británica) en consonancia con otras compañías tecnológicas de gran importancia para acercar a los colegios de manera económica los recursos necesarios para el aprendizaje básico de programación.

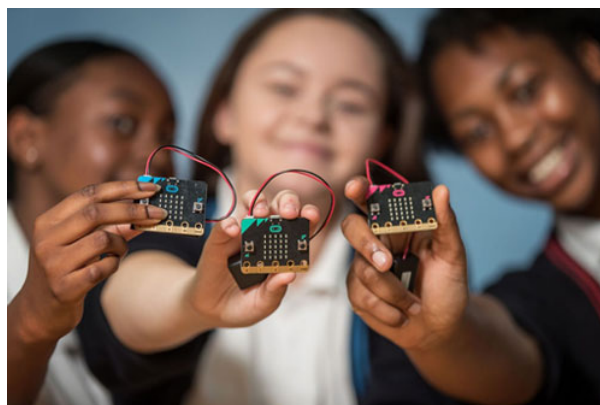


Figura 3.1: La tarjeta micro:bit como enseñanza de programación enfocada a niños.

3.2 Arquitectura

La tarjeta micro:bit (Figura 3.2) se trata de una microcontroladora de 43x52mm con procesador de 32 bits, 16 KB de memoria RAM y 256 KB de memoria flash. Además, cuenta con una antena Bluetooth de 2.4 GHz capaz de comunicarse con una aplicación móvil, y antena de radio para comunicar a varias tarjetas micro:bit entre ellas. En la Figura 3.3 se muestran los elementos que la componen.



Figura 3.2: La tarjeta micro:bit.

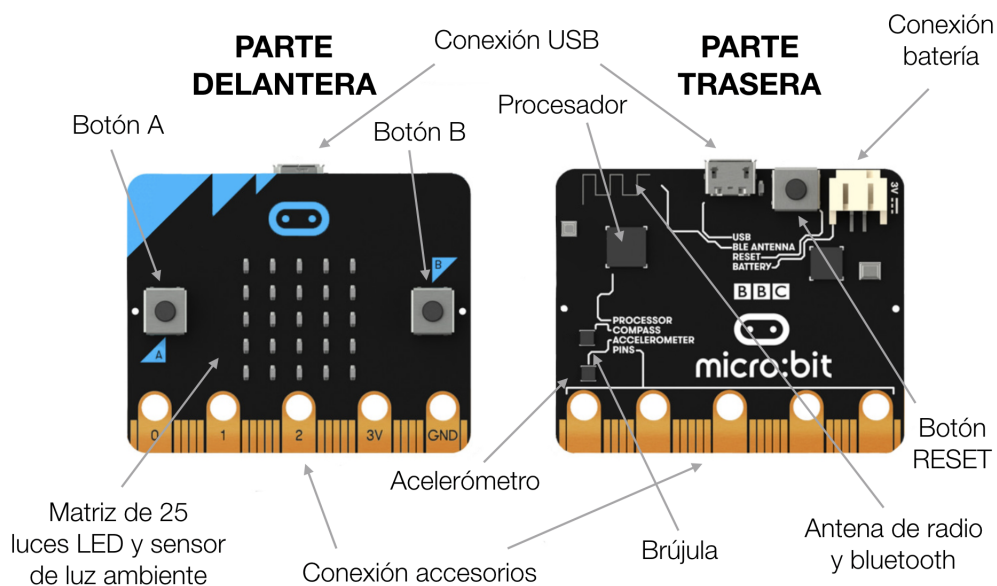


Figura 3.3: Elementos de la tarjeta micro:bit.

Cabe destacar que además de los sensores integrados en la tarjeta e indicados en la imagen de la Figura 3.3, la tarjeta micro:bit también es capaz de leer la temperatura del dispositivo a través del procesador.

3.3 Componentes externos

La tarjeta micro:bit se suele comercializar acompañada de una batería y un cable USB, tal y como se muestra en la Figura 3.4. Sin embargo, para el desarrollo de este trabajo se requieren componentes adicionales que trataremos en este apartado.

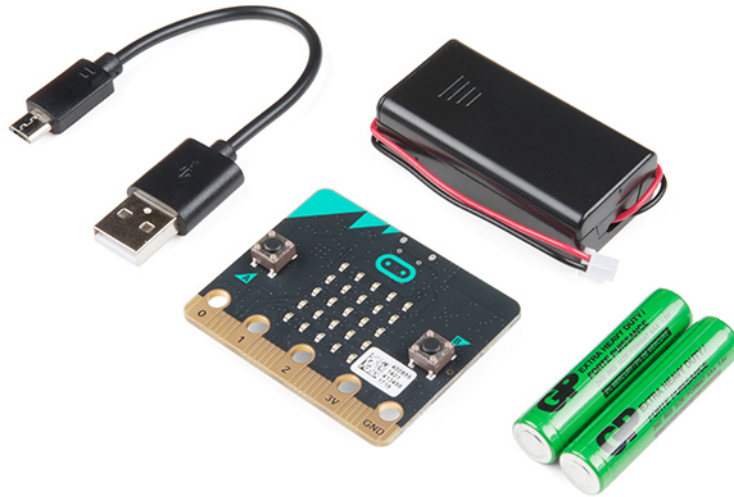


Figura 3.4: Kit básico de comercialización de la tarjeta micro:bit.

3.3.1. Placa Octopus:bit

Esta placa (véase la Figura 3.5), comercializada junto con la tarjeta micro:bit en la mayoría de kits que constan de componentes adicionales, es una placa que nos facilita la conexión de componentes externos. Provee al usuario de 16 pines en los que realizar una fácil conectividad de componentes en contraposición a las pinzas de cocodrilo que han de usarse al conectarlos directamente a los pines de la tarjeta micro:bit.

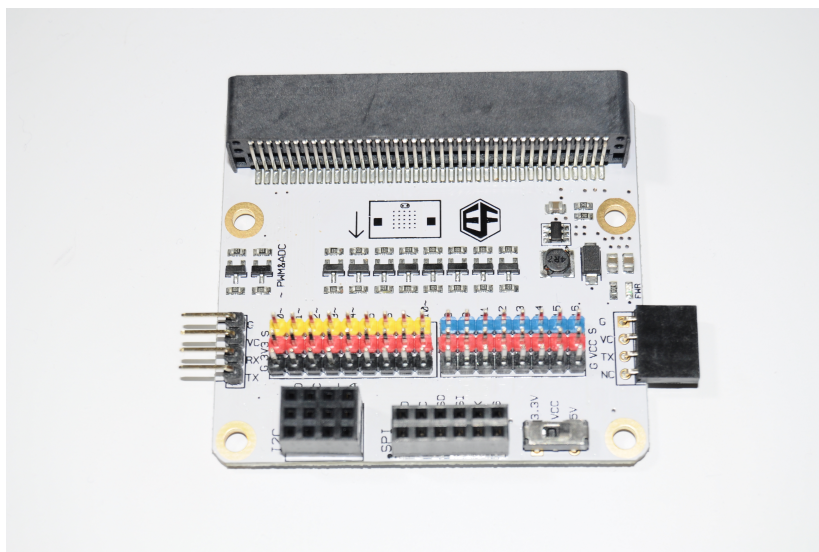


Figura 3.5: Placa Octopus:bit.

3.3.2. Pantalla OLED

Esta pantalla (Figura 3.6) se conecta a la tarjeta micro:bit a través de la placa Octopus:bit y nos permite la fácil lectura de textos (debido a la dificultad que supone la lectura de los misos a través de la matriz de LEDS incorporada en la tarjeta).

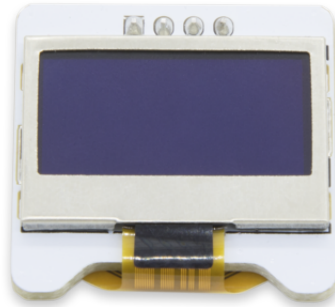


Figura 3.6: Pantalla OLED.

3.3.3. Módulo Wi-Fi ESP8266

El módulo ESP8266 contiene una antena *Wi-Fi* que nos permitirá conectar la tarjeta micro:bit a dichas redes. Para que el módulo pueda ser conectado a la tarjeta micro:bit a través de la placa Octopus:bit se precisa de un adaptador, tal y como muestra la Figura 3.7. Más adelante, concretamente en el Capítulo 4, se aborda de manera detallada cómo se ha de realizar la conexión.

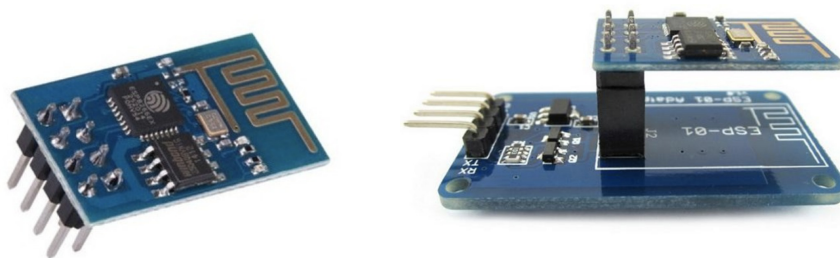


Figura 3.7: Módulo ESP8266 y módulo junto con su adaptador.

3.3.4. Sensor PIR

Sensor piroeléctrico (Figura 3.8) que, gracias a la detección de cambios de temperatura, es capaz de detectar movimientos, llegando a hacerlo incluso a 5 metros de distancia respecto al sensor. Se comercializa en algunos kits junto con la tarjeta micro:bit y se conecta a la misma a través de la placa Octopus:bit.

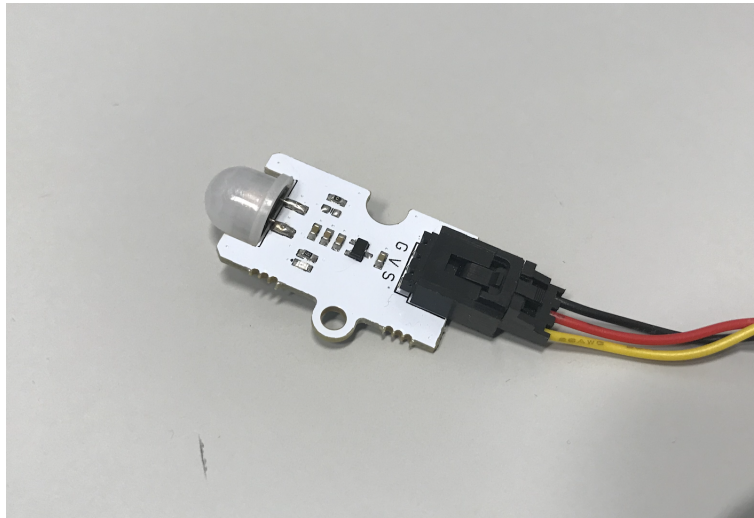


Figura 3.8: Sensor PIR.

3.3.5. Sensores emisor y receptor de luz infrarroja

Se precisan tanto sensores emisores de luz infrarroja (Figura 3.9) como sensores receptores de la misma (Figura 3.10). No se comercializan de manera habitual con la tarjeta micro:bit, pero los sensores de luz infrarroja compatibles con Arduino serán de igual manera compatibles con ella, conectándose también a través de la placa Octopus:bit.

