Facilitando la explicación de conceptos en programación con Wollok Game

Nahuel Palumbo[1], Nicolás Passerini[1], and Lucas Spigariol[2]

[1] DCyT - Universidad Nacional de Quilmes, Buenos Aires, Argentina
nahuel.palumbo@unq.edu.ar, npasserini@gmail.com
[2] Universidad Nacional de Hurlingham, Buenos Aires, Argentina
lspigariol@gmail.com

Resumen Los videojuegos constituyen un recurso frecuente en la enseñanza de la programación, ya que permiten construir entornos interactivos que facilitan la experimentación y comprensión de los programas. Este trabajo presenta avances sobre una línea de investigación en curso utilizando Wollok Game, un framework para el desarrollo de videojuegos en el lenguaje educativo Wollok.
Wollok propone un recorrido pedagógico que introduce gradualmente el paradigma de objetos, comenzando con programas compuestos por pocos objetos y postergando la aparición de clases. Las actividades propuestas utilizan el entorno de juego para introducir y reforzar conceptos centrales del paradigma, como identidad, polimorfismo, instanciación y *closures*.
El objetivo de estas experiencias es mostrar cómo el desarrollo de videojuegos puede integrarse en el recorrido didáctico propuesto por Wollok y simplificar la introducción de algunos conceptos clave de la programación orientada a objetos.

Keywords: Wollok · Game · Educación · Programación · Videojuegos

1. Introducción

La enseñanza de la programación mediante videojuegos ha sido ampliamente estudiada, especialmente en etapas tempranas de aprendizaje [6,2,9]. Sus beneficios han impulsado el desarrollo de propuestas en distintos contextos, incluyendo herramientas nacionales y entornos utilizados en el ámbito universitario [10,5,1,3].

Wollok es un lenguaje didáctico diseñado para la enseñanza de programación orientada a objetos [8]. Su recorrido pedagógico propone comenzar con programas compuestos por pocos objetos y postergar la introducción del concepto de clase [7]. Tanto el lenguaje como sus enfoques de enseñanza son utilizados en diversas universidades nacionales de Argentina.

Wollok Game es un *framework* para la creación de videojuegos en Wollok [11]. El objetivo es acompañar este recorrido pedagógico mediante la incorporación de elementos lúdicos que faciliten la comprensión y práctica de los conceptos.

Si bien forma parte de un trabajo de investigación en curso, este trabajo presenta tres propuestas didácticas desarrolladas en clase utilizando Wollok Game.

2. Videojuegos para enseñar a programar

El uso de entornos lúdicos para la enseñanza de la programación tiene una larga tradición. Uno de los antecedentes más influyentes es Logo, un lenguaje para controlar una "tortuga", desarrollado por Papert [6]. Logo introdujo la idea de aprender conceptos de computación mediante la experimentación de conceptos algorítmicos en un entorno visual interactivo.

Inspiradas en estas ideas, nacieron herramientas educativas incorporando entornos gráficos como Etoys [2] y Scratch [9]. Ambas proponen un lenguaje visual que permite crear historias interactivas, animaciones y videojuegos. Su diseño basado en bloques busca evitar errores sintácticos y facilitar el acceso temprano a la programación, principalmente enfocado a niños.

En Argentina también se han desarrollado herramientas didácticas orientadas a la enseñanza de la programación. Gobstones [5] es un lenguaje educativo con un modelo de tablero, celdas, y un cabezal, diseñado para introducir conceptos fundamentales de programación procedural. Pilas Bloques [10] propone un entorno basado en bloques y desafíos interactivos orientado a contextos educativos del ciclo inicial.

Dentro del área de la programación orientada a objetos, donde se encuentra este trabajo, aparecen otras propuestas. Alice [1] propone un entorno de programación visual basado en bloques para la creación de animaciones y mundos tridimensionales. Greenfoot [3] propone un entorno para la enseñanza de programación orientada a objetos mediante la construcción de videojuegos bidimensional y programación textual en Java, adoptando un enfoque centrado en clases desde etapas tempranas.

