

# Introducing programming concepts with mobiles and MIT App Inventor

Sergio Barrachina Mir and Germán Fabregat Lluca

**Abstract**—At the beginning of each course, we ask our students of the bachelor in Computer Engineering if they have ever developed a computer program. Surprisingly, the most frequent answer is negative. Those few students that have attended a Computer Science training module usually have some basic programming notions. However, most of our students arriving from high schools (freshmen) have never programmed. This lack of basic programming skills is a major drawback for those courses related with this discipline. This is especially the case for the course on Computer Organization, taught during the first semester of the first University year, as one of the main objectives for this course is to describe the processor architecture, and, therefore, a great part of it evolves around programming in assembly language.

To tackle this lack of basic computer programming competences, a workshop on mobile programming using MIT App Inventor is offered to freshmen. This workshop shows a high acceptance, is warmly appreciated by the students, and we believe that it has contributed to improve their performance in the courses related to programming, and in particular, on the Computer Organization course.

**Index Terms**—Programming basic concepts, mobile programming, MIT App Inventor

## I. INTRODUCTION

**L**A asignatura *Estructura de computadores* de primer curso, primer semestre, se imparte en los grados en Ingeniería Informática y Matemática Computacional de la Universitat Jaume I. Esta asignatura no exige conocimientos previos de programación. De hecho, teniendo en cuenta esta situación, en cursos previos se replanteó la asignatura y se desarrollaron dos recursos docentes [1][2] orientados a facilitar y motivar su estudio. Aunque estos cambios han mejorado las tasas de éxito y de rendimiento de la asignatura, los estudiantes sin unas competencias básicas de programación aún tienen que realizar un gran esfuerzo. Esto es debido a que en esta asignatura se detalla el funcionamiento a bajo nivel del computador, es decir, se muestra cómo un computador es capaz de ejecutar programas en código máquina. Por tanto, el estudiante sin unos conocimientos básicos de programación "tipos de datos, estructuras de datos, estructuras de control, codificación, etc." no podrá apoyarse en esta base para asimilar los mecanismos hardware que los hacen posibles.

Por otro lado, y pese a las declaraciones realizadas desde AENUI-CODDII [3] en favor de la inclusión de asignaturas específicas de ciencia y tecnología informática en los estudios básicos de la enseñanza secundaria y bachillerato, las asignaturas de informática siguen siendo actualmente optativas en estas enseñanzas y, además, solo una parte de estas asignaturas optan por enseñar programación. Esto conlleva que en la práctica, muchos de los estudiantes recién ingresados en el

grado en Ingeniería Informática, pese a haber optado por esta titulación, no posean competencias básicas de programación.

Así pues, y con el objeto de paliar esta situación, se ha ofertado un taller intensivo a los estudiantes de primero, justo al comienzo del semestre, con el objetivo de que puedan desarrollar una serie de competencias básicas de programación de una forma práctica e intuitiva, orientadas especialmente a mejorar el proceso de enseñanza/aprendizaje en la asignatura *Estructura de computadores*.

En la ponencia «¿Puedo programar mi móvil? Pero si acabo de llegar», publicada en las XXV Jornadas sobre Enseñanza Universitaria de la Informática (JENUI 2019), se presentó este taller y los resultados obtenidos [4]. Esta ponencia fue elegida como una de las dos mejores de JENUI 2019 y propuesta para su publicación en IEEE-RITA. Este artículo extiende esa ponencia detallando cómo se ha diseñado el experimento, describiendo con más detalle los proyectos elaborados, e indicando en qué aspectos contribuye cada proyecto a la consecución de los objetivos formativos del taller.

## II. OBJETIVO DOCENTE

El objetivo docente del taller propuesto es el de que los estudiantes de primer curso alcancen un conjunto de competencias básicas en programación que les permita mejorar el aprendizaje de las asignaturas relacionadas con esta materia, especialmente el de la asignatura *Estructura de computadores*. Como ya se ha comentado, esta asignatura, que se imparte en el primer semestre, introduce los conceptos de arquitectura de computadores relacionados con cómo el procesador es capaz de ejecutar programas, por lo que cuantas más nociones de programación se tenga, más fácil debería ser alcanzar muchos de los objetivos formativos de esta asignatura.

## III. DISEÑO DEL EXPERIMENTO

Para poder comprobar fehacientemente que los resultados obtenidos tras la realización del taller están relacionados con el hecho de haberlo realizado y que estos no sean debidos en realidad a otras variables, el grupo de estudiantes que realiza el taller debería haberse seleccionado de forma aleatoria entre todos los estudiantes de primer curso.

No obstante, puesto que considerábamos que el taller serviría a los estudiantes para mejorar su proceso de aprendizaje en aquellas asignaturas más relacionadas con la programación y que sería especialmente beneficioso para aquellos con pocos o ningún conocimiento sobre programación, se optó por ofertar el taller a todos los estudiantes, indicándoles cuáles eran los objetivos del taller y que era especialmente interesante que lo hicieran si tenían pocos o ningún conocimiento de programación. De esta forma, les dábamos a todos la oportunidad de realizarlo. Con este mismo objetivo, el taller se ofertó de forma gratuita y en un horario en el que todos los