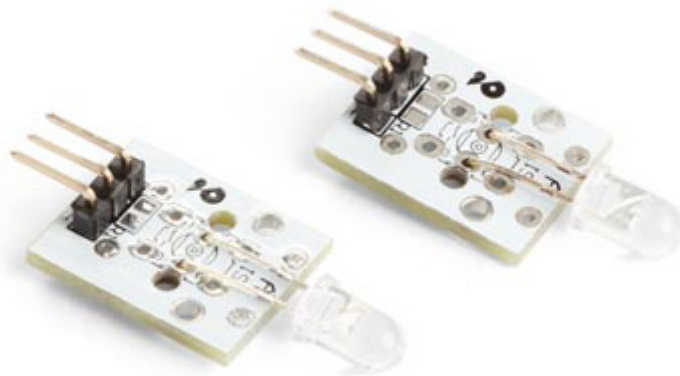


Figura 3.9: Sensor emisor IR compatible con Arduino.

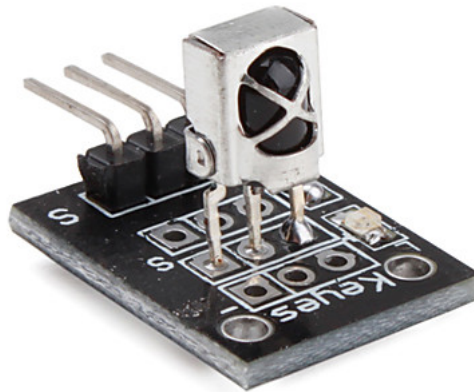


Figura 3.10: Sensor receptor IR compatible con Arduino

3.4 Entornos de desarrollo

El principal entorno de desarrollo es MakeCode¹ —véase en la Figura 3.11—, plataforma *online* diseñada por Microsoft para el desarrollo de código mediante programación por bloques y exportable a un fichero con el código codificado a hexadecimal (extensión **.hex**), de manera que pueda ser leído por sistemas embebidos como es el caso de la tarjeta micro:bit.

Además, ofrece la posibilidad de cambiar del entorno de programación por bloques a código en JavaScript, y permite la descarga de librerías adicionales, comúnmente subidas por desarrolladores de componentes externos compatibles con la micro:bit, para añadir nuevas funciones predefinidas al menú inicial.

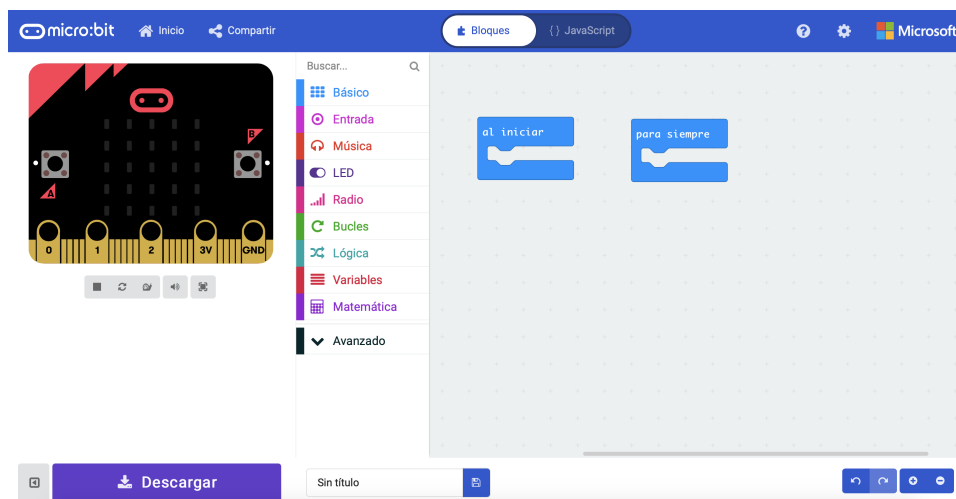


Figura 3.11: Entorno de programación MakeCode.

Pero la tarjeta micro:bit no se limita a este entorno, ya que existen compiladores que son capaces de transformar el código de lenguajes de programación comúnmente utilizados como bien son C++ y Python a ficheros hexadecimales compatibles con esta tarjeta.

¹Disponible en <https://makecode.microbit.org>

El más conocido de estos lenguajes de alto nivel es MicroPython, capaz de compilar código en lenguaje Python y que además cuenta con módulos que facilitan el acceso a los sensores integrados de la micro:bit así como sus pines. Todo ello se detallará de manera más específica en el Capítulo 4.

CAPÍTULO 4

La recolección de datos en la tarjeta micro:bit

En este capítulo se presentan dos alternativas para la lectura y almacenamiento de datos, bien mediante conexión USB y registrando la información en un *log*¹ o bien enviando la información a través de una red *Wi-Fi* —para lo que será necesario el módulo ESP8266 referido en los aspectos técnicos de la tarjeta micro:bit— a una plataforma *online* encargada de recibir y representar los datos.

4.1 Conexión USB y uso de *logs*

Existe una forma de recolectar los datos que perciben los sensores de la tarjeta micro:bit sin necesidad del uso de *Wi-Fi*, y es mediante la creación de un fichero *.csv* en el que se almacenan los datos.

Para este fin se ha empleado MicroPython —explicado en el apartado 3.4. *Entornos de desarrollo* del Capítulo 3— para la generación del fichero *.csv* y una herramienta llamada MicroFS [8] que, a través de una interfaz de línea de comandos, nos permite recuperar el fichero generado.

4.1.1. Código en MicroPython

Para este ejemplo hemos tomado como punto de partida el código descargado del proyecto 27 de la página <http://www.microbitsandbobs.co.uk/projects.html>, el cual se muestra a continuación:

```
#####  
#temp logger D Burrin #  
#19/6/16 #  
#Uses file system in Mu #  
#Limitaion 32k file storage #  
#Limition no append so run out of memory #  
#due to reading into a variable and this increasing #  
#in size o store append data and write back #  
#This example samples aprox 30sec at 0.5 sec per sample #  
#####
```

¹Log (registro) es un archivo de texto en el que constan cronológicamente los acontecimientos que han ido afectando a un sistema informático.

```
from microbit import *

#initialise base variables
logtime = running_time()
currtemp = temperature()
data2write=""

#sample rate in seconds
interval = 0.5
#set file name
filename = "datalogger.csv"

#functions
def readdata(filename):
    with open (filename,"r") as myfile:
        data = myfile.read()
        myfile.close()
    return(str(data))

def write2file(filename,data):
    with open (filename,'w') as myfile:
        data=str(data)
        myfile.write(data)
        myfile.close()

def get_time(logtime):
    logtime = (running_time() - logtime)//1000
    #logtime = round(logtime,2)
    return logtime

#main code
while True:
    #Start up message
    display.show("B")

    #exit main loop to enable download
    if button_a.is_pressed():
        display.show("X")
        sleep(100)
        display.clear()
        break

    #start logging
    if button_b.is_pressed():
        #logging data
        display.show("Y")

        #set header file for csv and write it
        data2write ="temp,time\r"
        write2file(filename,data2write)

        #set number of samples to be taken
```

```
for x in range (60):

    data2write = readdata(filename)
    currtemp = str(temperature())
    logtime = get_time(logtime)
    data2write = data2write + currtemp + "," + str(logtime) + "\r"
    write2file (filename,data2write)
    data2write = readdata(filename)
    #put interval in milliseconds
    sleep(interval*1000)

#test output to rpl
#print(data2write)

#exit loggin loop
if button_a.is_pressed():
    display.clear()
    break
```

Como podemos observar, este código está creando un fichero que recopila los datos del sensor de temperatura de la tarjeta micro:bit, almacenándolos en la variable **currtemp** mediante la función **temperature()**. Esta última función forma parte del módulo **micro:bit** importado al principio del código. En su documentación [1] podemos consultar el resto de funciones disponibles, mediante las cuales se puede leer toda la información que recibe la tarjeta micro:bit. Por ejemplo, si quisiéramos realizar un registro de un sensor externo situado en el pin 0, reemplazaríamos la función **temperature()** por **pin0.read_digital()** o **pin0.read_analog()** —en función de si quisiéramos leer el dato de manera digital o analógica.

Otras modificaciones que podemos realizar son las configuraciones del número de datos que queremos almacenar —en el código se realiza un bucle de 60 iteraciones, por lo que consistiría en la modificación de esta cifra— y el intervalo de medición del dato, almacenado en la variable **interval** en función del número de segundos —en el ejemplo se guarda el dato de la temperatura cada medio segundo, y es por ello que almacena un 0,5.

Por último, en la variable **filename** podemos indicarle qué nombre queremos que tenga nuestro fichero, manteniendo la extensión **.csv**. En el ejemplo vemos que se le ha dado el nombre **datalogger.csv**.

Una vez modificado el código, se compila de manera que pueda ser reconocido por la tarjeta micro:bit, es decir, codificado en hexadecimal (extensión **.hex**). Para ello podemos descargar el compilador MicroPython en nuestro equipo, pero también existen compiladores *online* que nos facilitan esta tarea. Un ejemplo de esto último es el compilador de Python para micro:bit², en el cual solo habría que pegar el código, realizar las modificaciones necesarias y descargarlo listo para ser insertado en la tarjeta micro:bit.

Finalmente, cuando el programa haya sido insertado, ejecutaremos la tarjeta micro:bit y se mostrará una "B", indicando que ha de pulsarse el botón B para iniciar el proceso de almacenamiento de datos. Durante el almacenamiento se mostrará una "Y", y una vez vuelva a aparecer la "B", se pulsará el botón A para concluir la ejecución del programa.

²Consultado en <https://python.microbit.org/v/1.1>

Este fichero permanecerá en la memoria de la tarjeta micro:bit aunque la reiniciemos, siempre que no sobrescribamos el programa actual de la micro:bit, en cuyo caso se borraría todo lo que contiene.

4.1.2. Recuperar los *logs* con MicroFS

Tras haber ejecutado el programa, y conectando la tarjeta micro:bit a través del cable USB al ordenador, si se accede a la carpeta destinada a la tarjeta veremos que no aparece ningún fichero. Esto se debe a que el modo de almacenamiento masivo está destinado a copiar archivos con código hexadecimal (extensión **.hex**) y no a los ficheros existentes en su sistema de archivos generados por MicroPython, y es por ello que para proceder a la recuperación del fichero se precisa la ayuda de MicroFS.

Para su instalación se seguirán los pasos citados en su documentación oficial [8], y ya instalado ejecutaremos el siguiente comando:

```
$ ufs get nombredelfichero.csv
```

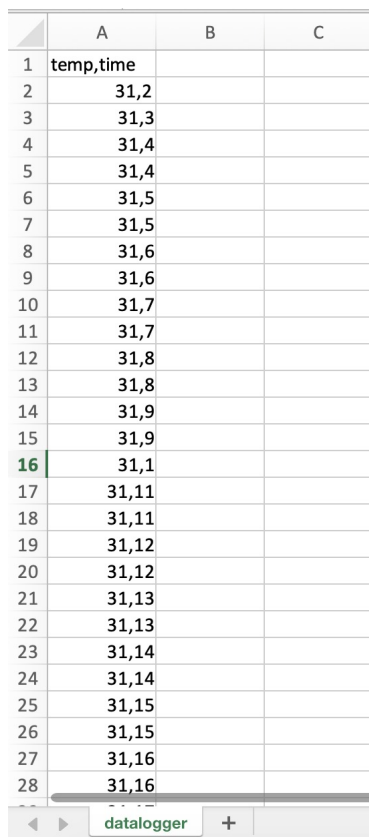
A continuación se muestra el código fuente de la función:

```
def get(filename, target=None, serial=None):
    """
    Gets a referenced file on the device's file system and copies it to the
    target (or current working directory if unspecified).

    If no serial object is supplied, microfs will attempt to detect the
    connection itself.

    Returns True for success or raises an IOError if there's a problem.
    """
    if target is None:
        target = filename
    commands = [
        "from microbit import uart",
        "f = open('{}', 'rb').format(filename)",
        "r = f.read",
        "result = True",
        "while result:\n result = r(32)\n if result:\n  uart.write(result)\n",
        "f.close()",
    ]
    out, err = execute(commands, serial)
    if err:
        raise IOError(clean_error(err))
    # Recombine the bytes while removing "b" from start and "" from end.
    with open(target, 'wb') as f:
        f.write(out)
    return True
```

Esta función nos permite la obtención de un fichero (tal y como muestra la Figura 4.1) existente en el sistema de archivos de la tarjeta micro:bit a través del nombre que recibe como parámetro, y lo almacena predeterminadamente en la carpeta desde la cual se ejecuta el comando.



