Programación educativa con Scratch

Programación para Todos

2010-02-01

Programación para Todos

> *Artículo escrito por el equipo de MIT, desarrollador de Scratch. En él, argumentan por qué es importante que los estudiantes aprendan a programar y explican las tres premisas básicas de este entorno de programación: ser flexible (thinkerable), significativo y socialmente interactivo.*

Artículo escrito por el equipo de MIT, desarrollador de Scratch. En él, argumentan por qué es importante que los estudiantes aprendan a programar y explican las tres premisas básicas de este entorno de programación: ser flexible (thinkerable), significativo y socialmente interactivo.

-

PROGRAMACIÓN PARA TODOS

Mitchel Resnick, John Maloney, Andrés Monroy-Hernández, Natalie Rusk, Evelyn Eastmond, Karen Brennan, Amon Millner, Eric Rosenbaum, Jay Silver, Brian Silverman, Yasmin Kafai.

http://cacm.acm.org/magazines/2009/11/48421-scratch-programming-for-all/fulltext

“La competencia digital” debe hacer referencia a diseñar, crear y remezclar; no simplemente a navegar, comunicar e interactuar.

Cuando Moshe Y. Vardi, Editor en Jefe de *Comunicaciones*, nos invitó a escribir este artículo recordó cómo conoció Scratch: “Uno de mis colegas, Profesor de ciencias de la computación [1], había intentado sin éxito que su hija de 10 años se interesara en programación, con lo único que pudo captar su atención fue con Scratch”.

Es esto precisamente lo que queríamos lograr cuando decidimos desarrollar Scratch hace seis años; desarrollar un acercamiento a la programación de computadores que atrajera a la gente que previamente no había imaginado que podía ser programador. Queríamos facilitar la programación a todas las personas, independientemente de su edad, experiencia previa e intereses, para que pudieran programar sus propias historias interactivas, juegos, animaciones y simulaciones y además, compartir sus creaciones con otros.

Desde su lanzamiento al público en Mayo de 2007, el sitio Web de Scratch (http://scratch.mit.edu) se ha convertido en una vibrante comunidad en línea, integrada por personas que comparten, intercambian ideas y remezclan programas propios con los de otros. Se ha llamado a Scratch “el YouTube de los medios interactivos”. Cada día, los “Scratchers” alrededor del mundo, suben al sitio más de 1.500 proyectos nuevos, con el código fuente de acceso libre para permitir compartir y remezclar. La colección de proyectos del sitio es muy diversa e incluye juegos de video, boletines de noticias interactivos, simulaciones de ciencias, tures virtuales, tarjetas de cumpleaños, concursos de animaciones de bailes y tutoriales interactivos, todos programados en Scratch.

La edad del grueso de la audiencia del sitio se ubica entre los 8 y los 16 años (con un pico en los 12 años) aunque un grupo considerable de adultos participa también. A medida que los “Scratchers” programan y comparten proyectos interactivos, aprenden importantes conceptos matemáticos y de computación; además de aprender a pensar creativamente, razonar sistemáticamente y trabajar colaborativamente: todas estas, habilidades esenciales para el Siglo XXl. De hecho, nuestro objetivo principal no es preparar a las personas para hacer carrera como programadores profesionales sino cultivar una nueva generación de pensadores creativos y sistemáticos que se sientan cómodos programando para expresar sus ideas.

En este artículo nos referimos tanto a los principios de diseño que sirven de guía a nuestro desarrollo de Scratch como a nuestras estrategias para hacer la programación accesible y enganchadora para todos. Pero primero, queremos dar una idea más clara de cómo se está usando Scratch describiendo una serie de proyectos desarrollados por una niña de 13 años que se registró en el sitio Web de Scratch con el nombre de

“BalaBethany”.

A “BalaBethany” le gusta dibujar personajes animados. Así que, cuando comenzó a utilizar Scratch, programó historietas animadas en las que aparecían estos personajes. Comenzó a compartir sus proyectos en el sitio Web de Scratch y otros miembros de la comunidad respondieron positivamente a sus publicaciones, escribiendo comentarios halagadores a sus proyectos. Decían, “Sorprendente” y “OMG I LUV IT!!!!!!"), acompañados por preguntas sobre cómo había logrado ciertos efectos visuales. Por ejemplo, ¿cómo logras que un Objeto se vuelva transparente? Estimulada, “BalaBethany” comenzó a crear y a compartir regularmente nuevos proyectos de Scratch, tales como episodios de series de Televisión.

