Escuela Superior
de Ingeniería y Tecnología
Universidad de La Laguna

Trabajo Fin de Grado

Pensamiento Computacional: Trazabilidad de los retos de entrenamiento

Computational Thinking: Traceability of Training Challenges

Carla Ramos Alonso

La Laguna, 10 de junio de 2019

Dra. **Coromoto León Hernández**, con N.I.F. 78.605.216-W, profesora Catedrático de Universidad del área de Lenguajes y Sistemas Informáticos, adscrita al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como tutora,

Dr. **Carlos Segura González**, con N.I.F. 78.404.244-S, profesor Investigador tipo C adscrito al Departamento de Ciencias de la Computación del Centro de Investigación Matemática (CIMAT) de México, como cotutor,

C E R T I F I C A (N)

Que la presente memoria titulada:

"*Pensamiento Computacional: Trazabilidad de los retos de entrenamiento*"

ha sido realizada bajo su dirección por D^{a}. **Carla Ramos Alonso**, con N.I.F. 54.059.713-F.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 10 de junio de 2019

Agradecimientos

A mi madre, mi mejor amiga, que siempre ha ordenado mi caos.

A mi hermana y compañera de vida.

A mi tía favorita, que siempre me ha allanado el camino.

A mi familia, continuamente celebrando mis logros:

Mi abuelo por ser el mejor consejero.

Mi primo Daniel, la alegría de esos días en los que no podía más.

Mi tío César por empujarme hacia la informática.

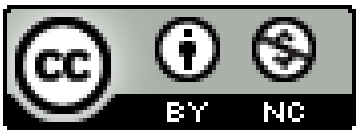
Mi tía Rosi y Ale por apoyarme siempre en esta travesía.

A mis amigos de la facultad, por compartir las penas pero sobre todo las alegrías.

A mi tutora Coromoto, por enseñarme la magia de la computación.

A todos los que creyeron en mí.

Licencia



© Esta obra está bajo una licencia de Creative Commons Reconocimiento-NoComercial 4.0 Internacional.

Resumen

El uso de las nuevas tecnologías está en el día a día de todos. Existen diversas capacidades cognitivas necesarias para hacer una correcta utilización de las herramientas que se nos ofrecen. Sin embargo, si vamos más allá, y nos planteamos no solo el consumo de esta ciencia, sino que analizamos también las aptitudes necesarias para comprenderla, diseñarla o crearla, obtendremos como resultado la aplicación del ahora cada vez más conocido: Pensamiento Computacional (PC). Este concepto abarca competencias tales como la aplicación de la lógica, la abstracción, la simplificación, la optimización y el pensamiento crítico aplicados a la resolución de problemas. La gran ventaja del desarrollo de esta destreza es que se puede aplicar en cualquier disciplina, no solo en el campo de la computación.

Existe un gran número de herramientas y plataformas que hacen más sencilla la tarea de entrenar el PC, aunque si deseamos tener una mayor implicación en la tarea de desarrollar dicha inteligencia, podemos involucrarnos en obtener una trazabilidad de los retos de entrenamiento. Obteniendo con ella las dificultades encontradas, los errores más comunes o los conceptos más complejos de interpretar. En ello se centra la finalidad de este trabajo: junto con la investigación del estado actual del tema, y el análisis de las plataformas ya existentes, se realiza la implementación de un reto de entrenamiento que además permite obtener una trazabilidad del mismo. Se ha desarrollado una aplicación web a la que puede acceder cualquier persona de forma gratuita.

Palabras clave: pensamiento computacional, reto de entrenamiento, aplicación web, trazabilidad, análisis de datos

Abstract

Nowadays, use of latest technologies is in everyone's daily life. There are different cognitive abilities that are necessary to make a correct use of the tools offered. However, going further, and not only considering this science use, but also analyzing necessary skills to understand, design or create it, we will obtain as a result the application of the, now increasingly well-known, Computational Thinking (CT). This concept covers skills such as the logic application, abstraction, simplification, optimization and critical thinking, applying them in problem solving. Advantages of developing that skill, is that it could be applied to any discipline, and not only in Computer Science.

There are a great number of tools and platforms only dedicated to make CT training easier, although if we want to be more involved in this intelligence development, we can try to obtain the training challengue traceability. We will get difficulties found during the challenge, most common mistakes and complex concepts. That's the purpose of that project: investigate the subject in our society, analyze existing platforms and create a training challenge that will let us know a traceability of it. It will be a free web application that can be used by anyone, anywhere.

Keywords: *computational thinking, training challenge, web application, traceability, data analysis*

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

A lo largo de las últimas décadas, la tecnología se ha hecho un hueco muy grande en nuestras vidas y actualmente la informática está implantada en casi todos los ámbitos de nuestra sociedad. Este hecho hace que nuestro día a día se vea influenciado de una manera u otra con la computación y todo lo que ella conlleva. La informática y sus funcionalidades han llegado a los rincones menos pensados, haciendo de nuestras tareas cotidianas o trabajos más complejos, simples órdenes a una máquina o a un conjunto de ellas. La implicación que debemos dedicar como consumidores de esta nueva ciencia, para adaptarnos a la rapidez de los avances, debe estar a la altura. Por ello, en este capítulo se define lo que es el Pensamiento Computacional y se destaca la importancia del aprendizaje del mismo en edades tempranas.

1.1. ¿Qué es el Pensamiento Computacional?

El Pensamiento Computacional (PC) puede definirse como la capacidad que tiene un individuo para la resolución de problemas mediante la división de un determinado reto en subproblemas o instrucciones más pequeñas y aplicar luego algoritmos computacionales para resolverlos. El concepto de PC fue formalizado por Jeannette M. Wing en 2006 [36], afirmando que esta habilidad es mucho más que saber programar una máquina, es conseguir realizar una abstracción del problema a resolver, aplicando la lógica. Wing defiende también este pensamiento como una habilidad fundamental para la sociedad en general, y no solo como una simple destreza que deben desarrollar los dedicados al mundo de las Ciencias de la Computación.

Dicho pensamiento se puede adquirir mediante el análisis y estudio de los problemas, tratando de buscar una forma estructurada y más sencilla de resolverlos. Además, ha de conocerse cuál es el algoritmo más indicado para resolverlo, así como tener la capacidad de diseñar e implementar un nuevo algoritmo si este aún no existe. Esto no implica que se deba pensar como los ordenadores, ya que

en realidad los humanos son los encargados de crear los modelos y diseñar los sistemas que las máquinas deben reproducir [36].

Wing afirma que el PC comenzará a ser una verdadera realidad para la humanidad y no una simple filosofía explícita cuando este sea parte integral de nuestros esfuerzos. Por tanto, se debe inspirar el interés por este campo y promover el PC como una ciencia igual de importante y necesaria que cualquier otra [36].

1.2. Importancia de la adquisición del PC en edades tempranas

Se deben llevar a cabo labores de concienciación de las ventajas que tiene el aprendizaje de esta ciencia desde que se comienza con la educación. Así, se formarán personas con notables capacidades de análisis, abstracción, síntesis y optimización. Disciplinas como la literatura, las matemáticas y la música deberían estar acompañadas de aquellas que nos inviten a pensar utilizando la lógica, aplicar el ingenio, el razonamiento y adquirir las habilidades necesarias para resolver problemas empleando el PC.

Existen a día de hoy numerosas herramientas que permiten a los jóvenes adquirir estas habilidades con retos, de manera que estas competencias serán obtenidas de forma entretenida y más sencilla desde que comienzan con su educación escolar. Además, los retos que pueden ser empleados para promover este aprendizaje están basados en juegos que hacen más atractiva la tarea de aprender. Se introduce entonces el concepto de Aprendizaje Basado en Juegos, del inglés *Game Based Learning (GBL)*. Estos han sido propuestos como un buen sistema pedagógico para desarrollar las habilidades que nos proporciona la aplicación del PC [22].

Dichos juegos pueden estar diseñados de tal forma que inviten a crear mentalmente las primeras aproximaciones a lo que puede ser la definición de un algoritmo básico. A través de una programación visual del reto, se puede facilitar el uso de sentencias empleadas en programación, que ayudarán al usuario a familiarizarse con el Pensamiento Computacional empleando bucles, condicionales, recursividad, etc. de forma implícita. Consistirá por tanto en presentar el reto como un problema a resolver, el cual hará que el usuario tenga que plantearse antes de aplicar una solución, cuál es la mejor manera de resolverlo.

1.3. Motivación y objetivos

Se conoce que la aplicación del GBL da buenos resultados [22], pero se desconoce si el uso de dichos juegos ayuda al estudiante a hacer un correcto análisis del reto, cumpliendo con los objetivos de promover el PC, se desconoce también qué concepto pudo haberle parecido más complicado de abordar, o si le ha resultado difícil abstraer las pautas a seguir para la resolución del mismo.

Por consiguiente, sería interesante obtener una retroalimentación del progreso del estudiante desde que inicia la realización de un reto sin conocimientos específicos de PC, hasta la finalización del reto, tras la supuesta adquisición de los mismos.

Por tanto, se propone en el desarrollo de este Trabajo Fin de Grado (TFG), la realización de un reto de entrenamiento del PC y adicionalmente la obtención de un informe detallado de las decisiones tomadas y pasos llevados a cabo tras haber finalizado el estudiante uno de dichos juegos o retos, para así alcanzar la trazabilidad del proceso. De este modo, se podría poner el foco en cuáles han sido los errores y dónde encuentran mayor dificultad los estudiantes a la hora de aplicar el pensamiento lógico. Así, una posible mejora de estos juegos educativos podría dirigirse a plantear de forma distinta los mismos para así facilitar el aprendizaje.

Los objetivos a cumplir para la realización de este TFG se recapitulan a continuación:

- Revisión bibliográfica. Haciendo un estudio de las investigaciones existentes acerca del PC y el estado actual en el que se encuentra el tema.
- Desarrollo de un reto de entrenamiento. En este, el estudiante podrá adquirir conocimientos básicos de programación que le ayuden a aplicar el PC.
- Trazabilidad del reto. Se obtendrán los resultados de la aplicación del reto para su posterior análisis.
- Creación de la documentación. Se realiza la redacción de la memoria del Trabajo en la que se incluyen los datos en relación al tema propuesto y los detalles del desarrollo.