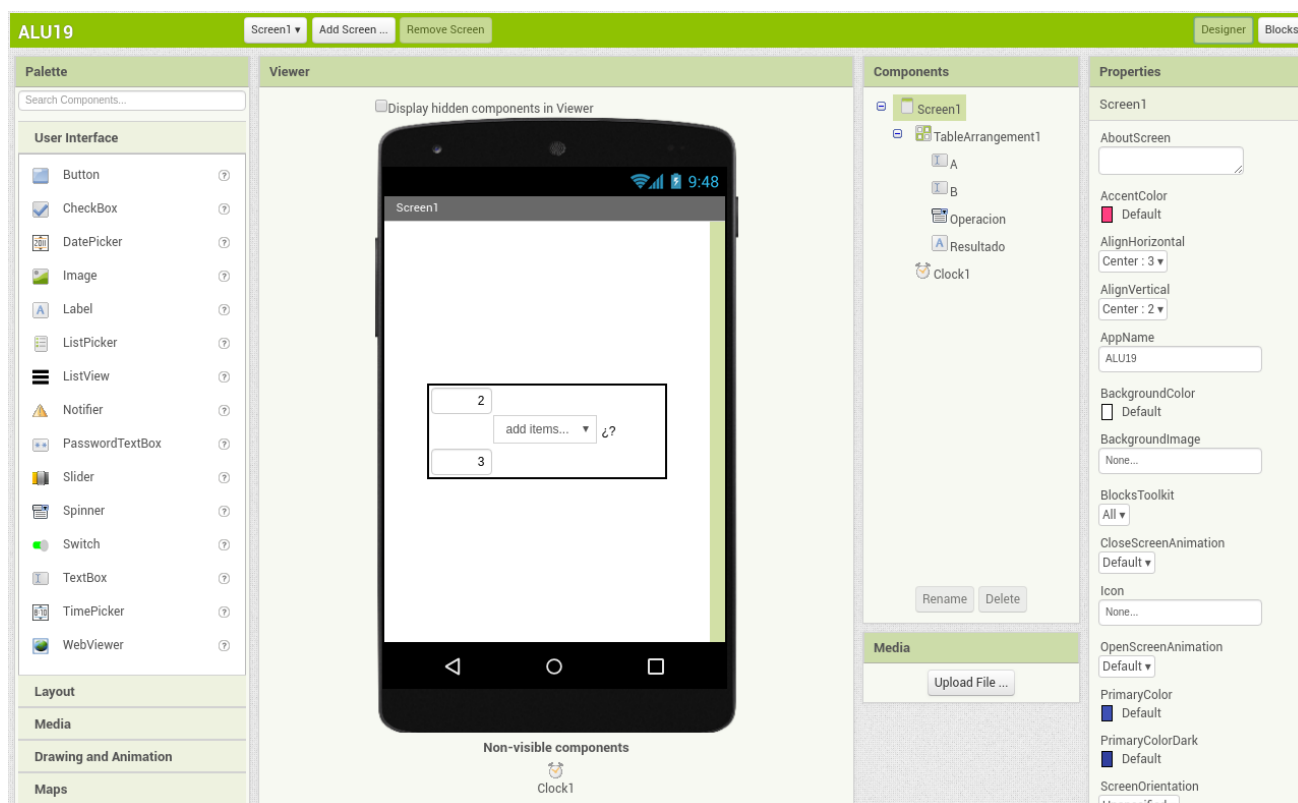


Figure 1: Editor de la interfaz gráfica de MIT App Inventor

estudiantes pudieran acudir independientemente del grupo en el que estuvieran matriculados.

Debido a que el proceso de admisión en el taller no ha sido aleatorio, conviene tener en cuenta que al comparar los resultados obtenidos por los estudiantes que han realizado el taller con los de los que no, las posibles diferencias podrían ser debidas en realidad a otras variables que no se han tenido en consideración en este estudio.

Por otro lado, los estudiantes fueron informados de que sus resultados académicos se iban a utilizar de forma agregada en esta investigación. Sin embargo, para evitar que este conocimiento pudiera condicionar sus resultados, se les informó de esto una vez pasada la evaluación de las distintas asignaturas.

#### IV. SELECCIÓN DEL ENTORNO DE PROGRAMACIÓN

Puesto que el objetivo del taller es que los estudiantes adquieran competencias básicas de programación al principio del curso de una forma práctica e intuitiva, y en unas pocas sesiones, es sumamente importante seleccionar el entorno de programación más adecuado a estos objetivos.

Para introducir rápidamente y de una forma gráfica los conceptos básicos de programación, la primera decisión que se tomó fue la de recurrir a un entorno visual de programación. De entre los entornos disponibles, evaluamos utilizar Scratch o MIT App Inventor.

Scratch [5] es un entorno de programación visual bastante conocido que permite crear aplicaciones arrastrando bloques en lugar de escribiendo código. Fue desarrollado para ayudar a la gente joven, principalmente con entre 8 y 16 años, a aprender a programar. Además, ha sido utilizado con éxito tanto en etapas previas, como en cursos introductorios a la programación en la universidad [6], [7], [8]. Otra característica

que hace interesante a Scratch es que su entorno de programación puede extenderse fácilmente. Existen, entre otras, extensiones que permiten utilizar Scratch para programar distintos dispositivos hardware: robots *Lego Mindstorms NXT* [9] o el hardware *ad-hoc* del proyecto educativo SUCRE4Kids [10].

MIT App Inventor<sup>1</sup>, por su parte, es un entorno visual de programación que presenta una interfaz muy similar a la de Scratch, pero que está orientado al desarrollo intuitivo de aplicaciones totalmente funcionales para móviles y tabletas. MIT App Inventor permite definir de forma visual tanto los elementos de la interfaz gráfica de la aplicación (véase la Figura 1), como su código (véase la Figura 2). Por lo tanto, permite desarrollar programas con la misma facilidad que Scratch, pero además, los estudiantes obtienen un producto tangible: una aplicación móvil<sup>2</sup> que pueden mostrar e instalar en los móviles de sus amigos y familiares, e incluso publicar en la tienda en línea Google Play. Esta orientación a la programación móvil, al desarrollo de un producto real que pueden enseñar, supone una motivación importante, y habitualmente ha inclinado la balanza en favor de MIT App Inventor frente a Scratch, como un entorno de introducción a la programación para estudiantes de grado [11][12][13].

Para el caso que nos ocupa, el de desarrollar competencias básicas de programación útiles para la asignatura *Estructura de computadores*, MIT App Inventor posee una ventaja añadida: permite interactuar con los sensores del móvil (cámara, temporizador, GPS...), por lo que también puede utilizarse para introducir de una forma fácil y aplicada los conceptos

<sup>1</sup><http://appinventor.mit.edu/explore/about-us.html>

<sup>2</sup>Aunque actualmente MIT App Inventor permite realizar aplicaciones solo para Android, está en desarrollo una versión para iOS. En cualquier caso, también existen entornos basados en MIT App Inventor que permiten publicar aplicaciones también para iOS, por ejemplo, Thunkable (<https://thunkable.com/>).

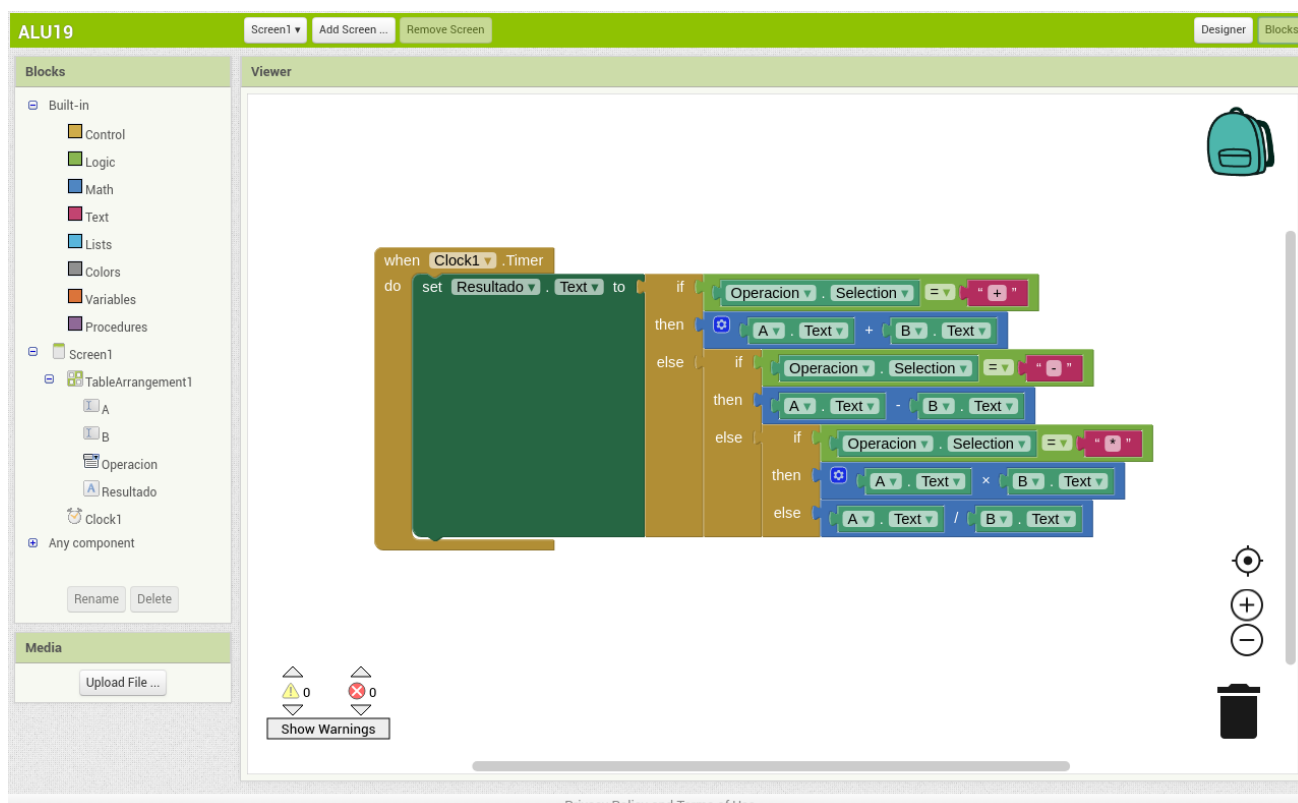


Figure 2: Editor de código de MIT App Inventor

de entrada/salida que posteriormente se desarrollarán en la asignatura.