Periódicamente agregaba nuevos personajes a sus series y en un momento dado se preguntó ¿por qué no involucrar a toda la comunidad de Scratch en el proceso? Ella creó y subió un nuevo proyecto en el que anunciaba un “concurso” en el que solicitaba a otros miembros de la comunidad, diseñar una hermana para alguno de los personajes (Ver Figura 1). El proyecto listaba una serie de requerimientos para el nuevo personaje, incluyendo “debe tener el pelo de color rojo o azul, favor escoger” y “debe parecer un gato o tener cachos de carnero o una combinación de ambos”.

El proyecto recibió más de 100 comentarios. Uno de ellos de un miembro de la comunidad que deseaba concursar pero decía no saber cómo dibujar personajes animados. Entonces BalaBethany generó otro proyecto de Scratch, con un tutorial paso a paso, indicando un conjunto de 13 pasos para dibujar y colorear personajes animados.

En el lapso de un año BalaBethany programó y compartió más de 200 proyectos de Scratch, cubriendo con ellos una serie de temas (cuentos, concursos, tutoriales y más). Sus habilidades tanto de programación como artísticas se acrecentaron y sus proyectos claramente tuvieron eco dentro de la comunidad de Scratch pues recibieron más de 12.000 comentarios.

Figura 1: Imágenes de pantalla de la serie animada, el concurso y el tutorial de "BalaBethany".

**

¿POR QUÉ PROGRAMAR?**

Se ha vuelto lugar común referirse a las personas jóvenes como "nativos digitales" debido a su aparente fluidez con las tecnologías digitales (TIC) [2]. En realidad muchos jóvenes se sienten muy cómodos enviando mensajes de texto, interactuando con juegos en línea y navegando la Web. ¿Pero los hace esto realmente competentes en TIC? Aunque interactúan con los medios digitales todo el tiempo, pocos son capaces de crear sus propios juegos, animaciones o simulaciones. Es como si pudieran "leer" pero no "escribir". Desde nuestro punto de vista, la competencia digital requiere no solamente tener habilidad para chatear, navegar o interactuar sino también la habilidad de diseñar, crear e inventar con los nuevos medios [3], tal como lo evidencia "BalaBethany" con sus proyectos.

Para hacer lo anterior usted necesita aprender algún tipo de programación. La habilidad para programar ofrece importantes beneficios. Por ejemplo, expande considerablemente el rango / posibilidades de lo que usted puede crear (y las posibilidades de auto expresión) con el computador. También expande el rango de lo que usted puede aprender. En particular, programar apoya el "pensamiento computacional", que ayuda a las personas a aprender estrategias importantes de solución de problemas y de diseño (tales como, modularización y diseño iterativo) que conducen a dominios externos a la programación [4]; Además, como programar involucra el crear representaciones externas de procesos para solucionar problemas, la programación ofrece oportunidades para reflexionar sobre el propio pensamiento y aún para pensar en el proceso mismo de pensar [5]. ****

INVESTIGACIÓN PREVIA

Cuando se introdujeron los computadores personales al final de la década de los años 70 y 80, hubo un entusiasmo inicial por enseñar a todos los niños a programar. Miles de colegios enseñaron a millones de estudiantes a escribir programas sencillos en Logo o en Basic [6]. En el libro *"Mindstorms"* [7] publicado en 1980 por Seymour Papert, se presentaba a Logo como la piedra angular para repensar los enfoques tanto para la educación como para el aprendizaje.

Aunque algunos niños y maestros se animaron y transformaron con esas nuevas posibilidades, la mayoría de las Instituciones Educativas pronto redireccionaron los computadores hacia otros usos. Desde entonces, los computadores se han vuelto omnipresentes en la vida de los niños, pero pocos de ellos aprenden a programar. Hoy en día muchas personas ven la programación de computadores como una actividad técnica puntual, apropiada solo para un pequeño segmento de la población.

¿Qué pasó con el entusiasmo inicial para que los niños aprendieran a programar? ¿Por qué Logo y otras iniciativas no cumplieron sus promesas iniciales? Hubo varios factores:

Los primeros lenguajes de programación eran muy difíciles de usar y muchos niños simplemente no pudieron dominar la sintaxis de la programación;

- Con frecuencia, la programación se presentaba con actividades como, generar listas de números primos y hacer dibujos lineales sencillos, que no tenían conexión con los intereses y experiencias de los jóvenes; y

- En muchos casos, la programación se presentaba en contextos en los que nadie podía u ofrecer orientación cuando se presentaban problemas o estimular una exploración más profunda cuando las cosas marchaban bien.