1.4. Metodología

La metodología de trabajo se estableció desde un principio en pequeñas tareas que había que alcanzar para llevar un orden en la correcta realización de este trabajo. La planificación de las prioridades y los tiempos se hicieron desde un

comienzo en las primeras reuniones con la tutora. Estas reuniones se llevaron a cabo asiduamente durante el desarrollo del proyecto, estableciendo siempre el siguiente paso a seguir y revisando lo hasta ese momento realizado. El procedimiento que se llevó a cabo para su precisa finalización, ha sido el expuesto a continuación:

Tarea 1. Revisión bibliográfica

Para poder comenzar con la realización de un proyecto basado en el PC, se debe realizar la búsqueda de una bibliografía que permita analizar el tema propuesto, además de todo lo relativo a los antecedentes y estado actual del tema.

Tarea 2. Estudio de las herramientas existentes

Debido a la existencia actual de gran cantidad de herramientas para el desarrollo del PC, se debe realizar antes de proceder al desarrollo de la correspondiente a este proyecto, una investigación de las que ya están implantadas para evaluar sus ventajas y desventajas, de manera que así se pueda analizar las posibles mejoras a llevar a cabo.

Tarea 3. Desarrollo de un reto de entrenamiento

Debido a que este trabajo consiste en la implantación y desarrollo del PC, se comienza con la implementación de un reto de entrenamiento para mejorar dichas capacidades. Se establecieron los objetivos de dicho reto y además se barajan distintas posibilidades de diseño y funcionamiento del mismo. Por tanto, en esta tarea se decide al comienzo cuales van a ser las características iniciales del reto para proceder con su posterior desarrollo.

Tarea 4. Implantación de la trazabilidad

Teniendo en cuenta la importancia que conlleva para este trabajo el seguimiento de la adquisición del PC, se deben establecer pautas para la correcta trazabilidad del reto de entrenamiento. Se fijan los requisitos que debe cumplir dicha trazabilidad y en qué momentos tendrá más importancia seguirla.

Tarea 5. Presentación de los resultados documentales

Dicho proyecto llevará consigo la documentación del procedimiento, por tanto, al finalizar con las anteriores tareas se procederá a la redacción de la memoria donde se hará la presentación del proyecto de forma escrita. En ella se redactan los pasos llevados a cabo y las herramientas utilizadas en las tareas anteriormente citadas. Además, se establece un tutorial para el usuario que requiera la

utilización de la herramienta para el desarrollo del PC.

El resto de este documento consta de:

Antecedentes y estado actual del tema en el Capítulo 2, donde se especifican los recursos bibliográficos a los que se ha acudido, y además se especifican las herramientas ya existentes en relación con el desarrollo del PC. En el Capítulo 3 se especifican los puntos más importantes para el desarrollo del reto y las herramientas que han hecho posible la obtención del mismo. El análisis de los resultados figura en el Capítulo 4, ya que superado el reto de entrenamiento se obtendrán los resultados de la ejecución del mismo, se trata el desarrollo del procedimiento de obtención de datos y los pasos siguientes tras recibir el informe de acciones. Se detalla un Modo de uso en el Capítulo 5, con un tutorial dónde se explican los pasos a seguir para el correcto funcionamiento de la aplicación, así como los objetivos a conseguir con la ejecución del mismo. Por último, en el Capítulo 6, se enumeran las Conclusiones y líneas futuras: se redactan las propuestas de continuación de este trabajo. Y además, en el Capítulo 7, se enumeran también en inglés. El presupuesto se especifica en el Capítulo 8.

Capítulo 2

Antecedentes y estado actual del tema

En este capítulo se detallan los recursos bibliográficos de los que se ha obtenido información y con los cuales se han contrastado datos. Además, se llevará a cabo un análisis de las herramientas existentes para el desarrollo del Pensamiento Computacional (PC).

2.1. Retos para el entrenamiento del PC

Tras realizar una revisión sistemática en diferentes bases de datos localizadas en el buscador de recursos de la Universidad de La Laguna, denominado PuntoQ, se obtuvo la bibliografía empleada en la realización de este proyecto. Estas búsquedas han sido realizadas en las siguientes bases de datos: ACM, Web of Science y Scopus.

En ellas se publican referencias bibliográficas y artículos científicos. Tras obtener un gran número de resultados buscando con palabras clave tales como: *'computational thinking'*, *'tracking video games'*, *'game based learning'* se detalla en los siguientes párrafos los **artículos de referencia** consultados para la investigación relativa a este proyecto.

En **"Computational Thinking"** [36] se comenzaron a marcar las primeras definiciones que se atribuyeron al concepto de Pensamiento Computacional. Su autora, **Jeannette M. Wing** define el PC como un método y modelo que permite resolver problemas y diseñar sistemas que nadie podría afrontar solo. Este hecho hace que se plantee la pregunta de qué o qué no, puede ser computable.

El PC es una habilidad fundamental para todos, dice la autora, no solo para los dedicados a la computación. Este nos hace plantearnos cuán de difícil es

resolver un problema y cuál es la mejor manera de abordarlo, de manera que se puede aplicar a casi cualquier ámbito de la vida cotidiana. Con su adquisición, se aprende a procesar información en paralelo, interpretar código como datos y viceversa, a juzgar un programa además de por su eficiencia por su diseño y simplicidad. Esta nos permite además ser capaces de pensar en múltiples niveles de abstracción ante un problema.

Wing concluye diciendo que el desarrollo del PC no consiste en llevar a las personas a pensar como máquinas, ya que estas sin la imaginación e inteligencia de los humanos no serían tan emocionantes. Por ello, anima a propagar el poder de las ciencias de la computación al resto del mundo, para que el Pensamiento Computacional sea una realidad y deje de ser una filosofía explícita.

Por otro lado, en **"Learning Programming at the Computational Thinking Level via Digital Game-Play"**, se indica que los juegos digitales son atractivos y enganchan a cualquier grupo de personas, es por ello que el GBL forma parte fundamental de la adquisición del Pensamiento Computacional desde edades muy tempranas.

Cagin Kazimoglu, en [22], describe los beneficiosos efectos que tiene sobre los estudiantes adquirir el PC mediante el uso de GBL. Se menciona también el uso de Scratch [23], Alice [32] y Agentsheets [3] como herramientas visuales para dar soporte a la educación introductoria de la programación.

Destaca la elección de este artículo porque en él se plantea la urgencia de obtener un mejor entendimiento del impacto del uso de juegos para la introducción temprana de la programación y el desarrollo del PC. Esta información da soporte a la implantación de la trazabilidad que se ha llevado a cabo en el desarrollo de este proyecto.

El GBL ayuda a la familiarización con las estructuras, sentencias, patrones y requerimientos que exige la programación y no solo con la sintaxis y semántica que se aplica en la misma.

Además, la implicación de dicho artículo en este proyecto cobra aún más importancia cuando se confirma en él, que para crear un reto efectivo se han de usar herramientas de programación visual, que permitan al usuario tener un repositorio de posibilidades y escoger cuáles son las necesarias y construir con ellas la solución correcta del mismo.

Por último, se hace referencia a las distintas habilidades que se adquirirán tras la superación de un buen reto para el aprendizaje y desarrollo del PC:

- **Resolución de problemas.** Empleando la lógica y utilizando modelos

computacionales llegar a la resolución de un problema.

- **Construcción de algoritmos.** Elaborar procedimientos paso a paso que resuelvan un problema en particular.
- **Depuración.** Analizar problemas y errores de un cierto algoritmo.
- **Simulación.** Construcción de un modelo, demostrando el funcionamiento del algoritmo.
- **Socialización.** Coordinación, cooperación y/o competición durante los estados de la resolución de un problema, construcción del algoritmo, depuración y simulación.

En **"How to learn and how to teach computational thinking: Suggestions based on a review of the literature"** [18] se pueden encontrar numerosas estadísticas y clasificaciones referentes a muchos de los ámbitos relacionados con el PC, además de indicar técnicas efectivas para la implantación del Pensamiento Computacional en las aulas. Una de las clasificaciones que se encuentran enumera los conceptos que se desarrollan gracias a la aplicación del PC, entre ellos figuran los siguientes: *abstracción, diseño de algoritmos, automatización, análisis de datos, descomposición, paralelización, simulación, reconocimiento de patrones, etc.*

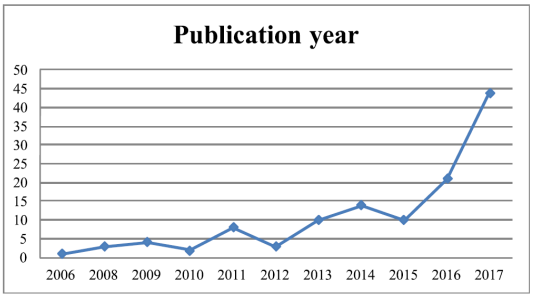


Fig. 1. The annual distribution of computational thinking papers.

Figura 2.1: Artículos relacionados con el PC publicados por año

Además, se publica en el artículo el histograma de la Figura 2.1 que muestra la cantidad de publicaciones referentes al PC, donde se ve un incremento notable en los últimos años. Este hecho confirma que el PC es, cada vez más, una importante habilidad a desarrollar.

Para concluir, muestra el número de estudios dedicados a cada estrategia para el aprendizaje del PC. En la Figura 2.2 se puede observar que los estudios dedicados al GBL se encuentran entre las primeras posibilidades más estudiadas para desarrollar el PC.

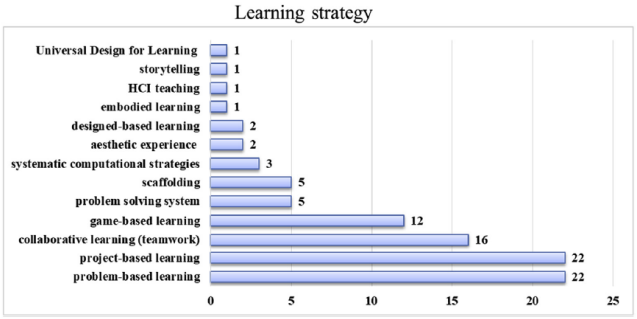


Fig. 3. The number of computational thinking studies for each learning strategy.

Figura 2.2: Estrategias de aprendizaje del PC

2.2. Herramientas para el entrenamiento del Pensamiento Computacional

Existen actualmente numerosas herramientas para ejercitar el desarrollo del PC, y como el principal objetivo de este trabajo es el desarrollo de una, sirven de referencia plataformas como **Code.org** [8], Figura 2.3, ya que ofrecen grandes facilidades para que se pueda aprender jugando, pero destacar sobre todo su gran implicación en la propuesta de un gran número de actividades, proyectos y cursos para el desarrollo del PC.

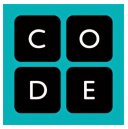


Figura 2.3: Logo Code.org

Valiéndonos del artículo titulado *"Education in the Knowledge Society"* [37], se extraen algunos datos de las herramientas para el desarrollo del PC que se relacionan en la Tabla 2.1.

