Rafael Alba-k idatzia
Asteazkena, 2008(e)ko maiatza(r)en 28-(e)an 15:38etan

There are no translations available.
Descubre varias herramientas de programación que pueden ser utilizadas desde edades tempranas, con algunas sugerencias de uso para cada una de ellas.

APRENDER A PROGRAMAR... ¿DESDE PEQUEÑOS?

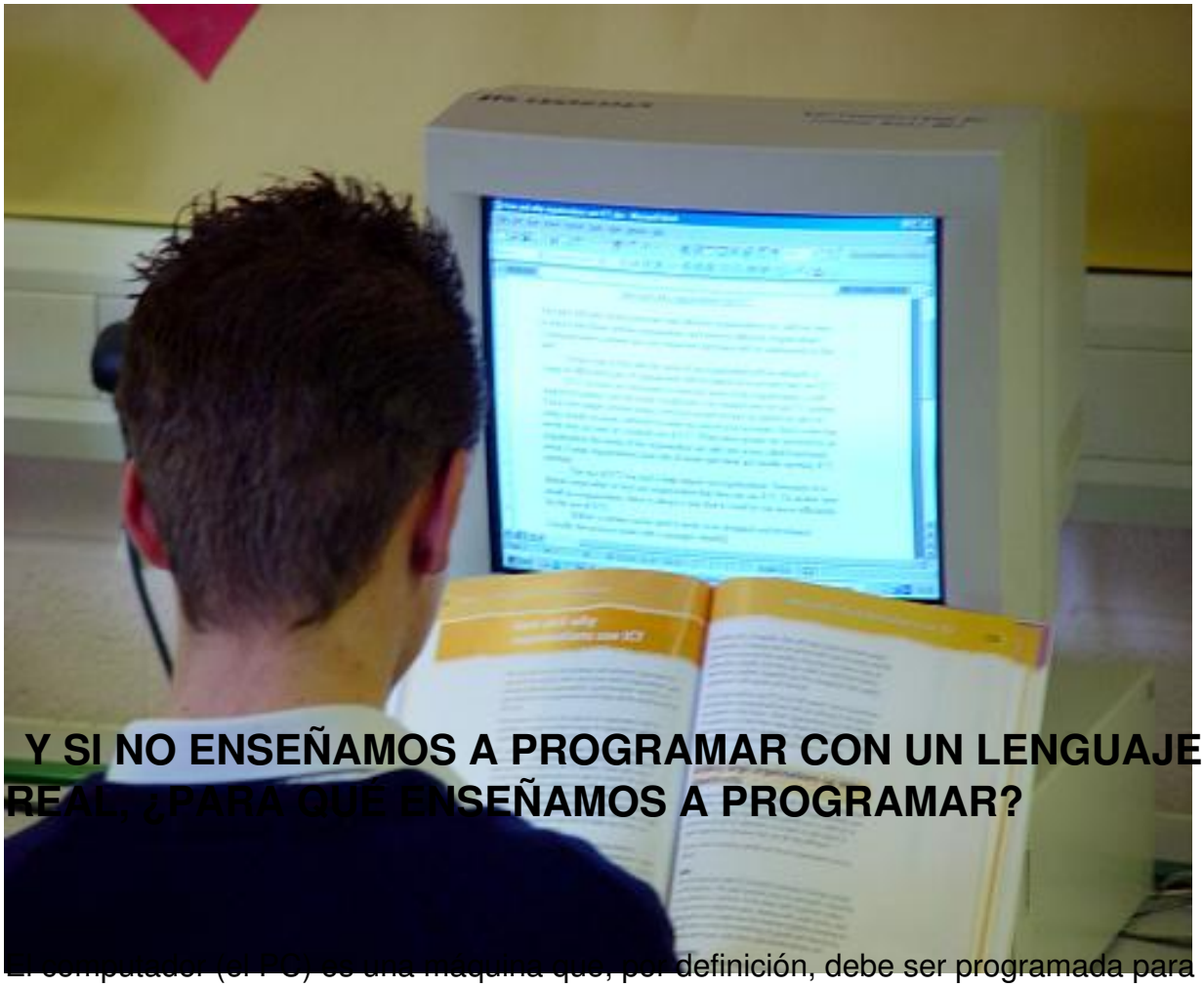
INTRODUCCIÓN

En los últimos años han comenzado a aparecer herramientas que ponen el mundo de la programación a disposición de un público infantil y juvenil.
Estas herramientas tienen como objetivo facilitar la iniciaci ón en la programación a través de un entorno amigable pero que, al mismo tiempo, ofrece diversas alternativas de aprendizaje de los elementos básicos de los lenguajes de alto nivel: variables, estructuras de control, sentencias, funciones, condiciones, operadores...
La mayor ía de ellos tienen en común que no pretenden necesariamente convertir a sus usuarios en programadores expertos, sino facilitar el desarrollo de diversas habilidades multidisciplinares que se ponen en marcha cuando se elaboran pequeños programas encaminados, por ejemplo, a la resolución de problemas, a la creación de juegos sencillos e incluso a la generación de escenarios complejos 3D.
De hecho, algunos expertos aseguran que ense ñar a programar simplemente con el fin de obtener mejores programadores en el futuro puede incluso ser contraproducente, por razones que veremos después.
Por ello, el planteamiento que se hace por parte de los creadores de estas herramientas es generar un entorno de programaci ón amigable, que incluya factores motivacionales, tanto internos como externos, y que pongan en marcha diferentes habilidades, no todas ellas basadas en la inteligencia matemática.

Rafael Alba-k idatzia
Asteazkena, 2008(e)ko maiatza(r)en 28-(e)an 15:38etan

¿Y POR QUÉ NO APRENDER A PROGRAMAR CON LENGUAJES AUTÉNTICOS EN LUGAR DE HACERLO CON HERRAMIENTAS ADAPTADAS?

Los que comenzamos a programar en los 80 nos encontramos un lenguaje de programación sencillo que nos permitía comenzar a escribir líneas de código casi inmediatamente: BASIC. A pesar de sus muchos detractores, con este lenguaje se podían hacer operaciones básicas y programas elementales sin tener unos conocimientos elevados de programación.