Papert argumentaba que los lenguajes de programación debían tener un "piso bajo" esto es ser fáciles de iniciar y un "techo alto" (ofrecer oportunidades para crear en el tiempo proyectos cada vez más complejos). Además, el lenguaje necesitaba "paredes amplias" que le permitieran soportar gran cantidad de proyectos diferentes de manera

que las personas con distintos intereses y estilos de aprendizaje quisieran utilizarlo. Satisfacer esta tripleta de piso bajo, techo alto y paredes amplias, no ha sido fácil [8].

En años recientes, nuevos abordajes han buscado interesar en programación a niños y jóvenes [9]. Algunos se han enfocado en lenguajes de programación profesionales como el ActionScript de Flash; otros utilizan lenguajes nuevos como Alice [10] y Squeak Etoys [11], desarrollados especialmente para programadores jóvenes. Estos han inspirado e influido en nuestro trabajo con Scratch. Pero no estábamos satisfechos con las opciones existentes. En particular, sentíamos que era importante bajar aún más el piso y hacer que las paredes fueran todavía más amplias pero que al mismo tiempo que se siguiera apoyando el desarrollo del pensamiento computacional [12].

Para lograr estas metas, establecimos para Scratch tres principios fundamentales: Hacerlo más "tinkerable" [13], es decir flexible, significativo y con mayor interacción social que otros entornos de programación [*Nota de Eduteka: la traducción más cercana en español del término tinkerable, para darle el sentido con que se usa en este documento, es ser flexible o elástico entendido como "que no se sujeta a normas estrictas, a dogmas o a trabas", "susceptible de cambios o variaciones según las circunstancias o necesidades", "acomodaticio, que puede ajustarse a muy distintas circunstancias", "que admite muchas interpretaciones"*]. En las secciones siguientes, explicaremos cada uno de estos principios que guiaron nuestro diseño de Scratch.

MÁS FLEXIBLE "TINKERABLE" [13]

Nuestro grupo de investigación en el "Life Long Kindergarten" (Kindergaden para toda la vida) del Laboratorio de Medios de MIT (http://llk.media.mit.edu) viene trabajando desde hace muchos años con la compañía Lego (http://www.lego.com/), ayudando a desarrollar el "Lego Mindstorms" y otros conjuntos de herramientas (kits) [14]. Siempre nos ha intrigado e inspirado la manera en que los niños juegan y construyen con las fichas/bloques/ladrillos de Lego. Si tienen una caja llena de estos, inmediatamente comienzan a experimentar sin restricciones (tinkering), ensamblando al azar unos pocos bloques y la estructura resultante les da inmediatamente nuevas ideas. A medida que juegan y construyen, evolucionan orgánicamente planes y metas y simultáneamente lo hacen estructuras e historias.

Quisimos que el proceso de programar computadores con Scratch ofreciera una experiencia similar. La gramática de Scratch se basa en un conjunto de “bloques gráficos de programación” que los niños ensamblan para crear programas (ver Figura2). Tal como con las fichas/bloques de LEGO, conectores en los bloques sugieren de qué manera pueden estos ensamblarse. Los niños pueden comenzar simplemente a experimentar con los ladrillos, ensamblándolos en diferentes secuencias y combinaciones para observar qué pasa. No hay nada de la oscura sintaxis o de la puntuación de los lenguajes de programación tradicionales. El piso es bajo y la experiencia lúdica.

Figura 2: Ejemplo de secuencias de comandos de Scratch

Los bloques de Scratch se diseñan para que solamente encajen de maneras que hagan sentido sintácticamente. Las estructuras de Control tales como, ***por siempre*** y ***repetir***, tienen forma de C y con ella se quiere sugerir que los bloques deben ponerse dentro de estas. Los bloques con valores de salida tienen formas acordes con el tipo de valor que retornan: óvalos para valores numéricos y hexágonos para valores Boleanos. Los bloques condicionales tales como ***si*** y ***repetir hasta que***, tienen espacios de forma hexagonal, para indicar que se requiere poner en ellos un Boleano.

El nombre mismo de “Scratch” resalta la idea de la experimentación flexible, pues provienen de la técnica de “rayar” (Scratching) utilizada por los “disc jockeys” de hip-hop que experimentan con la música, girando con sus manos para adelante y para atrás, discos de vinilo, para mezclar clips de música que juntan de maneras creativas. Al programar con Scratch, la actividad es similar pues mezcla gráficas, animaciones, fotos, música, sonido.