En este contexto se ubica **Wollok Game**, una propuesta educativa para la enseñanza del paradigma de objetos mediante la creación de videojuegos en Wollok [8,11]. Su propuesta pedagógica introduce progresivamente los conceptos del modelo de objetos, comenzando con programas compuestos por pocos objetos y postergando la introducción del concepto de clase [7]. Wollok Game se diferencia del resto de las herramientas mencionadas en que acompaña los conceptos principales del paradigma de objetos, como *polimorfismo* e *identidad*, manteniendo al mismo tiempo las ventajas pedagógicas de un entorno interactivo basado en juegos.

3. Wollok Game

Wollok Game es una propuesta que busca complementar el recorrido pedagógico de Wollok mediante la creación de videojuegos como recurso didáctico [8,7,11].

Wollok Game[3] permite desarrollar aplicaciones interactivas de manera sencilla manejando un mundo de cuadrícula donde cualquier objeto que entienda los mensajes `position()` e `image()` puede ser visualizado. Los programas en

[3] `https://www.wollok.org/documentation/wollok_game`

Wollok interactúan principalmente con un objeto `game`, ya definido en el lenguaje, para controlar el juego. La herramienta se usa manejando los elementos básicos de la programación orientada a objetos, tales como *objetos*, *mensajes* y *polimorfismo*, sin necesidad de lidiar con conceptos más avanzados, como *clases* y *herencia*, hasta etapas posteriores.

En esta sección, se presentan tres propuestas didácticas.

3.1. Programar en objetos es como contar historias

Durante la primera clase, se introduce la idea de cómo funciona un programa de objetos por medio de una historia interactiva. Los objetos son actores que se visualizan en la escena, a quiénes se les envía mensajes por consola para controlar lo que sucede.

En el ejemplo, se utiliza una historia basada en la película Titanic[4]. Comenzando con un mensaje `pelicula.iniciar()` el juego se abre y los objetos pueden verse en pantalla. Luego, se le envían mensajes a distintos objetos en la escena, como `rose`, `iceberg` o `tabla`, para ir contando la historia. Una posible historia se presenta en la Figura 1.

De esta forma, se busca la comprensión de cómo funciona un programa en objetos por medio de la experimentación y visualización inmediata de los cambios. La historia sucede a partir de los mensajes que se le envían a distintos objetos. A su vez, los objetos se envían mensajes entre ellos, como la `tabla` que puede usar *polimórficamente* tanto a `rose` como `jack` para pedir ayuda.

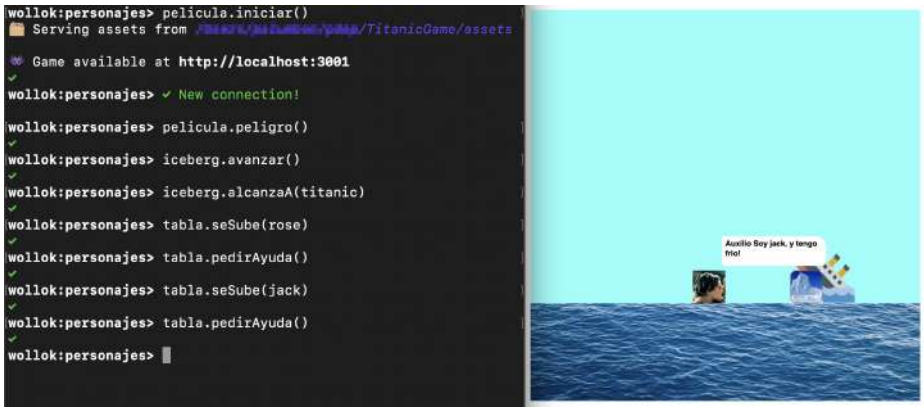

Figura 1. A la izquierda, la consola de Wollok donde el usuario envía mensajes a los objetos para contar la historia. A la derecha, se va visualizando la historia interactiva.

[4] `https://github.com/wollok/TitanicGame`

3.2. Teclas que envían mensajes

Un ejercicio a modo de tutorial[5], usado para mostrar las funcionalidades de la herramienta, propone crear un juego interactivo. Para ello se abandona la consola, y se desarrolla un programa Wollok que configure e inicie el juego.