Herramienta	Gratis	Online	Dificultad	Método empleado
Logo (Turtle Academy)	Sí	Sí	Baja	Basada en texto
Scratch	Sí	Sí	Baja	Basada en bloques
Snap!	Sí	Sí	Baja	Basada en bloques
Alice	Sí	No	Media	Basada en bloques
Looking Glass	Sí	No	Media	Basada en bloques
App Inventor	Sí	Sí	Baja	Basada en bloques
GreenFoot	Sí	Sí	Alta	Basada en texto
Pencil Code	Sí	Sí	Baja	Ambos
AgentSheets	No	Sí	Medium	Basada en bloques
AgentCubes	No	Sí	Medium	Basada en bloques
AgentCubes Online	Sí	Sí	Medium	Basada en bloques

Tabla 2.1: Herramientas para desarrollar el Pensamiento Computacional

Tras la revisión de las herramientas citadas en la Tabla 2.1, se procede a detallar más en profundidad dos de las cuales en ella figuran, como son **Scratch** [23] y **Alice** [32]. Además se estudiará una de las actividades que ofrece **Code.org** [8] con *"La Hora del Código (Hour of Code)"* [9].

2.2.1. Scratch

Scratch [23] (Figura 2.4) es una herramienta desarrollada por el grupo Lifelong Kindergarten en el MIT Media Lab. Consiste en una aplicación web en la que se pueden crear juegos, animaciones, o retos y compartirlo con el resto, o utilizar los que ya están creados.

Figura 2.4: Logo Scratch

Entre las posibilidades que ofrece, se encuentra la de poder crear nuestro propio reto o juego. Este está basado en bloques, como se mencionó anteriormente en la Tabla 2.1. Los bloques ofrecen la posibilidad de ejecutar una porción de código sin que te percates de ello, ya que están representados con una interfaz gráfica que les hace ser más comprensibles. El entorno ofrecido para la creación de un reto es muy intuitivo, separado en diversas secciones, donde podemos ver una sección para los bloques a utilizar, otra denominada *'Workspace'* o *'Espacio de trabajo'* y una previsualización del contenido que se va generando, véase

la Figura 2.5. Con aplicaciones como esta podemos crear un reto basado en bloques que pueden utilizar después otras personas. Sin embargo, este entorno nos permite crear el objetivo con bloques, y luego hacer uso de él, pero no es el ideal para la creación de un reto basado en bloques en el que el propio usuario deba hacer uso de los bloques creados para ejecutar una determinada acción.

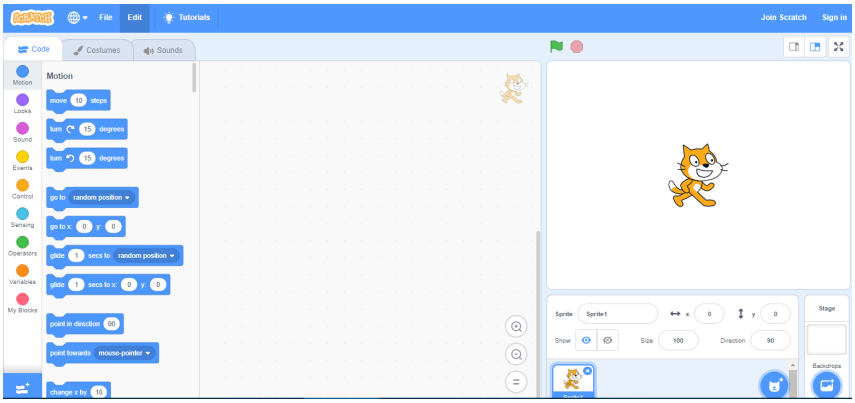


Figura 2.5: Creación de un reto con Scratch

2.2.2. Alice

La plataforma Alice [32] (Figura 2.6) es usada por profesores para su aplicación en distintos niveles, tanto en primaria como universitarios. Es un entorno de programación basado en bloques que permite crear animaciones, o programas en 3D. Ayuda al aprendizaje a través de la creatividad, incitando al usuario a aplicar la lógica y la programación orientada a objetos.

Figura 2.6: Logo Alice

Tal y como se indica en la Tabla 2.1, esta herramienta no está disponible de manera online, por tanto es necesario descargarse el software [32]. La interfaz que encontramos esta vez es ligeramente diferente, véase la Figura 2.7, pero siempre teniendo en cuenta que podremos encontrar en las aplicaciones destinadas a este tipo de aprendizaje lo siguiente:

- Una sección dónde se encuentran los bloques, o porciones de código que pueden utilizarse.

- Un espacio en blanco dónde se podrán colocar los bloques necesarios para ejecutar una determinada acción.
- Una visualización o previsualización del resultado que se obtendrá tras ejecutar lo establecido en el espacio anterior.

En el caso de Alice, lo primero que se nos pide es elegir un escenario en el que trabajar. Una vez elegido el escenario, podremos pasar a crear un reto, o animación, empleando lo que se nos dispone en dicha herramienta, es decir, los espacios que se han mencionado anteriormente.

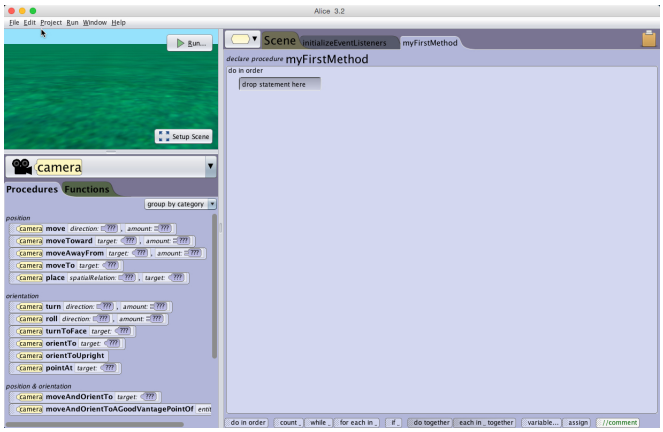


Figura 2.7: Creación de un reto con Alice

2.2.3. Hour of Code

Por último, destacar Hour of Code [9] (Figura 2.8), que sin lugar a dudas es una perfecta herramienta para el desarrollo del PC. Es un movimiento creado por Code.org [8] que permite realizar retos y tutoriales de una hora a cualquier persona en cualquier lugar, ya que está disponible en más de 45 idiomas.

Figura 2.8: Logo Hour of Code

Se ofrecen una gran cantidad de posibilidades para aprender jugando, y aunque la gran mayoría se basan en una implementación en bloques, existen otros tantos en los que, por ejemplo, se puede aprender más acerca de la seguridad informática, como el cifrado de seguridad César [33], desarrollar capacidades para comprender los condicionales, y además aprender a emplear técnicas que utilizan los equipos para comprimir un determinado fichero de texto, véase la Figura 2.9.

Figura 2.9: Retos en Hour of Code

Analizando gran parte de los retos basados en bloques con los que cuenta este espacio, la gran mayoría consisten en colocar una secuencia de bloques (instrucciones) en el área de trabajo, y al ejecutarlo un determinado objeto se moverá hasta alcanzar el objetivo/posición que se pide (Figura 2.10). Son retos que comienzan siendo sencillos, pero que a medida que el nivel aumenta, la dificultad se ve afectada obligando a aplicar un mayor nivel de abstracción ante el problema. En estos, se comienzan a entender conceptos como *'instrucción'*, *'condicional'* y *'bucle'*, esenciales para el comienzo temprano de la adquisición del PC.

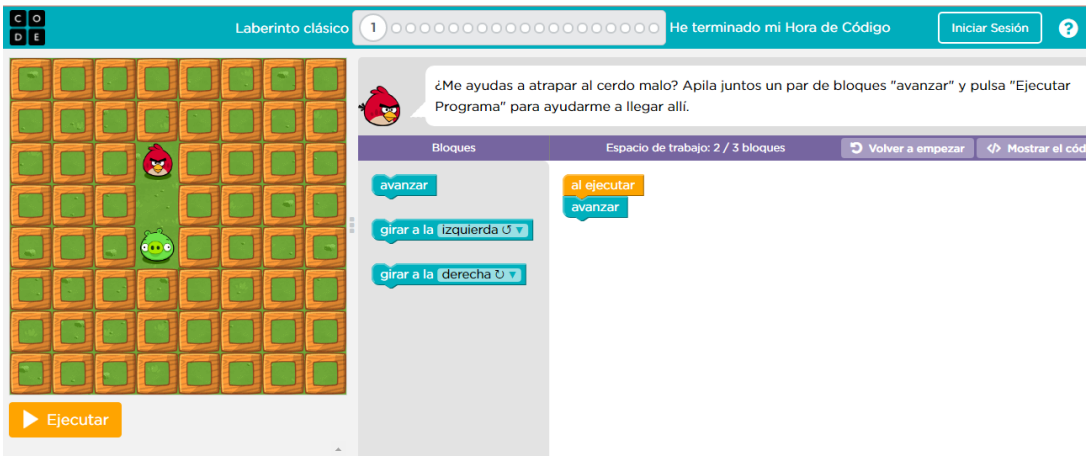


Figura 2.10: Completar un reto con Hour of Code

Capítulo 3

Desarrollo

El desarrollo del reto para la adquisición del Pensamiento Computacional (PC) de este trabajo está basado en bloques, y con él podrán adquirirse conceptos básicos de programación y se enriquecerán las capacidades de abstracción y de aplicación de la lógica.

3.1. Herramientas y tecnologías empleadas

Para la realización de este reto, se decidió optar por las tecnologías web, para que tras un despliegue de la aplicación, esta pudiese ser accesible a cualquier persona.

3.1.1. Blockly

Blockly [15] es una biblioteca de código abierto perteneciente a Google [17], que se utiliza para crear lenguajes con bloques visuales. Comenzó siendo diseñada para Javascript pero debido a la gran demanda de la misma se ha ido adaptando a una gran cantidad de lenguajes. Blockly fue creado en el año 2011, su lanzamiento se produjo en 2012 y puede generar código Javascript, Python, PHP o Dart. Esta funciona desde el lado del cliente, y añade un editor a la aplicación donde puedes intercalar bloques que representan instrucciones, por tanto, estos son equivalentes al código. Devuelve código sintácticamente correcto a elección del creador de la aplicación. Mientras, para el usuario consiste en arrastrar bloques, Blockly generará el código, y la aplicación hará algo con ese código.

