

Reiterando

por

PEDRO LATORRE GARCÍA

(CPEPA Gómez Lafuente, Zaragoza)

Este artículo se divide en dos partes. En la primera, haré una pequeña reflexión sobre el uso de recursos digitales en el aula y comentaré mi experiencia utilizando IAs para programar pequeñas aplicaciones interactivas; en la segunda, presentaré mi último programa.

Creación de recursos digitales con ayuda de IAs

He llegado a la sorprendente conclusión de que la comunidad educativa y especialmente el Departamento de Educación no valora y mucho menos incentiva la competencia digital de los docentes. Lo mismo ocurre con la creación de recursos educativos. Se trata de tareas cuya ejecución se delega en un tercero. En muchos casos, la editorial de turno facilitará el correspondiente libro de texto incluyendo los pertinentes recursos digitales. Mis aspiraciones de adquirir notoriedad entre mis colegas como creador de manipulables parecen inalcanzables.

Buscando información en los medios parece confirmarse mi valoración.

Actualmente resulta muy complejo que el profesorado se anime a enriquecer sus clases usando sus propios recursos digitales. El aumento de la carga administrativa y la pérdida de poder adquisitivo generan desgaste profesional y reducen el tiempo y la energía disponibles para innovar o crear materiales propios. En este contexto, la elaboración de recursos digitales —que requiere tiempo, formación y experimentación— queda relegada frente a las tareas del día a día.

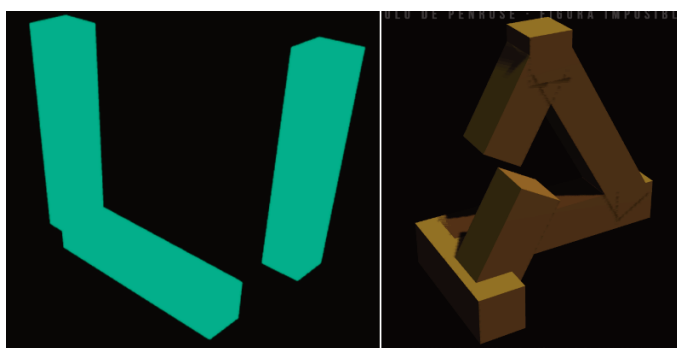
La administración educativa, con su infinita sabiduría, impulsa la digitalización, pero no se acompaña de incentivos —ya sean económicos, de reducción de carga lectiva o de reconocimiento profesional— para que el profesorado diseñe recursos. Como consecuencia, algunos docentes optan por utilizar materiales ya existentes en lugar de crearlos. Otros muchos no usan manipulables en el aula. Para corroborar lo expuesto, el último curso anunciado por el CP sobre recursos para pantallas digitales está organizado por una editorial.

Resulta muy significativo que los últimos cursos subidos a la plataforma de Educación Secundaria a distancia ESPAD de Aragón carezcan de recursos interactivos. Hace unos años, se redujo la jornada a un equipo de profesores para elaborar los materiales de la ESPAD, incluyendo escenas realizadas con DescartesJS. Nuestra compañera María José García fue una de los líderes del proyecto. En el [enlace](#) se encuentra uno de sus últimos trabajos.

Si a pesar de todo lo expuesto anteriormente te animas a desarrollar recursos digitales, las actuales IAs harán que tu trabajo sea mucho más sencillo. Desde luego no todo es perfecto. Veamos un primer ejemplo. Actualmente uso las versiones gratuitas de ChatGPT y Claude. Les he pedido que dibujen un [tribar](#) usando la biblioteca Three.js:

Con Three.js crea una página html que dibuje un tribar.

ChatGPT es incapaz de realizarlo después de varios intentos. Claude casi lo consigue, véase [enlace](#).



Se ha producido un cambio de paradigma en la programación de aplicaciones. Nuestras amigas inhumanas nos facilitan el código en el lenguaje que elijamos y con instrucciones claras de cómo ejecutarlo. Ya no hace falta dedicar largas horas de estudio para aprender con profundidad un lenguaje de programación. La prioridad, en cambio, pasa a ser la capacidad de formular instrucciones precisas —los llamados *prompts*— que permitan a la IA generar exactamente lo que necesitamos.

Como ejemplo, para crear una aplicación que simule el juego de Conecta 4, se puede partir de una instrucción como la siguiente:

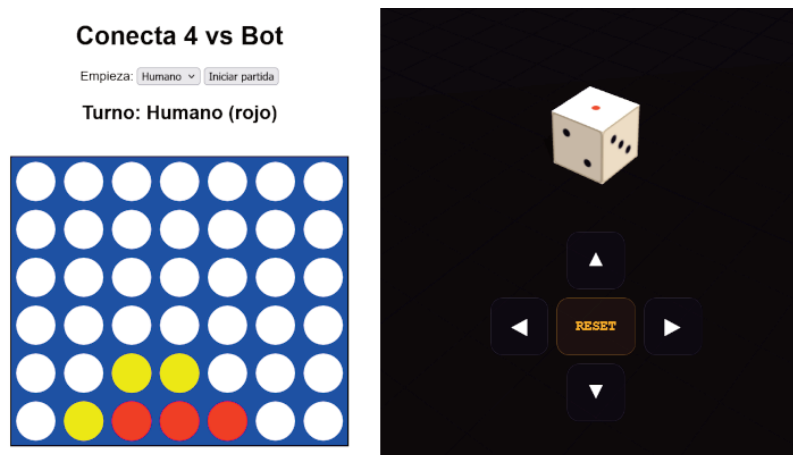
Crea un bot que juegue al Conecta 4 contra un humano con una estrategia ni demasiado simple ni demasiado compleja. El objetivo es que la estrategia pueda descubrirse tras un tiempo de reflexión. Impleméntalo en HTML, canvas y JavaScript.