Scratch se diseñó para ser altamente interactivo. Simplemente con hacer clic en una pila de bloques (guión) esta inmediatamente empieza a ejecutar su código. Usted puede además hacer cambios en la pila cuando se está ejecutando, de manera que

sea fácil experimentar con nuevas ideas de manera incremental e iterativa. ¿Quiere crear hilos paralelos? Simplemente genere múltiples pilas de bloques. Nuestra meta es hacer la ejecución paralela tan intuitiva como la ejecución secuencial.

Figura 3: Interfaz de usuario de Scratch

El área de programas en la interfaz de Scratch está pensada para usarse como un escritorio físico (ver Figura 3). Usted puede hasta dejar en él bloques o pilas extras que puede necesitar más adelante. El mensaje implícito es que es aceptable ser un poco desordenado y experimentar.

La mayoría, tanto de lenguajes de programación, como de clases de ciencias de la computación, privilegian la planeación de arriba hacia abajo versus la experimentación flexible de abajo hacia arriba. Con Scratch queremos que los cacharreros (experimentadores) se sientan tan confortables como las personas que planean.

El énfasis en diseño iterativo, incremental se alinea con nuestro propio estilo de desarrollo al crear Scratch. Escogimos “Squeak” como lenguaje de implementación ya que se adapta bien para realizar prototipos rápidos y diseños iterativos. Antes de lanzar Scratch en el 2007, continuamente hicimos ensayos de campo con prototipos en entornos del mundo real, revisándolo una y otra vez en base a retroalimentación y

sugerencias provenientes de pruebas de campo [15].

MÁS SIGNIFICATIVO

Nosotros sabemos que las personas aprenden mejor y disfrutan más, cuando trabajan en proyectos que tienen para ellas significado personal. Así que en el desarrollo de Scratch le dimos prioridad alta a dos criterios de diseño:

- *__Diversidad__**. *Para que pudiera usarse en proyectos muy diversos (historias, juegos, animaciones, simulaciones), de manera que las personas dentro de un rango de intereses muy variado pudieran todas trabajar en proyectos que les interesaran; y

- *__Personalización__**. *Facilitarque las personas personalizaran sus proyectos de Scratch importando fotos y clips de música, grabando voces y creando gráficas [16].

Estas prioridades influenciaron muchas de nuestras decisiones de diseño. Decidimos por ejemplo enfocarnos en imágenes 2D en lugar de las de 3D, pues es mucho más fácil para las personas crear, importar y personalizar trabajos artísticos en 2 dimensiones (2D). Aunque algunas personas puedan considerar el estilo 2D de los proyectos de Scratch como algo pasado de moda, colectivamente los proyectos de Scratch exhiben una diversidad visual y una personalización, ausentes de los ambientes de autor para 3D.

El valor de la personalización está muy bien expresado en la siguiente entrada de blog hecha por un Científico de la Computación (computer scientist) quién presentó Scratch a sus dos hijos: “*debo admitir que al principio no entendí por qué un lenguaje de programación para niños debía estar centrado en medios (audio, video, texto), pero después de ver a mis hijos interactuando con Scratch esto se hizo para mi mucho más claro. Una de las mejores cosas que observé con Scratch es que personalizaba la experiencia de desarrollo de nuevas maneras pues facilitaban a mis hijos agregar contenido personalizado y participar activamente en el proceso. No solamente, podían desarrollar programas abstractos para hacer cosas sencillas como un gato, una caja,*

etc. Sino que podían agregar sus propias fotografías y voces al entorno de Scratch; esto les ha dado muchas horas de diversión y los ha llevado a aprender".

Nos continúa sorprendiendo la diversidad de los proyectos que se suben al sitio Web de Scratch. Era de esperarse que hubiera muchos juegos, que fueran desde recrear con mucho esfuerzo versiones de los videojuegos favoritos, tales como "*Donkey Kong*" , hasta otros totalmente originales. Pero también se encuentran proyectos de muchos otros tipos (ver Figura 4). Algunos de ellos documentan experiencias de vida, tales como vacaciones en familia; otros, documentan experiencias imaginarias que se desearía tener, como hacer un viaje para conocer a otros "Scratchers". Algunos de los proyectos, tales como tarjetas de cumpleaños y mensajes de aprecio, tienen por objeto cultivar relaciones. Otros se diseñan para aumentar la conciencia sobre temas sociales tales como calentamiento global o maltrato animal. Durante la elección presidencial norteamericana en el 2008, un gran número de proyectos tuvieron como personajes a Barack Obama y a John McCain. Posteriormente, otra serie de proyectos promovieron algunos miembros de la comunidad de Scratch en línea para el cargo, no bien definido, de "Presidente de Scratch".