Por todo lo anterior, se decidió usar la plataforma MIT App Inventor para realización de este taller.

#### V. PROGRAMACIÓN MÓVIL CON MIT APP INVENTOR

El taller Programación móvil con MIT App Inventor está disponible en la siguiente dirección web para cualquier profesor que esté interesado en ofertarlo o que quiera recomendarlo a sus estudiantes como herramienta de auto-aprendizaje: <http://lorca.act.uji.es/curso/mit-app-inventor/>.

Está orientado a estudiantes recién ingresados en el grado en Ingeniería Informática, especialmente a aquellos que no hayan recibido formación previa en programación y, una vez completado, el estudiante debería ser capaz de:

- OF1 Diferenciar entre valores y variables.
- OF2 Reconocer y saber para qué se pueden utilizar diferentes estructuras condicionales.
- OF3 Reconocer y saber para qué se pueden utilizar diferentes estructuras iterativas.
- OF4 Reconocer algunos de los objetos proporcionados por MIT App Inventor.
- OF5 Diferenciar entre propiedades y funciones de un objeto.
- OF6 Encapsular partes del código utilizando funciones.
- OF7 Interactuar con la entrada/salida de un dispositivo móvil (temporizador y cámara).
- OF8 Diseñar interfaces de usuario simples.
- OF9 Reconocer distintos tipos de eventos (click al pulsar un botón, alarmas, inicialización de la pantalla y respuesta de la cámara) y vincularlos con una parte del código.
- OF10 Desarrollar aplicaciones sencillas para móviles utilizando MIT App Inventor.

Puesto que la duración del taller debe ser corta para que pueda impartirse completamente al principio del curso sin

sobrecargar en exceso a los estudiantes, se debe tener en cuenta que los anteriores objetivos se alcanzarán a un nivel muy básico. En base a esto, se recomienda que la duración del taller sea de 10 horas repartidas, por ejemplo, en 4 sesiones de 2 horas y media cada una, a razón de una sesión por semana.

Siguiendo las recomendaciones para cursos introductorios de MIT App Inventor<sup>3</sup>, en la primera sesión se presenta el entorno de programación, se accede a la aplicación, se configuran correctamente los móviles de los estudiantes y se realizan, de una forma guiada, los cuatro tutoriales básicos proporcionados por MIT App Inventor que desarrollan de forma gradual: i) una aplicación que habla, ii) una extensión de la anterior, que habla cuando se agita el móvil, iii) una aplicación que permite mover una bola en pantalla utilizando el dedo, y iv) una aplicación de dibujo que permite tomar fotos.

Para las siguientes sesiones, en lugar de recurrir a algunos de los tutoriales del repositorio de MIT App Inventor<sup>4</sup>, se optó por desarrollar proyectos propios que persiguieran los objetivos formativos del taller e introdujeran, además de nociones básicas de programación, conceptos relacionados con la arquitectura de computadores.

Cada uno de los proyectos propone el desarrollo de forma incremental, mediante una serie de pasos, de una determinada aplicación. Los proyectos se han estructurado de esta forma para que el estudiante pueda tener cuanto antes una versión funcional, aunque se trate de una versión muy básica e incompleta de la aplicación final. Con cada paso, el estudiante irá ampliando el proyecto, pero, de nuevo, la idea es que tenga cuanto antes una nueva versión funcional en la que pueda comprobar las ampliaciones que ha realizado. De esta forma,

<sup>3</sup><http://appinventor.mit.edu/explore/teach.html>

<sup>4</sup><http://explore.appinventor.mit.edu/ai2/tutorials>

TABLE I  
RELACIÓN ENTRE LOS PROYECTOS Y LOS OBJETIVOS  
FORMATIVOS DEL TALLER

	1	2	3	4	5	6	7	8	9	10
Simulador ALU	•	•		•			•	•	•	
Banco registros	•	•	•	•	•	•	•	•	•	
RGB foto	•		•	•		•	•	•		•
Memorización	•	•	•	•	•	•	•	•	•	•
El tiempo	•	•	•	•	•	•	•	•	•	•

se pueden introducir paulatinamente los distintos componentes y bloques de código necesarios para desarrollar la aplicación.

Por otra parte, y para poder atender a la diversidad, se han tomado dos tipos de medidas. Por un lado, al final de cada uno de los pasos se proporciona un enlace que permite descargar una versión correcta del proyecto tras haber realizado dicho paso. De esta forma, si un estudiante no consiguiera realizar por su cuenta un determinado paso, siempre puede proceder con el siguiente partiendo de una versión correcta. Por otro lado, al final de muchos de los pasos se proponen posibles extensiones relacionadas con la ampliación que acaba de realizarse. De esta forma, si un estudiante va muy por delante del resto de compañeros, siempre puede explorar por su cuenta las extensiones propuestas.

Los proyectos se han diseñado para abordar de forma incremental los objetivos formativos del taller que son importantes para la asignatura *Estructura de computadores* (OF1, OF2, OF3, OF6, OF7 y OF9) y para introducir conceptos relacionados con esta asignatura: partes del computador y acceso a la información (*Simulador de una ALU y de un banco de registros*), codificación y entrada/salida (*RGB foto*), estructuras de datos complejas (*Juego de memorización*) e interrupciones (*El tiempo*). La relación entre los distintos proyectos y los objetivos formativos del taller se muestra en el Cuadro I.

#### A. Simulador de una ALU

El primero de los proyectos propuestos consiste en el desarrollo de una aplicación móvil que simula una Unidad Aritmética Lógica (ALU) (véase la Figura 3) capaz de realizar una operación aritmética "+—" que puede seleccionarse de entre cuatro: suma, resta, multiplicación o división"— sobre dos operandos numéricos y de mostrar el resultado obtenido. El desarrollo de este proyecto se estructura en los siguientes pasos: i) un sumador, que dadas dos entradas actualiza la salida cuando se pulsa un botón; ii) una ALU que permite seleccionar la operación a realizar y que actualiza el resultado cuando se pulsa un botón; y finalmente, iii) una ALU que actualiza periódicamente su resultado en función del valor de sus entradas y de la operación seleccionada.

En este proyecto se muestra por primera vez el uso de elementos de disposición gráfica, de etiquetas, de listas desplegables y del temporizador del móvil.

En cuanto a la parte de programación, se muestra por primera vez cómo modificar propiedades de objetos, consultar el elemento seleccionado de una lista desplegable, el uso de estructuras de control condicionales y cómo responder ante un evento del temporizador.