	A	B	C
1	temp,time		
2	31,2		
3	31,3		
4	31,4		
5	31,4		
6	31,5		
7	31,5		
8	31,6		
9	31,6		
10	31,7		
11	31,7		
12	31,8		
13	31,8		
14	31,9		
15	31,9		
16	31,1		
17	31,11		
18	31,11		
19	31,12		
20	31,12		
21	31,13		
22	31,13		
23	31,14		
24	31,14		
25	31,15		
26	31,15		
27	31,16		
28	31,16		

Figura 4.1: Fichero generado por la tarjeta micro:bit con datos de temperatura.

4.2 Conexión *Wi-Fi*

Esta segunda alternativa consiste en el envío de los datos que recibe la tarjeta micro:bit a través de una red *Wi-Fi*, por lo que precisaremos tanto del adaptador como del módulo ESP8266 —tratados en el apartado 3.3. *Componentes externos* del Capítulo 3.

4.2.1. Plataformas de envío de datos

Existen dos plataformas compatibles con el entorno de programación por bloques MakeCode: ThingSpeak y The Things Network. Para ambas se precisa de la descarga de una librería adicional, que descargaremos desde el menú de *Extensiones* de la esquina derecha superior (véase la figura 4.2).

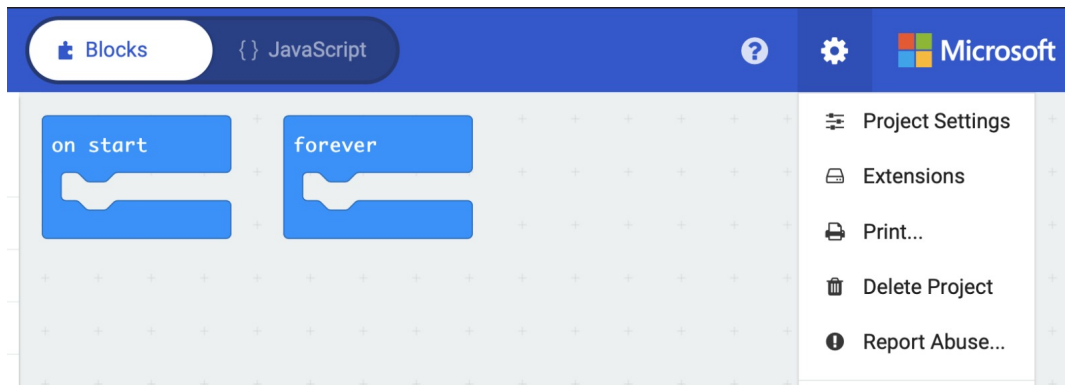


Figura 4.2: Menú para la descarga de librerías en MakeCode.

Para el uso de la plataforma The Things Network se precisa la descarga de la librería **iot-lora-node** así como del componente de Lora para micro:bit, cuyo precio aproximado está alrededor de los 30-40 €.

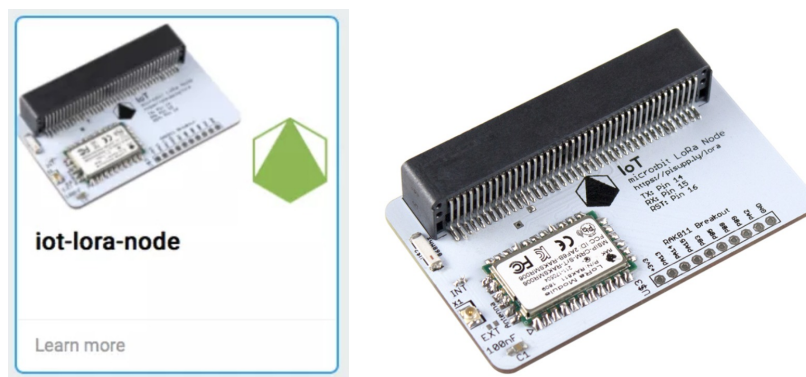


Figura 4.3: Librería **iot-lora-node** para MakeCode y componente de Lora para micro:bit.

Por otro lado, para emplear la plataforma ThingSpeak precisaremos de la librería **iot-environment-kit** además del módulo ESP8266 (tanto adaptador como antena, tal y como se muestra en la Figura 3.7). Estas dos piezas se pueden obtener por 10-15 €, lo que supone una reducción notable respecto al precio del componente de Lora, y es por ello que para este trabajo se ha decidido realizar el envío de datos a través de esta última plataforma.

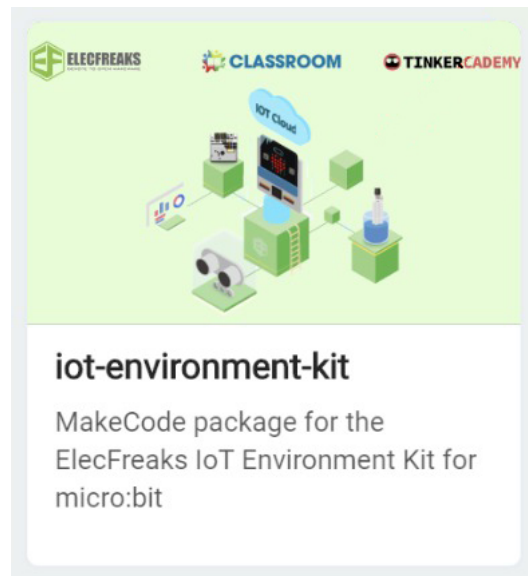


Figura 4.4: Librería `iot-environment-kit` para MakeCode.

ThingSpeak³ (véase Figura 4.5) es una plataforma de código abierto mediante la cual se pueden generar canales a los cuales enviar datos y poder visualizarlos, almacenarlos y tratarlos. Además, está integrada con MATLAB y permite la visualización y tratamiento de datos haciendo uso del mismo sin requerir su licencia.



Figura 4.5: Plataforma ThingSpeak.

Se puede crear una cuenta de manera gratuita, permitiendo la creación de un máximo de cuatro canales. Para este ejemplo se ha creado el canal *Pruebas módulo Wi-Fi* (véase en la figura 4.6) que recibirá un solo parámetro, *Temperatura °C*, con el valor de la temperatura de la tarjeta micro:bit.

³Disponible en <https://thingspeak.com>.

Pruebas módulo wi-fi

Channel ID: 812059
 Author: mnjmaria
 Access: Private

Private View Public View Channel Settings Sharing API Keys

Channel Settings

Percentage complete 30%

Channel ID 812059

Name

Description

Field 1

Figura 4.6: Configuración del canal de ThingSpeak.

Una vez finalizada la configuración, en el apartado *API Keys* encontraremos la clave de escritura, que necesitaremos a la hora de programar nuestra micro:bit y que aparece subrayada en la Figura 4.7.

Private View Public View Channel Settings Sharing API Keys Data Import / Export

Write API Key

Key PFMCZBJAFD1U6FPL

[Generate New Write API Key](#)

Read API Keys

Key ICS0SCD4PX0BXSMS

Note

[Save Note](#) [Delete API Key](#)

Help

API keys enable you to write data to a channel or read data from a private channel. API keys are auto-generated when you create a new channel.

API Keys Settings

- **Write API Key:** Use this key to write data to a channel. If you feel your key has been compromised, click **Generate New Write API Key**.
- **Read API Keys:** Use this key to allow other people to view your private channel feeds and charts. Click **Generate New Read API Key** to generate an additional read key for the channel.
- **Note:** Use this field to enter information about channel read keys. For example, add notes to keep track of users with access to your channel.

API Requests

Update a Channel Feed

```
GET https://api.thingspeak.com/update?api_key=PFMCZBJAFD1U6FPL&field1=0
```

Get a Channel Feed

```
GET https://api.thingspeak.com/channels/812059/feeds.json?api_key=ICS0SCD4PX0BXSMS&results=2
```

Figura 4.7: Apartado con las claves API del canal de ThingSpeak.

4.2.2. Diseño del código

Tras descargar la librería *iot-environment-kit* (Figura 4.4), veremos que en el menú lateral de MakeCode nos aparecen nuevas entradas de menú con funciones adicionales. Para este apartado haremos uso de la entrada *ESP8266_IoT*, cuyas funciones se muestran en la Figura 4.8.

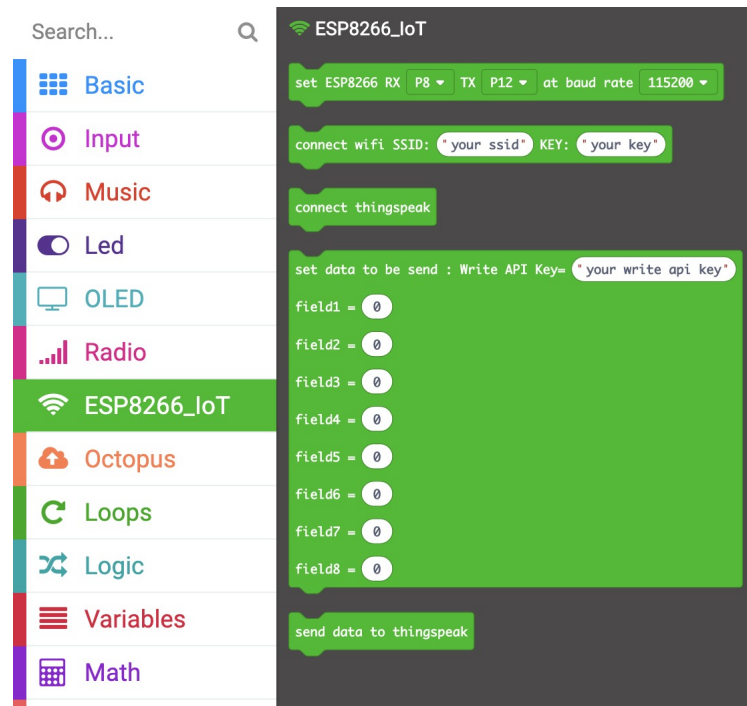


Figura 4.8: Menú *ESP8266_IoT* en MakeCode.

En primer lugar, al iniciar la ejecución del programa emplearemos la primera de las funciones del menú, asociando el RX al pin 0 y TX al pin 1.

Después, en la función **connect wifi** escribiremos en SSID el nombre de nuestra red *Wi-Fi* y en KEY su contraseña. Una vez conectado al *Wi-Fi* podremos realizar el envío, por lo que emplearemos la función **connect thingspeak** seguida de **set data to be send**, en la que introduciremos la clave API de escritura a la que se hace referencia en el subapartado anterior (4.2.1 *Plataformas de envío de datos*, Figura 4.7) y la temperatura en el primer parámetro del envío (**field1**).

Una vez modificada la función de preparación del envío únicamente quedaría hacer uso de la función **send data to thingspeak** para enviar los datos. Todo ello se inserta en un bucle que se repite para siempre con tal de estar continuamente enviando datos.