Figura 4: Imágenes de ejemplos de proyectos Scratch

Algunos de los proyectos atendieron actividades escolares. Para una clase de Ciencias de la Tierra, un muchacho Indio de 13 años, creo un proyecto en el cual un personaje animado viajaba al centro de la tierra, con una grabación de voz sobrepuesta (voz en off) que describía las diferentes capas que iba encontrando en su camino. Cómo parte

de una clase de Ciencias Sociales, un joven de 14 años de New Jersey (USA), creó una simulación de la vida en la isla de Pascua (Rapa Nui), con el objeto de ayudar a otros a aprender sobre la cultura y la economía locales.

A medida que los “Scratchers” trabajan en proyectos con significado personal, encontramos que estaban listos y motivados para aprender conceptos matemáticos y de computación importantes, relacionados con sus proyectos.

Veamos el caso de Raúl, un joven que estaba creando un juego interactivo en un centro de jornada escolar complementaria [17]. Generó las gráficas y las acciones básicas para este pero no sabía cómo llevar el puntaje del resultado. Cuando uno de los investigadores de nuestro equipo visitó ese centro, Raúl le pidió ayuda. El investigador le indicó cómo crear una variable en Scratch y él inmediatamente entendió cómo podía usarla para llevar el puntaje del juego. Comenzó a jugar con los bloques para incrementar las variables y luego lo buscó emocionado y le estrechó la mano diciendo: “¡Gracias, gracias, gracias!!”. El investigador se preguntó ¿cuántos profesores de álgebra de octavo grado reciben un agradecimiento tan efusivo de sus estudiantes por enseñarles variables?.

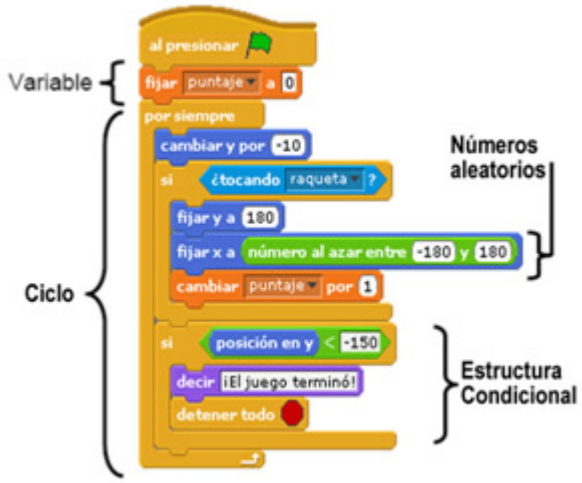

Figura 5: Ejemplo de de una secuencia de comandos (del juego Pong), en el que se resaltan conceptos matemáticos y de computación.

MÁS SOCIAL

Los desarrollos en el lenguaje de programación Scratch van de la mano con el desarrollo de su sitio Web [18]. Para que Scratch tuviera éxito, el lenguaje necesitaba

enlazarse con una comunidad en la que las personas pudieran apoyarse, colaborar y criticarse, además de construir usando el trabajo realizado por otros [19].

El concepto de compartir, hace parte integral de la interfaz de usuario de Scratch la cual tiene, en la parte superior de la página, tanto un menú como un icono de "Compartir" claramente visibles. Si se da clic sobre el icono de Compartir, su proyecto sube al sitio Web de Scratch y se muestra en el primer lugar de la sección de "Proyectos más Recientes" de la página. Una vez subido el proyecto al sitio Web, cualquiera puede ejecutarlo en un navegador, usando un reproductor basado en Java, hacerle comentarios, votar por él (presionando el botón "¿Me encanta?") o descargarlo para mirarlo y revisar sus códigos. (Todos lo proyectos compartidos en el sitio están amparados por licencia Creative Commons).

Tres principios básicos del diseño de Scratch: hacerlo más flexible (tinkerable), más significativo y más social que otros entornos de programación

Figura 6: Sitio Web de Scratch

En los 32 meses siguientes al lanzamiento de Scratch, se compartieron más de 665.000 proyectos en el sitio Web [20]. Para muchos “Scratchers” la oportunidad de tener una audiencia amplia para sus proyectos y recibir tanto retroalimentación como consejos de otros “Scratchers”, es una motivación fuerte. La extensa librería de proyectos con la que cuenta el sitio, sirve también de inspiración. Explorando los proyectos en ella existentes los “Scratchers” tienen ideas para nuevos proyectos y aprenden nuevas técnicas de programación.

Marvin Minsky dijo una vez que Logo tenía una magnífica gramática pero no mucha literatura [21]. Así como los escritores jóvenes se inspiran muchas veces con la lectura de obras maestras de la literatura, no existía el equivalente para proyectos de Logo excelentes que sirvieran de inspiración a los jóvenes programadores. El sitio Web de Scratch constituye el inicio de esa “literatura” para Scratch.