#### B. Simulador de un banco de registros

En este segundo proyecto se propone el desarrollo de un simulador gráfico de un banco de 8 registros (véase la

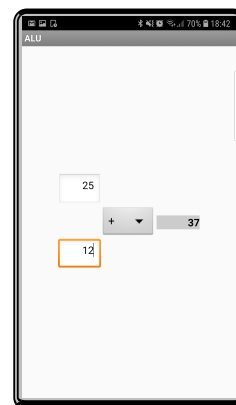


Figure 3: Simulador de una ALU

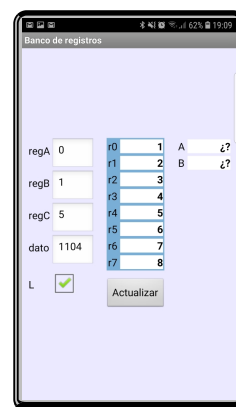


Figure 4: Simulador de un banco de registros

Figura 4), que permite la lectura simultánea de dos de sus registros y la escritura de un valor numérico en uno de ellos. Al final de este proyecto se propone como ampliación añadir el simulador de ALU anterior para operar con el contenido de los registros.

El primer paso de este proyecto consiste en la creación de la interfaz gráfica que muestra el contenido de los registros, el valor de los cuales viene dado por los elementos una lista. El segundo paso consiste en inicializar mediante código la anterior lista. El tercero implementa la funcionalidad de lectura. El cuarto, la de escritura. El último paso consiste en realizar la operación de lectura o escritura en función de si la correspondiente casilla está marcada o no.

Este proyecto, desde el punto de vista de la interfaz gráfica, introduce el uso de casillas y un uso más complejo de los elementos de disposición gráfica, incluyendo la personalización de los colores de fondo.

Desde el punto de vista de la programación, se introduce la estructura de datos tipo lista y sus operaciones básicas "+— declaración, inicialización, añadir elementos y leer y escribir elementos"—; las variables globales; las estructuras de control repetitivas y los procedimientos.

#### C. RGB foto

El tercer proyecto consiste en el desarrollo de una aplicación móvil que permite seleccionar un color mediante sus componentes RGBA, tomar una foto y variar su color al color previamente seleccionado (véase la Figura 5). El valor de cada una de las componentes del color puede variarse utilizando un control deslizante y se muestra como un número hexadecimal.

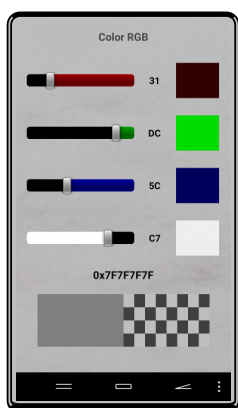


Figure 5: RGB foto

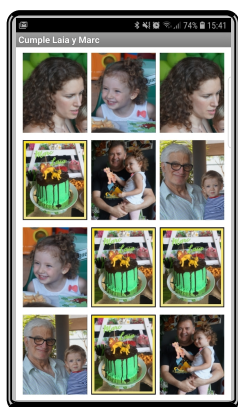


Figure 6: Juego de memorización

El color seleccionado se muestra de tres formas distintas: como un número hexadecimal; variando el color de un cuadro sin tener en cuenta la transparencia; y variando el color de un cuadro en el que se aplica el nivel de transparencia indicado, lo que permite mostrar en mayor o menor medida un patrón tipo tablero de ajedrez que se encuentra debajo de este.

Este proyecto consta de cuatro pasos: i) definir la interfaz gráfica; ii) seleccionar el valor de las componentes del color por medio de controles deslizantes; iii) componer el color a partir de sus componentes; y iv) obtener una foto y variar su color al color seleccionado.

En este proyecto se muestra por primera vez el uso de controles deslizantes y el acceso a la cámara del móvil para la captura de fotos.

Desde el punto de vista de la programación, se ahonda en el uso de procedimientos, de estructuras de control y se introducen aspectos de representación "+representación de números en hexadecimal"+ y de codificación de la información "+conversión de reales (posición del controlador deslizante) a enteros y composición del color (4 bytes) a partir de sus componentes (1 byte por componente)"+.

#### D. Juego de memorización

Consiste en el desarrollo de un juego de memorización en el que hay que localizar 6 parejas de cartas pudiendo levantar como máximo dos cada vez (véase la Figura 6).

Este proyecto está dividido en los siguientes pasos. El primero consiste en crear la interfaz gráfica del juego, cargando además las imágenes correspondientes a los anversos y reversos de las cartas y añadiendo el código necesario

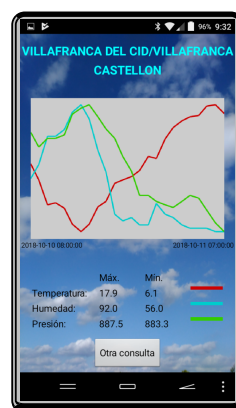


Figure 7: El tiempo

para voltear hasta tres de las cartas "+que en realidad se implementan por medio de botones"+. En el segundo paso se muestra cómo añadir sonidos al juego y cómo hacer que cuando se voltee una carta se reproduzca un sonido de volteo. El tercer paso muestra cómo utilizar un procedimiento que permita voltear una de las cartas en base al valor de un parámetro "+en lugar de tener que copiar y pegar el mismo código para cada una de las cartas"+. El cuarto paso ilustra cómo una estructura de datos "+una lista de listas"+ puede recoger la información requerida para la lógica del juego y simplificar la llamada al anterior procedimiento. El quinto y último paso especifica el algoritmo que debe implementarse para que el juego sea completo "+hasta este momento se podían voltear cartas, pero no se aplicaban las reglas del juego de memorización"+. En el último paso se incluye también la reproducción de sonidos ligados al emparejamiento de dos cartas, al fallo al emparejar dos cartas y a la finalización del juego "+cuando se han emparejado todas las cartas"+. Al final del proyecto se incluye un paso opcional adicional en el que se especifican posibles ampliaciones del juego.

Desde el punto de vista de la interfaz, el proyecto no incorpora nuevos elementos, aunque sí muestra cómo es posible utilizar botones para simular otro tipo de elementos, como en este caso las cartas del juego, sin más que cambiar su imagen de fondo.

Desde el punto de vista de programación, se introduce: i) una estructura de datos más compleja "+una lista de listas"+ y cómo acceder a sus distintos elementos; ii) la utilización de variables locales a un procedimiento; y, por último, iii) cómo implementar un algoritmo más complejo que los vistos hasta este momento.

#### E. El tiempo

En este proyecto se propone el desarrollo de una aplicación que obtiene y visualiza datos meteorológicos de la Agencia Estatal de Meteorología (AEMET) (véase la Figura 7). Este proyecto es el más complejo de todos y muestra cómo obtener información de un servicio web y cómo trabajar con respuestas asíncronas. Además, frente a los proyectos anteriores que solo contaban con una única pantalla, la aplicación que se desarrolla en este proyecto permite al usuario navegar por una serie de pantallas, donde cada una solicita o muestra una determinada información.

El proyecto está estructurado en los siguientes pasos. En el primero se describe la API de OpenData de la AEMET, el formato JSON y la extensión de MIT App Inventor que



permite trabajar con datos tipo JSON. En el segundo paso se describe el componente web y cómo se puede utilizar para realizar peticiones web y para reaccionar cuando se reciba la respuesta a una determinada petición. En el tercer paso se muestra cómo utilizar un lienzo para representar gráficamente los datos obtenidos. En el cuarto paso se muestra cómo es posible transferir información entre las distintas pantallas de la aplicación. En el último paso se introducen los mecanismos de gestión de errores que permiten tratar de la forma más adecuada posible tanto los errores previstos como los imprevistos.

Desde el punto de vista de la interfaz, en este proyecto se ve por primera vez cómo crear una aplicación formada por varias pantallas y la utilización de lienzo para la representación de gráficas.

