

Damo's Quest como entorno gamificado para aprender a programar

IDE gamificado para la enseñanza de la programación destinado a niños y adolescentes

Emilio Ormeño¹, Cintia Ferrarini Oliver²

*1. Instituto de Informática, 2. Departamento de Informática,
Facultad de Ciencias Exactas, Físicas y Naturales, UNSJ
eormeno@gmail.com, ferrarinicintia@gmail.com*

RESUMEN

Damo's Quest es un entorno de programación gamificado, desarrollado en el Instituto de Informática con el objetivo de ofrecer una experiencia lúdico-educativa sobre programación, en la que los aprendices deben resolver retos en historias de fantasía mediante la generación de código. Esta propuesta pretende el desarrollo de habilidades vinculadas al pensamiento computacional como: abstracción, reconocimiento de patrones, algoritmos, gestión de proyectos y el uso de consola de comandos, por parte de los estudiantes.

CONTEXTO

El trabajo forma parte del proyecto CICITCA "Framework USIM para el desarrollo de Aplicaciones Web", aprobado según resolución 2902-CS-UNSJ.

1. INTRODUCCIÓN

En el mundo digital actual, la programación se ha convertido en una habilidad esencial. Sin embargo, la mayoría de las herramientas educativas existentes han adoptado enfoques basados en bloques [1] que, aunque facilitan la introducción al código [2], no necesariamente promueven el desarrollo de habilidades cruciales relacionadas con el aprendizaje computacional [3]–[5]. En este sentido, el uso exclusivo de bloques puede limitar la capacidad de jóvenes estudiantes para visualizar estructuras complejas y comprender interrelaciones y patrones

fundamentales dentro de un sistema informático [6]. Considerando que es necesario proponer herramientas que promuevan el pensamiento abstracto en los futuros programadores, se propone como una alternativa a Damo's Quest: un entorno de desarrollo gamificado que promueve la escritura de código real, la gestión de proyectos, y el uso de consola de comandos; herramientas que se consideran indispensables como interfaz de usuario, para potenciar algunas dimensiones del pensamiento computacional.

1.1. LA HERRAMIENTA DAMO'S QUEST

Damo's Quest [7] es un entorno de desarrollo integrado (IDE) gamificado diseñado especialmente para ser aplicado por profesores de computación de nivel primario y nivel secundario que enseñan programación a niños y adolescentes. Este IDE le presenta al usuario/jugador el desafío de resolver una serie de situaciones problemáticas a través de la programación.

El IDE en línea incorpora los elementos básicos de entornos de desarrollo profesionales, pero de una manera intuitiva y amigable para estudiantes que están dando sus primeros pasos en el mundo de la programación. En su versión actual, la herramienta cuenta con los siguientes componentes:

- Un panel con un árbol de directorio,
- Una consola de comandos que imita algunos comandos de Linux,
- Un panel con el escenario,
- Un editor de scripts.



Figura N° 1. Captura del IDE con los principales elementos de su interfaz de usuario.

Para poder jugar en Damo's Quest los profesores y sus estudiantes deben registrarse en un servidor web que brinda soporte de almacenamiento y compilación de scripts. Una vez registrados, tendrán a su disposición un sistema de almacenamiento de scripts y carpetas para sus proyectos.

1.2. ARQUITECTURA DE DAMO'S QUEST

Damo's Quest está implementado siguiendo una arquitectura cliente-servidor (Figura N°2). El cliente, es decir el IDE, está implementado en el motor de videojuegos gratuito Godot [8]. Para ejecutarse, necesita estar conectado con un servidor. Éste está implementado en Laravel [9] y brinda soporte de:

- Registro y autenticación de usuarios.
- Almacenamiento en MySQL. A cada usuario que se registra, el sistema le asigna un sistema de archivos con carpeta raíz, el cual inicialmente cuenta con un conjunto mínimo de tutoriales y ejemplos.
- Compilación del código. Para ello, cada vez que el usuario ejecuta un

código GALA, éste se envía a un endpoint del servidor, que se encarga de realizar la interpretación y la detección de errores de sintaxis. Finalmente, si el código fue compilado exitosamente, genera un código intermedio [10] que será ejecutado por el cliente.

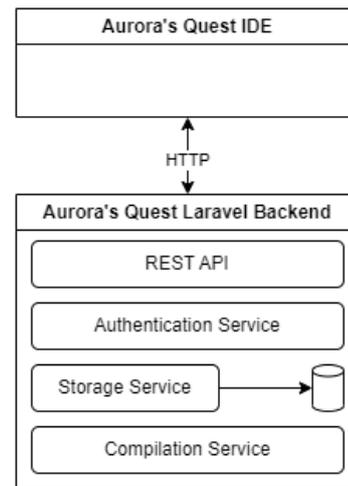


Figura N°2. Arquitectura del software.