El sitio es también tierra fértil para la colaboración. Los miembros de la comunidad constantemente prestan, adaptan y construyen con las ideas, imágenes y programas de otros. Más del 15% de los proyectos son re-mezclas de otros proyectos que están en el sitio. Por ejemplo, hay docenas de ejemplos del juego “Tetris”, pues los “Scratchers” continúan adicionándole nuevas funcionalidades y tratando de mejorar su “Jugabilidad”. También hay docenas de proyectos de muñecas para vestir, solicitudes y concursos, todos adaptados de proyectos ya existentes.

Inicialmente, algunos “Scratchers” se molestaron cuando otros re-mezclaron sus proyectos, quejándose de que se los estaban “robando”. Esto generó discusiones en los foros del sitio Web sobre el valor de compartir y sobre las ideas subyacentes en las comunidades que usan código abierto. Nuestra meta es generar una cultura en la que los “Scratchers” (usuarios de Scratch) se sientan orgullosos y no molestos, cuando otros adaptan y re-mezclan sus proyectos. Continuamente le estamos adicionando al sitio nuevas funcionalidades que apoyen y estimulen esta forma de pensar. Ahora, cuando alguien re-mezcla un proyecto, el sitio automáticamente adiciona un enlace al proyecto original, para darle crédito al autor original. Además, cada proyecto incluye enlaces a sus “derivados” (proyectos re-mexclados de este) y los mejores proyectos resultantes de estas “Re-mezclas” se ubican en lugar destacado de la página de inicio de Scratch.

Algunos proyectos se enfocan en el sitio mismo, comentan y analizan otros proyectos que existen en él. Uno de los primeros ejemplos se denominó SNN, sigla en inglés para Red de Noticias de Scratch. Usando el gato, imagen por defecto de Scratch, se daban noticias sobre Scratch de manera muy parecida a como las da un presentador de CNN. Inicialmente nos pareció que "se estaba simulando un noticiero" pero pronto nos dimos cuenta que era un verdadero noticiero que ofrecía noticias de interés a una comunidad real, la comunidad de Scratch en línea.

Este proyecto inspiró otros y generó la proliferación de Boletines en línea, revistas y programas de Televisión, todos ellos programados en Scratch y dirigidos a la comunidad formada en torno de este.

Otros "Scratchers" conformaron "compañías" en línea para crear en conjunto proyectos, que individualmente ninguno de sus miembros hubiera podido hacer. Una de estas compañías se inició cuando una joven Inglesa de 15 años, con el seudónimo de BeeBop, creó un proyecto lleno de objetos programables (sprites) y animó a otros a usarlos en sus proyectos o a hacerle pedidos especiales de objetos animados específicos. Estaba iniciando un negocio gratuito de consultoría. A una niña de 10 años, también inglesa y con el seudónimo de "MusicalMoon", le gustaron las animaciones de BeeBop y le solicitó crear un fondo para uno de sus proyectos.

Esta colaboración dio nacimiento a "Mesh Inc." compañía auto catalogada como "compañía miniatura" para producir juegos de "alta calidad" en Scratch. Unos días después, un muchacho de 14 años de New Jersey, USA, con el seudónimo de "Hobbit" descubrió la galería de esa compañía y ofreció sus servicios diciendo "Soy bastante buen programador, puedo ayudarles a depurar los programas y a otras cosas" . Más adelante un niño Irlandés de 11 años, con el seudónimo de "Marty", se incorporó al staff de Mesh Inc. aportando su experticia en dar movimiento a los escenarios.

Este tipo de colaboraciones ofrecen oportunidades para muchos y diferentes tipos de aprendizaje. A continuación una muchacha de 13 años de California (USA), que comenzó una compañía de Scratch llamada "Blue Elk Productions" describe su experiencia:

> *“Lo que es divertido tanto de Scratch como de organizar una compañía para hacer con otros guiones para juegos, es que he conocido mucha gente y he aprendido una cantidad de cosas nuevas. He aprendido mucho sobre diferentes tipos de programación mirando juegos de otros que tienen efectos interesantes, bajándolos y modificándoles los códigos y los objetos programables. Realmente me gusta la programación! Además, cuando comencé a usar Scratch yo no pensaba que era una buena artista. Pero desde eso, con solo mirar los proyectos de arte de otros, hacerles preguntas y practicar dibujo con programas como Photoshop y el editor de pinturas de Scratch he mejorado mucho como artista.... y otra cosa que he aprendido al organizar Blue Elk es cómo mantener a un grupo de personas motivadas y trabajando en equipo... A mi me gusta más Scratch que los blogs o los sitios de redes sociales como Facebook porque con el creamos juegos y proyectos interesantes con los que es divertido jugar, mirar y descargar. A mi no me gusta simplemente hablar en línea con otras personas, a mi me gusta hablar sobre algo nuevo y creativo”. El sitio Web de Scratch se ha convertido en una comunidad en línea vibrante, con personas que comparten, dialogan y re-mezclan proyectos de otros*