Para interactuar con el juego hay que asignar ciertas acciones a eventos del teclado. Las acciones se configuran pasándole un *closure* a una tecla específica. La Figura 2 muestra el IDE de Wollok (VSCode) presentando el código del programa, una forma de correrlo y mirar los logs, y el juego interactivo donde las teclas ejecutan los *closures* correspondientes.

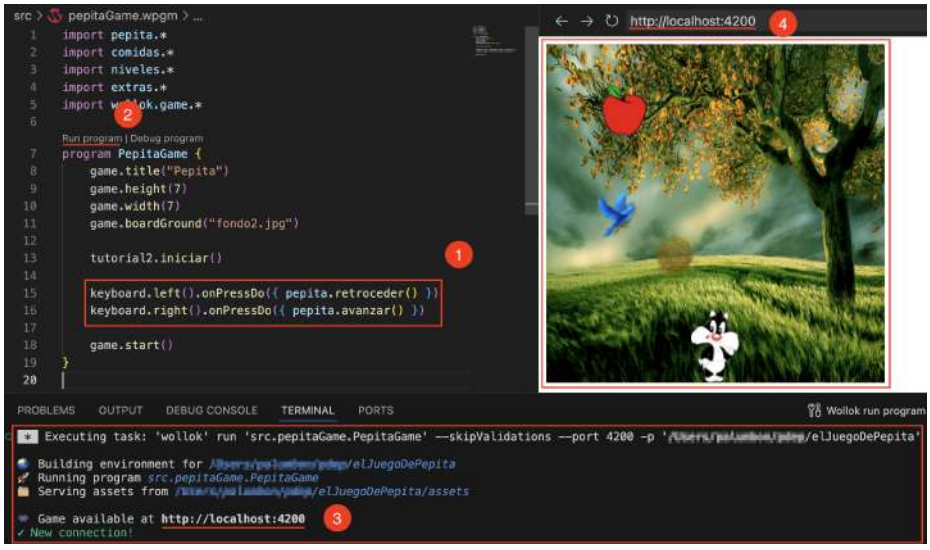

Figura 2. Ejemplo del tutorial de Pepita. 1) Se escribe un programa que usa *closures* para configurar las teclas del teclado. 2) Se corre el programa desde el editor. 3) Los logs muestran información sobre dónde está servido el juego. 4) El juego es accesible desde un navegador web, pudiendo ejecutarse en el propio IDE.

Este ejercicio continúa explorando el uso de *closures* en distintos contextos, como la respuesta a eventos temporales y el manejo de colisiones. A lo largo de la actividad se presentan diferentes variantes, incluyendo *closures* con y sin parámetros, así como aquellas que se ejecutan una única vez o de manera repetida.

Esta didáctica permite introducir la idea de *closure* como código que se ejecuta en momentos posteriores a su creación.

[5] `https://github.com/wollok/elJuegoDePepita`

3.3. Manejando varias instancias

Finalmente, si bien el recorrido propuesto por Wollok posterga la aparición de clases, éstas son presentadas luego de trabajar con objetos *autodefinidos*. La última didáctica de este trabajo propone terminar un juego en el que un granjero planta y riega cultivos, por ejemplo maíz[6].

Se introduce el concepto de clase para darle a todos los maíces plantados el mismo comportamiento. A su vez, se asocia la acción de plantar a *instanciar* un objeto de esa clase que se visualiza en el juego. Por último, al regar una instancia particular, se refuerza la distinción entre instancias de una misma clase, que no solo están en posiciones distintas del juego sino que crecen a distinto ritmo según como se rieguen. La Figura 3 presenta un ejemplo de esta dinámica.

Figura 3. Diferencias entre la clase `Trigo` y sus instancias, dado por la posición y el crecimiento por riego.

El ejercicio continúa agregando otros tipos de cultivos para abordar el polimorfismo entre clases.