Adicionalmente, y si se cuenta con una pantalla OLED, se podrá programar de manera que muestre por pantalla toda la trazabilidad de nuestro programa. El código resultante se muestra en la Figura 4.11.

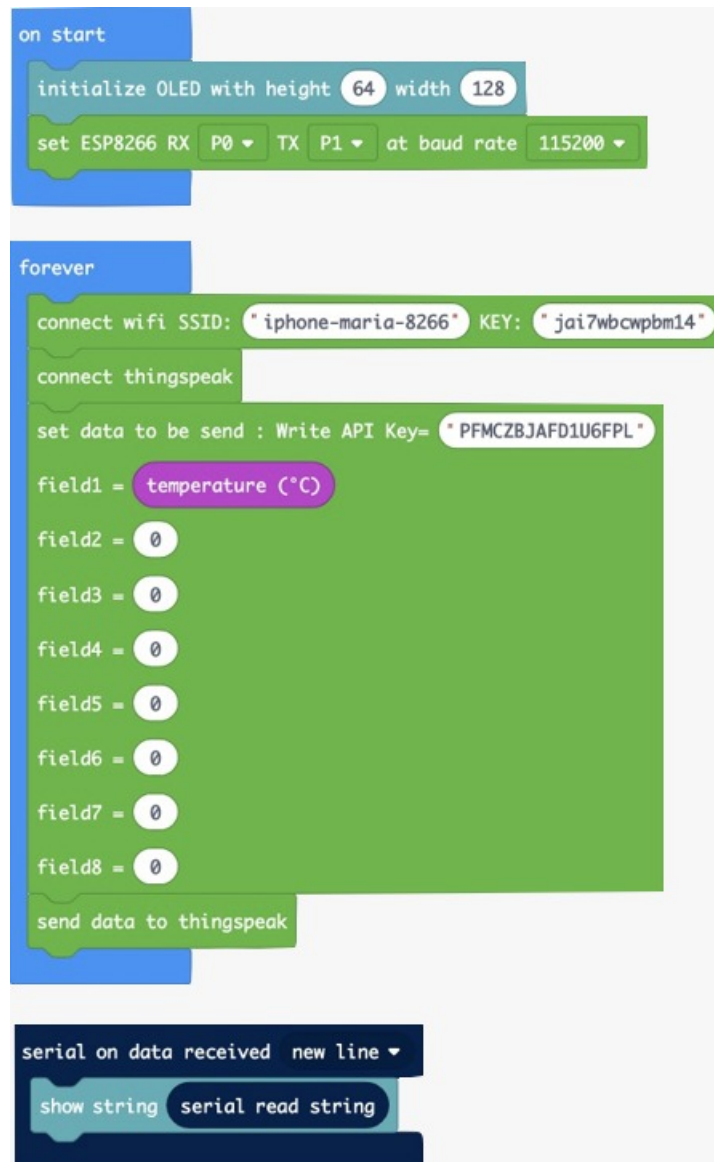


Figura 4.9: Código para el envío de datos de la tarjeta micro:bit vía Wi-Fi.

Además, MakeCode habilita un simulador de manera que nos muestra una consola con lo que leeremos en la OLED una vez ejecutado nuestro programa, que en este caso debería ser lo siguiente:

```

AT+RST
AT+CWMODE=1
AT+CWJAP="iphone-maria-8266","jai7wbcwpbm14"
AT+CIPSTART="TCP","api.thingspeak.com",80
AT+CIPSEND=107
GET /update?key=PFMCZBJAFD1U6FPL&field1=21&field2=0&field3=0&field4=0
&field5=0&field6=0&field7=0&field8=0
  
```

Por último, para poder consultar el código fuente de todas las funciones extraídas de la librería descargada, se puede cambiar de la visualización para programación por bloques a la visualización de programación en JavaScript (véase Figura 4.11), de manera que debajo del simulador nos aparecerá un explorador con todo el código en JavaScript que podremos consultar e incluso modificar.

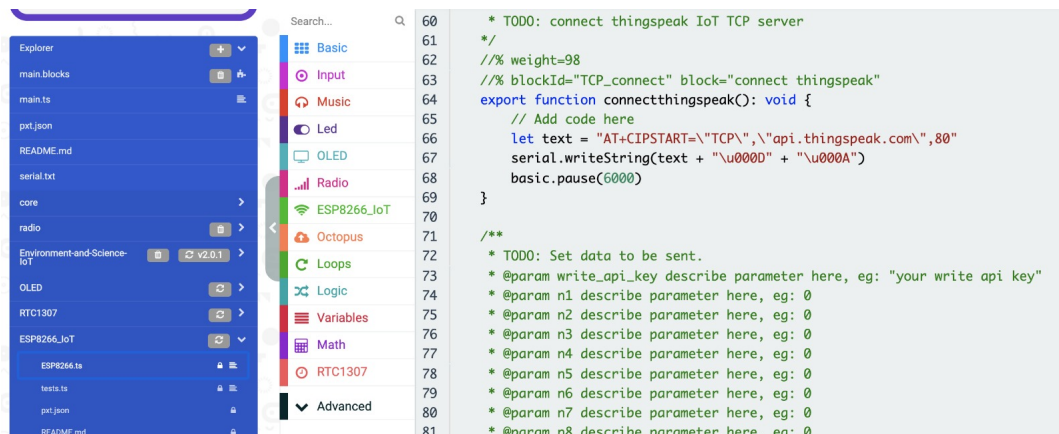


Figura 4.10: Menú de exploración y código fuente de la función **connect thingspeak** en JavaScript.

4.2.3. Conexión de los componentes

Una vez insertado el programa en la tarjeta micro:bit, el último paso que queda por realizar antes de proceder a la ejecución del programa es la conexión de los componentes.

A través de la placa Octopus:bit conectaremos la OLED y el adaptador del módulo ESP8266. Para este último, necesitaremos cables individuales, conectando RX, V y GND al pin 0 de la Octopus:bit, mientras que TX irá al pin 1, quedando el resto del pin vacío. Una vez conectado debería quedar de la siguiente forma:

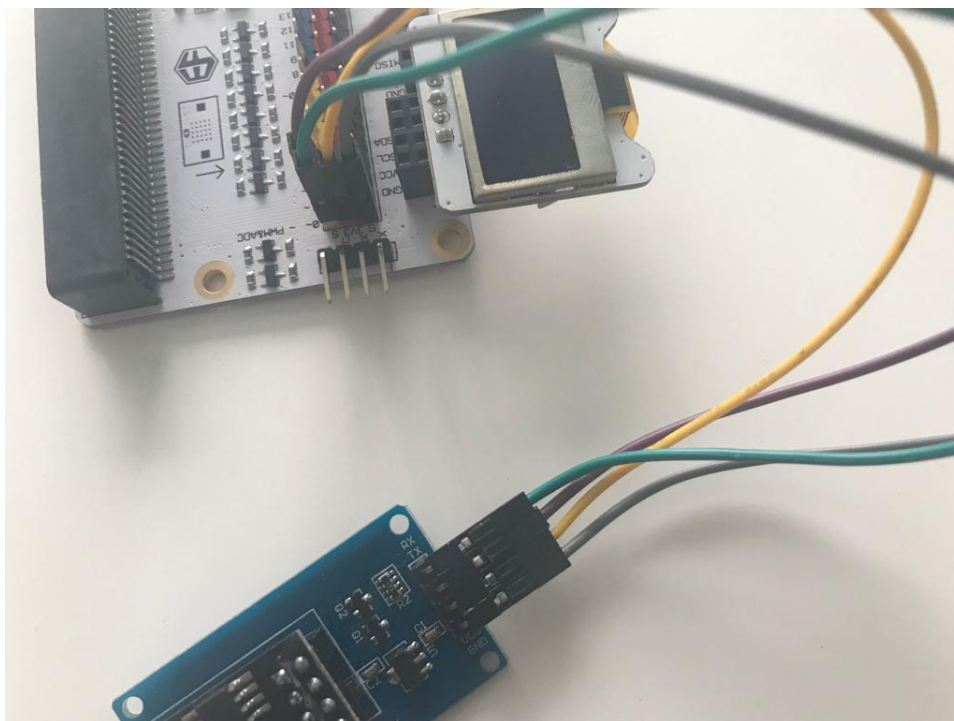


Figura 4.11: Conexión del módulo ESP8266 a la placa Octopus:bit.

Un dato a destacar es que la fuente de alimentación a la que conectemos la tarjeta micro:bit es de vital importancia, ya que el voltaje de la conexión USB al ordenador es muy débil y debido a esto el módulo *Wi-Fi* puede no funcionar, tal y como nos indican en su

blog⁴ los proveedores de ElecFreaks, principales desarrolladores de la librería descargada con la que estamos trabajando.

Para solucionar esto, se ha de verificar que la pestaña de voltaje de la placa Octopus:bit está situada en 5V y que se utiliza un adaptador de corriente con el voltaje mínimo necesario. En este caso se ha hecho uso de un adaptador de carga de iPhone, con salida de 5V (véase la Figura 4.12).

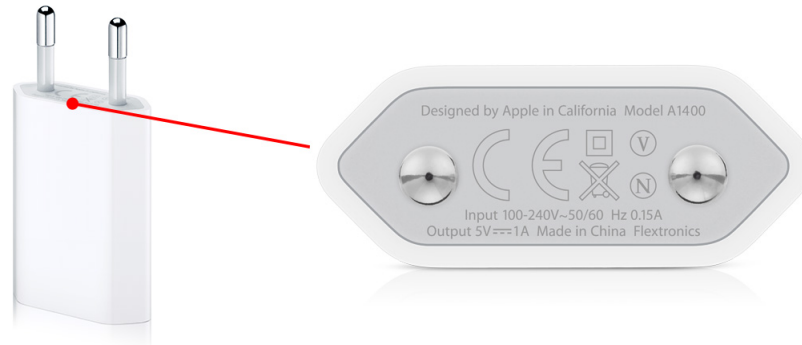


Figura 4.12: Adaptador de corriente con 5V de salida.

4.2.4. Visualización de datos

Una vez realizada la conexión de los componentes y tras ejecutar el programa, la tarjeta micro:bit enviará datos indefinidamente a nuestro canal, en este caso la temperatura. Estos datos los podremos observar desde la pestaña de visualización privada de nuestro canal, mostrada en la Figura 4.13.

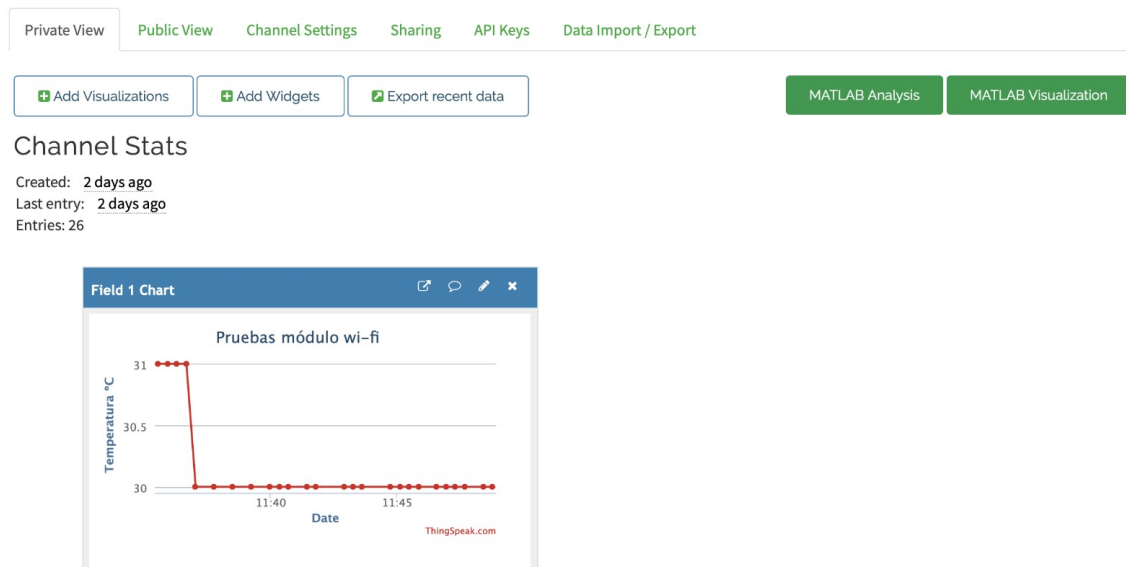


Figura 4.13: Visualización de datos de temperatura enviados por la tarjeta micro:bit.