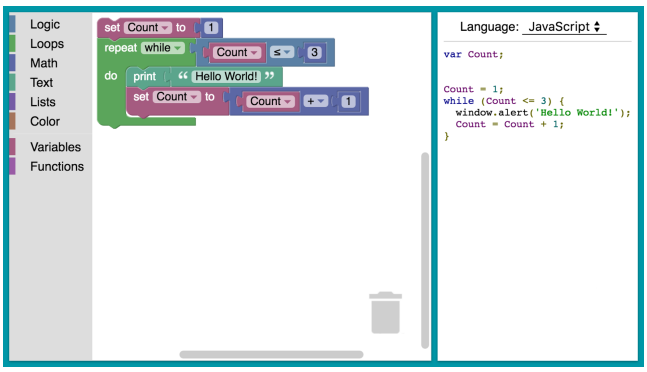


Figura 3.1: Apariencia de Blockly

Como puede verse en la Figura 3.1, lo que encuentra el usuario cuando hace uso de Blockly en una determinada aplicación, es un entorno bien diferenciado, como se explicó ya anteriormente con las otras herramientas. A la izquierda están a disposición del usuario los bloques ofrecidos para completar el reto, ordenados por categoría (si procede), espacio que se denomina *'Caja de herramientas'* o *'Toolbox'*, y a la derecha el *'Espacio de trabajo'* o *'Workspace'*, donde estos pueden ser colocados para su posterior ejecución. Además, se puede observar que en la esquina inferior derecha hay una papelera donde introducir los bloques si ya no se necesitan. Esta opción y otras muchas más, son personalizables, de manera que se podrá crear un Espacio de trabajo a exigencia del programador.

Blockly es una biblioteca muy bien documentada, en la página oficial [16] puede encontrarse toda la documentación necesaria para añadirlo a tu aplicación, hacer que funcione, y aplicar cualquiera de las funcionalidades que este nos permite. Es importante tener en cuenta que Blockly no es un lenguaje de programación como tal, si no que es una biblioteca que nos permite crear un lenguaje de programación a través del uso de bloques.

Visualmente recuerda a Scratch, como vimos en el Capítulo 2.2.1, pero realmente Blockly se emplea para crear un lenguaje propio, haciendo uso de los bloques, mientras que Scratch es un lenguaje de programación visual, que nos permite programar algo en concreto sin tener que emplear código de manera explícita. Por tanto ambas herramientas tienen un uso bien diferenciado, y no deben confundirse.

Esta biblioteca es cada vez más conocida y usada en grandes proyectos. Actualmente se emplea en algunos conocidos como son *'App Inventor'* del MIT, para crear aplicaciones para Android, *'Blockly Games'*, que es un conjunto de juegos educativos para enseñar conceptos de programación que también pertenece a Google, o Code.org, plataforma que ya se ha mencionado anteriormente.

Hasta ahora se han hecho simples referencias a los bloques, por ello se procede a detallar un poco más su uso. Blockly ofrece un gran número de bloques que ya están diseñados y representan porciones de código que vienen por defecto preestablecidas (véase la Figura 3.2).

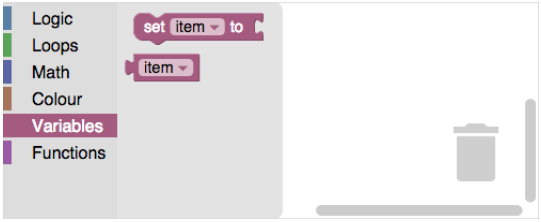


Figura 3.2: Categorías de la Caja de Herramientas

Sin embargo, cuando se pretende crear una aplicación que cumpla unas características determinadas, lo más probable es que los bloques ofrecidos por Blockly no cumplan los requisitos, y se necesiten bloques extras. Estos son representados en ficheros con formato .json o .js, y siguen una nomenclatura concreta, que puede modificarse manualmente. Esta tarea, sin embargo, no es tan sencilla dada la alta complejidad que puede adquirir un bloque desde su diseño. Por tanto, Blockly ofrece una herramienta de desarrollo para crear nuevos bloques, **Blockly Developer Tools**. En ella, el proceso de crear o modificar un bloque es una tarea mucho más sencilla.

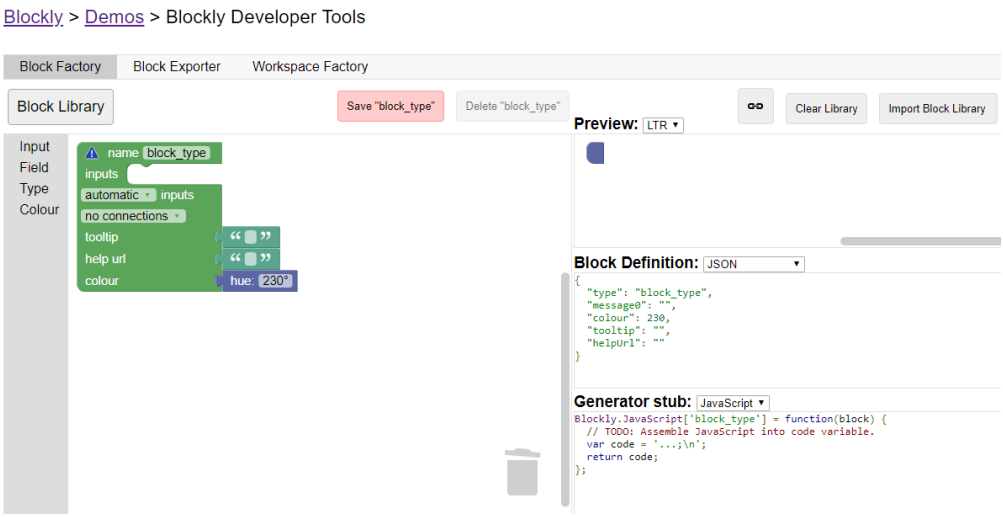


Figura 3.3: Blockly Developer Tools

Como puede verse en la Figura 3.3, en la columna izquierda se encuentra un espacio donde poder enlazar unos bloques con otros, añadiendo las características que se buscan en el bloque. Al lado derecho, se podrá previsualizar el resultado final del bloque a medida que se ven modificadas las características.

Además, se deberá copiar la definición del bloque que se ofrece, ya sea en formato .json o .js. Por último, se dispondrá también del generador del bloque, donde se añadirá el código que será ejecutado con cada instancia de ese bloque.

3.1.2. Javascript, HTML y CSS

Como lenguajes para el desarrollo web de la aplicación, se han empleado Javascript (JS), HTML y CSS (Figura 3.4). HTML es un lenguaje de marcas, empleado para formar la estructura DOM de la web. Como hojas de estilo se ha empleado CSS, realizando las tareas de presentación de la página. Y por último, como lenguaje interpretado se encuentra Javascript, encargado de dar toda la funcionalidad a la herramienta.

Figura 3.4: Logo HTML, JavaScript y CSS

3.1.3. jQuery

El lema de jQuery [21] (Figura 3.5) cita *'Write less, do more'*, de manera que esta biblioteca multiplataforma de código abierto de Javascript, nos permite simplificar la interacción con el DOM y manejar todo tipo de eventos de forma más sencilla. Fue presentada en 2006 y tras un análisis se ha coronado como la biblioteca de JS más utilizada del mundo [30].

El uso de jQuery simplifica y hace más legible el código. Un ejemplo de uso de dicha librería puede ser el siguiente:

```
document.getElementById("glass");
```

Y utilizando jQuery resulta en lo siguiente:

```
$('#glass')
```

Como puede observarse, la utilización de esta biblioteca simplifica bastante la tarea de desarrollo con JavaScript, por ello en este trabajo es una herramienta muy útil.

Figura 3.5: Logo jQuery

3.1.4. JS-Interpreter

Muchas de las funcionalidades que proporciona Blockly son posibles gracias a las aportaciones que vuelca JS-Interpreter [12] sobre el mismo. Esta biblioteca de JS fue diseñada de manera independiente de Blockly, sin embargo se creó precisamente para Blockly. El intérprete permite ejecutar los bloques de forma ordenada, y determinar la velocidad de ejecución. Además, permite poder realizar una ejecución paso a paso, además de poder pausarla en caso de que se requiera.

Esta biblioteca ha sido realmente útil en este proyecto debido a que realiza las labores de iluminación del bloque que esté ejecutándose, obligando a estos a ejecutarse uno a uno. Este requisito es indispensable para una aplicación dirigida al entrenamiento del PC en edades tempranas, ya que sirve de guía para seguir el procedimiento.

Un ejemplo de uso de esta biblioteca puede ser el que se muestra a continuación para ejecutar los bloques [7]:

```
function nextStep() {
  if (myInterpreter.step()) {
     window.setTimeout(nextStep, 10);
 }
}
nextStep();
```

Una instancia del intérprete podría crearse de la siguiente manera:

```
var myInterpreter = new Interpreter(code,initApi);
```

Siendo *code* el código del bloque a ejecutar e *initApi* la API donde estarán definidos todos aquellos bloques que interactúan con el espacio distinto a Blockly.

3.1.5. Nodejs y Npm

Node.js [11] (Figura 3.6) es un entorno Javascript en tiempo de ejecución multiplataforma, que trabaja en el lado del servidor utilizando un modelo asín-

crono y dirigido por eventos. Es de código abierto y fue creado en 2009. Node.js es bastante ligero a la vez que potente, ya que es capaz de mantener gran cantidad de conexiones abiertas. En la sección 3.3 de este capítulo, se mencionará más a fondo las funciones de estas bibliotecas utilizadas en la herramienta desarrollada.

Node.js trabaja utilizando paquetes que pueden instalarse de manera local en un proyecto o de manera global. Cuando se instala un paquete de forma local se añade automáticamente al directorio `node_modules`, y esto es gracias a **npm**.

Node Package Manager [24] es un gestor de paquetes, que facilita la tarea de trabajar con Node.js, ya que este nos permite obtener cualquier biblioteca de manera muy sencilla, y además administra los módulos y agrega las dependencias.

Figura 3.6: Logo Node.js

La nomenclatura establecida para descargar un determinado paquete tras haber realizado la instalación de Node.js es la siguiente:

```
npm install nombre_paquete@version
```

Aunque realmente indicar la versión es opcional.

3.1.6. Pure

Pure [19] (Figura 3.7) es un framework CSS desarrollado por Yahoo. Es una librería bastante ligera, legible y escalable. Es muy útil para construir una web desde cero, aportando bastantes posibilidades para el diseño de la estructura. Siendo el más apropiado para aplicaciones sencillas y minimalistas.

Figura 3.7: Logo Pure

La funcionalidad principal que se ha obtenido de Pure en este proyecto ha sido el diseño responsivo. Adaptando a la web a cualquier tamaño de pantalla, desde un móvil hasta grandes monitores.

