9945 PROGRAMACIÓN EN ENTORNOS VISUALES, TÉCNICAS PARA LA INICIACIÓN A LA PROGRAMACIÓN Y UNA REFERENCIA A EXPERIENCIAS REALIZADAS CON DISTINTAS FRANJAS ETARIAS

Mauro Nelson Ziehlke[(1)(2)], Carlos A. Talay[(1)(3)]

[(1)]Universidad Nacional de la Patagonia Austral

Río Gallegos, Argentina

[(2)]mauroziehlke@gmail.com

[(3)]carlostalay@yahoo.com.ar

Resumen: En este trabajo detallaremos aspectos de la programación en entornos visuales basados en las herramientas de desarrollo Scratch y App Inventor. Estas herramientas fueron probadas con un variado segmentos de usuarios que abarcaron público con edades que van de los 9 a 50 años, sin experiencia alguna en programación, y desarrollados en la localidad de Río Gallegos.

Palabras Clave: PROGRAMACIÓN, ENTORNOS VISUALES, SCRATCH, APPINVENTOR.

Introducción

Los métodos tradicionales de enseñanza de programación, implican un gran trabajo previo antes de realizar el primer programa.

Para introducir los principios de orden secuencial, de instrucciones y las estructuras de control básicas de programación como “if”, “for”, “while”, se utiliza pseudocódigo y diagramas de flujos, por lo que el alumno deberá soslayar un buen tiempo comprendiendo estos conceptos antes de llegar a probar los primeros ejemplos en una computadora, sin tener una plena conciencia de lo que significa programar.

En paralelo o a continuación de esto se deben incorporar algunos conceptos básicos de números binarios, comprender definiciones como bit, byte, Word, etc. entendiendo sus diferencias y usos, para poder inicializar variables que luego utilizará.

Una vez incorporados estos conceptos, los alumnos pasan a escribir sus primeros programas. Aquí comienzan otra serie de dificultades, deben aprender la nomenclatura de cada instrucción a utilizar; tener especial atención a la sintaxis que impone el lenguaje utilizado y también comprender como se realiza la interacción con el compilador de ese lenguaje, debiendo interpretar los mensajes de error que seguramente tendrán al proceder a depurar un programa.

Una vez superados todos estos pasos, llevados adelante con un poco de esfuerzo, seguramente el resultado será la elaboración de su primer programa: en la pantalla de computador, aparecerá el texto “Hola Mundo” y eso será el inicio del alumno al mundo de la programación.

Es así que podemos inferir que con un mundo en el que la demanda de empleos vinculados a la informática, y en particular a profesionales que puedan programar de

manera fluida, esta forma de enseñar es demasiado lenta, pone a prueba la paciencia y el interés de los alumnos haciendo poco dinámica la manera de aprender a programar.

En los últimos años han aparecido nuevos lenguajes, con entornos gráficos, donde casi no se requiere escribir texto, tener conocimientos de número binarios, ni preocuparnos por las sintaxis que obligan a arduos trabajos de depuración, por lo menos en los comienzos.

Los lenguajes

Scratch 2.0

Figura 1.

Scratch [1] es un lenguaje de programación con una sencilla interfaz gráfica, donde se puede crear historias interactivas, juegos y animaciones. Los usuarios pueden compartir sus creaciones con otros usuarios alrededor del mundo. En el proceso de diseño de proyectos y programación, los principiantes aprenden a pensar creativamente, razonar sistemáticamente y trabajar de manera colaborativa [2].

La primera versión de Scratch, disponible solo en versión de escritorio, fue desarrollada y publicada en 2003 conjuntamente por el MIT Media Lab, dirigido por Mitchel Resnick, y la compañía Playful Invention Company, con sede en Montreal y cofundada por él junto a Brian Silverman y Paula Bonta. El propósito fue ayudar a la gente joven, principalmente con edades a partir de los ocho años, a aprender a programar. Desde 2013, Scratch 2 está disponible en línea y como aplicación de escritorio para Windows, OS X y Linux [3].

MIT App Inventor

MIT App Inventor [4] es un una manera innovadora de introducir a principiantes en la programación y creación de aplicaciones para celulares. Este transforma el complejo lenguaje de codificación basado en texto, en un entorno visual con bloques constructivos que se insertan con solo arrastrar y soltar.