1.3. SERVICIO DE COMPILACIÓN

La Figura N°3 muestra el detalle de cómo opera el servicio de compilación, desde que recibe un código fuente, hasta que es compilado y transformado en un lenguaje intermedio, que puede ser ejecutado por el componente ejecutor del IDE.

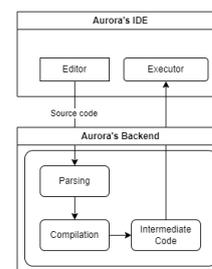


Figura N° 3. Componentes del servicio de compilación.

1.4. ÍTEMS Y ARREGLOS

En el futuro planteamos el desarrollo de inventarios e ítems, que son elementos que el personaje debe ir consiguiendo mientras recorre un laberinto. Estos ítems podrían residir en un inventario representado como un “arreglo”, el cual es una estructura de datos fundamental en la programación que permite almacenar múltiples valores en una sola variable.

El arreglo de ítems actuaría como un inventario para el personaje, donde cada ítem obtenido se iría añadiendo. La gestión de este arreglo es una tarea esencial que el jugador debe aprender. Esto incluye operaciones como consultar un ítem en una posición específica del arreglo, insertar un nuevo ítem en el arreglo, y ordenar los ítems en el arreglo según ciertos criterios.

Por ejemplo, si el personaje recoge un ítem “llave”, este ítem se añadiría al final del arreglo. Si el jugador necesita usar la “llave” para abrir una puerta, podría buscar en el arreglo para ver si tiene una. Además, el jugador podría querer ordenar sus ítems por tipo o por frecuencia de uso, lo que implicaría aprender sobre algoritmos de ordenación.

Además, el arreglo de ítems podría ser utilizado para introducir conceptos más avanzados de programación. Por ejemplo, si cada ítem tiene atributos asociados, como un valor de “poder” o “peso”, el arreglo de ítems se convierte en un arreglo de objetos, introduciendo el concepto de programación orientada a objetos.

En resumen, el uso de un arreglo para gestionar ítems en el juego proporciona una forma práctica y atractiva de aprender sobre estructuras de datos y algoritmos en la programación. A medida que los jugadores avancen en el juego, se espera que mejoren sus aptitudes en la escritura de código. La

Figura N°4 presenta una posible configuración de ítems en posiciones de un arreglo.

| | |
|---|---|
| 0 |  |
| 1 |  |
| 2 |  |
| 3 |  |

Figura N°4. Ejemplo de ítems y arreglo.
Fuente Elaboración propia

1.5. ESCENARIOS, MAPAS Y DESAFÍOS

En Damo’s Quest, los planteos y consignas a ser resueltos por los estudiantes se implementan en escenarios. En ellos se desarrolla una historia y se proponen desafíos a ser solucionados por el jugador. Estas historias, a futuro, podrán ser desarrolladas por los docentes en el mismo entorno.

La propuesta es que las prácticas de programación se desarrollen resolviendo situaciones problemáticas y desafíos planteados en un mapa o escenario. Estos mapas, diseñados específicamente para profesores de computación, consisten básicamente en una o más capas de imágenes con un árbol de nodos interconectados que hacen referencia a lugares específicos del mismo denominados Nodos. Estos nodos pueden poseer restricciones para ser accedidos (por ejemplo, un nodo llamado “puerta” podría necesitar que el personaje haya conseguido un ítem “llave”).

La visión a futuro es que con el tiempo las dinámicas de enseñanza/aprendizaje, sean compartidas (o comercializadas) bajo la forma de mapas predefinidos.

A modo de otro ejemplo, la Figura N°5 muestra un mapa con nodos de diferentes tipos en lugares estratégicos que deben ser recorridos para llegar desde el inicio hasta la

salida.

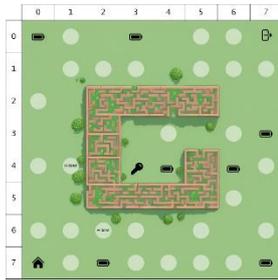


Figura N°5. Nodos lógicos de diverso tipo.

2. LÍNEAS DE INVESTIGACIÓN Y DESARROLLO

La línea de investigación y desarrollo de Damo's Quest se centra en la creación de un entorno de programación gamificado que promueva el aprendizaje de la programación en niños y adolescentes. Este proyecto se basa en la premisa de que la gamificación puede fomentar la motivación y el compromiso de los estudiantes, lo que a su vez puede conducir a un mejor aprendizaje.

El desarrollo de Damo's Quest implica la integración de varios componentes clave, incluyendo un panel con un árbol de directorio, una consola de comandos, un panel con el escenario y un editor de scripts. Estos componentes están diseñados para ser intuitivos y amigables para los estudiantes que están dando sus primeros pasos en el mundo de la programación.

Además, la investigación y desarrollo de Damo's Quest también se centra en la implementación de una arquitectura cliente-servidor. El cliente, es decir, el IDE, está implementado en el motor de videojuegos gratuito Godot. Para ejecutarse, necesita estar conectado con un servidor, que está implementado en Laravel y brinda soporte de registro y autenticación de usuarios, almacenamiento en MySQL y compilación del código.