Con la creciente complejidad de los ambientes gráficos que hicieron su aparición en la década de los 90, el aprendizaje de la programación se complicó en el mismo grado, alejándose del público general. Los PCs se generalizaron, pero su utilización se centró en la manipulación de herramientas ofimáticas. No en la programación.
Actualmente, los ambientes de programación se basan en lenguajes orientados a objetos, y la complejidad que los rodea para comenzar a escribir un simple programa (el famoso "hola mundo"), los aleja de su utilización en la escuela (entendiendo la escuela en el sentido amplio de la escolarización obligatoria).
Enseñar a programar desde esta perspectiva está por tanto reservado a un alumnado con conocimientos avanzados en el ámbito de las matemáticas y otras áreas del currículo, y por supuesto con una capacidad de abstracción que no está al alcance, normalmente, de chavales de Educación Primaria, y no siempre de los primeros niveles de la Secundaria Obligatoria.
Las experiencias en este ámbito, según se manifiesta en diversos análisis, o bien están condenadas al fracaso o bien terminan siendo contraproducentes: la programación como concepto se aleja de los intereses y, a veces, de las capacidades del alumnado de Primaria y Secundaria Obligatoria.
De hecho, dichas experiencias se suelen centrar fundamentalmente en el aprendizaje de la sintaxis del lenguaje. Y cuando llegamos a producir un pequeño programa, éste se suele basar en texto; es decir, obtenemos un resultado "pobre" a los ojos de un alumnado cuyo pensamiento es más bien multimedia y que espera unos resultados con gráficos, movimiento e incluso sonido.
Conseguir mover un pequeño "sprite" (un objeto gráfico con capacidad de ser animado) por la pantalla del ordenador con un lenguaje de alto nivel como Java, C, C++ o C#, Visual Basic .NET, etc., resulta tan complejo que el esfuerzo invertido difícilmente merece la pena en un aula estándar.

