Recta Numérica: Una Actividad de Códimo

Luciano Graziani, Gabriela Anahí Cayú, Matías Emanuel Sanhueza y Enrique Molinari

Universidad Nacional de Río Negro, Sede Atlántica, Viedma, Río Negro
http://www.unrn.edu.ar/
{lgraziani, gcayu, msanhueza, emolinari}@unrn.edu.ar

Abstract. El Pensamiento Computacional resulta de utilidad en situaciones de la vida cotidiana [9]. Introducirlo en la educación primaria y secundaria permitirá despertar el interés de los estudiantes en continuar sus estudios en ciencias de la computación, a la vez que posibilitará que desarrollen estas habilidades desde temprano para que puedan aplicarla en su vida cotidiana. En este trabajo presentamos una actividad diseñada para Códimo [5] y denominada Recta Numérica [1] que fue pensada para alumnos entre 6 y 8 años, y con la cual se busca incentivar de manera transparente uno de los aspectos del Pensamiento Computacional: el diseño de algoritmos. Los alumnos deben jugar a ubicar un número en la recta numérica, haciéndolo avanzar por un laberinto, mediante el uso de bloques (o comandos) simples. El alumno refuerza un conocimiento adquirido en clases (lo números naturales, enteros y su ubicación en la recta), utilizando el pensamiento computacional.

Keywords: códimo, educación, pensamiento-computacional, objetos-de-aprendizaje, aprendiendo-a-programar, javascript.

1 Introducción

Desde el año 2014, la Universidad Nacional de Río Negro realiza distintas actividades para llevar la programación a las escuelas primarias y secundarias de la región. Durante todas ellas, el objetivo de las mismas fue contribuir a despertar vocaciones tempranas en carreras afines a las Ciencias de la Computación. A su vez, el Pensamiento Computacional es una herramienta de gran utilidad para todas las personas, no solo para aquellos decididos a continuar su vida profesional dentro de las Ciencias de la Computación [9].

Luego de dos años de experiencia llevando la programación a las escuelas utilizando herramientas como Alice, entendiendo las dificultades de docentes y alumnos, decidimos construir una herramienta que permitiera reforzar los conceptos dictados en clase utilizando habilidades del Pensamiento Computacional. A esta herramienta la nombramos Códimo.

En las actividades planteadas en Códimo, los alumnos refuerzan temas aprendidos en clases, ya sean de ciencias exactas, naturales, sociales u otra, aplicando conceptos de Ciencias de la Computación.

Este trabajo continúa en la sección 2 con la descripción de la propuesta, donde describimos en detalle cómo se juega y se aprende utilizando la actividad llamada “Recta Numérica”. En la sección 3 se describen detalles técnicos del desarrollo de Códimo; y en la sección 4 planteamos diferentes caminos posibles para continuar este trabajo.

2 “Recta Numérica”: Programar para Aprender

Es una actividad pensada para alumnos de primer ciclo de primaria (6 a 8 años). Se encuadra dentro del área de matemáticas, más concretamente luego de aprender números naturales, enteros, relaciones de orden y la recta numérica.

El objetivo siempre es ubicar el número dado en la recta numérica. La actividad presenta diferentes niveles de complejidad. En el nivel inicial se utilizan únicamente números naturales de una cifra; en el siguiente nivel números naturales de dos cifras; y en el último nivel, el más avanzado, se utilizan números enteros.

La consigna es desplazar cada número a través de un laberinto y ubicarlo dentro de la recta numérica, con la restricción de que solo es posible mover el número utilizando un conjunto limitado bloques. Cada bloque tiene una función específica, como mover el número en una dirección, o hacer que salte. Los bloques pueden encastrarse formando un conjunto de instrucciones que se aplicarán sobre el número. De esta forma, utilizando uno de los aspectos del Pensamiento Computacional, el diseño de algoritmos, los alumnos realizan la actividad.

El uso de bloques para programar tiene un fundamento y ventaja pedagógica por sobre la utilización de sentencias escritas de un pseudo lenguaje de programación. Los bloques siempre generan programas sintácticamente correctos, lo cual permite a los alumnos enfocarse únicamente en comprender la semántica o sea el efecto que producen los bloques sobre el número, en este caso, el movimiento del número a través del laberinto.

La Fig. 1 muestra uno de los ejercicios de nivel avanzado de la actividad “Recta Numérica”.