En términos de desarrollo futuro, la línea de investigación y desarrollo de Damo's Quest buscará continuar mejorando y expandiendo las características y funcionalidades del IDE para proporcionar una experiencia de aprendizaje aún más enriquecedora y efectiva para los estudiantes. Esto puede incluir la incorporación de nuevas funcionalidades, la mejora de la interfaz de usuario y la optimización del rendimiento del sistema. Además, se realizarán esfuerzos para evaluar la eficacia de Damo's Quest como herramienta de aprendizaje y para recopilar comentarios de los usuarios para informar las iteraciones futuras del proyecto.

3. RESULTADOS

Uno de los resultados esperados más significativos de este proyecto es la aplicación y el estudio del Modelo de Aceptación de Tecnología (TAM) [11] por parte de los profesores de informática de nivel secundario. Este estudio permitirá evaluar la aceptación y la utilidad percibida de Damo's Quest en el contexto educativo, proporcionando información valiosa para futuras iteraciones y mejoras del sistema.

Además, se espera realizar una serie de mejoras en el lenguaje de programación utilizado en Damo's Quest. Estas mejoras incluyen:

1. **Manejo de estructuras de arreglos:** Se busca incorporar la capacidad de manejar estructuras de arreglos, para gestionar inventarios del personaje, lo que permitirá a los estudiantes trabajar con conjuntos de datos y comprender conceptos fundamentales de la programación.
2. **Manejo de estados y componentes:** Se propone incluir la capacidad de gestionar estados [12] y componentes,

lo que permitirá a los estudiantes desarrollar aplicaciones más complejas y dinámicas.

3. **Manejo de librerías:** Se espera incorporar la capacidad de utilizar librerías, lo que permitirá a los estudiantes ampliar la funcionalidad de sus programas y aprender sobre la reutilización de código.
4. **Creación de mapas y desafíos desde el mismo entorno:** Se busca permitir a los estudiantes crear sus propios mapas y desafíos dentro del entorno de Damo's Quest, lo que se espera promueva el desarrollo de la creatividad y el pensamiento crítico.

Estos resultados esperados no sólo mejorarán la funcionalidad y la utilidad de Damo's Quest como herramienta educativa, sino que también contribuirán a la formación de habilidades de programación y pensamiento computacional en los estudiantes.

5. BIBLIOGRAFÍA

- [1] F. Tejera-Martínez, D. Aguilera, and J. M. Vilchez-González, "Lenguajes de programación y desarrollo de competencias clave. Revisión sistemática," *Revista electrónica de investigación educativa*, vol. 22, doi: 10.24320/redie.2020.22.e27.2869. Available: https://www.scielo.org.mx/scielo.php?script=sci_arttext&pid=S1607-40412020000100127. [Accessed: Feb. 21, 2024]
- [2] M. U. Bers, "Coding as another language: a pedagogical approach for teaching computer science in early childhood," *Journal of Computers in Education*, vol. 6, no. 4, pp. 499–528, 2019.
- [3] J. M. Wing, "Computational thinking," *Commun. ACM*, vol. 49, no. 3, pp. 33–35, 2006.
- [4] S. Papavlasopoulou, M. N. Giannakos, and L. Jaccheri, "Exploring children's learning experience in constructionism-based coding activities through design-based research," *Comput. Human Behav.*, vol. 99, pp. 415–427, 2019.
- [5] E. Schanzer, S. Krishnamurthi, and K. Fisler, *Blocks versus text: Ongoing lessons from bootstrap*. Institute of Electrical and Electronics Engineers Inc., 2015.
- [6] S. Grover and R. Pea, "Computational Thinking in K-12: A Review of the State of the Field," *Educ. Res.*, vol. 42, no. 1, pp. 38–43, 2013.
- [7] "Idei Game." Available: <https://damogame.com/>. [Accessed: Feb. 20, 2024]
- [8] J. Linietsky and A. Manzur, "Godot Engine - Free and open source 2D and 3D game engine," *Godot Engine*. Available: <https://godotengine.org/>. [Accessed: Feb. 22, 2024]
- [9] "Laravel - The PHP Framework For Web Artisans." Available: <https://laravel.com/>. [Accessed: Feb. 22, 2024]
- [10] T. Æ. Mogensen, "Intermediate-Code Generation," in *Introduction to Compiler Design*, T. Æ. Mogensen, Ed., Cham: Springer International Publishing, 2017, pp. 127–156.
- [11] V. V. H. Bala, "Technology Acceptance Model 3 and a Research Agenda on Interventions," *Decision Sciences*, vol. 39, no. 3, pp. 273–315, May 2008.
- [12] E. Domínguez, B. Pérez, Á. L. Rubio, and M. A. Zapata, "A systematic review of code generation proposals from state machine specifications," *Information and Software Technology*, vol. 54, no. 10, pp. 1045–1066, 2012.