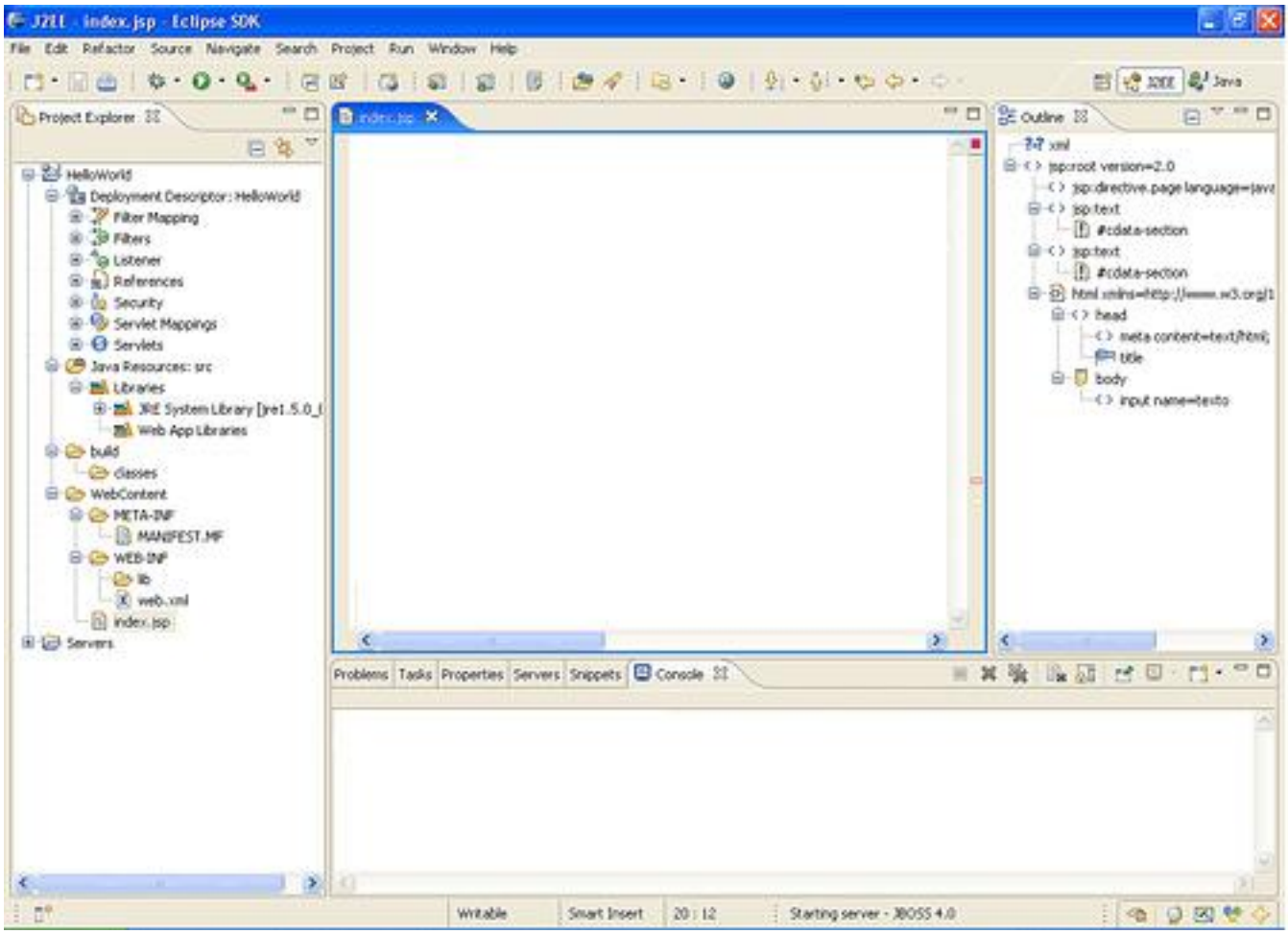


Y SI NO ENSEÑAMOS A PROGRAMAR CON UN LENGUAJE REAL, ¿PARA QUÉ ENSEÑAMOS A PROGRAMAR?

El computador (el PC) es una máquina que, por definición, debe ser programada para funcionar. Actualmente, los PCs suelen ser utilizados, como decíamos más arriba, para emplear programas ofimáticos (y algunos de diseño gráfico), navegar por internet, localizar y manipular información, interactuar con otros a través de la red y manejar herramientas de mensajería instantánea...
En definitiva, se está utilizando sólo una parte de lo que podríamos llamar la tecnología digital : la manipulación de la información, su tratamiento y uso.
Hay otra parte que también es relevante: la tecnología digital como medio de construcción. Programar un PC implica estructurar el pensamiento, las ideas, convertirlas en un proyecto de construcción para generar algo nuevo, no solamente para manipular lo que ya existe.
Desde esta perspectiva, utilizar los PCs (entendidos como herramientas elementales de acceso a la tecnología digital, pero no las únicas) para crear proyectos genera algunas ventajas sobre el uso habitual de los mismos:

- Permite desarrollar el pensamiento abstracto;
- Favorece el desarrollo del pensamiento algorítmico;
- Pone en marcha procesos creativos que pueden ser realizados a través de grupos de trabajo (aprendizaje cooperativo);
- Aglutina la utilización de diferentes "inteligencias" en proyectos compartidos: lingüística, matemática, artística, espacial, musical, interpersonal e intrapersonal.