Para estimular el compartir y colaborar internacionalmente, le damos prioridad alta a traducir Scratch a múltiples idiomas. Creamos una infraestructura que permite que los bloques de programación de Scratch se traduzcan a cualquier idioma sin importar el tipo de caracteres que este tenga. Una red de voluntarios a nivel global ha realizado la traducción a más de 40 idiomas.

Los niños, alrededor del mundo, comparten ahora unos con otros programas de Scratch y cada uno puede ver los bloques de programación en su propio idioma.

DIRECCIONAMIENTO FUTURO

Un número creciente de Instituciones Educativas de Básica y Media (K – 12) de todas partes del mundo y aún algunas universidades, incluyendo a Harvard y a la Universidad de California en Berkeley (USA) [22], utilizan Scratch como introducción a la programación más formal. La pregunta que surge entonces es ¿qué se sigue? En los foros de discusión de Scratch, se dan debates permanentes respecto al lenguaje de

programación que debe usarse después de Scratch. Recibimos solicitudes permanentes para adicionarle funcionalidades más avanzadas tales como “Herencia” y “Listas de estructuras recursivas”, con el ánimo de que Scratch mismo sea el “paso siguiente”.

Sin embargo, nosotros pensamos mantener nuestro enfoque inicial de un bajo umbral de inicio (fácil de comenzar) y posibilidades amplias para realizar gran diversidad de proyectos, sin aumentar la complejidad. Para algunos “Scratchers”, en especial aquellos que piensan hacer carrera en ciencias de la computación, es importante moverse a otros lenguajes. Pero para muchos otros “Scratchers”, que ven a la programación como medio de expresión y no como un camino o carrera, Scratch es todo lo que necesitan. Con el pueden seguir experimentando nuevas formas de auto-expresión, produciendo un amplio rango de proyectos a medida que profundizan su comprensión de un núcleo fundamental de ideas computacionales. Con un poquito de programación se puede llegar muy lejos.

A medida que desarrollamos versiones futuras, nuestra meta es hacer Scratch todavía más flexible [13], significativo y social. Con nuestro Tablero de Sensores de Scratch (Scratch Sensor Board), las personas pueden crear proyectos de Scratch que sienten y reaccionan a eventos que ocurren en el mundo físico. También estamos desarrollando una versión de Scratch que corre en dispositivos móviles y otra basada en la Web que permite a las personas acceder datos y programar actividades en línea.

Probablemente, los mayores retos que enfrente Scratch no son tecnológicos sino culturales y educativos [23]. Scratch ha sido un éxito entre los primeros que lo adoptaron pero tenemos que ofrecer un mejor apoyo educativo para que llegue a más personas [24]. Recientemente lanzamos una nueva comunidad en línea llamada Scratch-Ed en la que los educadores comparten sus ideas experiencias y proyectos de clase en los que usan Scrach. Mirando más ampliamente, debe haber un cambio en lo que la gente piensa respecto a la programación y a los computadores en general. Es necesario expandir la noción de “Fluidez o competencia digital” para que incluya diseñar y crear, no solamente navegar e interactuar. Solo en ese momento tendrán iniciativas como Scratch la posibilidad de desplegar todo su potencial.

**

NOTAS Y REFERENCIAS:**

[1] Nota Eduteka: “Computer Science” o Ciencias de la Computación equivale en nuestros países a una Ingeniería de Sistemas.

[2] Prensky, M. Digital natives, digital immigrants. On the Horizon 9, 5 (Oct. 2001), 1–6.

[3] Resnick, M. Sowing the seeds for a more creative society. Learning and Leading with Technology (Dec. 2007), 18–22.

[4] Wing, J. Computational thinking. Commun. ACM 49, 3 (Mar. 2006), 33–35.

[5] diSessa, A. Changing Minds: Computers, Learning, and Literacy. MIT Press, Cambridge, MA, 2000.