He copiado el código generado por la ChatGPT en un archivo de texto y lo he guardado con la extensión html, teniendo la precaución de usar la codificación UTF-8 para que las tildes se muestren correctamente. Al abrirlo con el navegador Firefox he verificado el correcto funcionamiento de la aplicación.

A partir de ahí, podemos ir mejorando progresivamente el resultado obtenido. En nuestro caso:

Añade un selector para elegir quién empieza el juego y un botón de inicio. Que aparezca un mensaje indicando de quién es el turno.

La [aplicación obtenida](#) ya está lista para usarse en el aula. A partir de ella, comprobaremos si a los alumnos les motiva usarla y si descubren la estrategia de la máquina pronto, tarde o nunca. Según su respuesta, podemos seguir actualizándola.



No siempre las cosas funcionan bien a la primera. Os invito a probar con vuestra IA favorita el *prompt*:

Genera una aplicación simulando el movimiento de un dado, con las caras numeradas del 1 al 6, en un plano rotando con respecto a la arista correspondiente. Con unos botones se elegirá la dirección del movimiento. Impleméntalo con Three.js.

El código generado por chatGPT está tan alejado de mi petición que ya no me molestó en intentar mejorarlo: las caras del dado no están numeradas y no sé con respecto a qué eje o punto gira. La respuesta inicial de Claude está cercana a mi solicitud, pero el cubo solo gira correctamente en una dirección y, en la perpendicular, se coloca debajo del plano. Al intentar mejorarlo, después de dos instrucciones del tipo:

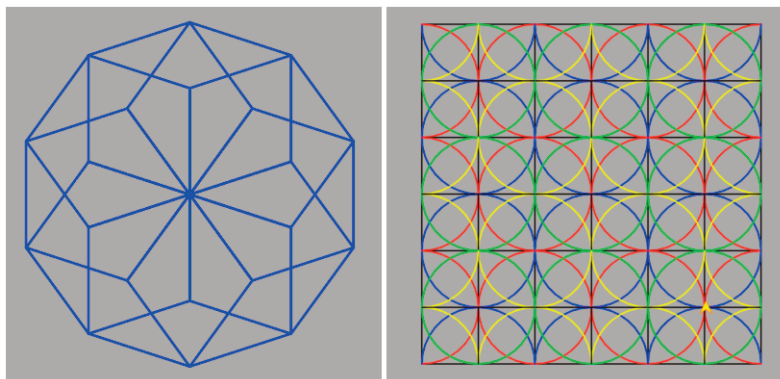
No está bien, el dado tiene que estar siempre encima del plano y ahora las flechas superior e inferior lo colocan debajo.

se termina mi plan gratuito sin haber conseguido una aplicación inicial operativa. Después de unas cuatro horas de espera y una última sencilla instrucción, alcanzamos el *éxito*.

Este es el gran problema de Claude, genera un código mucho mejor para mis peticiones que ChatGPT, pero habitualmente tarda mucho en responder y puedo hacerle pocas preguntas consecutivas sin que haya una larga interrupción.

El rincón de Petrus

Reiterando ha sido la última novedad del [rincón de Petrus](#). Es una adaptación de un intérprete del lenguaje de programación Logo escrito en el lenguaje Javascript realizado por [LWH-21](#). La aplicación ha sido probada con los navegadores Firefox y Chrome en un PC y para su empleo se recomienda una pantalla con una resolución mínima de 1024×768 . Logo es un lenguaje de programación, desarrollado a finales de los años 70 por un equipo de matemáticos en la Universidad de California, con el fin de entrenar el pensamiento lógico. Originalmente, el lenguaje Logo estaba destinado a controlar el movimiento de un vehículo robotizado, con aspecto de tortuga, controlado desde un ordenador.



Hemos probado la aplicación en el [Taller de Talento Matemático](#) y a la mayoría de los alumnos les ha gustado. Pensamos que puede usarse con alumnos desde el último ciclo de Primaria. Desde el punto de vista educativo, el uso de Logo constituye una herramienta para el desarrollo del pensamiento computacional. A través de la programación de movimientos de la «tortuga», el alumnado trabaja de manera natural conceptos como la secuenciación de instrucciones, la identificación de patrones y la depuración de errores. Además, su carácter visual y geométrico facilita la conexión con contenidos matemáticos —como ángulos, polígonos o simetrías—, favoreciendo un aprendizaje activo en el que el estudiante experimenta, formula hipótesis y ajusta sus estrategias.

Director: Ricardo Alonso Liarte (IES Salvador Victoria, Monreal del Campo)

Consejo de Redacción: Alberto Elduque Palomo (Departamento de matemáticas de la Universidad de Zaragoza), Julio Sancho Rocher (IES Avempace, Zaragoza), Daniel Sierra Ruiz (CPI El Espartidero, Zaragoza).

Entorno Abierto es una publicación digital trimestral que se edita en Zaragoza por la Sociedad Aragonesa «Pedro Sánchez Ciruelo» de Profesores de Matemáticas. *Entorno Abierto* no se identifica necesariamente con las opiniones vertidas en las colaboraciones firmadas.

Envío de colaboraciones a <sapmciuelos@gmail.com>

Blog: <<http://sapmatematicas.blogspot.com.es/>>

Twitter: @SAPMciuelos



Abril de 2026
ISSN: 2386-8821e