¿Podríamos utilizar otras herramientas, además de la programación, para conseguir estos objetivos? Probablemente, sí: herramientas de creación de multimedias, de edición de vídeo... Pero todas ellas, normalmente, pueden ir acompañadas de "scripts", de programación, en definitiva.
Resumiendo lo anterior, aprender a programar, por tanto, tiene un uso directo en la adquisición de la competencia digital, y este uso ha sido tradicionalmente olvidado, debido normalmente a que se intentaba convertir al alumnado en programadores expertos, lo que estaba normalmente condenado al fracaso, dado que, a este punto, sólo debería llegar aquel sector del alumnado que realmente desee desarrollar su labor profesional en dicho ámbito o ampliar su formación específica en el mismo.



HEMOS DICHO QUE HAN PROLIFERADO VARIAS HERRAMIENTAS... ¿CUÁLES SON?

La lista que se propone a continuación no es exhaustiva, ni pretende recoger todas las herramientas disponibles. Lo que sí se ha intentado es que todas las que se mencionan tengan, al menos, una versión "gratuita" de la misma, y que estén lo suficientemente soportadas y extendidas como para saber que pueden perdurar lo suficiente como para invertir tiempo en ellas.
El hecho de que existan versiones gratuitas va a garantizar que, si se decide utilizarlas en el aula, no vamos a obligar a nadie a comprar un producto comercial. El alumnado podrá usar el producto en el Centro, pero también en su casa, sin necesidad de invertir recursos económicos específicos para ello.
Veamos, por tanto, algunas sugerencias que cumplen las características mencionadas.