En cuanto a la parte de la programación, se ve cómo transferir información entre distintas pantallas de una aplicación, cómo utilizar componentes para obtener información web de forma asíncrona y cómo gestionar los posibles errores que se pueden producir durante la ejecución de una aplicación.

## VI. RESULTADOS

El taller se ofertó a los 164 estudiantes matriculados en la asignatura *Estructura de computadores*, especificando que estaba especialmente indicado para aquellos estudiantes que no tuvieran conocimientos de programación. Se matricularon 46 estudiantes y lo completaron 42 (el 91%). No obstante, hay que tener en cuenta que el haber sido ellos quienes eligieron realizar el taller puede haber influido en su motivación por completarlo, y que eso explique el alto porcentaje de finalización.

La actitud de los estudiantes en el taller fue extremadamente buena, participando activamente y completando los ejercicios propuestos. Además, pese a que los proyectos incluían puntos de reenganche tras cada uno de los pasos, no recurrieron a ellos, salvo uno de los grupos en una ocasión. Aún sabiendo que disponían de esa ayuda, preferían alcanzar la solución por su cuenta. Conviene tener en cuenta que la alta motivación y la predisposición para aprender que demostraron en el taller también puede haber influido en sus resultados académicos.

Los estudiantes valoraron el taller por medio de un formulario estándar de satisfacción<sup>5</sup> en el que se les preguntaba sobre el nivel de esfuerzo requerido, los conocimientos adquiridos, las habilidades y dedicación del profesor, el contenido del taller, qué aspectos consideraban útiles y sugerencias de mejora. Los cuatro primeros aspectos se valoraron por medio de preguntas que los estudiantes contestaban utilizando una escala tipo Likert de cinco puntos y los dos últimos, mediante preguntas de respuesta abierta. En los cuatro primeros aspectos se obtuvieron respuestas mayoritariamente positivas. A modo de muestra, se detallan a continuación las preguntas correspondientes al contenido del taller:

P1: Los objetivos del taller estaban claros.

P2: El contenido del taller estaba bien organizado y planificado.

P3: La carga de trabajo del taller fue la adecuada.

P4: Los alumnos pudieron participar activamente en el taller.

Como se puede observar en la Figura 8, la mayor parte de los estudiantes estaba de acuerdo o totalmente de acuerdo con cada una de estas preguntas.

<sup>5</sup>Como formulario de satisfacción se utilizó la plantilla «Valoración del curso» de la aplicación de formularios de Google.

En cuanto a la primera de las preguntas de respuesta abierta (¿qué aspectos de este taller te resultaron más útiles?), los estudiantes han incidido en: que esté orientado a programar para móviles, el haber aprendido a diseñar y a programar, el haber visto cómo relacionar conceptos mientras piensan cómo construir un programa, y el haber adquirido conocimientos básicos de programación.

De la segunda pregunta (¿cómo mejorarías este taller?) conviene destacar las siguientes respuestas: más horas para poder realizar más prácticas y para tocar más temas, y desarrollar más juegos.

Por otro lado, antes de comenzar el taller se pasó un cuestionario con 15 preguntas tipo test muy sencillas sobre conceptos básicos de programación. Este cuestionario se volvió a pasar una vez completado el taller. Como se puede observar en la Figura 9, las notas obtenidas después del taller (mediana: 8.0) fueron mejores que las obtenidas al principio (mediana: 6.67).

Para comprobar si la variación positiva entre las medias de los resultados obtenidos antes y después del taller se puede considerar significativa, se ha realizado la prueba T de Student para muestras relacionadas<sup>6</sup> sobre estas calificaciones, obteniendo un t-estadístico de 9.26 y un p-valor de  $0.84 \cdot 10^{-11}$ , por lo que se puede concluir que es muy probable que el incremento en las calificaciones que se ha constatado sea significativa.

También se han comparado los resultados académicos obtenidos por los estudiantes en las cinco asignaturas de primer curso, primer semestre, en función de si han aprovechado el taller o no. Para ello, se ha partido de los listados de calificaciones de estas asignaturas y de un listado adicional que combina los resultados de las cuatro asignaturas que no son *Inglés*. En este último listado, formado únicamente por los estudiantes que se han presentado a estas cuatro asignaturas, se ha calculado la nota de cada estudiante como la media de sus notas en esas asignaturas.

Para realizar este estudio se han dividido los estudiantes en dos grupos: los estudiantes que han aprovechado el taller y el resto (los que no lo han aprovechado o no lo han hecho). Se ha considerado que un estudiante ha aprovechado el taller si en el cuestionario básico de programación ha obtenido una nota igual o superior a la mediana de las notas de ese cuestionario (un 8). De los 41 estudiantes que han completado el taller, 30 cumplen esta condición (el 73%).

El Cuadro II muestra para cada asignatura y distinguiendo entre los estudiantes que han aprovechado el taller y el resto: el número de estudiantes, la mediana de las notas, el porcentaje de aprobados sobre presentados y los resultados del test de normalidad de Shapiro-Wilk<sup>7</sup>.

Como se puede observar, el número de estudiantes varía ligeramente en cada asignatura en función de cuántos se han presentado. Es interesante destacar que aunque se han presentado alrededor de 120 estudiantes a cada una de las asignaturas, solo 83 estudiantes se han presentado a las cuatro que no son *Inglés*.

En el Cuadro II también se puede constatar que las medianas de las calificaciones obtenidas por los estudiantes que han

<sup>6</sup>Para realizar la prueba T de Student para muestras relacionadas se ha utilizado la función `ttest_rel` del módulo `stats` de la biblioteca SciPy de Python.

<sup>7</sup>Para realizar la prueba de normalidad de Shapiro-Wilk se ha utilizado la función `shapiro` del módulo `stats` de la biblioteca SciPy de Python.

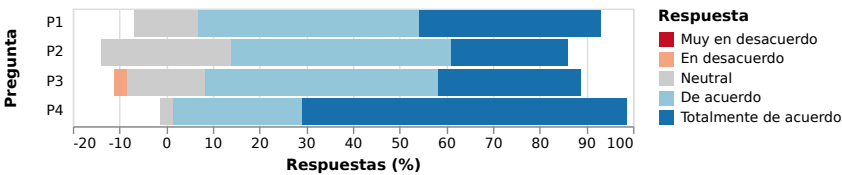


Figure 8: Porcentaje de estudiantes que ha seleccionado cada una de las cinco posibles respuestas, centrando las respuestas neutrales "±en gris"± sobre el 0, para cada una de las cuatro preguntas realizadas sobre los contenidos del taller

TABLE II  
NÚMERO DE ESTUDIANTES, MEDIANA DE LAS NOTAS Y SU INTERVALO DE CONFIANZA AL 95%, PORCENTAJE DE APROBADOS SOBRE PRESENTADOS Y RESULTADOS DEL TEST SHAPIRO-WILK PARA CADA UNO DE LOS LISTADOS DE CALIFICACIONES