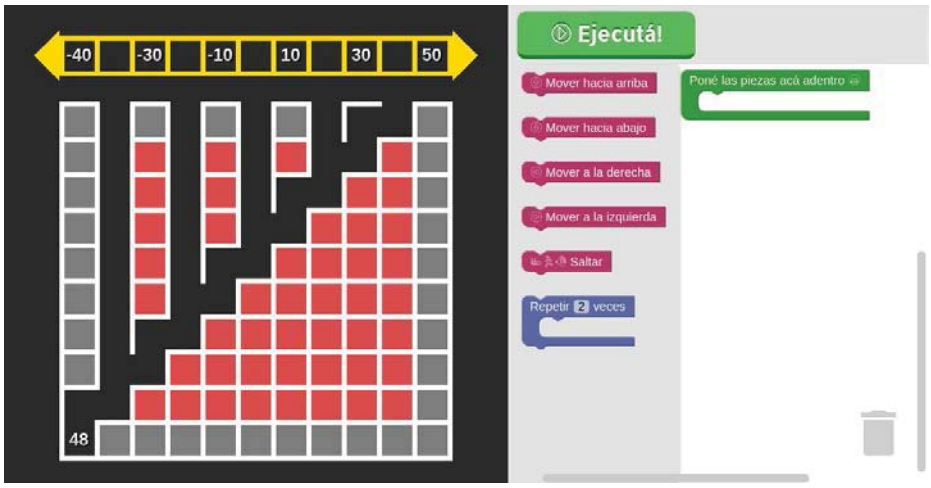

Fig. 1. Actividad Nivel Avanzado

Sobre la izquierda de la Fig. 1 se puede observar el laberinto y la recta numérica encima de él. La actividad inicia siempre con varios números sobre la recta y un número dentro del laberinto que hay que mover y llevar hacia la recta. En el ejemplo de la Fig. 1, el número que hay que llevar hacia la recta es el 48. La recta numérica posee huecos en los cuales el número puede ser ubicado. A partir del último ejercicio del nivel inicial en adelante, la recta numérica posee huecos donde el número se ubicará correctamente y donde no.

El objetivo de la actividad es llevar el número a través del laberinto hasta una posición de salida, aquellas donde termina el laberinto, y allí saltar hacia la recta.

Sobre la derecha de la Fig. 1 se encuentran los bloques que se utilizaran para mover el número a través del laberinto. Cada bloque posee una función específica que produce un efecto diferente sobre el movimiento que realizará el número. Dividimos las funciones de los bloques en tres categorías:

- Simples:
  - **Mover hacia arriba**: mueve el número hacia arriba una posición.
  - **Mover a la derecha**: mueve el número una posición hacia la derecha.
  - **Mover a la izquierda**: mueve el número una posición hacia la izquierda.
  - **Mover hacia abajo**: mueve el número hacia abajo una posición.
- Compuestos:
  - **Repetir N veces**: ejecuta N veces los bloques simples que estén dentro de él.
- Especiales de la Actividad:
  - **Saltar**: una vez que se llega a una posición de salida del laberinto, con este bloque se ubica el número en la recta numérica.

Los bloques, cualquiera sea su categoría, se deben ubicar dentro del contenedor de bloques, el cual se identifica por su color verde y su descripción *"Poné las piezas acá adentro"*.

Al oprimir el botón "Ejecutar", Códimo interpreta los bloques uno a uno, realizando el movimiento del número a través del laberinto. Terminada la ejecución, si el número finalizó dentro del hueco correcto de la recta numérica, se sugiere continuar con la siguiente actividad, en caso contrario se sugiere intentarlo nuevamente. Al reiniciar el ejercicio, el número volverá a su posición inicial y (probablemente) cambiará su valor dentro del rango de números posibles correspondientes al hueco que actualmente es el correcto. Esto así planteado permite que no sea un ejercicio sistemático, aun cuando la solución sea constante.

Los autores entienden que es importante destacar el hecho de que "Recta Numérica" es una actividad de matemática y no de programación. Al utilizar los bloques simulando ser sentencias en un lenguaje de programación, se está ejercitando uno de los aspectos del pensamiento computacional: el diseño de algoritmos; pero esto no forma parte del objetivo principal de la actividad que es afianzar el conocimiento sobre los números naturales, enteros y la recta numérica.

Todas las actividades que se pretenden generar con Códimo tendrán esa misma particularidad, lo que hace que puedan ser utilizadas por los docentes sin tener que desviarse de su currícula para introducir el Pensamiento Computacional en sus clases.

Al utilizar esta actividad, el alumno tiene que programar con los bloques para ejercitarse y afianzar este contenido específico de su currícula escolar. Por tal motivo decimos que no se está aprendiendo a programar, sino que se programa para aprender.

3 "Recta Numérica": descripción técnica

Códimo es una aplicación web escrita únicamente en Javascript. Funciona en las últimas versiones de Firefox, Chrome, Edge y Safari. Además, como es de código abierto, no solo puede verse cómo está desarrollado, sino también es posible colaborar al informar sobre errores, proponer ideas, o incluso realizar modificaciones. El repositorio se encuentra en: https://github.com/lgraziani2712/codimo.