Logo

Todos conocemos, sin duda, el primer lenguaje que se creó con una finalidad específicamente

pedagógica y escolar: LOGO. Su creador (entre otros), Seymour Papert, tiene publicados varios artículos y libros (cuya lectura recomendamos encarecidamente, por el indudable contenido pedagógico de los mismos) en los que defiende el aprovechamiento de las TICs en la educación, en los dos ámbitos que mencionamos anteriormente. Véase la bibliografía para más información.
LOGO es un lenguaje que sigue vivo y vigente, y ha sido migrado a los nuevos entornos y sistemas operativos existentes en la actualidad. Algunos de sus intérpretes han sido traducidos al castellano (y se puede por tanto escribir programas en esta lengua), y existen versiones escritas como software libre .
Sin embargo, sus características lo han convertido en una herramienta utilizada fundamentalmente para actuar como interfaz de comunicación con diversas controladoras electrónicas. Por ello, su uso ha quedado normalmente relegado a los niveles de Secundaria Obligatoria donde se imparte Tecnología.
El CNICE tiene una sección del Observatorio dedicada específicamente a este Lenguaje, desde donde se puede descargar el intérprete MSWLOGO 6.5a, manuales, etc.

Scratch

SCRATCH es un lenguaje de programación orientado específicamente a niños y adolescentes, basado en SQUEAK, con un concepto muy didáctico basado en la utilización de "bloques" que se unen para formar pequeños fragmentos de código ("scripts"), y que permiten crear historias interactivas, animaciones, juegos, piezas musicales y artísticas...
Dos elementos motivacionales fundamentales de SCRATCH son:

- en primer lugar, permite añadir cualquier creación artístico-digital del "programador" en el entorno de una manera muy sencilla (así que podemos incorporar nuestros propios dibujos, nuestros propios elementos multimedia, etc.);
- en segundo lugar, podemos compartir nuestras creaciones con otros "programadores" en la misma página web del proyecto SCRATCH, aprendiendo de las creaciones de otros y

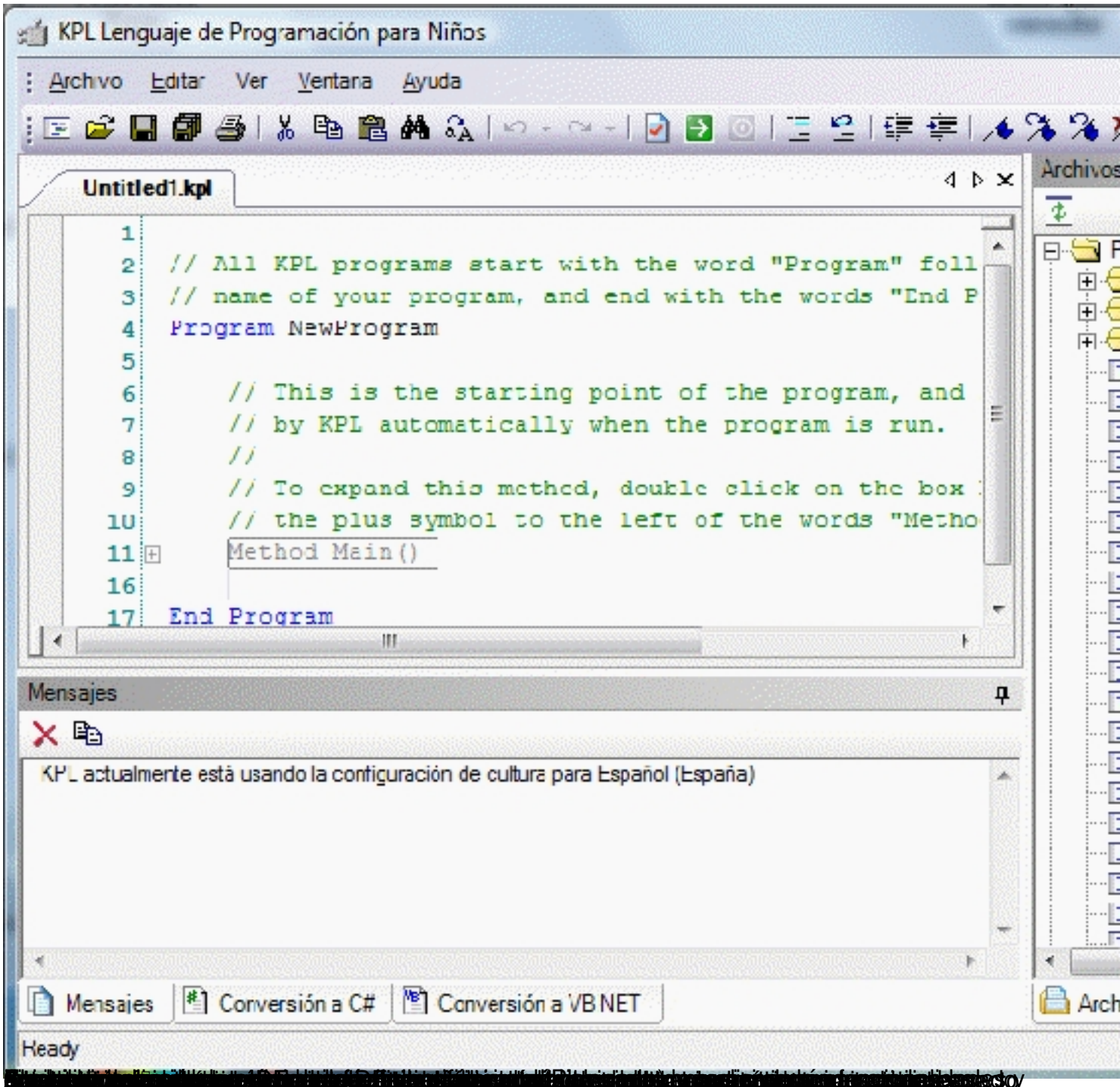
poniendo a disposición de los demás las nuestras.