Además de la visualización básica, se puede modificar el tipo de gráfica, analizar y visualizar los datos a través de MATLAB, añadir diversos *widgets* y exportar los datos de

⁴Post disponible en <https://www.electfreaks.com/store/blog/post/how-to-send-microbit-data-to-iot-platform.html/>

un solo parámetro o de todos los parámetros del canal en los formatos que se muestran en la Figura 4.14.

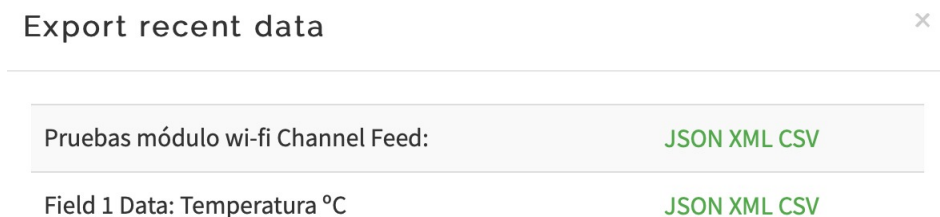


Figura 4.14: Exportar datos de un canal de ThingSpeak.

4.3 Conclusiones

Las soluciones presentadas en este apartado son válidas y funcionales, sin embargo, existen limitaciones para ambas. En primer lugar, la solución de generación de *logs* presenta dos inconvenientes de gran relevancia:

- El primero de ellos es la complejidad, ya que una de las principales metas de este trabajo es la divulgación de soluciones sencillas e intuitivas dirigidas a cualquier persona sin requerir conocimientos previos, y la programación requerida para la generación de alternativas en MakeCode o cualquier otro entorno de programación por bloques.
- Además, nos vemos limitados espacialmente, ya que el tamaño máximo de ficheros generados por MicroPython para ser almacenados en la tarjeta micro:bit ha de ser de 32 KB, según indica la documentación del sistema persistente de archivos de MicroPython para micro:bit⁵.

Estos dos inconvenientes son resueltos por la segunda solución presentada, ya que el envío vía *Wi-Fi* se puede programar a través de MakeCode y no existe límite espacial. No obstante, requiere tanto de librerías adicionales como de la compra de componentes externos, recursos que no precisaba la primera solución, que únicamente necesita una tarjeta micro:bit y un cable USB.

Aun así, se considera que las desventajas que presenta la primera alternativa son de mayor relevancia, y es por ello que para el diseño de las soluciones que forman este trabajo se almacenará la información a través del envío por *Wi-Fi*.

⁵Consultado en <https://microbit-micropython.readthedocs.io/en/latest/filesystem.html> y en el artículo con la documentación de MicroPython referenciado en la bibliografía [9].

CAPÍTULO 5

Gestión de residuos con la tarjeta micro:bit

Este capítulo presenta y describe con detalle las diferentes soluciones propuestas para la gestión de residuos en entornos de interior empleando la tarjeta micro:bit, así como el coste de su aplicación en un entorno real.

5.1 La micro:bit como incentivo al reciclaje

Esta primera solución requiere del uso de la tarjeta micro:bit junto con un sensor de presencia PIR —véase en el apartado 3.3. *Componentes externos* del Capítulo 3. Dicho sensor, a través de cambios de presión y temperatura, permite identificar movimientos en un radio de 5 metros, por lo que puede captar la atención de la gente que pase próxima a contenedores de reciclaje y encender luces para llamar su atención con respecto a los contenedores normales.

Además, esta información se enviará a un canal de ThingSpeak, de manera que se puede analizar esta información para conocer qué zonas son más frecuentadas a según qué horas, y con ello lograr la posible recolocación estratégica de papeleras de reciclaje.

5.1.1. Diseño del código

Partiremos como base del código implementado en la Figura 4.11 del Capítulo 4. En esta ocasión nuestro canal también recibe un único parámetro, pero esta vez no se tratará del valor de la temperatura, sino del valor del sensor PIR.

Realizaremos la lectura de este sensor digitalmente, de manera que si detecta movimiento enviará un 1, mientras que si no detecta movimiento enviará un 0.

Adicionalmente, leeremos de manera constante el dato del sensor PIR y en función de su valor encenderemos o apagaremos la matriz de luces LED de la tarjeta micro:bit. El código final para esta solución se muestra en la Figura 5.1.

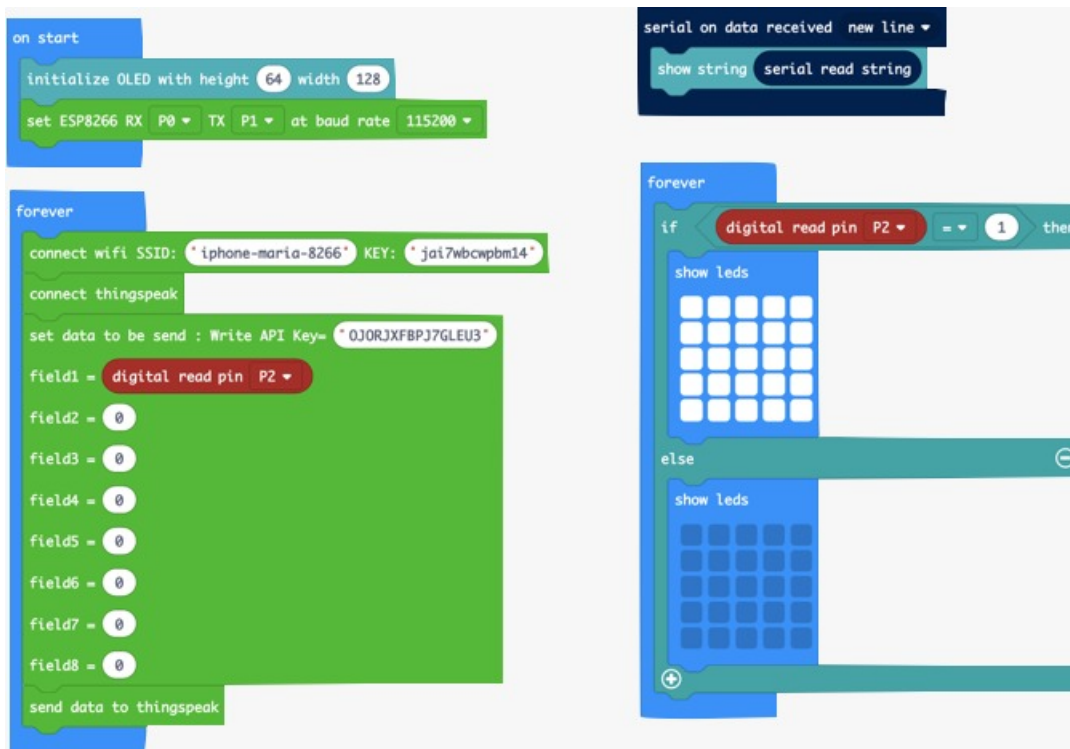


Figura 5.1: Código para la solución con el sensor PIR.

5.1.2. Conexión de los componentes

En primer lugar conectaremos el módulo *Wi-Fi* tal y como se explica en el apartado [4.2.3. Conexión de los componentes](#) del Capítulo 4. Seguidamente, se conectará el sensor PIR al pin 2, pin del cual se realiza la lectura en el código.

Es importante recordar que la fuente de alimentación ha de ser de un voltaje de 5V. Una vez conectados todos los componentes, el resultado será el mostrado en la Figura [5.2](#).

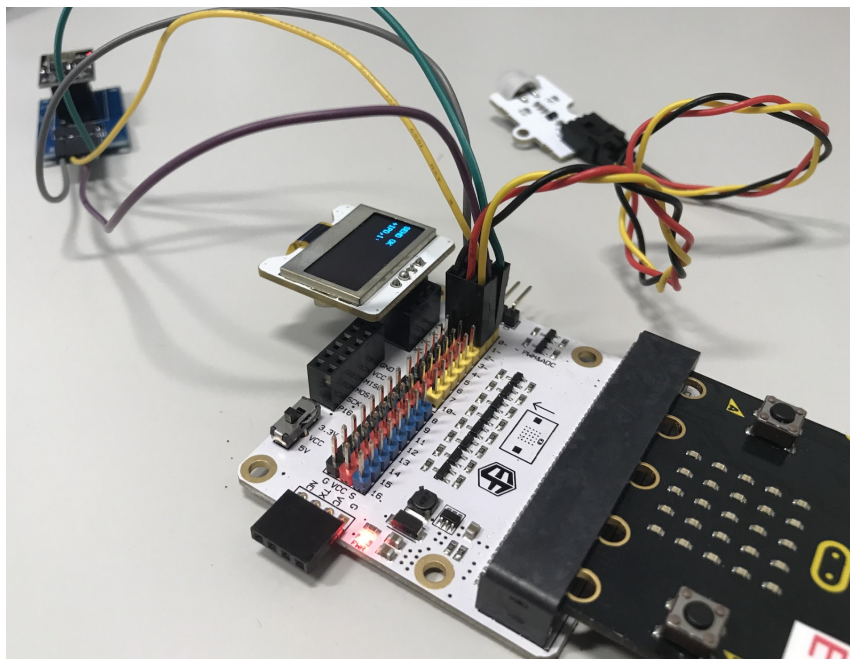


Figura 5.2: Conexión del PIR y el módulo ESP8266 a la placa Octopus:bit.

5.1.3. Visualización de datos

En la gráfica de la Figura 5.3 se muestran los datos recibidos a raíz de la ejecución del programa durante unos 10 minutos.

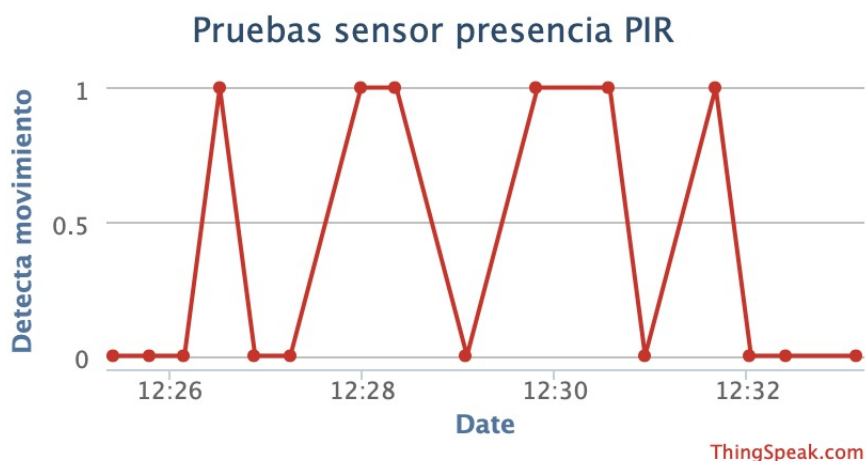


Figura 5.3: Datos recibidos por el sensor PIR.