3.1 Tecnologías utilizadas

Para el desarrollo de Códimo fueron necesarias un conjunto variado de herramientas, tanto para facilitar la experiencia del desarrollo, como para asegurar un buen diseño de los componentes. A continuación se listan las más importantes y qué función cumplió cada una en la implementación.

Parte de la aplicación

- **Blockly**: "es una librería para diseñar editores visuales de programación" [4]. Permitió implementar una interfaz de programación con bloques.
- **PixiJS**: "es un renderizador 2D para Javascript" [6]. Cumple la función de motor gráfico 2D, es decir, todas las animaciones del juego de la "Recta Numérica" están construidos sobre esta herramienta.

- **React**: "es una librería Javascript para construir interfaces gráficas" [3]. Se utilizó para poder crear la página web, donde fue posible desarrollar rutas dinámicas sin tener que construir un servidor. Permitió integrar PixiJS con Blockly.

En la Fig. 2 se puede apreciar de manera general cómo está diseñada la arquitectura de Códimo:

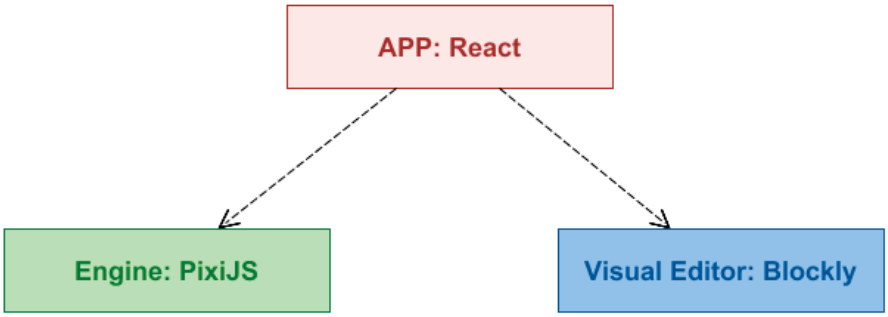

Fig. 2. Arquitectura general de Códimo

Como se evidencia en la Fig. 2, la arquitectura está integrada por tres componentes principales:

- **APP**: se encarga de manipular las URL de la aplicación, como la carga y eliminación de cada dependencia relativa a la actividad que se seleccionó.
- **Engine**: se encarga de renderizar la actividad y de animar cada objeto según la acción que recibe. Posee una interfaz que es utilizada por la APP para poder comunicarse.
- **Visual Editor**: se encarga de listar todas las posibles acciones, como también ofrecer un espacio de trabajo donde poder ir creando el programa. Ofrece una interfaz con la APP para poder comunicarse.

Cada vez que la APP intenta cargar una actividad, llama al *Engine* y le pide los elementos que necesita. Al mismo tiempo, llama al *Editor Visual*, y le pide que cree un editor con los elementos necesarios para la actividad en particular.

El usuario siempre interactúa con la APP y es ésta la que delega las tareas que recibe en los otros dos componentes.

Parte de la experiencia en el desarrollo

- **Storybook**: "es un entorno para desarrollo de Interfaces de Usuario" [8]. Con esta herramienta fue posible diseñar y desarrollar cada componente visual por separado.
  Para Códimo, un componente visual no sólo hace referencia al texto o los botones que acompañan a la actividad, sino el motor gráfico en sí. Así, fue posible diseñar, por ejemplo, cada animación del número de manera aislada.

- **Babel**: permite compilar código Javascript de las últimas versiones, a código que puede correr en las versiones más recientes de todos los exploradores. Esta tecnología nos permite confirmar que Códimo funciona, al menos, para las dos últimas versiones de Firefox, Chrome, Edge y Safari.
- **ESLint**: esta herramienta evalúa el código Javascript de manera estática en búsqueda de errores de sintaxis, de estructura y de estilo.
- **FlowType**: "es un analizador estático de tipos de datos para Javascript" [2].
- **Jest**: es un framework de testing para Javascript innovador en ciertos elementos.
  Analiza qué archivos fueron modificados desde el último commit al repositorio (generalmente de tipo GIT), y ejecuta únicamente los tests relacionados a los mismos.
  Durante la ejecución de los tests es posible filtrar por archivo, por nombre de test o por una expresión regular.
- **Snapshots**: este nuevo tipo de testing toma una variable y serializa el contenido en un archivo. Cada vez que se ejecuta el test, comprueba el contenido de la variable con lo serializado en el archivo, y si hay una diferencia, falla el test e informa la diferencia de la misma manera en que lo haría una herramienta de versionado de archivos. Esto es realmente útil cuando se quieren hacer tests unitarios de componentes HTML sin tener que hacer chequeos de cada elemento de la estructura.
- **Webpack**: es un module bundler, una herramienta que toma todos los archivos de un proyecto y los compila en único archivo. En particular Webpack puede considerar cualquier tipo de archivo (desde imágenes, json, de estilo, como css, entre otros) como un módulo de Javascript; siempre y cuando esté configurado para que sepa cómo manipular cada uno de ellos.
  Además, ofrece un servidor de desarrollo realmente sencillo de configurar donde es posible ir modificando el programa y ver los cambios en vivo sin tener que refrescar ninguna página.
  La versión de producción de Códimo, con archivos que se descargan de manera asincrónica a medida que se requieren (por ejemplo, cada uno de los ejercicios), está generada a través de Webpack.