[6] Nota Eduteka: Recomendamos consultar el artículo del Dr. Gary Stager, pionero en aprendizaje, quien describe las oportunidades que para la educación ofrece el uso creativo de los computadores. Él cuestiona el haber excluido la programación de computadores de la formación de los estudiantes. Defiende con argumentos de peso su reinclusión, aduciendo razones como el desarrollo de capacidades intelectuales de orden superior. https://eduteka.icesi.edu.co/modulos/9/272/224/1

[7] Papert, S. Mindstorms: Children, Computers, and Powerful Ideas. Basic Books, New York, 1980.

[8] Guzdial, M. Programming environments for novices. In Computer Science Education Research, S. Fincher and M. Petre, Eds. Taylor & Francis, Abingdon, U.K., 2004, 127–154.

[9] Kelleher, C. and Pausch, R. Using storytelling to motivate programming. Commun. ACM 50, 7 (July 2007), 58–64.

[10] Kelleher, C. and Pausch, R. Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programmers. ACM Computing Surveys 37, 2 (June 2005), 83–137.

[11] Kay, A. Squeak etoys, children, and learning.

[12] Nota Eduteka: Según la CSTA, el Pensamiento Computacional (Computational Thinking) se define como “la integración del poder del pensamiento humano con las capacidades de los computadores”.

[13] Nota de Eduteka: la más cercana traducción al español del término tinkerable, en el sentido en el que se utiliza en este documento, es ser flexible o elástico, entendido como “que no se sujeta a normas estrictas, a dogmas o a trabas”, “susceptible de cambios o variaciones según las circunstancias o necesidades”, “acomodaticio, que puede ajustarse a muy distintas circunstancias”, “que admite muchas interpretaciones” (fuente: diccionario de la RAE para los terminus flexible y elástico).

[14] Resnick, M. Behavior construction kits. Commun. ACM 36, 7 (July 1993), 64–71.

[15] Kafai, Y., Peppler, K., and Chiu, G. High-tech programmers in low-income communities: Seeding reform in a community technology center. In Communities and Technologies, C. Steinfield, B. Pentland, M. Ackerman, and N. Contractor, Eds. Springer, New York, 2007, 545–564.

[16] Peppler, K. and Kafai, Y. From SuperGoo to Scratch: Exploring creative media production in informal learning. Journal on Learning, Media, and Technology 32, 7 (2007), 149–166.

[17] Maloney, J., Peppler, K., Kafai, Y., Resnick, M., and Rusk, N. Programming by choice: Urban youth learning programming with Scratch. ACM SIGCSE Bulletin 40, 1 (Mar. 2008), 367–371.

[18] Monroy-Hernández, A. and Resnick, M. Empowering kids to create and share programmable media. Interactions 15, 2 (Mar.–Apr. 2008), 50–53.

[19] Bransford, J., Brown, A., and Cocking, R. How People Learn: Mind, Brain, Experience, and School. National Academies Press, Washington, D.C., 2000.

[20] Nota Eduteka: Recomendamos consultar las estátisticas de utilización del sitio Web de Scratch disponibles en http://stats.scratch.mit.edu/

[21] Minsky, M. Introduction to LogoWorks. In LogoWorks: Challenging Programs in Logo, C. Solomon, M. Minsky, and B. Harvey, Eds. McGraw-Hill, New York, 1986.

[22] Malan, D. and Leitner, H. Scratch for budding computer scientists. ACM SIGCSE Bulletin 39, 1 (Mar. 2007), 223–227.

[23] Margolis, J. Stuck in the Shallow End: Education, Race, and Computing. MIT Press, Cambridge, MA, 2008.

[24] En respuesta a la creciente comunidad de educadores que trabajan con Scratch en el aula, se desarrollo el sitio Web ScratchEd. Disponible al public desde Julio de 2009, ScratchEd es una comunidad en línea en la que los docents pueden compartir experiencias relacionadas con Scratch, intercambiar recursows, hacer preguntas y encontrar otros docents que están trabajando con Scratch (http://scratched.media.mit.edu/)

CRÉDITOS:

Artículo escrito por Mitchel Resnick, John Maloney, Andrés Monroy-Hernández, Natalie Rusk, Evelyn Eastmond, Karen Brennan, Amon Millner, Eric Rosenbaum, Jay Silver, Brian Silverman, Yasmin Kafai. Publicado en la revista Communications of the ACM - Vol. 52 No. 11, Páginas 60-67.

Traducción al español realizada por Eduteka con autorización de los autores.

Créditos adicionales a iStockPhoto.com y Scratch projects.

*Publicación de este documento en EDUTEKA: Febrero 01 de 2010.

Última modificación de este documento: Febrero 01 de 2010.*