Evita los típicos problemas con los errores de sintaxis al no permitir que piezas que no pueden ir unidas se junten. De esta manera, antes de "ejecutar" nuestro programa, sabemos si va a funcionar o no, al menos desde el punto de vista de la corrección sintáctica.
Introduce de una manera básica pero elegante muchos de los conceptos basados en objetos.
Ha sido desarrollado por el "Lifelong Kindergarten Group" en el MIT Media Lab. Existe una argumentación teórica muy importante sobre la que se sustenta, y es una herramienta que, pudiendo ser utilizada por niñas y niños de corta edad, demuestra también su validez con estudiantes que se están iniciando en (o que van a estudiar) informática, en sus distintas posibilidades universitarias.

KPL

KPL (Acrónimo de "Kids Programming Language", o "Lenguaje de Programación para Niños) ha sido desarrollado bajo el auspicio de personal que ha trabajado o trabaja en ambientes "Microsoft" y, de alguna manera, es una adaptación de Visual Basic .NET.

Rafael Alba-k idatzia
Asteazkena, 2008(e)ko maiatza(r)en 28-(e)an 15:38etan



Alice

ALICE es una potentísima herramienta que, como sus creadores promocionan, representa una propuesta "límpia y fresca" al aprendizaje de las "Ciencias de la Computación".

¿CUÁNDO PODEMOS COMENZAR A USAR CADA HERRAMIENTA?

A continuación incluimos un pequeño cuadro que representa una posible sugerencia de uso por edades de las herramientas mencionadas.
No debe tomarse como una orientación excluyente dado que no existen estudios científicos que lo avalen.
Sin embargo, está elaborado partiendo de las características de cada uno de los lenguajes y las habilidades estándar de las edades tomadas como referencia.

BIEN, EXISTEN ALTERNATIVAS VIABLES, PERO, ¿NO REPRESENTAN UNA CARGA LECTIVA EXTRA CON RESPECTO A LAS MATERIAS QUE SÍ SON OFICIALES Y ESTÁN EN EL CURRÍCULO?