Asignatura	Taller MIT	N	Mediana	IC 95%	Dif	% Aprobados	Dif	Shapiro-Wilk W	p-valor
Estructura de computadores	Sí	30	7.87	[7.22, 8.52]	1.43	86.67	-1.18	0.98	0.053
	No	107	6.44	[6.11, 6.77]		87.85		0.92	0.020
Informática básica	Sí	26	6.90	[6.11, 7.69]	1.50	73.08	16.09	0.96	0.003
	No	93	5.40	[5.00, 5.80]		56.99		0.94	0.120
Inglés	Sí	28	6.90	[6.48, 7.32]	0.18	92.86	8.34	0.98	0.157
	No	84	6.72	[6.42, 7.03]		84.52		0.94	0.085
Matemáticas	Sí	27	5.80	[4.78, 6.81]	1.41	55.56	11.23	0.96	0.005
	No	97	4.39	[3.86, 4.92]		44.33		0.93	0.056
Programación	Sí	26	7.35	[6.34, 8.36]	0.85	76.92	12.92	0.93	0.000
	No	100	6.50	[5.96, 7.04]		64.00		0.90	0.014
Todas las informáticas	Sí	24	7.30	[6.51, 8.10]	0.83	79.17	12.99	0.95	0.012
	No	68	6.47	[6.01, 6.93]		66.18		0.93	0.086
Todas salvo inglés	Sí	23	7.04	[6.32, 7.76]	1.15	82.61	20.94	0.98	0.423
	No	60	5.88	[5.43, 6.34]		61.67		0.97	0.746

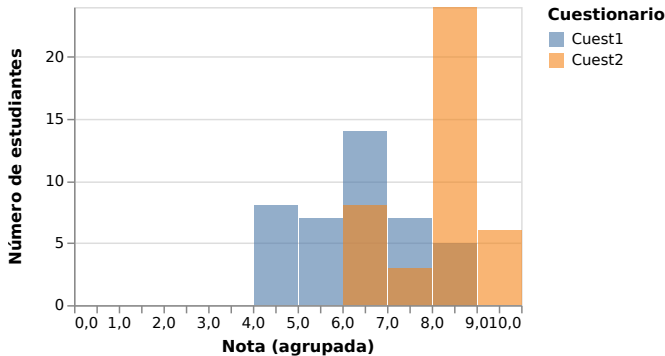


Figure 9: Resultados del cuestionario sobre conocimientos básicos de programación, antes y después de realizar el taller

aprovechado el taller son superiores a las de los que no lo han hecho. Conviene señalar que justamente en la asignatura *Inglés*, que no debería verse influida por si se ha aprovechado o no el taller, es donde se observa una menor diferencia entre las medianas de los dos grupos de estudiantes.

Por otro lado, los porcentajes de aprobados de los estudiantes que han aprovechado el taller también son mayores en todos los casos, salvo en el de la asignatura *Estructura de computadores*, que es ligeramente inferior (86.67% frente a 87.85%). No obstante, el porcentaje de aprobados de esta asignatura ha sido este curso más elevado que el del curso pasado (que fue del 76%) y que los del resto de asignaturas de este curso, a excepción de *Inglés*, que habitualmente suele obtener resultados similares. Así pues, teniendo en cuenta que: i) las tasas de aprobados de ambos grupos están por encima del 85%; ii) que la nota mediana de los que han

aprovechado el taller está por encima de la de los que no, y iii) que en la distribución de las notas de ambos grupos, que se presenta más adelante, se aprecian mejores resultados para aquellos que han aprovechado el taller, consideramos que el taller sí ha tenido un impacto positivo en la asignatura, pese a haber obtenido un porcentaje de aprobados ligeramente inferior. Por último, es interesante observar que el porcentaje de aprobados sobre presentados que se obtiene al combinar las calificaciones de los estudiantes presentados a las asignaturas de matemáticas e informática es un 21% más alto para el grupo de los estudiantes que han aprovechado el taller.

Para poder valorar con más detalle cómo se han distribuido las calificaciones de cada asignatura, estas se han representado por medio de gráficas de cajas y bigotes (véase la Figura 10), que agrupan las calificaciones por sus cuartiles. Como se puede ver, en todos los casos, salvo para *Inglés*, los estudiantes que han aprovechado el taller parten de notas más altas y obtienen mayoritariamente calificaciones más altas que sus compañeros que no lo han hecho. Esta diferencia en cómo se han distribuido las calificaciones es especialmente significativa en las asignaturas *Estructura de computadores* e *Informática básica*, donde más de la mitad de los estudiantes que han aprovechado el taller obtienen mejores notas que las tres cuartas partes de los estudiantes que no lo han hecho.

Por otro lado, para visualizar el porcentaje de estudiantes que ha obtenido una determinada calificación en cada asignatura, en la Figura 11 se muestran los histogramas normalizados de las calificaciones obtenidas por los estudiantes que han aprovechado el taller y los que no, para todas las asignaturas de primer curso, primer semestre, y para la combinación de todas estas asignaturas salvo *Inglés*. En el caso de las asignaturas *Estructura de computadores* e *Informática básica*, la mayor

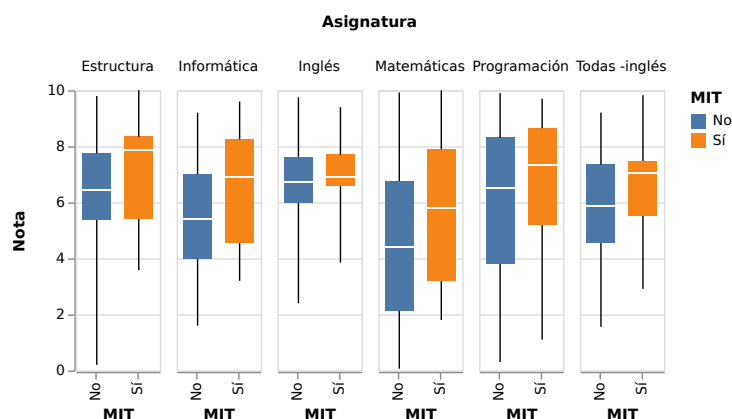


Figure 10: Distribución de las calificaciones de los estudiantes que han aprovechado el taller (naranja) y de los que no (azul), para cada asignatura y para la combinación de todas las asignaturas menos *Inglés*

diferencia porcentual se concentra en las calificaciones altas: entre 8 y 10 para la primera; y entre 7 y 8, pero especialmente, entre 9 y 10, para la segunda. En el caso de *Inglés*, se observan dos histogramas muy similares para ambos grupos de estudiantes. Para *Matemáticas* y *Programación*, pese a que se observan diferencias porcentuales en notas altas, también hay picos en notas bajas, aunque más acusados en *Matemáticas* que en *Programación*. Por último, al combinar las notas de las asignaturas que no son *Inglés*, se puede observar cómo el histograma correspondiente a los que han aprovechado el taller es similar al principio al de los que no, pero desplazado a la derecha, presentando, además, diferencias porcentuales elevadas en las calificaciones entre 6 y 8, y entre 9 y 10.

Para poder evaluar mediante pruebas estadísticas si la diferencia entre las medias de los distintos grupos de calificaciones que se muestran en la Figura 11 son significativas se puede utilizar la prueba T de Student o la de Welch. Para poder utilizar estos estadísticos, es necesario comprobar previamente si las muestras que se quieren comparar provienen de poblaciones normalmente distribuidas.

Como se puede observar en el Cuadro II, las calificaciones que superan el test de normalidad de Shapiro-Wilk (esto es, que obtienen un p-valor  $\geq 0.05$ ) son las de la asignatura *Inglés* (con p-valores de 0.157 y 0.085, para los que han aprovechado el taller y los que no, respectivamente), y las que combinan los resultados de las asignaturas que no son *Inglés* (con p-valores de 0.423 y 0.746, respectivamente). En los otros casos, en al menos uno de los dos grupos de estudiantes, se obtienen p-valores inferiores a 0.05, por lo que se debe rechazar la hipótesis nula de que esas muestras correspondan a una distribución normal (y, por tanto, se deba aceptar que probablemente no estén normalmente distribuidas).