Parte del proceso de integración continua

- **CircleCI**: herramienta de integración continua donde es posible configurar eventos a partir de ejecutar distintas acciones cada vez que se hace un commit, un *pull request* o un merge al proyecto.
- **Netlify**: integra Códimo un proceso de deployment continuo.

4 Trabajo futuro

Los autores desean dar continuidad a este proceso de introducir el Pensamiento Computacional en las escuelas a través de las siguientes actividades:

- Completar todo el proceso de enseñanza, proveyendo un tutorial introductorio y un proceso de evaluación final. Aplicando FLOM [7] como arquitectura de diseño de actividades.
- Mejorar Códimo con la intención de brindar más y mejores funcionalidades junto a un conjunto de abstracciones que permitan a cualquier desarrollador implementar diferentes actividades reduciendo los tiempos de desarrollo.
- El Pensamiento Computacional abarca otros aspectos, no solo el diseño de algoritmos —el cual es ejercitado por medio de bloques—, como en la actividad descrita en este trabajo. El pensamiento abstracto, el reconocimiento de patrones, la descomposición de un problema complejo en múltiples más simples, son otros de sus aspectos relevantes. Es importante poder seguir generando actividades bajo el criterio de "programar para aprender", que permitan evidenciar la utilidad de los diferentes aspectos del Pensamiento Computacional.

5 Conclusión

En este trabajo se presentó la actividad "Recta Numérica", implementada con Códimo con el objetivo de poder encontrar evidencia sobre cómo la implementación del paradigma del Pensamiento Computacional a situaciones de aprendizaje específicas permite reforzar el conocimiento en sí a la vez que se desarrolla una manera de abordar problemas que será de utilidad para diversas situaciones. Introducir el Pensamiento Computacional en la educación primaria, brinda herramientas tanto al alumno como al docente. A los primeros porque permitirá despertar el interés de los estudiantes en el estudio de ciencias de la computación, a la vez que posibilitará que desarrollen estas habilidades desde temprano, para que puedan aplicarla en su vida cotidiana. Para los docentes es una forma entretenida de incorporar tecnología digital en el dictado de la materia, manteniendo el interés de los alumnos y brindando una forma novedosa y atractiva para afianzar las temáticas de cada asignatura.

6 Referencias

1. Códimo. (2017) ¡Donde el código, la imaginación y la motivación se encuentran! Recuperado de https://codimo.netlify.com/. Vigente junio 2017.
2. Facebook. (2017). Flow is a static type checker for javascript. Recuperado de https://flow.org/. Vigente junio 2017.
3. Facebook. (2017). React: A Javascript Library for Building User Interfaces. Recuperado de https://facebook.github.io/react/. Vigente junio 2017.
4. Google. (2017). A library for building visual programming editors. Recuperado de https://developers.google.com/blockly/. Vigente junio 2017.
5. Graziani, L. y otros (2016). Códimo: Desarrollo del pensamiento computacional en las escuelas. II Jornadas Argentinas De Tecnología, Innovación y Creatividad 2016 - Workshop sobre Innovación en Centros Educativos y de Investigación

(WICEI). Recuperado de http://jatic2016.ucaecemdp.edu.ar/trabajos/WICEI394TE-GrazianiJATIC2016.pdf. Vigente junio 2017.
6. PixiJS. (2017). The HTML5 Creation Engine. Recuperado de http://www.pixijs.com/. Vigente junio 2017.
7. Sanchez, A. J., Perez-Lezama, C. y Starostenko O. (2015). A Formal Specification for the Collaborative Development of Learning Objects. Procedia - Social and Behavioral Sciences. 182, 726–731. Recuperado de http://www.sciencedirect.com/science/article/pii/S1877042815030955. Vigente junio 2017.
8. Storybook. (2017). Storybook: The UI Development Environment You'll ♥ to use. Recuperado de https://storybook.js.org/. Vigente junio 2017.
9. Wing, J. M. (2006). Computational thinking. Communications of the ACM, 49:33–35. [versión electrónica]. Recuperado de: https://goo.gl/CHS4Yp. Vigente junio 2017.