4. Discusiones y trabajo futuro

Wollok como herramienta educativa, y su camino pedagógico que pospone la introducción de las clases en la enseñanza de la programación orientada a objetos, ha sido presentado en trabajos anteriores [8,4,7]. Actualmente, el lenguaje es utilizado en varias universidades nacionales, entre ellas UTN, UNQ, UNSAM y UNDAV, involucrando a decenas de docentes y alrededor de mil estudiantes por año.

El impacto de Wollok Game en las evaluaciones de trabajos finales de cursos universitarios en Argentina también fue analizado previamente [11]. En contraste, este trabajo explora un uso diferente de la herramienta: su incorporación a lo largo del camino pedagógico para acompañar la construcción progresiva de los conceptos de programación orientada a objetos.

Este trabajo presenta tres propuestas didácticas para integrar Wollok Game en distintas etapas de ese recorrido. Las experiencias fueron implementadas en cursos universitarios por docentes que participan activamente tanto en el diseño del lenguaje como en su enseñanza. Si bien la evaluación sistemática de los efectos de estas propuestas constituye una línea de trabajo futura, la experiencia obtenida hasta el momento resulta alentadora y evidencia el potencial de la herramienta para enriquecer el proceso de aprendizaje. Se espera que las actividades presentadas puedan servir como guía e inspiración para otros docentes interesados en incorporar este tipo de herramientas en sus cursos.

[6] `https://github.com/wollok/granjavilla`

Referencias

1. Cooper, S.: The design of alice. ACM Trans. Comput. Educ. **10**(4) (Nov 2010). https://doi.org/10.1145/1868358.1868362, https://doi.org/10.1145/1868358.1868362
2. Kay, A.: Squeak etoys, children & learning. online article **2006**, 1–8 (2005)
3. Kölling, M.: The greenfoot programming environment. In: Proceedings of the 15th Annual Conference on Innovation and Technology in Computer Science Education. pp. 14–17. ACM (2010). https://doi.org/10.1145/1822090.1822099
4. Lombardi, C., Passerini, N.: Wollok: reconciliando didáctica e industria en un lenguaje educativo para poo. In: XIV Congreso Nacional de Tecnología en Educación y Educación en Tecnología (TE&ET 2019),(Universidad Nacional de San Luis, 1 y 2 de julio de 2019). (2019)
5. Martínez López, P.E., Ciolek, D., Arévalo, G., Pari, D.: The gobstones method for teaching computer programming. In: 2017 XLIII Latin American Computer Conference (CLEI). pp. 1–9 (2017). https://doi.org/10.1109/CLEI.2017.8226428
6. Papert, S.: Mindstorms: Children, Computers, and Powerful Ideas. Basic Books, New York, NY (1980)
7. Passerini, N., Lombardi, C.: Postponing the concept of class when introducing oop. In: Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education. p. 152–158. ITiCSE '20, Association for Computing Machinery, New York, NY, USA (2020). https://doi.org/10.1145/3341525.3387369, https://doi.org/10.1145/3341525.3387369
8. Passerini, N., Lombardi, C., Fernandes, J., Tesone, P., Dodino, F.: Wollok: Language + ide for a gentle and industry-aware introduction to oop. In: 2017 Twelfth Latin American Conference on Learning Technologies (LACLO). pp. 1–4 (2017). https://doi.org/10.1109/LACLO.2017.8120933
9. Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B., Kafai, Y.: Scratch: Programming for all. Communications of the ACM **52**(11), 60–67 (2009). https://doi.org/10.1145/1592761.1592779
10. Sanzo, A., Schapachnik, F., Factorovich, P., O'Connor, F.S.: Pilas bloques: A scenario-based children learning platform. In: 2017 Twelfth Latin American Conference on Learning Technologies (LACLO). pp. 1–6 (2017). https://doi.org/10.1109/LACLO.2017.8120926
11. Spigariol, L., Palumbo, N., Passerini, N.: Competencias en programación orientada a objetos. SADIO Electronic Journal of Informatics and Operations Research **20**(1), 77–101 (Jun 2021), https://revistas.unlp.edu.ar/ejs/article/view/17646