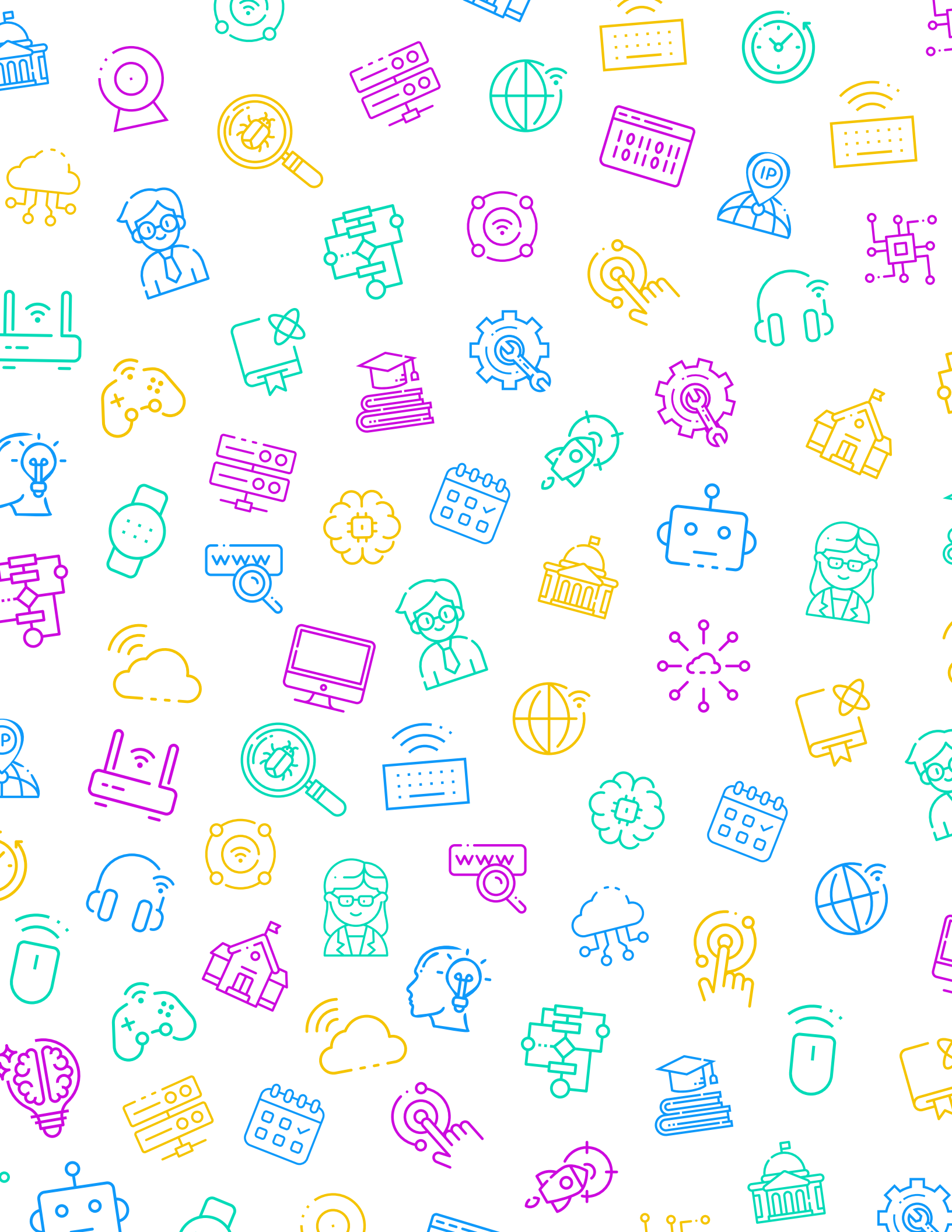
APRENDO A PROGRAMAR

Primero Medio

Alineado curricularmente con Tecnología
Horas de Libre Disposición

Ideo Digital


WWW
WWW

Aprendo a Programar

Editores

María Cristrina Cid Cartes

Loraine Schlotterbeck Byrne

Curadores

Sebastián Godoy Reyes

Marisel Mateluna Jorquera

Diseñadora Gráfica

Tamara Oyanedel

IdeoDigital

Proyecto país originado gracias a una alianza estratégica entre Fundación Kodea y BHP Foundation, que busca crear las condiciones necesarias para implementar las Ciencias de la Computación en el sistema escolar público chileno, para que miles de niños, niñas y adolescentes, se conviertan en protagonistas de la sociedad digital del siglo XXI.