3.1.7. Github y Overleaf

Github [14] (Figura 3.8) y Overleaf [25] (Figura 3.9) no podían faltar en la redacción de esta memoria debido a la importancia que han cobrado durante la realización de este trabajo como herramientas de soporte.

Github es la herramienta por excelencia para llevar un control de versiones de los proyectos en ejecución que podremos almacenar en repositorios. Está basada en **Git** [13] y ayuda a llevar un registro de los cambios ejecutados en el código fuente, permitiendo así conservar de manera segura el proyecto. Admite también crear copias de código mediante una bifurcación del mismo, denominando a estas *ramas*, por tanto se podrá trabajar en estas sin afectar al trabajo realizado hasta el momento.

Figura 3.8: Logo Github

Por otro lado, el editor online **Overleaf** permite la edición de artículos, documentos, publicaciones científicas, etc. escritas utilizando LaTeX [34], también de manera colaborativa. La gran ventaja que ofrece esta plataforma es la previsualización en tiempo real y de manera automática de la compilación del proyecto, de esta manera se puede ver el resultado en formato .pdf antes incluso de exportarlo.

Figura 3.9: Logo Overleaf

3.2. Primeros Pasos

Para comenzar con el desarrollo del reto de entrenamiento, fue necesario plantear desde un comienzo cuales eran los objetivos que este debía cumplir y además en que dinámica iba a basarse. Como ya se mencionó anteriormente, teniendo en cuenta que Code.org es una de las plataformas más importantes que trabajan para la adquisición del PC, este proyecto se ha fundamentado en uno de los cursos impartidos por Code.org, denominado *My Robotic Friends* [10] (Figura 3.10).

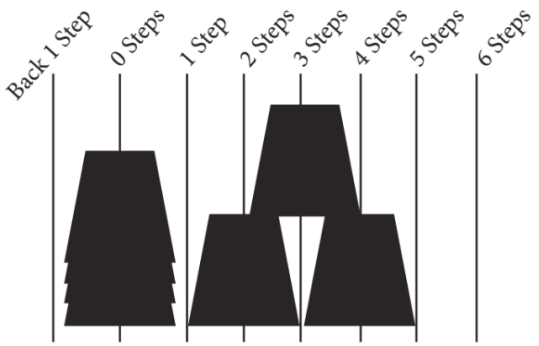


Figura 3.10: Actividad 'My Robotic Friends'

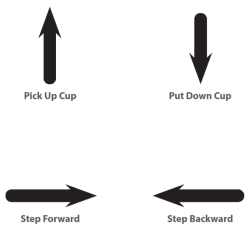


Figura 3.11: Instrucciones

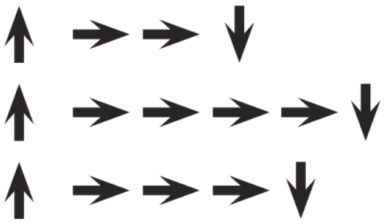


Figura 3.12: Ejemplo de secuencia

Como se observa en las Figuras 3.11 y 3.12, las instrucciones de este reto pretenden que el usuario las utilice en secuencia a medida que las vaya necesitando. Se quiere realizar una simulación en la que un brazo robot sujeta un vaso cuando se hace uso de la instrucción *'Pick up cup'*, y se moverá a derecha o izquierda tantas veces como las instrucciones correspondientes se repitan. Por último, el brazo soltará el vaso en cuanto se haga uso de la instrucción *'Put down cup'*, y se tendrá que indicar al robot que debe volver a la pila de vasos para continuar con otro vaso o finalizar el reto.

3.3. Creación de un reto para el entrenamiento del Pensamiento Computacional

En los siguientes puntos se desarrollará el procedimiento seguido durante la creación del reto para el entrenamiento del Pensamiento Computacional, al cual se ha denominado **'Glass'**:

3.3.1. Interfaz

La interfaz del reto es sencilla e intuitiva, diseñada así especialmente para el público al que va destinada (véase la Figura 3.13). En ella podemos encontrar en la parte superior:

- **Título** de la herramienta.
- **Niveles de dificultad** enumerados del 1 al 6.
- **Nivel** en el que se encuentra el usuario (Nivel 1 por defecto antes de comenzar el reto).
- **Descripción del reto** a alcanzar en dicho nivel de dificultad.
- Imagen con el **objetivo** que se debe alcanzar haciendo uso de los bloques, para mover los vasos.

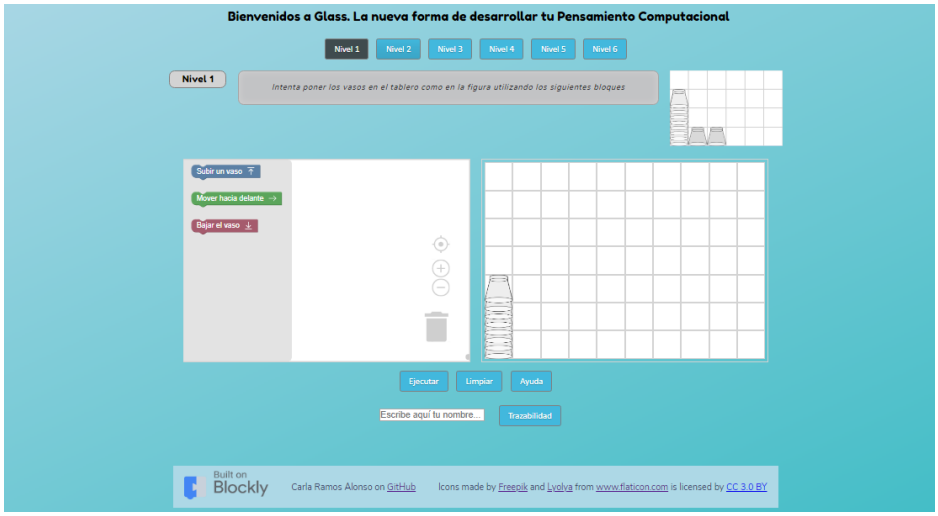


Figura 3.13: 'Interfaz de Glass'

En la segunda sección diferenciada de la interfaz, se encuentra ubicado el **espacio de trabajo de Blockly**, al lado izquierdo, y el **tablero** donde se encuentra la pila de vasos, a la derecha. Además en la parte inferior de ambos espacios, encontramos 4 botones. Los tres primeros tienen la funcionalidad siguiente:

- **Ejecutar:** Ejecutará el código que se encuentre en ese momento en el espacio de trabajo. El resultado se verá en el tablero, ejecutándose cada bloque paso a paso, de manera que se visualiza todo el proceso.
- **Limpiar:** Da la posibilidad al usuario de eliminar los bloques del espacio de trabajo y los vasos del tablero si los hubiera.
- **Ayuda:** Ofrece al usuario vídeos de ayuda con una breve explicación de los pasos a dar para poder superar el nivel. Cada nivel consta de un vídeo diferente.

El último botón, corresponde a la obtención de la trazabilidad que se buscaba tras la realización de este reto. Introduciendo el nombre de usuario en el campo

de texto, y haciendo click sobre el botón, se descarga automáticamente un fichero de texto con cada uno de los bloques y acciones que ha aplicado el usuario sobre el espacio de trabajo. Esta funcionalidad se desarrollará con más detalle en el Capítulo 4.

Por último, en el **pie de página** figuran:

- Logo de atribución a Blockly
- Nombre de la autora y enlace al Github del proyecto
- Atribución a los iconos utilizados en la realización del reto, con enlace a la fuente.

Para establecer la estructura de la aplicación, se ha hecho uso de la herramienta ya mencionada en el punto 3.1.6, Pure.css. Gracias a este framework, se establecieron las filas y columnas de la aplicación, de manera que tiene implantada un diseño responsivo, que se adaptará al tamaño de pantalla en la que sea visualizada.

Se definieron las filas asignando al contenedor o div HTML la clase `<div class='pure-g'>`, y dentro de cada fila, se dividieron las columnas asignando la clase `<div class='pure-u-lg-1-2 pure-u-md-1-2'>` a cada contenedor de la fila, siendo este ejemplo el de una clase correspondiente a una fila de dos columnas. La nomenclatura de las clases se establece dependiendo del tamaño de la pantalla, Pure.css aplicará el diseño correspondiente tras evaluar el tamaño de la misma. Esta herramienta consta de 5 filas diseñadas con Pure.css.

La imagen de fondo fue obtenida de la web 'Pexels' [1], la imagen del vaso encontrada en YA-webdesign [2] y los iconos insertados en los bloques y en la pestaña de la página descargados de Flaticon [29].

3.3.2. Ejecución y colisión de los vasos

Las animaciones e interacción de la aplicación han sido posible gracias a la utilización de Javascript y Node.js. Para realizar una explicación más exacta de los procedimientos llevados a cabo, se detallará lo programado en cada fichero .js del proyecto.

Los bloques creados gracias a la herramienta de Blockly, Developer Tools mencionada en puntos anteriores, se encuentran en el fichero denominado *blocks.js*. En la definición de cada bloque (véase un ejemplo en la Figura 3.14), se llama además a la función que contiene el código que ejecutará el bloque.

```
    Blockly.Blocks['step_forward'] = {
  init: function() {
    this.appendDummyInput()
        .appendField("Mover hacia delante")
        .appendField(new Blockly.FieldImage("images/right-arrow.png", 15, 15, " "));
    this.setPreviousStatement(true, null);
    this.setNextStatement(true, null);
    this.setColour(120);
 this.setTooltip('Utiliza este bloque si quieres mover el vaso hacia la derecha');
 this.setHelpUrl("");
  }
};

Blockly.JavaScript['step_forward'] = function(block) {

  var code = 'step_forward()\n';

    return code;
  };
```

Figura 3.14: Bloque 'Mover hacia delante'

Se continúa con la definición del fichero *script.js*, el cual contiene las funciones mencionadas anteriormente, las cuales ejecutan los bloques. Además, este fichero consta de la declaración de la clase 'Glass' (Figura 3.15), la cual tiene los atributos relativos a la posición del vaso, su tamaño y si está rotado o no.

```
// Glass Class
class Glass {
    constructor(n) {
        this.n = n, // number of glass
        this.y = parseInt($("#fixed").css("bottom")); // glass vertical axis position
        this.x = parseInt($("#fixed").css("left")); // glass horizontal axis position
        this.w = parseInt($("#fixed").css("width"));
        this.h = parseInt($("#fixed").css("height"));
        this.neighbors = []; // neighbors positioned below the glass
        this.rotate = false;
    }
}
```

Figura 3.15: Clase 'Glass'

Para realizar la animación de los vasos se ha utilizado el método **.animate()** [20] de jQuery, modificando las propiedades del CSS. Además, se almacena un nuevo objeto en un vector de vasos cada vez que el usuario sube un vaso con el bloque definido para ello, de manera que tras sacar un vaso nuevo de la pila, se trabajará siempre con ese hasta tomar otro o ejecutar el código.