Una vez se ha comprobado que las calificaciones de *Inglés* y de la media de todas salvo *Inglés* representan a poblaciones normalmente distribuidas, el siguiente paso consiste en comprobar si las varianzas de las poblaciones correspondientes a los estudiantes que han hecho el taller y los que no para ambos casos son iguales o no. Si las varianzas son iguales, se utilizará la prueba T de Student y si no, la de Welch.

Al aplicar el test de Levene<sup>8</sup> sobre los resultados obtenidos en la asignatura *Inglés* por los estudiantes que han aprovechado el taller y los que no, se ha obtenido un t-estadístico de 2.55 y un p-valor de 0.12 ( $> 0.05$ ), por lo que

<sup>8</sup>Para realizar la prueba de Levene se ha utilizado la función `levvene` del módulo `stats` de la biblioteca SciPy de Python.

TABLE III  
RESULTADOS DE LA PRUEBA T DE STUDENT SOBRE LAS CALIFICACIONES DE *Inglés* Y LAS CALIFICACIONES AGREGADAS DE TODAS LAS ASIGNATURAS SALVO *Inglés*, ASÍ COMO EL INTERVALO DE CONFIANZA AL 95% DE LA DIFERENCIA ENTRE LAS MEDIANAS DE LOS QUE HAN REALIZADO EL TALLER Y LOS QUE NO.

Asignatura	Prueba T de Student		IC 95%
	t-estadístico	p-valor	
<i>Inglés</i>	0.89	0.187	[−0.32, 0.84]
Todas salvo <i>Inglés</i>	1.80	0.038	[−0.08, 1.67]

se puede asumir que las varianzas de ambas poblaciones son similares. Al aplicar este test sobre las medias de las notas obtenidas por los que han aprovechado el taller y los que no en todas las asignaturas salvo *Inglés*, se ha obtenido un t-estadístico de 0.56 y un p-valor de 0.46 ( $> 0.05$ ), por lo que en esta caso, también se puede asumir que las varianzas de ambas poblaciones son similares.

Aplicando la prueba T de Student<sup>9</sup> sobre las calificaciones a las que puede aplicarse, a las de *Inglés* y a las de todas las asignaturas salvo *Inglés*, se han obtenido los resultados que se muestran en el Cuadro III. Aplicando el nivel de significación anterior ( $\alpha = 0.05$ ) se puede concluir: i) que en el caso de la asignatura *Inglés* (p-valor = 0.187  $> 0.05$ ) no hay una diferencia estadísticamente significativa entre haber realizado el taller o no, que era lo esperado; y ii) que en el caso de las calificaciones agrupadas (p-valor = 0.038  $< 0.05$  y t-estadístico = 1.8  $> 0$ ) es posible rechazar la hipótesis nula y aceptar la hipótesis alternativa de que las calificaciones obtenidas si se aprovecha el taller son significativamente mejores a las que se obtienen en caso contrario.

Conviene recordar, no obstante, que fueron los propios estudiantes los que eligieron participar o no en el taller. Por este motivo, se debe tener en cuenta que el que los estudiantes que han aprovechado el taller hayan obtenido mejores resultados que sus compañeros podría ser debido en realidad a otros factores externos a este estudio.

<sup>9</sup>Para realizar la prueba T de Student para muestras independientes se ha utilizado la función `ttest_ind` (del módulo `stats` de la biblioteca SciPy de Python) indicando que las varianzas de ambas poblaciones son iguales.



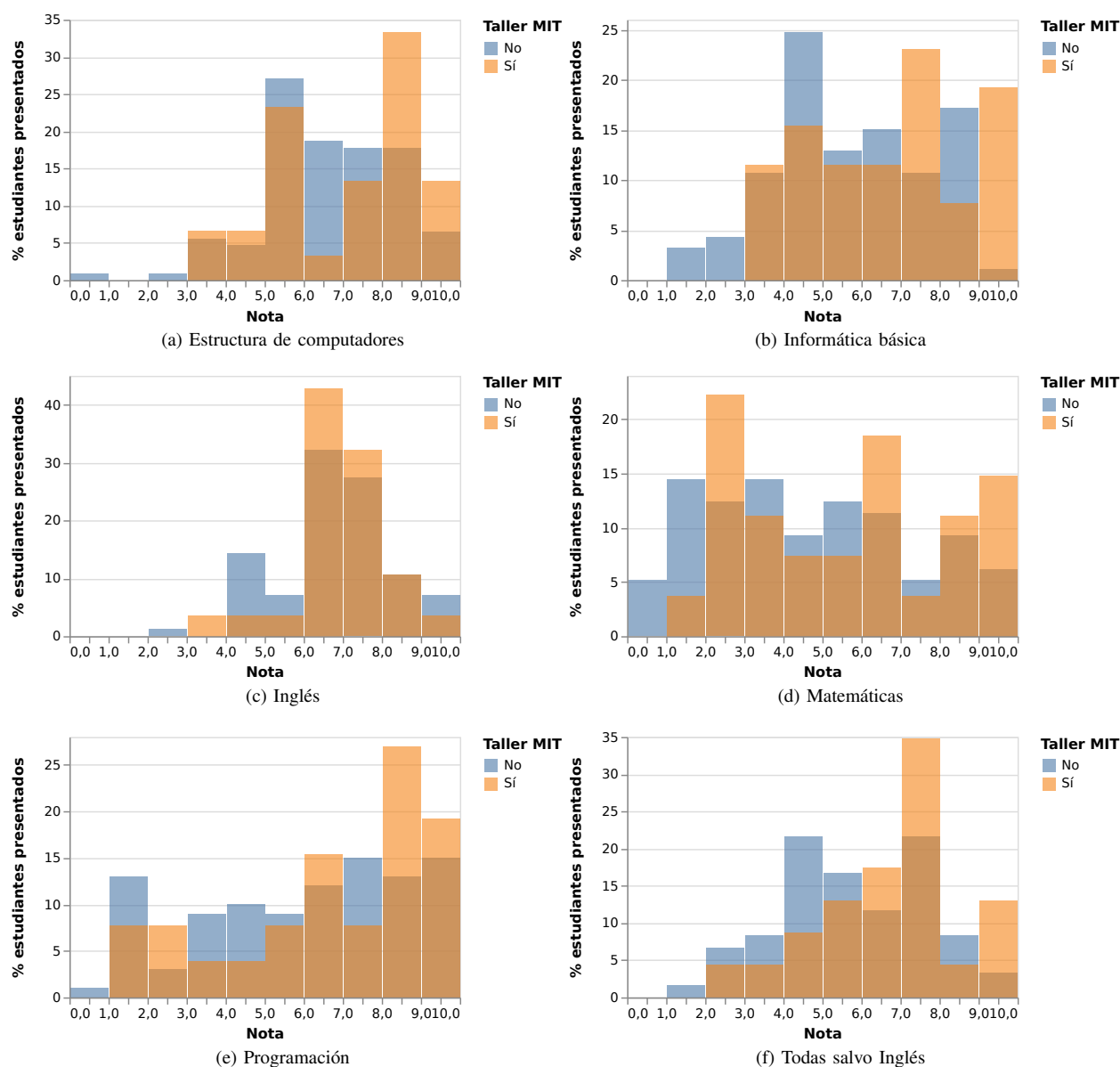


Figure 11: Histogramas normalizados de las calificaciones obtenidas en las distintas asignaturas de primer curso, primer semestre, por los estudiantes que han aprovechado el taller (en naranja) y el resto de estudiantes (en azul)

## VII. CONCLUSIONES Y TRABAJO FUTURO

Se ha creado un taller de programación móvil con MIT App Inventor para estudiantes de primer curso de grado, que tiene por objeto el desarrollo de un conjunto de competencias básicas en programación que permitan mejorar su proceso de aprendizaje, en particular en la asignatura *Estructura de computadores*. Este taller sigue la filosofía general de los tutoriales de MIT App Inventor, que es la de permitir que los estudiantes puedan ver cuanto antes el resultado de lo que están realizando. Para ello, se proponen cinco proyectos, en orden creciente de dificultad, estructurados en una secuencia de pasos, lo que permite que las competencias necesarias se vayan adquiriendo y comprobando de forma paulatina.