Líder de Proyecto

Claudia Jaña
Gerente de Educación- Fundación Kodea

IdeoDigital

Ciencias de la Computación en el aula

Desarrolla

fundaciónkodea

Apoya y respalda

BHP Foundation

APRENDO A PROGRAMAR

Primero Medio

Alineado curricularmente con Tecnología
Horas de Libre Disposición

Ideo Digital

Índice

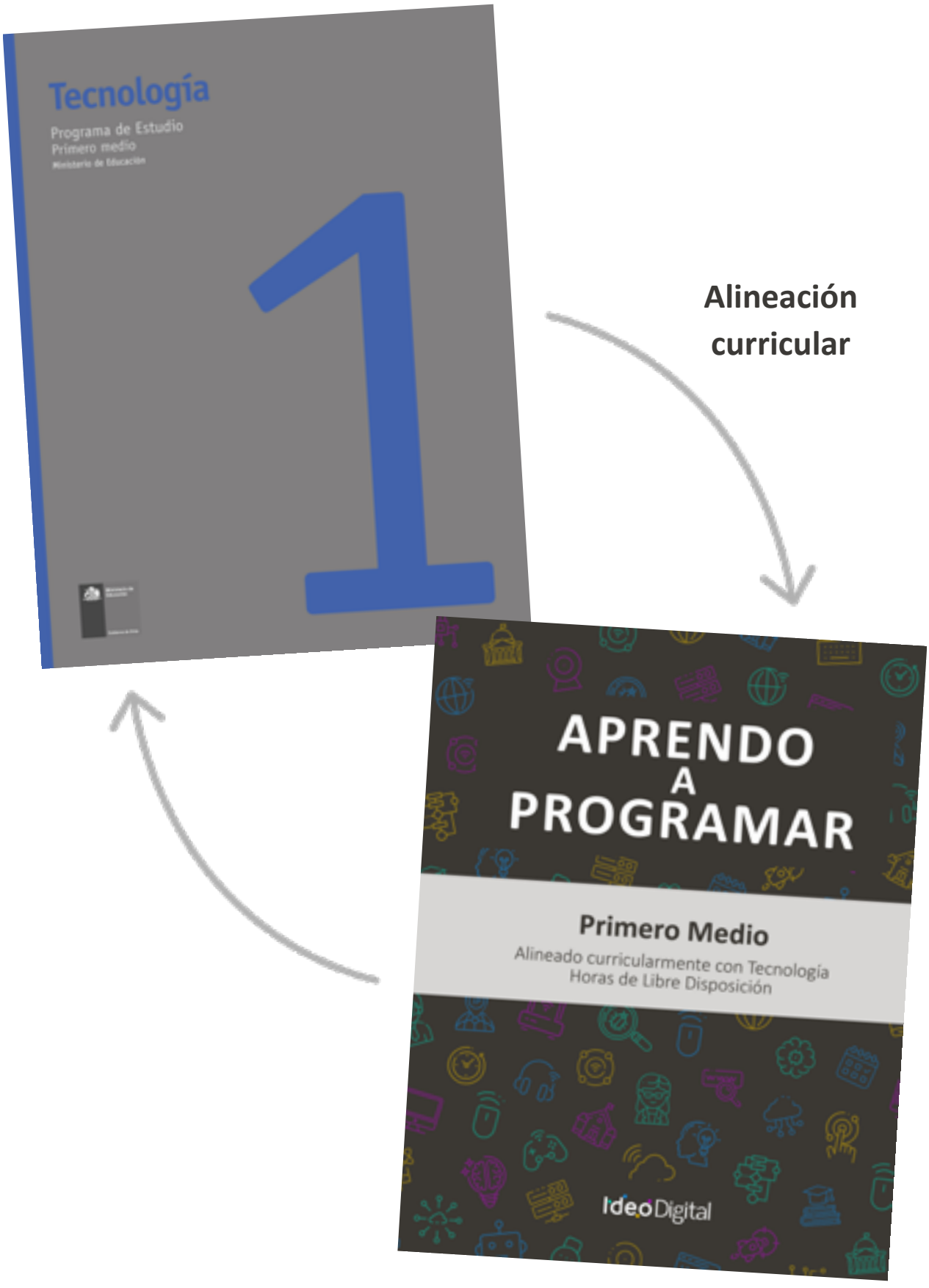
Introducción

Presentación del programa

Aprendo a programar: programación integrada con Pensamiento computacional y programación pone el foco en relevar los Objetivos de Aprendizaje de las Bases Curriculares desde la lógica de la programación para Primero Medio.