En relación a las colisiones, la función denominada *collition(g, mov)* recibe el objeto vaso que se está moviendo, y el movimiento que va a realizar en el

siguiente paso. Dependiendo de estos datos la colisión se determinará con una condición distinta. En caso de que la colisión se vaya a hacer efectiva, el vaso parará de moverse, de manera que nunca un vaso se introducirá dentro de otro. Esta función compara la posición del vaso pasado por parámetro con las posiciones del resto de vasos del tablero, recorriendo el vector de vasos.

La definición del tablero, de la caja de herramientas con los bloques, la función para la iluminación de los bloques y el ya mencionado intérprete de JS se encuentran en el fichero *workspace.js*.

Al cambiar de nivel, se cambia el número del nivel actual, la descripción y objetivo del mismo, y además se modifican los bloques necesarios para ese nivel. El fichero *levels-control.js* maneja los cambios de nivel, y los botones, dejando en otro color el del nivel seleccionado. Y además, para la comprobación del nivel, se dispone de seis ficheros con la nomenclatura `levelX.js` siendo `X` el nivel correspondiente. En cada uno de ellos se comprueba si se cumple con el objetivo siguiendo los requisitos que se describen a continuación:

- Se comprueba si en el tablero hay el mismo número de vasos que exige el ejercicio, si no, retorna un mensaje informando al usuario.
- La posición de los vasos debe ser la misma que se muestra en la imagen objetivo, si no lo es, se informa al usuario.
- Además de la posición, se comprueba también en aquellos niveles que así lo requieren, el uso de los bucles y condicionales propuestos. Si no se utilizan, se informa al usuario.
- En caso de que se cumplan los requisitos anteriores, se informa al usuario del éxito al superar el ejercicio, y se le propone continuar con el siguiente nivel.

Por último, se implementa la trazabilidad del reto en *trace.js*, funcionalidad que se detalla en el Capítulo 4.

3.3.3. Obtención de los datos

Debido al cambio de una cantidad considerable de información cada vez que se avanza de nivel, se optó por insertar estos datos de forma dinámica en el fichero *index.html*, de manera que así, la información estática de la página sea únicamente aquella que es invariable para todos los niveles. Se ha creado un servidor utilizando las funcionalidades de Node.js, en el fichero contenido en la carpeta raíz del proyecto, denominado *server.js*. En él se realizan las peticiones HTTP al servidor, para obtener los archivos de manera dinámica.

```
$.getJSON("/levels/level"+level_number+".json", function( data ) {
        $('#level-title').html(data.definition[0].title);
        $('#level-description').html(data.definition[0].description);
        $('#image-container').html(data.definition[0].image);
        resize();
    });
```

Figura 3.16: Obtener datos de nivel

```
$.getJSON("/levels/level"+level_number+".json", function( data ){
    for(var i = 0; i<data.definition[1].blocks.length; i++){
        xml_structure = xml_structure+'<block type=
        "'+data.definition[1].blocks[i]+'"></block>'
    }
});
return xml_structure+'</xml>';
```

Figura 3.17: Obtener datos de los bloques

Los ficheros .json que contienen los datos relativos a cada nivel, se encuentran alojados en la carpeta denominada *levels* (Figuras 3.16 y 3.17). La estructura que siguen estos .json, se muestra en la Figura 3.18.

```
{
    "definition": [
        {
            "title": "<p>Nivel 2</p>",
            "description": "<p>Ya sabes colocar los vasos, ¡ahora vamos
            a aprender a rotarlos!</p>",
            "success": "¡Perfecto! ¿Lo complicamos?",
            "wrong_position": "¿Seguro que los vasos están colocados donde deben?",
            "wrong_glass_number": "¿Seguro que estás usando los vasos necesarios?",
            "image": "<img id='level-goal-image' src='/images/level2.png' alt=''>",
            "help":"Coloca el bloque de rotación antes o después de bajar el vaso
            que quieras rotar"
        },
        {
        "blocks": ["pick_up","step_forward","put_down","rotate"]
        },
        {
            "identifier": "2"
        }
    ]

}
```

Figura 3.18: Definición del nivel 2

Los datos de estos ficheros se obtienen como se ha mostrado en las Figuras 3.16 y 3.17. Además de recoger toda la información como título, descripción, y los mensajes que se muestran al usuario dependiendo del éxito o error al resolver un nivel, se obtienen también:

- Un mensaje de ayuda, que se muestra junto con un vídeo explicativo cuando el usuario pulsa el botón de 'Ayuda'. Los vídeos mencionados se encuentran alojados en la carpeta llamada *videos*.

- Un array con los bloques, que se colocarán de forma dinámica en el Toolbox cada vez que se cambie de nivel.

3.3.4. Despliegue de la aplicación

Para realizar el despliegue de la aplicación se ha optado por el uso de la plataforma gratuita de alojamiento **Heroku** [28]. Heroku es una plataforma como servicio (PaaS), en la que ofrecen alojamiento gratuito para tu aplicación, junto con el mantenimiento de los servidores y configuración de los mismos. Los usuarios de este servicio solo deben preocuparse por el mantenimiento y actualización de sus herramientas. Heroku ofrece también el dominio al desarrollador, pudiendo a partir de ese momento compartir su aplicación con cualquiera que disponga del mismo.

Para llevar a cabo el despliegue, se debe comprobar antes la correcta instalación de Node.js, Npm y Git. Tras ello, se puede proceder a instalar lo siguiente:

```
$ npm install -g heroku
```

A continuación se debe crear una cuenta en Heroku. Después, podrá realizarse la conexión a esa cuenta a través de la línea de comandos en el repositorio donde se ha realizado la instalación:

```
$ heroku login
```

Y para terminar, se debe crear una aplicación Heroku e indicar que nuestro proyecto está basado en Node.js, para después poder realizar el push a este repositorio remoto:

```
$ heroku create
$ heroku buildpacks:set heroku/nodejs
$ git push heroku master
```

Destacar que para que esto funcione, Heroku buscará los ficheros *package.json* y *package-lock.json* en el repositorio raíz del proyecto. Además se añadirá también al mismo nivel un fichero llamado *Procfile* con la información siguiente: `web: node server.js`. En el caso de este proyecto, para poder cargar la pá-

gina en un navegador web se debe ejecutar el servidor, por tanto se indica a Heroku este funcionamiento.

Tras la realización de estos pasos la plataforma ofrecerá el dominio donde poder encontrar la aplicación web. Siempre que se desee realizar un cambio en la misma se deberá realizar la actualización (`push`) del repositorio remoto, habiendo guardado previamente los cambios en el repositorio y realizado la confirmación (`commit`).

3.3.5. Control de versiones

Para llevar un control de versiones del proyecto, se ha alojado en la plataforma ya descrita, Github (Figura 3.19). En ella se almacena el código íntegro de la aplicación así como el historial de versiones y cambios [5].

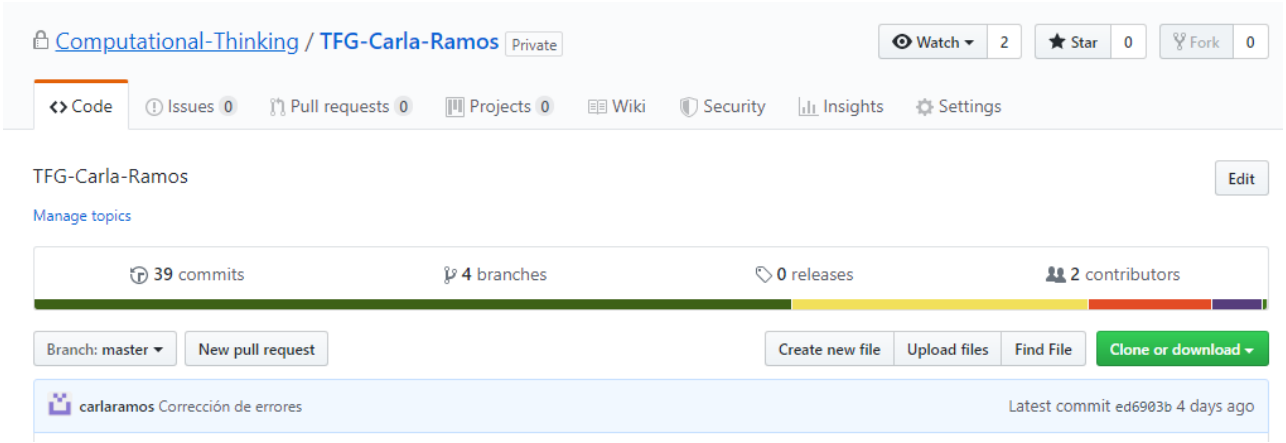


Figura 3.19: Repositorio de Github

Capítulo 4

Análisis de los resultados

Como se estableció desde un comienzo, y siguiendo el título de este proyecto, se realiza una trazabilidad del reto. Los motivos por los cuales se ha decidido seguir los pasos del usuario durante la realización del reto de entrenamiento, es porque esto podría ayudar a mejorar el planteamiento del reto, y además se podrá analizar cuáles han sido las dificultades con las que se han encontrado los usuarios.

4.1. Tecnologías Empleadas

Blockly es una biblioteca muy completa, en la cual no podía faltar la posibilidad de manejar el espacio de trabajo ('Workspace'). Este framework ofrece un gran número de métodos para trabajar con el espacio de trabajo [6], entre los cuales se destacan los empleados para la realización de la trazabilidad.

4.1.1. Workspace.addChangeListener()

El método .addChangeListener() detecta cualquier tipo de cambio en el Workspace, desde un bloque que está siendo arrastrado al mismo, o un nuevo bloque que ha sido colocado, eliminado, o cambiado de sitio. Esta función permite que cada vez que el usuario realice un movimiento sobre el espacio de trabajo, se llame a otra función que nos retorna los bloques que están siendo utilizados en ese momento, como se verá a continuación.

4.1.2. Workspace.getAllBlocks()

Este método retorna todos los bloques que se encuentran en el Workspace en el momento de la llamada al mismo. Como se mencionó en el punto inmediatamente anterior, se aprovechó la función detectora de cambios para llamar a .getAllBlocks(), para almacenar todos y cada uno de los cambios realizados. El resultado que devuelve tras su ejecución es un array que almacena de manera

ordenada (de izquierda a derecha y de arriba a abajo) los bloques que detecta en el espacio de trabajo. Sin embargo, solo se almacenará el campo *'type'* de cada bloque, en el cual se indica el nombre del mismo, el resto de información que contiene un bloque es irrelevante para los objetivos actuales de este proyecto.