La sencilla interfaz gráfica otorga incluso a un principiante inexperto la capacidad de crear aplicaciones para celulares, básicas y totalmente funcionales, en una hora o menos [5].

Figura 2.

Mark Friedman, de Google, y el profesor del MIT, Hal Abelson, encabezaron el desarrollo de App Inventor en 2009. Otros colaboradores de ingeniería de Google fueron Sharon Perl, Liz Looney y Ellen Spertus. App Inventor funciona como un servicio web administrado por el personal del Centro para el Aprendizaje Móvil del MIT, una colaboración del Laboratorio de Informática e Inteligencia Artificial (CSAIL) del MIT y el Laboratorio de Medios del MIT [6].

En 2015, la comunidad MIT App Inventor constaba de casi 3 millones de usuarios que representan a 195 países. Más de 100.000 usuarios semanales activos han construido más de 7 millones de aplicaciones para Android. Como una herramienta de código abierto que busca que la programación y la creación de aplicaciones sean accesibles a una amplia gama de audiencias [7].

Otros lenguajes

Existen otros lenguajes o entornos visuales para programación como miniBloq [8], probots (Mis Ladrillos) [9], S4A [10], etc.

Se decidió utilizar Scratch y App Inventor por la mayor similitud entre instrucciones de lenguajes tradicionales de código y estos nuevos entornos.

Experiencia desarrollada

Se han realizado cursos con el objeto de enseñar a programar aplicaciones para teléfonos celulares, video juegos y robots con la plataforma de prototipos electrónica

de código abierto (open- source) Arduino [11]; y los lenguajes de programación de entorno gráfico, diseñado por el MIT, Scratch y App Inventor.

De los cursos, 7 fueron realizados con la app inventor, destinado a personas de 15 a 50 años sin ninguna experiencia en programación y otros 7 fueron realizados con Scratch para alumnos de 9 a 14 años.

Los cursos tenían una duración variable, de 8 a 12 hs. reloj de cursado, con una modalidad casi en su totalidad, de tipo práctica. Es decir que la teoría la realizamos instantes antes de realizar la práctica.

App Inventor

Las primeras horas se trabaja sobre el concepto de orden secuencial para realizar un programa y la ejecución de acciones al trabajar por eventos. Se utilizan herramientas de uso directo que no requieren algoritmos complejos para su uso.

El segundo paso es incorporar algunas estructuras de control como “if” y “for”, para poder darle opciones de control sobre el programa.

Se suman a esto, en primer lugar el manejo de variables para guardar datos y para uso como índice con incremento +1 (ej.: variable=variable+1;)

Una vez que el usuario maneja el concepto de variable única, se incorpora el concepto de “pila” o “lista”, que también se utiliza para almacenar información.

Al incorporar el uso de variables y listas, se aplica también el concepto de inicialización de variables.

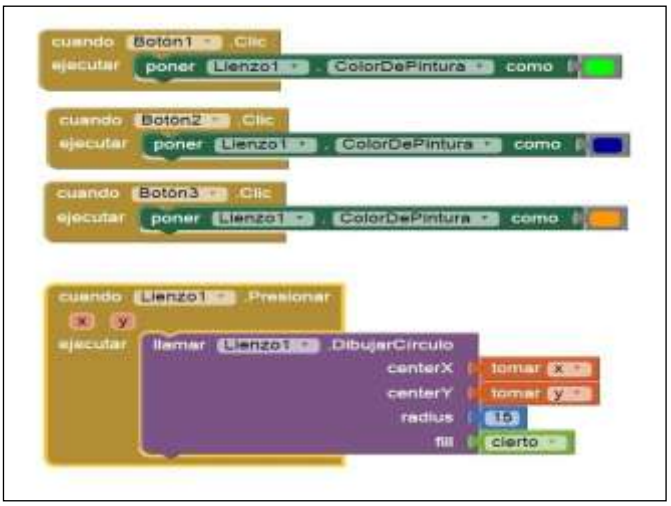

Figura 3.

Para poder almacenar información en tiempo de ejecución se utiliza una herramienta de guardado de datos en base de datos, por lo que también se dan las primeras orientaciones en manejo de información en base de datos.

Como etapa final del curso se incorporan herramientas para realizar video juegos o animaciones que ya vienen incluidas en el software.

En muchas de estas herramientas, si bien no se logra una gran profundidad en el nivel de conocimiento, se dan las primeras bases para que el programador comience a trabajar y luego profundice a medida que lo necesita.