Si recopilamos estos mismos datos durante un mayor periodo de tiempo y situamos varios sensores en diferentes lugares, al conocer tanto la hora de envío como el estado de los sensores, podemos realizar un estudio de las zonas que, en un mismo periodo de tiempo, tienen mayor tendencia de recibir un 1 que un 0 y viceversa, delimitando así las zonas que requieren de una mayor cantidad de papeleras.

5.2 La tarjeta micro:bit como contador de residuos

Para esta segunda solución se emplea el par de sensores infrarrojos (emisor y receptor) especificados en el apartado 3.3. *Componentes externos* del Capítulo 3.

Este par de sensores se coloca en extremos opuestos de la parte superior del contenedor de reciclaje, y se envían datos de manera constante para que al tirar un residuo corte el haz de luz infrarroja y el receptor no reciba señal, pudiendo así llevar un conteo de los residuos que se han tirado en el contenedor.

Este conteo se muestra en pantalla a modo de incentivo al reciclaje a los usuarios —técnica motivacional seguida, por ejemplo, para alentar a los ciudadanos a emplear los carriles bici en la ciudad de Valencia -véase en la Figura 5.4.



Figura 5.4: Incentivo al uso del carril bici en la ciudad de Valencia.

5.2.1. Diseño del código

Para el desarrollo de esta solución inicializaremos una variable a 0 que será nuestro contador y que para siempre se le mostrará al usuario.

El componente emisor de luz infrarroja estará para siempre enviando un 1 (que escribiremos digitalmente en el pin que haya sido conectado), mientras que, también para siempre, se realizará una lectura analógica del componente receptor de luz.

En caso de que el valor leído sea inferior a 1000 significará que no ha leído el envío del emisor porque algo ha interceptado el haz de luz infrarroja, y es en este momento cuando incrementaremos en uno el contador. El código resultante de esto es el que se muestra en la Figura 5.5.

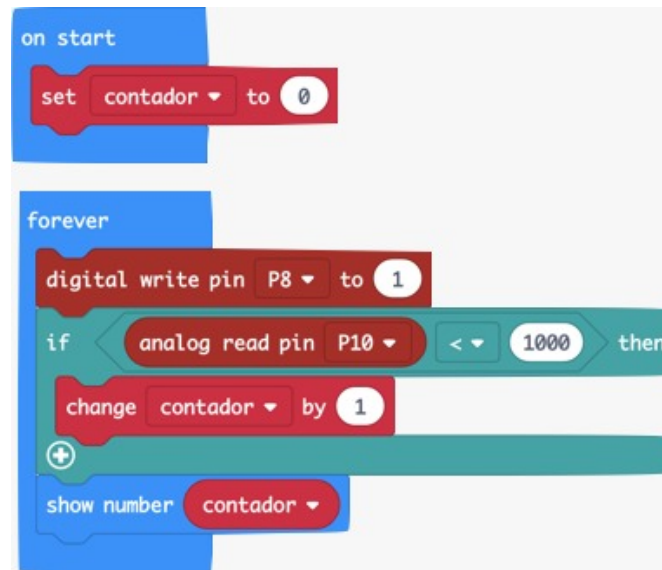


Figura 5.5: Código para la solución con sensores infrarrojos.

5.2.2. Conexión de los componentes

Para esta solución necesitaremos conectar la tarjeta micro:bit a la placa Octopus:bit, y a ella conectaremos los sensores infrarrojos emisor y receptor a los pines 8 y 10 respectivamente.

Se han elegido estos pines debido a que el pin 8 se trata de un pin exclusivamente de escritura, mientras que el pin 10 se puede leer analógicamente.

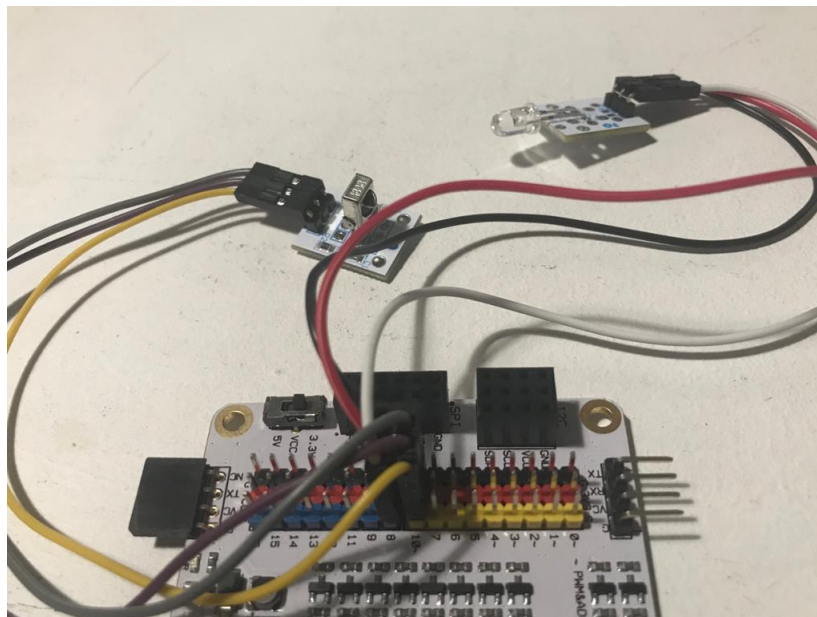


Figura 5.6: Conexión de los sensores infrarrojos.

5.3 La tarjeta micro:bit como herramienta de gestión de residuos en entornos de interior

Esta última solución presenta un prototipo que combina las soluciones vistas con anterioridad con el fin de sacar el máximo partido a la información generada, y es por ello que combina tanto el sensor de presencia como los sensores de emisión y recepción de luz infrarroja, además de hacer uso del módulo Wi-Fi.

Se aplica a un conjunto de papeleras, considerando que haya situadas juntas la papeleras destinada a residuos de papel y cartón, la destinada a plásticos y finalmente la orgánica. La idea es potenciar el uso de estos conjuntos de papeleras frente a las papeleras orgánicas que no están situadas junto a papeleras de reciclaje y no ofrecen al usuario la opción de reciclar.

Cada una de ellas contendría una tarjeta micro:bit con sensores de luz infrarroja, mientras que adicionalmente habría una cuarta micro:bit recibiendo los datos de cada una de ellas a través de la antena de radio y enviándolos vía *Wi-Fi*, además de tener conectado el sensor PIR.

Las tarjetas micro:bit situadas en cada una de las papeleras las llamaremos micro:bit emisora, mientras que nos referiremos a la tarjeta micro:bit que estará conectada al *Wi-Fi* y al sensor PIR como micro:bit receptora.

Las funcionalidades son las siguientes:

- El sensor de presencia de la tarjeta micro:bit receptora se encarga de encender la matriz de LEDs, llamando la atención de la gente a hacer uso de estos conjuntos de papeleras.
- Los sensores infrarrojos de las tarjetas micro:bit emisoras nos permiten llevar un conteo de los residuos que se tiran en la papeleras en la que están situadas. Además, en pantalla se agradece a cada usuario que las emplea.
- De manera periódica la tarjeta micro:bit receptora envía mediante el módulo *Wi-Fi* y a través de la plataforma *ThingSpeak* el conteo actual de residuos en cada una de las papeleras del conjunto y el estado del sensor de presencia, permitiéndonos saber tanto las horas del día a las que se frecuentan los alrededores del contenedor como su uso.
- En caso de que el usuario se encuentre una de las papeleras llena, puede pulsar el botón A de la tarjeta micro:bit emisora situada en la misma para informar a los servicios de recogida, también a través de la plataforma *ThingSpeak*. El servicio de recogida, una vez vaciada dicha papeleras, reinicia el contador pulsando B.

5.3.1. Diseño del código

Comenzaremos con las funcionalidades de la emisora, partiendo de base con el código implementado para la solución con sensores infrarrojos del apartado anterior (Figura 5.5). A este código le añadiremos al iniciar la ejecución del programa el grupo de radio por el que se realizarán las comunicaciones, que deberá ser el mismo que en la tarjeta micro:bit receptora.

Se realizarán envíos de radio de tipo clave-valor, de manera que mandaremos un identificador que denotará si lo que enviamos es el contador de residuos o la indicación de que la papeleras debe ser vaciada, junto con el valor.

En este caso se han escogido los identificadores **contador** y **vaciar** ya que se trata de una tarjeta micro:bit emisora genérica, pero en una aplicación real de la solución el identificador debería contener también información sobre qué papelerera está situada la tarjeta micro:bit emisora, es decir: **contadorPlastico**, **contadorPapel**, **contadorOrganico**, **vaciarPlastico**, **vaciarPapel**, **vaciarOrganico**.

De manera continua se realizará un envío del valor del contador con el identificador **contador** a la tarjeta micro:bit receptora, y solo en caso de que el usuario presione alguno de los botones se realizará un envío con el identificador **vaciar** con el valor:

- 1 si el usuario ha presionado el botón A, indicando que la papelerera está llena y se precisa del servicio de recogida.
- 0 si el servicio de recogida ha vaciado la papelerera y requiere que el contador de residuos se vacíe.

El código final de la tarjeta micro:bit emisora será el presentado en la Figura 5.7.

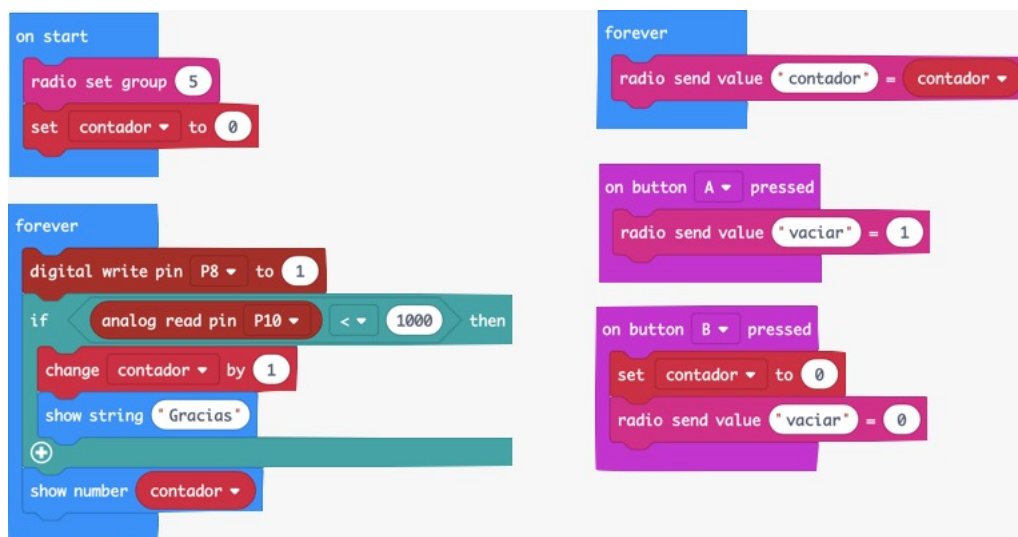


Figura 5.7: Código de la tarjeta micro:bit emisora para la solución final.

Por otro lado tenemos la tarjeta micro:bit receptora, en cuyo caso partiremos del código implementado para la primera solución de este capítulo (Figura 5.1).