Para efectos pedagógicos significativos y coherentes con los intereses de los estudiantes, la integración de aprendizajes está enfocada en lecciones con y sin conexión, con la intención de incrementar instancias de aprendizaje que necesitan ser resueltas por medio de la programación, la interacción con equipos computacionales y el uso de materiales y recursos educativos que permiten al estudiantes avanzar desde ideas concretas o otras más abstractas. Asimismo, se busca una integración coherente con los conocimientos y habilidades propias de Tecnología para este nivel, como son la búsqueda análisis de información, la adaptabilidad y flexibilidad, la creación, el manejo de materiales y herramientas, el trabajo en equipo y la comunicación.

En cada una de las lecciones se favorece la adquisición de un lenguaje de programación que facilita la resolución de problemas en diferentes ambientes de enseñanza aprendizaje.


Tecnología
Programa de Estudio
Primero medio
Ministerio de Educación
1
Alineación
curricular
APRENDO
A
PROGRAMAR
Primero Medio
Alineado curricularmente con Tecnología
Horas de Libre Disposición
IdeoDigital

Propósito de Aprendo a Programar

El propósito de las lecciones es relevar estrategias didácticas asociadas al aprendizaje de la programación y el desarrollo del pensamiento técnico y tecnológico, que permiten abordar de manera simultánea los objetivos de aprendizaje prescritos en las Bases Curriculares vigentes para la asignatura de Tecnología de I medio.

Esta propuesta de aprendizaje busca dar continuidad al Programa Fundamentos de la Computación de 1° a 8° básico disponible en Curriculum Nacional, fortaleciendo conceptos y habilidades del pensamiento computacional, la resolución de problemas tecnológicos, el diseño sistemas y la comprensión del mundo a través de la tecnología, el ambiente y la sociedad.

Para el uso e implementación de las lecciones se sugieren propósitos, secuencias de aprendizaje, momentos de preparación, vocabulario y los objetivos de cada una, las cuales potencian e integran activamente los aprendizajes de los estudiantes a través de la resolución de problemas, el desarrollo de habilidades y la aplicación creativa de la programación, teniendo en cuenta el tiempo disponible y las particularidades de cada contexto escolar.

Las actividades consolidadas en cada lección se suman a una serie de experiencias de aprendizaje, con el objetivo de profundizar y afianzar el conocimiento de los contenidos vistos, así como también fortalecer las habilidades abordadas en cada unidad. Además, propone desafíos a los estudiantes, que los docentes podrán utilizar como ticket de salida y parte de la evaluación de proceso, según la pertinencia y atingencia de los avances de sus estudiantes, ya que la idea es facilitar el proceso de enseñanza aprendizaje de los y las estudiantes.[1]

Cada lección será valorada, como una evaluación de proceso, con el fin de enriquecer los conocimientos adquiridos, a través de desafíos que serán considerados como una estrategia de evaluación de salida, retroalimentando de manera constante, así como también desarrollando la metacognición y metaevaluación de los estudiantes.

Aprendo a programar entrega una serie de sugerencias al docente, recomendaciones de recursos didácticos complementarios, como videos, tutoriales y bibliografía dispuesta tanto para profesores como para los y las estudiantes.[2]

Para poder abordar las lecciones, en cada una de ellas se sugiere la mencionada secuencia de aprendizaje, la cual está asociada a tiempos para cada instancia, siendo flexible de acuerdo con la asignación de horas de libre disposición que considere cada establecimiento:

Conocimiento inicial → **Ampliación del conocimiento** → **Transferencia del conocimiento** → **Evaluación**

[1] y [2] https://www.curriculumnacional.cl/

Tabla de relaciones entre Objetivos de Aprendizaje de la asignatura de y las lecciones recomendadas

Unidad 1: Iniciación al Pensamiento Computacional y el Impacto de las tecnologías en nuestra sociedad

Horas pedagógicas sugeridas: 21

Objetivos de Aprendizaje Tecnología 1ro Medio		**Lecciones que cubren OA**
OA 01	Identificar oportunidades o necesidades personales, grupales o locales que impliquen la creación de un servicio, utilizando recursos digitales u otros medios.	1 a la 3
OA 02	Desarrollar un servicio que implique la utilización de recursos digitales u otros medios, considerando aspectos éticos, sus potenciales impactos y normas de cuidado y seguridad.	1 a la 3
OA 05	Analizar las formas en que los productos tecnológicos y los entornos evolucionan, caracterizando los diversos factores que influyen en ese cambio.	1 a la 3
OA 06	Inferir, basándose en la evolución de los productos tecnológicos y los entornos, los efectos positivos o negativos que estos han tenido en la sociedad.	1 a la 3
OAA A (actitudes)	Respetar al otro y al medio ambiente, lo que se expresa en los requerimientos del trabajo colaborativo exigido en la producción de soluciones tecnológicas, en la reflexión y el debate sobre el análisis de productos tecnológicos, en la conservación de los recursos y del bien común, entre otros.	1 a la 3