El taller ha tenido una buena acogida entre los estudiantes. Lo han completado casi todos los inscritos y ha sido valorado muy positivamente. Además, se pasó a los estudiantes un cuestionario sobre conceptos básicos de programación antes y después de realizar el taller y los resultados evidencian que han mejorado de forma significativa.

Por otra parte, tras comparar las calificaciones obtenidas en las asignaturas de primer curso por los estudiantes que han aprovechado el taller con las del resto de estudiantes, se puede apreciar un mayor porcentaje de estudiantes con una calificación más alta, especialmente en las asignaturas más relacionadas con programación.

Como trabajo futuro se pretende reeditar el taller aplicando algunas de las recomendaciones realizadas por los estudiantes: distribuir mejor el tiempo dedicado a los distintos proyectos, sustituir alguno de los proyectos actuales por algún tipo de juego y realizar una explicación inicial de los recursos ofrecidos por MIT App Inventor. También aprovecharíamos esta nueva edición para recabar más información sobre su efecto en el desempeño académico de quienes lo cursen.

## AGRADECIMIENTOS

En primer lugar nos gustaría dar las gracias a los desarrolladores de MIT App Inventor y a la comunidad educativa que se ha constituido alrededor de esta herramienta. Consideramos

que se ha creado una plataforma excelente para la difusión de las STEM que beneficiará a la sociedad.

También nos gustaría dar las gracias a nuestros compañeros de primer curso, que nos han cedido amablemente las calificaciones de las asignaturas de las que son responsables: Juan Carlos Amengual Argudo, Fernando Javier Hernando Carrillo, José Luis Llopis Borrás, Ana María Lluch Peris, Marina Murillo Arcila, y María Luisa Renau Renau.

Por último, nos gustaría agradecer igualmente el apoyo recibido por parte del Departamento de Ingeniería y Ciencia de los Computadores de la Universitat Jaume I.

#### REFERENCES

- [1] S. Barrachina, G. Fabregat, C. Fernández, and G. León, "Utilizando ARMSim y QtARMSim para la docencia de Arquitectura de Computadores," *ReVisión*, vol. 8, no. 3, p. 2, 2015.
- [2] S. Barrachina, G. Fabregat, and J. V. Martí, "Utilizando Arduino DUE en la docencia de la entrada/salida," in *Actas de las XXI Jornadas de la Enseñanza Universitaria de la Informática*. Universitat Oberta La Salle, 2015, pp. 58–65.
- [3] X. Canalet, F. Sánchez, I. Jacob, Á. Velázquez, and M. Marques, "Declaración AENUI-CODDII por la inclusión de asignaturas específicas de ciencia y tecnología informática en los estudios básicos de la enseñanza secundaria y bachillerato," *Actas de las XX Jornadas sobre Enseñanza Universitaria de la Informática*, pp. 229–236, 2014.
- [4] S. Barrachina Mir and G. Fabregat Lluca, "¿Puedo programar mi móvil? Pero si acabo de llegar," *Actas de las XXV Jornadas sobre Enseñanza Universitaria de la Informática*, vol. 4, no. 0, 2019.
- [5] J. Maloney, M. Resnick, N. Rusk, B. Silverman, and E. Eastmond, "The Scratch programming language and environment," *ACM Transactions on Computing Education (TOCE)*, vol. 10, no. 4, pp. 16:1–16:15, 2010.
- [6] D. Topalli and N. E. Cagiltay, "Improving programming skills in engineering education through problem-based game projects with Scratch," *Computers & Education*, vol. 120, pp. 64–74, 2018.
- [7] D. Ozoran, N. Cagiltay, and D. Topalli, "Using Scratch in introduction to programming course for engineering students," in *2nd International Engineering Education Conference (IEEC2012)*, vol. 2, 2012, pp. 125–132.
- [8] S. P. Roche and N. M. Martínez, "Evaluación de entornos de programación para el aprendizaje," *Actas de las XVII Jornadas sobre Enseñanza Universitaria de la Informática*, pp. 83–90, 2011.
- [9] R. Muñoz, T. S. Barcelos, R. Villarroel, M. Barría, C. Becerra, R. Noel, and I. Frango Silveira, "Uso de Scratch y Lego Mindstorms como apoyo a la docencia en fundamentos de programación," in *Actas de las XXI Jornadas de la Enseñanza Universitaria de la Informática*. Universitat Oberta La Salle, 2015, pp. 248–254.
- [10] S. Trilles and C. Granell, "SUCRE4Kids: El fomento del pensamiento computacional a través de la interacción social y tangible," *Actas de las XXIV Jornadas sobre Enseñanza Universitaria de la Informática*, vol. 3, no. 0, pp. 303–310, 2018.
- [11] S. Papadakis, M. Kalogiannakis, V. Orfanakis, and N. Zaranis, "Novice programming environments. Scratch & App Inventor: a first comparison," in *Proceedings of the 2014 Workshop on Interaction Design in Educational Environments*. ACM, 2014, pp. 1–7.
- [12] S. A. Nikou and A. A. Economides, "Transition in student motivation during a Scratch and an App Inventor course," in *2014 IEEE Global Engineering Education Conference (EDUCON)*. IEEE, 2014, pp. 1042–1045.
- [13] D. Wolber, "App Inventor and real-world motivation," in *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education*. ACM, 2011, pp. 601–606.



**Sergio Barrachina Mir** es Ingeniero de Telecomunicaciones especialidad Electrónica por la Universidad Politécnica de Valencia (1995) y Doctor en Ingeniería Informática por la Universitat Jaume I de Castellón (2003). Es Profesor Titular de Universidad en el Área de Arquitectura y Tecnología de Computadores desde 2012.

Ha impartido docencia principalmente en el primer y segundo ciclo de las antiguas titulaciones de informática y en los actuales grados en Ingeniería Informática y en Matemática Computacional.

Es miembro del grupo de investigación High Performance Computing & Architectures (HPC&A) en el que ha participado en numerosos proyectos relacionados con la computación y arquitecturas de altas prestaciones.



**Germán Fabregat Lluca** es licenciado en Física, especialidad Electricidad, Electrónica e Informática, por la Universidad de Valencia desde 1989 y doctor en la misma disciplina desde 1996. Es profesor de la Universidad Jaume I desde 1991, siendo Titular del Área de Arquitectura y Tecnología de Computadores desde 2001.

Sus trabajos se han desarrollado en tolerancia a fallos, sistemas empujados y redes de sensores, con especial interés en sus aplicaciones a la industria. Su labor docente se caracteriza además por el continuo desarrollo de aplicaciones de soporte a la docencia como dispositivos para las prácticas, simuladores, entornos de programación de sistemas, etcétera.