La diferencia reside en que añadiremos también al iniciar la ejecución del programa el grupo de radio correspondiente a las tarjetas micro:bit emisoras, además de inicializar dos nuevas variables que se enviarán junto con el estado del sensor PIR: una en la que recibiremos los datos correspondientes al contador y otra en la que recibiremos si requiere del servicio de recogida. En caso de tener tres papeleras necesitaríamos 6 variables, dos correspondientes a cada una de ellas.

El código de la micro:bit receptora se muestra en la Figura 5.8.

```

on start
  radio set group 5
  initialize OLED with height 64 width 128
  set ESP8266 RX P0 TX P1 at baud rate 115200
  set datosRecibidos to 0
  set vaciar to 0

on radio received name value
  if name == 'vaciar' then
    set vaciar to value
  else
    set datosRecibidos to value

serial on data received new line
  show string serial read string

forever
  connect wifi SSID: 'iphone-maria-8266' KEY: 'jai7bcwpbm14'
  connect thingspeak
  set data to be send : Write API Key= 'YXT30H1GWHY2LRGZ'
  field1 = digital read pin P2
  field2 = datosRecibidos
  field3 = vaciar
  field4 = 0
  field5 = 0
  field6 = 0
  field7 = 0
  field8 = 0
  send data to thingspeak

forever
  if digital read pin P2 == 1 then
    show leds
  else
    show leds
  
```

Figura 5.8: Código de la tarjeta micro:bit receptora para la solución final.

5.3.2. Aplicación de la solución y pruebas

Para la aplicación de la solución colocaremos la tarjeta micro:bit receptora en la pared donde se encuentra el conjunto de papeleras que controla. Para su situación lo ideal sería la protección mediante algún tipo de carcasa, sin embargo en la imagen de la Figura 5.9 se ha situado a modo de referencia sobre cómo irían ubicados los elementos.



Figura 5.9: Tarjeta micro:bit receptora con sensor PIR.

Por otro lado, la situación de la emisora sería una por cada una de las papeleras que formen el conjunto, colocando los sensores infrarrojos en la zona superior de la papelera (véase Figura 5.10), idealmente agujereándola de manera que quedaran atravesados en ella y fueran inamovibles.



Figura 5.10: Tarjeta micro:bit emisora con sensores infrarrojos.

Para realizar una trazabilidad (véase Figura 5.11) de lo que sería el curso de nuestra solución, se han tirado varios residuos a la papelera hasta llevar el contador a 9, momento en el que hemos pulsado el botón A para informar a los servicios de recogida, y el botón B una vez ha sido vaciada por los mismos.

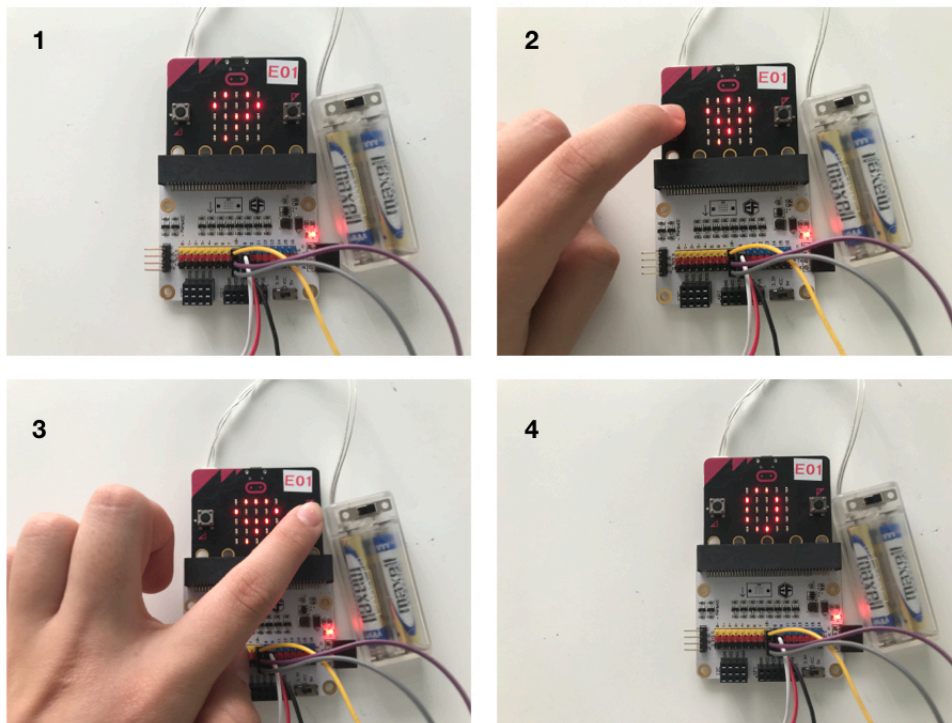


Figura 5.11: Trazo de la solución.

Los datos recogidos en el canal de ThingSpeak asociado a nuestra micro:bit receptora tras la traza especificada se pueden ver en la Figura 5.12.

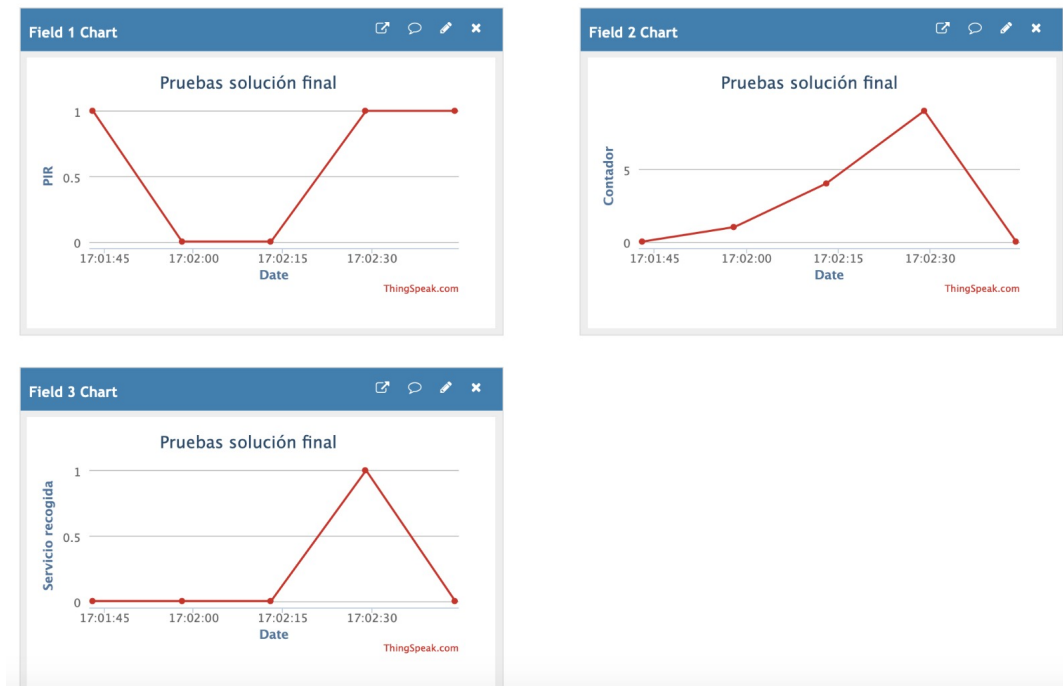


Figura 5.12: Resultados tras la traza de la solución final.

En conclusión, los resultados de aplicar esta solución son la incentivación al reciclaje así como la gestión eficiente de residuos en entornos de interior, localizando las horas y zonas más propicias a la acumulación de residuos y mejorando tanto la ubicación de basuras como la gestión de recogida de las mismas.

5.4 Presupuesto de la solución

En este último apartado se realiza un análisis de los costes de la aplicación de este trabajo en un entorno real. El entorno elegido para este estudio es el edificio 1E de la Escola Tècnica Superior d'Enginyeria Informàtica de la Universitat Politècnica de València.

A continuación, se muestran los planos (Figuras 5.14, 5.15 y 5.16) del edificio mencionado, situando la ubicación y tipo de las papeleras que se hallan en cada planta, representadas a través de la simbología mostrada en la Figura 5.13.

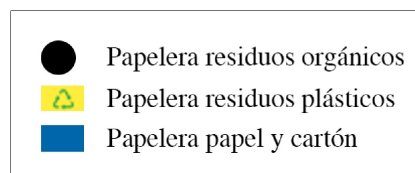


Figura 5.13: Simbología tipos de papeleras

Con tal de facilitar el análisis, se considerarán únicamente las papeleras dedicadas a residuos plásticos, las papeleras dedicadas a papel y cartón y las papeleras de residuos orgánicos (excluyendo así contenedores de grandes dimensiones y las zonas dedicadas a residuos RAEE ¹).