4.2. Obtención de los resultados

El proceso llevado a cabo para la obtención de los resultados ha consistido en la utilización conjunta de los métodos explicados en el apartado 4.1. En el fichero de definición del Workspace se añade el método para detectar los cambios:

```
workspace.addChangeListener(function(event){
 update_file(null);
});
```

Dentro de la función denominada `update_file(null)`, se encuentra la llamada al método para obtener todos los bloques. En un array se irá introduciendo el nivel actual, el cual se actualizará cada vez que el usuario cambie de nivel, y a medida que se vayan añadiendo bloques al espacio de trabajo, se introducirán en el mismo, vectores con los bloques actualmente en el Workspace. Solo se añadirán a este array en el caso de que la nueva composición de bloques sea distinta a los introducidos en la llamada inmediatamente anterior. Además, cada vez que se pulse sobre el botón de *'Ejecutar'*, se añadirá al array si se ha terminado el nivel con éxito o si debe volver a intentarse.

4.3. Informe de acciones

Se deduce tras la lectura de los puntos anteriores, que se obtendrá un informe de acciones con los resultados de los bloques empleados, en cada nivel donde se haya intentado, y además si se ha ejecutado con éxito o no cierta secuencia de bloques. Debido a la gran complejidad que hubiese supuesto la creación de una base de datos para almacenar la trazabilidad, requiriendo con ello un control de acceso para los usuarios, además del almacenamiento de resultados por nivel, se acordó guardar los mismos en formato de texto plano, y obtenerse mediante la descarga de un fichero con extensión *.txt* [27]. Si el usuario introduce su nombre en el campo de texto que figura junto al botón de 'Trazabilidad', se descargará el fichero con el mismo como nombre de archivo, si por el contrario no lo introduce, se descargará con un nombre por defecto.

Dicho informe, tiene la estructura que se muestra en la Figura 4.1.

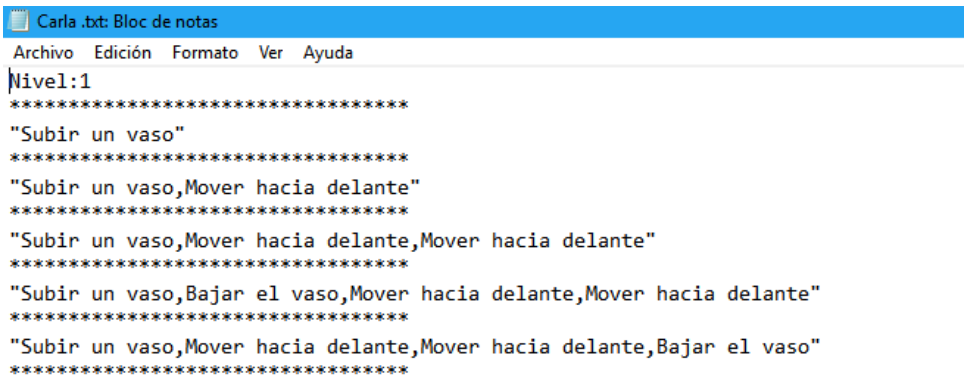


Figura 4.1: Informe de acciones

En algunos casos, se detecta también que un bloque esté siendo movido por encima de la secuencia de bloques ya colocada en el espacio de trabajo, es por ello que se ve modificada la secuencia en ciertos puntos. Este hecho no origina ningún inconveniente, debido a que la secuencia final se muestra en la línea posterior.

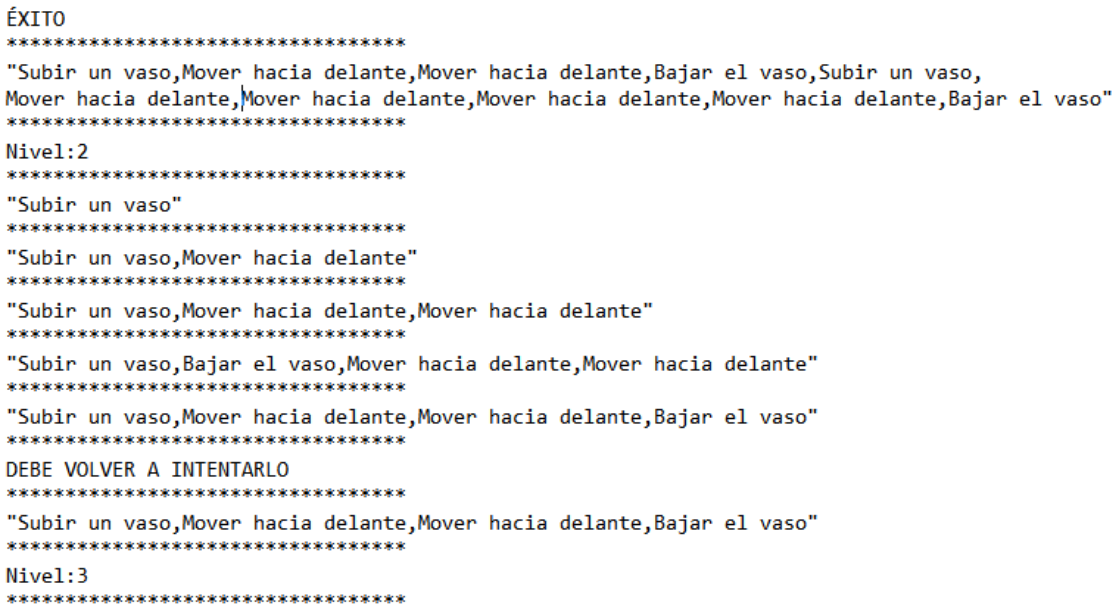


Figura 4.2: Informe de acciones

En los ejemplos que se visualizan en las Figuras 4.1 y 4.2 se puede observar un resultado correcto ante la realización del Nivel 1, siendo la secuencia tras la palabra 'ÉXITO' la que ha originado este resultado. Sin embargo, al intentarlo con el Nivel 2, no se cumple con el objetivo y se muestra el mensaje 'DEBE VOLVER A INTENTARLO'. Además se observa también un cambio de nivel al 3, en el cual no se ha producido ningún intento.

Capítulo 5

Modo de uso de 'Glass'

Tras detallar el procedimiento llevado a cabo para su realización, se presenta el reto para el entrenamiento del Pensamiento Computacional, que se ha denominado **Glass** [4].

Al acceder a la aplicación, la primera ventana que se muestra es la del Nivel 1. El funcionamiento consistirá siempre en empezar con 'Subir un vaso', y terminar los movimientos a realizar con ese vaso antes de utilizar 'Bajar el vaso'. Tras tomar un vaso nuevo, cualquier bloque utilizado tras ese bloque, afectarán a ese vaso en cuestión. Una vez el vaso caiga, se debe 'Subir un vaso' o 'Ejecutar' el ejercicio para su comprobación. Los vasos no deben manejarse, solo podrán moverse utilizando los bloques. Este reto consta de 6 niveles distintos:

- **Nivel 1:** En este nivel se plantea al usuario comenzar a entender la dinámica del reto. Es importante que se comprenda desde un principio que al emplear el bloque 'Mover hacia delante' el bloque sólo se moverá media posición en el tablero, de manera que se deberá utilizar este dos veces para que efectúe un movimiento completo. Además, se podrá comprender en este nivel que al subir o bajar un vaso, estos ejecutan el movimiento completo hasta el borde superior o inferior del tablero, respectivamente. El objetivo del nivel debe resolverse con la utilización de varias instancias de los bloques que se relacionan en la Figura 5.1.

Figura 5.1: Bloques de movimiento

- **Nivel 2:** Se aumenta en este nivel la dificultad del reto, añadiendo otro bloque a la caja de herramientas (Figura 5.2). Con este cuarto bloque se podrá girar un vaso cuando sea necesario. El giro podrá realizarse antes de bajar el vaso, o tras la bajada del mismo.

Figura 5.2: Bloque rotación

- **Nivel 3:** A este nivel le corresponden los mismos bloques que al Nivel 1, pero aunque esto sea así, la dificultad se ve incrementada ya que uno de los vasos deberá ir colocado encima de otros dos, empleando la cantidad correcta de tres bloques 'Mover hacia delante' en este último vaso a colocar (Figura 5.3).

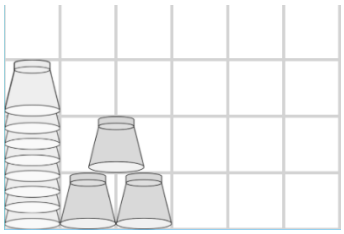


Figura 5.3: Objetivo del Nivel 3

- **Nivel 4:** Este nivel tiene un objetivo similar al del nivel anterior (Figura 5.4), con la salvedad de que hay que añadir un bloque 'Rotar vaso 180º' antes de bajar el último vaso. Además, se añade un bloque que representa a un bucle (Figura 5.5), en el interior del cual se pueden colocar aquellos bloques que se quieran repetir, e indicar cuantas veces se debe repetir dicha instrucción o conjunto de ellas. Por tanto, se valorará que el usuario haga uso de estos bucles para evitar repetir demasiadas veces de manera consecutiva el bloque de 'Mover hacia delante'.

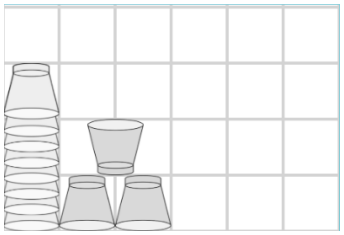


Figura 5.4: Objetivo del Nivel 4

Figura 5.5: Bloque de repetición

- **Nivel 5:** Debido a la mayor complejidad de este nivel (Figura 5.6), el usuario se verá obligado a utilizar varias veces el bloque 'Repetir X veces' (Figura 5.5), ya que se podrán colocar los dos vasos de la columna izquierda con las mismas instrucciones, introduciendo los bloques necesarios dentro del bucle. Se deberá tener en cuenta que el vaso superior está rotado, mientras que el inferior no, de manera que habrá que colocar la instrucción de rotación fuera del bucle, de manera que actúe solamente con el último vaso. Para la segunda columna

el procedimiento es el mismo, teniendo en cuenta que habrá que aumentar el número de repeticiones en el bucle que alberga 'Mover hacia delante'. Destacar que los bucles podrán contener más bucles dentro de ellos.