Estos cursos se dictaron sobre el rango etario más amplio y se generaron algunas aplicaciones basadas en el interés de los alumnos, lo que facilitó el aprendizaje y la participación.

Scratch

De forma similar al app inventor, primero se inicia enseñando los conceptos de secuencialidad para realizar acciones y el concepto de ejecución de tareas por eventos. Estos dos conceptos son claves para poder programar en estos lenguajes.

Para los primeros programas se genera un evento que desencadena todas las demás acciones que luego se generaran en forma secuencial. Estos programas son enfocados en controlar movimientos simples del personaje principal de Scratch. Primeramente se le enseña un algoritmo simple de 2 a 5 instrucciones para ejecutar en forma secuencial, luego se le pide al alumno que realice alguna modificación a algún parámetro de ese algoritmo simple para ajustar los movimientos y por último se le pide que modifique el algoritmo para ejecutar otra rutina, sin la asistencia del profesor.

En segunda instancia se incorpora el concepto de repetición de instrucciones, instrucciones “for” o “while” en programación tradicional. En el caso de Scratch, la instrucción se llama “repetir” y se puede indicar la cantidad de ciclo a repetir o bien hacer un caso llamado “repetir siempre”, que realiza un lazo infinito. Este caso se incorpora muy rápido con en ejercicio donde se hace caminar al personaje principal de Scratch.

El siguiente paso es enseñar el concepto “if” o “si”, que nos permiten tomar decisiones sobre el código.

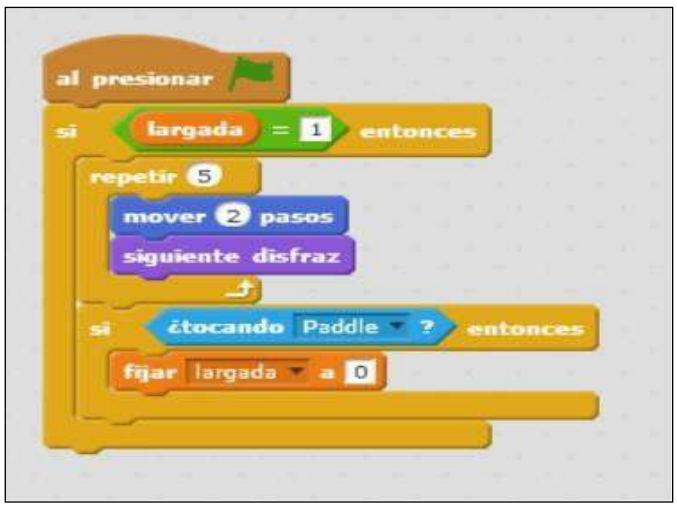

Figura 4.

Estas instrucciones de control se enseñan a través de ejemplos de control de movimiento del personaje principal, otra vez aplicando el método de hacer un ejercicio explicado por el profesor, otro semi-guiado y el último que sea modificado por el alumno sin intervención del profesor.

Tanto en Scratch como en app inventor, todos estos pasos pueden enseñarse sin la necesidad de crear variables, por lo que resulta muy didáctico.

Al momento de crear variables, normalmente es un concepto bastante difícil de explicar y aplicar de forma rápida, pero con estos entornos no se requiere explicar/especificar el tipo de dato a utilizar, sino que se declara la variable y se la utiliza con cualquier tipo de dato, por lo que la comenzamos a utilizar haciendo un juego. La variable es el lugar donde se almacenan las vidas y otra para los puntos. Con esto se logra introducir rápidamente el concepto en los niños.

También se utilizan las variables para incorporar los conceptos de “bit de estado” o “banderas”, para saber el estado de un proceso o comunicar dos objetos, por ejemplo en un juego para indicar que una pelota choca un objeto. Si bien Scratch tiene algunas herramientas ya diseñadas para esto, se prefiere enseñar con una variable, para hacerlo más parecido a los lenguajes de programación tradicionales.

Duración de las Clases

Para el dictado de los cursos se fueron probando varias metodologías y se fue variando la duración de los encuentros, buscando mejorar la performance.

Los primeros cursos se dictaron en 4 encuentros, 1 vez a la semana, de 2 horas de duración. Con personas adultas, sin conocimientos previos. La cantidad de horas de duración era buena pero al pasar una semana entre una clase y la otra, se debía retomar algunos conceptos de las clases anteriores.