¹Residuos de Aparatos Electrónicos y Eléctricos

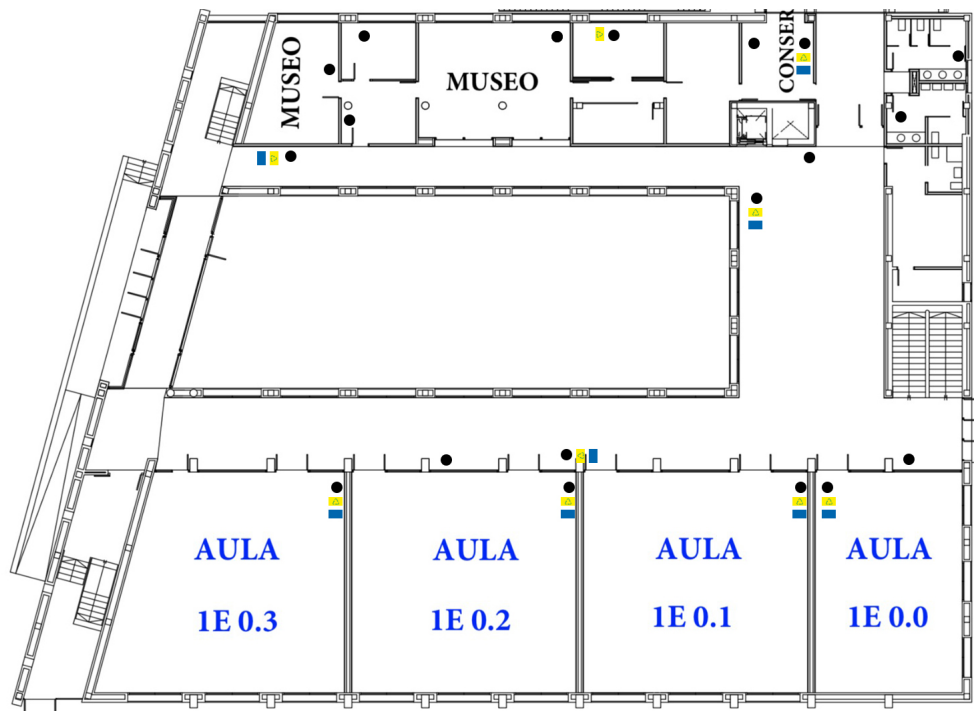


Figura 5.14: Plano de la planta 0 del edificio 1E

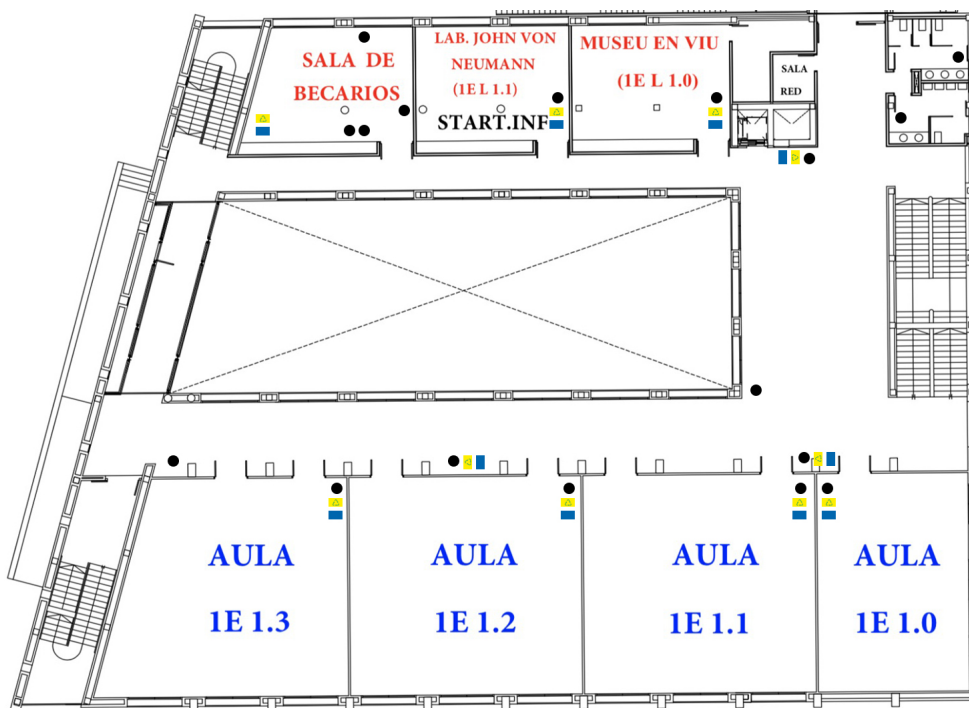


Figura 5.15: Plano de la planta 1 del edificio 1E

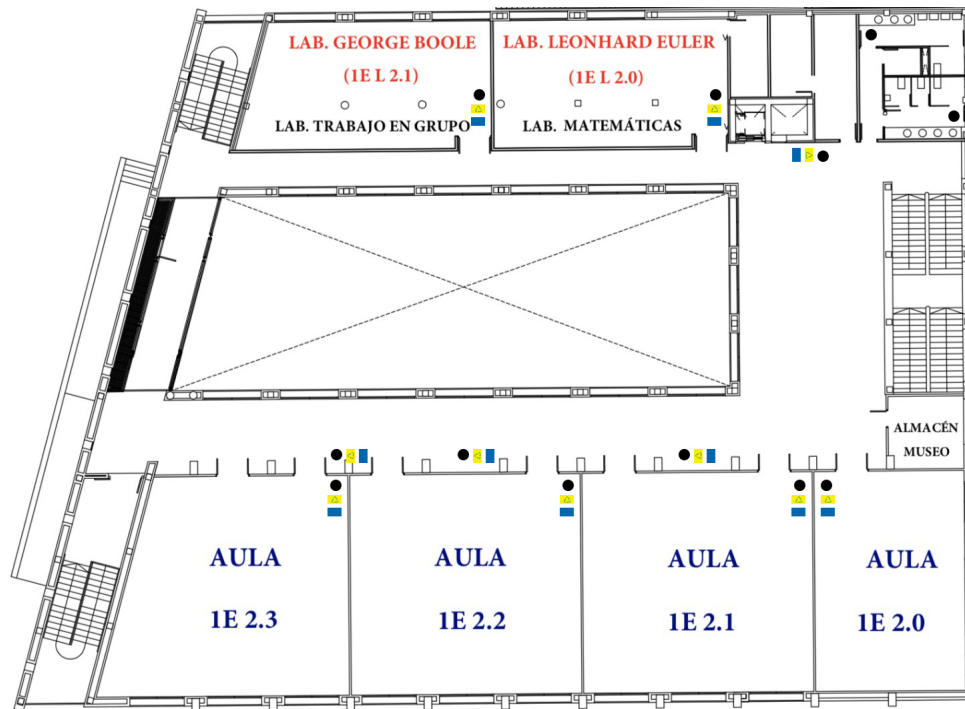


Figura 5.16: Plano de la planta 2 del edificio 1E

De los planos expuestos podemos establecer que existen un total de 105 papeleras distribuidas como se muestra en la Tabla 5.1.

Planta	Papeleras orgánicas	Papeleras plástico	Papeleras papel	Total
Planta 0	19	9	8	36
Planta 1	17	10	10	37
Planta 2	12	10	10	32
Total	48	29	28	105

Tabla 5.1: Recuento de papeleras del edificio 1E

Tal y como se ha especificado previamente en los objetivos de la solución, uno de ellos es el favorecer el uso de conjuntos de papeleras de reciclaje frente al uso de papeleras orgánicas aisladas que no ofrezcan alternativa al reciclaje. Es por ello que para la aplicación de esta solución, únicamente se tendrán en cuenta conjuntos de papeleras (véase Tabla 5.2) formados por el combo de tres, esto es, papel, plástico y orgánica.

Planta	Conjuntos de papeleras
Planta 0	8
Planta 1	9
Planta 2	10
Total	27

Tabla 5.2: Recuento de conjuntos de papeleras de reciclaje del edificio 1E

Con tal de aplicar la solución final necesitaríamos 4 tarjetas micro:bit por conjunto, lo que nos da un total de 108 tarjetas micro:bit, de las cuales 27 harían de receptoras y 81 de emisoras. Un pack de micro:bit + Octopus:bit cuesta alrededor de 35 €, por lo que partiríamos de un presupuesto de 3.780 €.

A esto habría que añadirle el precio de componentes externos, tal y como presenta la Tabla 5.3.

Producto	Cantidad	Precio (€)
Pilas AAA	162 (2 por cada micro:bit emisora)	40
Par de sensores infrarrojos	81 (un par para cada micro:bit emisora)	810
Sensor PIR	27 (uno para cada micro:bit receptora)	135
Módulo ESP8266 junto con adaptador	27 (uno para cada micro:bit receptora)	405
Total		1.390

Tabla 5.3: Presupuesto de componentes externos a la tarjeta micro:bit.

Con todo ello obtendríamos un presupuesto total de 5.170 € para la compra del material necesario requerido para aplicar la solución en todos los conjuntos de papeleras de reciclaje del edificio 1E. Cabe destacar que los conjuntos deberán estar cerca de una toma de corriente para conectar la tarjeta micro:bit receptora.

No se han tenido en cuenta gastos de instalación y material para la misma (como bien pueden ser carcasas protectoras para las tarjetas micro:bit, o adaptadores de corriente para las tarjetas micro:bit receptoras) ni de reparación. Tampoco se han considerado los gastos de la creación de una cuenta de ThingSpeak sin limitaciones por el exceso de canales requeridos para el envío de información.

Por último, dentro del presupuesto tampoco se han considerado los gastos de consumo de *Wi-Fi* ni electricidad, teniendo en cuenta que no supondrían un coste adicional notable ya que existen ambos servicios en el edificio.

CAPÍTULO 6

Conclusiones

6.1 Cumplimiento de objetivos

En primer lugar, se considera que el trabajo ha cumplido con los objetivos establecidos, favoreciendo el cumplimiento de los Objetivos de Desarrollo Sostenibles 11 y 12 (detallados en la motivación del trabajo) mediante el aporte de soluciones funcionales para la gestión eficiente de residuos:

- de fácil programación (con un entorno de programación por bloques) y por tanto aplicables a los talleres impartidos por la Cátedra de Tecnologías Cívicas y Empoderamiento,
- que promueven el reciclaje, contribuyendo al desarrollo de tecnologías sostenibles,
- que recolectan información enviando a ThingSpeak los datos recogidos por los sensores de presencia e infrarrojos, detectando tanto el llenado de basuras como las zonas frecuentadas y contribuyendo a un a mejor gestión de los residuos en el entorno en que sean aplicadas.

6.2 Dificultades encontradas

Las dificultades encontradas han sido especialmente en encontrar el equilibrio entre soluciones sencillas a nivel de programación pero que sean aplicables a un entorno real, mejorando en este caso la gestión de residuos de la Escola Tècnica Superior d'Enginyeria Informàtica de la Universitat Politècnica de València.

Otras dificultades adicionales de menor relevancia han sido localizar los componentes externos adecuados para la medición de residuos, tratando siempre de economizar en la medida de lo posible las soluciones para que cualquier persona pueda tener acceso a ellas.

6.3 Trabajo a futuro

Cabe destacar que las soluciones que se presentan son prototipos iniciales, tratándose de un punto de partida para la realización de trabajo a futuro en materia de diseño y pruebas así como un estudio con mayor detalle tanto de su aplicabilidad como del coste que ella implica para convertirse en una solución de uso en la vida real.

Además, este trabajo no solo aporta las soluciones que lo componen, sino que abre una nueva vía de investigación en lo que respecta a la gestión inteligente de residuos, proponiendo el cambio de las soluciones desarrolladas hasta día de hoy destinadas a ciudades por desarrollos enfocados a la mejora de entornos de interior, como bien son hospitales y centros educativos.

En conclusión, a pesar de que queda mucho trabajo a futuro por desempeñar, la realización de este trabajo ha aportado un gran aprendizaje más que a nivel de programación a nivel humano, a través de la concienciación respecto a la realización de desarrollos comprometidos con el medio ambiente y con los Objetivos del Desarrollo Sostenible, y de la importancia de divulgar el conocimiento, poniendo en manos de la ciudadanía los recursos necesarios para hacer del mundo un lugar más sostenible.

Bibliografía

- [1] API del módulo micro:bit para MicroPython. Disponible en https://microbit-micropython.readthedocs.io/en/latest/microbit_micropython_api.html, consultado por última vez el 23 de junio de 2019.
- [2] Cátedra Govern Obert, dirigida por Diego Álvarez. *Datos generados por la ciudadanía desde el contexto valenciano*. © Ayuntamiento de Valencia, primera edición, 2018.
- [3] Centro de innovación The Circular Lab. Disponible en <https://www.thecircularlab.com>, consultado por última vez el 5 de junio de 2019.
- [4] Ernesto Martínez de Carvajal Hedrich. *Robótica Educativa. 50 proyectos con micro:bit*. © Ernesto Martínez de Carvajal Hedrich, primera edición, enero, 2008.
- [5] Gareth Halfacree. *The Official BBC micro:bit User Guide*. © Wiley, primera edición, 6 de octubre de 2017.
- [6] IBM Sales and Distribution. IBM Intelligent Waste Management Platform. *Using the power of analytics to profit from the move to a circular economy.*, diciembre, 2015. Disponible en <https://www.ibm.com/downloads/cas/41Q1DY06>
- [7] José F. Muñoz. *Manual de Programación micro:bit & Octopus:bit*. Microes.org (Comunidad micro:bit en España), primera edición, noviembre, 2018.
- [8] MicroFS. Disponible en <https://microfs.readthedocs.io/en/latest/>, consultado por última vez el 23 de junio de 2019.
- [9] Multiple authors. BBC micro:bit MicroPython. Documentation. *Release 1.0.1.*, 13 de diciembre, 2018. Disponible en <https://buildmedia.readthedocs.org/media/pdf/microbit-micropython/latest/microbit-micropython.pdf>.
- [10] Proyecto Smart Waste de Ecoembes. Disponible en <https://economiecircularverde.com/smart-waste/>, consultado por última vez el 5 de junio de 2019.
- [11] Simon Monk. *Programming the BBC micro:bit. Getting Started with MicroPython*. © McGraw-Hill Education TAB, primera edición, 6 de enero de 2018.