La respuesta a esta pregunta es: depende. Cuando empleamos una herramienta que podemos utilizar desde diversos ámbitos, y esos mismos ámbitos van a recibir las ventajas de haberla utilizado, no es una carga lectiva extra, sino que va a ahorrarnos tiempo y esfuerzos. Obviamente, sea cual sea el nivel educativo en el que nos encontremos, y sea cual sea la

herramienta que podamos elegir, es obvio que va a requerir algún tiempo para familiarizarnos con ella y con sus posibilidades.
Pero si observamos globalmente las herramientas que hemos presentado, todas ellas nos permiten trabajarlas desde diversas áreas, y esa versatilidad es precisamente su ventaja. Una vez que nuestro alumnado conozca la herramienta, la va a poder utilizar desde las áreas que deseemos.
Este principio multidisciplinar es obviamente más fácil de aplicar en Educación Primaria, dado que probablemente la misma o el mismo docente imparte más de un área. Sin embargo, es perfectamente trasladable a primer ciclo de ESO y, con algunas adaptaciones, al segundo ciclo.
Sin entrar en detalles específicos de organización de cada Centro, desde mi experiencia personal puedo afirmar que la utilización de una misma herramienta en diferentes áreas nos permite aunar esfuerzos, ganar tiempo y generar sinergias que posteriormente van a incidir en todas las áreas que han participado en el trabajo colaborativo.
Así, mientras que desde las áreas de ciencias podemos incidir en la utilización de la programación como herramienta de resolución de problemas y facilitadora del pensamiento algorítmico, desde las áreas de letras podemos desarrollar su potencialidad como editores o presentadoras de historias y relatos, y desde las áreas artísticas como poderosas herramientas multimedia en las que dar sentido a nuestros diseños tanto gráficos como musicales. Y así, al mismo tiempo, difuminar de alguna manera la separación entre áreas que solemos remarcar con frecuencia.
El trabajo cooperativo tiene también sentido en un planteamiento como éste, dado que el alumnado con diferentes habilidades puede liderar el trabajo que se plantee desde cada área en cada momento, encontrando que cada una o cada uno tiene su lugar o momento de participación más específica, sin que por ello deje de participar en todos los ámbitos y sugerencias de trabajo que se hagan desde cada una de las áreas.
Espero poder desarrollar este aspecto en un artículo específico al respecto.
¿Nos quedan dudas sobre cómo descargar y comenzar a probar las herramientas sugeridas? En posteriores artículos vamos a intentar analizar más detalladamente algunas de sus funcionalidades.

BIBLIOGRAFÍA y REFERENCIAS

Logo:
htt p://es.wikipedia.org/wiki/Logo_%28Lenguaje_de_programaci%C3%B3n%29
http://recursostic.educacion.es/observatorio/web/images/upload/ob_innovacion/mswlogo65a_sp.zip
http://es.wikipedia.org/wiki/Seymour_Papert http://neoparaiso.com/logo/seymour-papert.html

Scratch:
http://scratch.mit.edu/
http://www.linux-magazine.es/issue/28/078-082_ScratchLM28.crop.pdf
http://sushiknights.org/2007/01/ensenar_programacion_a_los_ninos_con_scratch.html

Squeak:
http://www.squeak.org/ http://squeak.educarex.es/Squeakpolis

KPL:
http://www.kidsprogramminglanguage.com/espanol/parents.php
http://theschwartz.files.wordpress.com/2007/03/introductorycswithgameswithdemo.pdf
http://blogs.msdn.com/coding4fun/archive/2006/10/31/912456.aspx

Alice:
http://www.alice.org/ http://www.alice.org/index.php?page=what_is_alice/what_is_alice

General:
http://es.wikipedia.org/wiki/Howard_Gardner http://es.wikipedia.org/wiki/Sprite_(videojuegos)
http://es.wikipedia.org/wiki/Programaci%C3%B3n_orientada_a_objetos
http://es.wikipedia.org/wiki/BASIC