Se dictaron cursos con 3 encuentros semanales de 1:30 horas de duración, durante 3 semanas. También en personas adultas. Estos cursos tuvieron muy buena respuesta de los alumnos, adquiriendo bien los conocimientos.

Se dictaron cursos a adultos en un solo encuentro de 9 horas, con un descanso para almorzar. Resulto muy desgastante para los alumnos. Se recomienda solo para casos especiales donde no se pueda asistir en forma regular.

Se dictaron cursos a niños, de 2 encuentros semanales de 2 horas, durante 3 semanas, también con buenos resultados en cuanto a conocimientos, pero con un poco de dispersión de los chicos, ya que no resulta sencillo que se mantengan enfocados durante todo el tiempo de clase.

Se dictaron cursos a niños, de 3 encuentros consecutivos de 3 horas cada uno. Se logró un rápido aprendizaje, pero también se observaba dispersión de los alumnos en la clase.

Se dictaron cursos a niños, en 2 encuentros semanales de 1 horas, durante 4 semanas, observando una muy buena respuesta de los alumnos.

En todos los casos se observó muy poca práctica luego de las clases, por lo que se debió reforzar la práctica en clase, acotando el tiempo de los ejercicios.

Trabajo grupal

Para el dictado se trabajó con alumnos en forma individual, con grupos de 2 alumnos por PC y con grupos de 3 alumnos por PC.

En los grupos con alumnos individuales se marcaban mucho las diferencias en la velocidad de aprendizaje, provocando que los alumnos más avanzados se aburran en los momentos en que se debía repasar algunos conceptos con otros.

En los grupos de 2 alumnos, se logró un trabajo y avance parejo de todos los grupos, lográndose la mejor performance.

En los grupos de 3 alumnos se observaba que 2 de ellos trabajaban bien, pero el tercero se dispersaba y no lograba incorporar la suficiente práctica.

Luego de los primeros cursos se incorporó un sistema cartas con desafíos, para que los alumnos que avanzaran más rápido se mantengan ocupados, mientras se repasa conceptos con otros.

Conclusiones

Como vemos en forma constante las tecnologías cambian, nuevos entornos de aprendizaje se desarrollan, la educación evoluciona y la aparición de herramientas de aprendizaje como estas nos plantea nuevos desafíos a los cuales debemos adaptarnos a fin de intentar mejores resultados en el proceso de aprendizaje de los alumnos.

Los lenguajes con entornos visuales rompen con las estrategias tradicionales de enseñanza, facilitando la incorporación de los complejos requerimientos que requiere la programación.

Con su estructura colorida, icónica y con instrucciones intuitivas, resulta atractiva y atrapante para el alumno principiante. Las comunidades en línea permiten trabajar colaborativamente a escala mundial.

La modalidad de clase práctica permite captar la atención del alumno durante más tiempo. Para esta modalidad se recomienda trabajar en grupos de 2 alumnos (principalmente cuando se trate de niños), en clases de 1 a 2 horas de duración e incorporar algún sistema para darle nuevos desafíos a los alumnos que avanzan a mayor velocidad.

La duración del curso en 8 a 12 horas no resulta suficiente para incorporar de forma fehaciente todos los conceptos vertidos, se recomienda realizar cursos de mayor duración.

Referencias

[1] Lifelong Kindergarten research group. Getting Started with Scratch version 2.0. MIT Media Lab https://llk.media.mit.edu/. 2013.

[2] https://scratch.mit.edu/info/faq. 2017.

[3] Gil Alcántara, E. Scratch en el Aula. Microsoft en la educación. https://education.microsoft.com/Story/Lesson. 2017.

[4] Wolver, D., Abelson, H., Spertus, E., Looney, L., App Inventor 2: Create your own Android Apps. 2a Edición. O´really Media Inc.2015.

[5] http://appinventor.mit.edu/explore/about-us.html. 2017.

[6] http://appinventor.mit.edu/explore/about-us.html. 2017.

[7] http://appinventor.mit.edu/explore/about-us.html. 2017.

[8] http://blog.minibloq.org. 2017.

[9] http://misladrillos.com/magento/index.php/educativa.2017.

[10] http://s4a.cat/index_es.html. 2017.

[11] https://www.arduino.cc/. 2017.