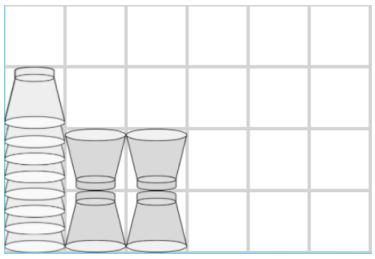


Figura 5.6: Objetivo del Nivel 5

- **Nivel 6:** En este último nivel, se añade la funcionalidad del bloque condicional (Figura 5.7). Este deberá ser utilizado para rotar el vaso. El vaso puede sin embargo rotarse sin hacer uso de este bloque, pero el usuario obtendrá un mensaje que indique que no es la solución óptima en caso de no utilizarlo. Se deberá indicar si el vaso con el que colisiona está del revés o no, y el único bloque que se podrá utilizar para colocar dentro será el de 'Rotar vaso 180º'

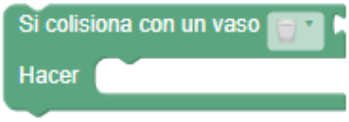


Figura 5.7: Bloque condicional

Para facilitar la realización del reto, en cada nivel el usuario tendrá la posibilidad de pulsar sobre el botón de 'Ayuda', que mostrará un vídeo explicativo con algunas indicaciones para poder resolver el nivel sin inconvenientes.

Capítulo 6

Conclusiones y líneas futuras

El Pensamiento Computacional (PC) es hoy en día definitivamente parte esencial de nuestra sociedad. Aumentan cada vez más las investigaciones dedicadas a descubrir los grandes beneficios de este razonamiento y generalmente, los resultados apuntan a que el PC será parte indispensable de nuestra educación en muy poco tiempo. La implantación temprana del estudio y aprendizaje de esta ciencia es trabajo de todos, debiendo mostrar al mundo lo importante que es aplicar el pensamiento lógico, crítico, y la resolución de problemas de forma óptima aplicando habilidades de abstracción.

Ya dijo Wing [36] que debemos inspirar el interés de la gente en la aventura intelectual de esta ciencia. Por ello debemos propagar la admiración y el poder de la computación, animando a hacer del PC un lugar común. La computación será por tanto una realidad cuando deje de ser una filosofía explícita y la palabra algoritmo forme parte de nuestro vocabulario.

Aunque en este proyecto el reto vaya dirigido a edades tempranas de aprendizaje, nunca es tarde para interesarse por los nuevos avances y formarse, de manera que con este trabajo se anima a cualquier persona que tenga curiosidad por el mundo de la computación, a entrenar su inteligencia hacia el PC.

Destacar que aunque se haya escogido la filosofía de juego basado en bloques para la realización de este reto, cualquier otra es igual de válida y útil para el entrenamiento. Especificando que la gran ventaja de un reto como el propuesto es la posibilidad de obtener una trazabilidad para el posterior análisis y mejora del reto.

Finalmente, cabe mencionar que los conocimientos requeridos para el desarrollo de este proyecto, han sido adquiridos durante los estudios del Grado en Ingeniería Informática, aplicando las nociones alcanzadas en asignaturas como

'Sistemas y Tecnologías Web', *'Usabilidad y Accesibilidad'* y *'Desarrollo de Sistemas Informáticos'* para la creación de la herramienta. También *'Lenguajes y Paradigmas de la Programación'*, como asignatura impulsora del PC en el Grado junto con la utilización del Control de Versiones con Git [13].

Tras la finalización de este proyecto, se quedan sin poner en marcha distintas ideas y propuestas recogidas durante la realización del mismo. Es por ello, que para su mejora, se exponen los siguientes puntos:

- Ampliación del rango de aprendizaje con la incorporación de más niveles de dificultad, en los que entren en juego un mayor número de vasos y bloques. Además se podrá añadir en estos últimos niveles, más complejidad a la hora de implantar los conceptos relativos a la computación, aplicando ideas como recursividad, polimorfismo, distintos tipos de bucle, etc.

- Crear un sistema de control de usuarios, de manera que para poder acceder se necesite registrarse, para así poder llevar un control de los datos del usuario para su posterior análisis. Los datos requeridos que realmente son importantes para la mejora del proyecto son la edad, estudios y país de residencia, por ejemplo.

- Mejora de la trazabilidad, albergando los datos en una base de datos. En ella se podría almacenar la información del usuario, relacionándola con los resultados del reto. Estos resultados se podrían clasificar a su vez por niveles para una mejor organización para el posterior análisis.

- Llevar a cabo un análisis exhaustivo de los datos obtenidos de la trazabilidad, para extraer información útil y luego poder aplicar las mejoras que el reto requiera. Estos datos podrían ser representados mediante tablas, diagramas o con distintos modelos de representaciones gráficas.

- Realizar un despliegue continuo de la aplicación, de manera que no se deba realizar un gran número de acciones para que se actualicen en el servidor las modificaciones.

Capítulo 7

Summary and Conclusions

Nowadays Computational Thinking (CT) is an essential part of our society. Research is increasing to discover the great benefits of this reasoning, and generally, results suggest that CT will be an essential part of education in a very short period. Early training of this science is everyone's work, we must show people how important is to apply logical and critical thinking, and problem solving applying abstraction abilities.

Wing [36] said we should look to inspire the public's interest in the intellectual adventure of the field. We'll thus spread the joy, awe, and power of computer science, aiming to make Computational Thinking commonplace. CT will have become a reality when it disappears as an explicit philosopy, and *algorithm* is part of everyone's vocabulary.

Even though the challenge proposed in this project has been made for people at the young age, is never too late for being interested in progess and training, so with this project we aim people who is interested in computation, to train their intelligence towards Computational Thinking.

Finally, we decided to use blocks based games to train CT, but any other ideas or methods are accepted and useful for training. Specifying that the great advantage of a challenge like the one proposed in this project is the possibility of obtaining a traceability for analysing it and then improving the challenge.

Capítulo 8

Presupuesto

Este capítulo recoge un posible presupuesto tras finalizar con la realización de este proyecto.

Tareas	**Horas empleadas**	**Precio**
Revisión bibliográfica	25h	5€/h
Estudio antecedentes y estado actual del tema	25h	5€/h
Desarrollo de un reto de entrenamiento	120h	15€/h
Trazabilidad del reto	40h	15€/h
Infraestructura y despliegue	10h	10€/h
Pruebas del correcto funcionamiento	20h	8€/h
Creación de la documentación	20h	10€/h
TOTAL	**260h**	**3110€**

Tabla 8.1: Presupuesto de desarrollo

Como se puede observar en la Tabla 8.1, este proyecto está valorado con un precio de 3110€ en consecuencia de 260 horas de trabajo.

Bibliografía

[1] Pexels. https://www.pexels.com/es-es/.

[2] Ya-webdesign. https://ya-webdesign.com/editor.html?id=1693545.

[3] Inc. AgentSheets. Agentsheets. http://agentsheets.com/.

[4] Carla Ramos Alonso. Glass. https://glass-computationalthinking.herokuapp.com/.

[5] Carla Ramos Alonso. Pensamiento computacional: Trazabilidad de los retos de entrenamiento. https://github.com/Computational-Thinking/TFG-Carla-Ramos.git.

[6] Blockly. Blockly.workspace. https://developers.google.com/blockly/reference/js/Blockly.WorkspacegetAllBlocks.

[7] Blocky. Js-interpreter tutorial. https://developers.google.com/blockly/guides/app-integration/running-javascript.

[8] Code.org. Code.org. https://code.org/.

[9] Code.org. Hour of code. https://studio.code.org/hoc/1.

[10] Code.org. My robotic friends. https://curriculum.code.org/csf-1718/courseb/6/.

[11] Node.js Foundation. Nodejs. https://nodejs.org/es/.

[12] Neil Fraser. Js-interpreter. https://github.com/NeilFraser/JS-Interpreter.

[13] Git. Git. https://git-scm.com/.

[14] Inc Github. Github. https://github.com/.

[15] Google. Blockly. https://developers.google.com/blockly/.

[16] Google. Blockly tutorial. https://developers.google.com/blockly/guides/get-started/web.

[17] Google. Google. https://www.google.es/.

[18] Ting-Chia Hsu, Shao-Chen Chang, and Yu-Ting Hung. How to learn and how to teach computational thinking: Suggestions based on a review of the literature. *Computers Education*, 126:296 – 310, 2018.

[19] Present Yahoo! Inc. Pure.css. https://purecss.io/.

[20] jQuery. .animate(). https://api.jquery.com/animate/.

[21] The jQuery Foundation. jquery library. https://jquery.com/.

[22] Cagin Kazimoglu, Mary Kiernan, Liz Bacon, and Lachlan MacKinnon. Learning programming at the computational thinking level via digital game-play. *Proceedings of the International Conference on Computational Science, Iccs 2012*, 9:522–531, 2012.

[23] Lifelong Kindergarten. Scratch. https://scratch.mit.edu/.

[24] npm Enterprise. Npm. https://www.npmjs.com/.

[25] Overleaf. Overleaf. https://es.overleaf.com/.

[26] Nikolaos Pellas and Spyridon Vosinakis. The effect of simulation games on learning computer programming: A comparative study on high school students' learning performance by assessing computational problem-solving strategies. *Education and Information Technologies*, 23(6):2423–2452, Nov 2018.

[27] Roimergarcia. .txt download. http://jsfiddle.net/roimergarcia/XLqsf/.

[28] Salesforce.com. Heroku. https://www.heroku.com/.

[29] Freepik Company S.L. Flaticon. www.flaticon.com.

[30] World Wide Web Technolog Surveys. Most popular javascript libraries. https://w3techs.com/.

[31] Darwin González Suárez. Sistemas informáticos para la evaluación del entrenamiento del pensamiento computacional. https://riull.ull.es/xmlui/handle/915/8523.

[32] Carnegie Mellon University. Alice. http://www.alice.org/.

[33] Wikipedia. Cifrado césar wikipedia, la enciclopedia libre. https://es.wikipedia.org/w/index.php?title=Cifrado_C%C3%A9saroldid=115421389, 2019.

[34] Wikipedia. Latex wikipedia la enciclopedia libre. https://es.wikipedia.org/w/index.php?title=LaTeXoldid=116018974, 2019.

[35] J. Wing. Computational thinking and thinking about computing. In *2008 IEEE International Symposium on Parallel and Distributed Processing*, page 1, 2008. ID: 2.

[36] Jeannette M. Wing. Computational thinking. *Commun. ACM*, 49(3):33–35, March 2006.

[37] Eduardo SEGREDO y Gara MIRANDA y Coromoto LEÓN. Hacia la educación del futuro: El pensamiento computacional como mecanismo de aprendizaje generativo. *Education in the Knowledge Society (EKS)*, 18(2), 2017.

[38] Rafael Herrero Álvarez. Una aproximación al pensamiento computacional a través de la nutrición. http://riull.ull.es/xmlui/handle/915/5123.