Unidad 1: Iniciación al Pensamiento Computacional y el Impacto de las tecnologías en nuestra sociedad

Horas pedagógicas sugeridas: 21

Objetivos de Aprendizaje Tecnología 1ro Medio		**Lecciones que cubren OA**
OAA B (actitudes)	Valorar las potencialidades propias y del otro, en relación con el desarrollo de capacidades técnicas y tecnológicas, de los desarrollos tecnológicos en virtud de su aporte al mejoramiento de la calidad de vida, con todo lo que su producción requiere.	1 a la 21
OAA C (actitudes)	Trabajar colaborativamente, lo que se refleja en el compromiso por la prosecución de los objetivos del equipo; en asumir responsabilidades en el grupo y establecer maneras de trabajo eficiente; en aceptar consejos y críticas, escuchando y respetando al otro para llegar a acuerdos; en tomar conciencia y superar las dificultades personales y del trabajo; en aprender de los errores; en solicitar y prestar ayuda a sus pares para el cumplimiento de las metas del trabajo.	1 a la 21
OAA D (actitudes)	Demostrar disposición hacia la prevención de riesgos y el autocuidado, entendidos como la capacidad progresiva de valorar la vida, del cuerpo, el bienestar y la salud; así como el desarrollo de prácticas y hábitos para mejorar la propia seguridad y la de los demás y, con ello, prevenir riesgos.	1 a la 3
OAH A (habilidades)	Búsqueda y análisis de información: comprendidas como habilidades relacionadas con la identificación de una variedad de tipos de fuentes; el acceso a estas; su examen, para luego aceptarlas o rechazarlas; y el análisis e interpretación de la información que dichas fuentes proveen.	1

Unidad 1: Iniciación al Pensamiento Computacional y el Impacto de las tecnologías en nuestra sociedad

Horas pedagógicas sugeridas: 21

Objetivos de Aprendizaje Tecnología 1ro Medio		**Lecciones que cubren OA**
OAH C (habilidades)	Creación: vinculada con habilidades para proponer y diseñar un nuevo objeto, sistema o servicio tecnológico como alternativa de solución frente a problemas personales o colectivos asociados a la tecnología usando lenguajes técnicos.	1 a la 3
OAH E (habilidades)	Manejo de materiales, recursos energéticos, herramientas, técnicas y tecnología: referidas al conjunto de habilidades asociadas a la capacidad de transformar y adaptar recursos tangibles e intangibles en pos de producir soluciones tecnológicas.	1 a la 21
OAH F (habilidades)	Trabajo en equipo: entendido como la capacidad de centrarse en los objetivos y coordinar acciones con otros, gestionar el tiempo, debatir y escuchar para llegar a acuerdos, solicitar y prestar cooperación para el cumplimiento de tareas habituales o emergentes.	1 a la 3
OAH G (habilidades)	Comunicación: referida a un conjunto de habilidades asociadas a informar diseños, planes y resultados de su trabajo en procesos tecnológicos; contribuir productivamente en su discusión y/o elaboración; escuchar, comprender y responder en forma constructiva; y utilizar una variedad de formatos de comunicación.	1 a la 3

Unidad 1: Iniciación al Pensamiento Computacional y el Impacto de las tecnologías en nuestra sociedad **Horas pedagógicas sugeridas**: 21		
Objetivos de Aprendizaje Tecnología 1ro Medio		**Lecciones que cubren OA**
OAH H (habilidades)	Reflexión crítica y responsable: referida a un conjunto de habilidades asociadas a la capacidad de reflexionar sobre tecnología, considerando criterios de impacto social y ambiental, de calidad, de efectividad, de respeto y ética.	1 a la 3

Unidad 2: Desarrollo e Implementación de soluciones con TIC **Horas pedagógicas sugeridas**: 17		
Objetivos de Aprendizaje Tecnología 1ro Medio		Lecciones que cubren OA
OA 01	Identificar oportunidades o necesidades personales, grupales o locales que impliquen la creación de un servicio, utilizando recursos digitales u otros me-dios.	27 a la 38
OA 02	Identificar oportunidades o necesidades personales, grupales o locales que impliquen la creación de un servicio, utilizando recursos digitales u otros me-dios.	27 a la 38
OA 03	Evaluar el servicio desarrollado considerando criterios propios, técnicos y valóricos, y proponer mejoras asociadas tanto a los procesos como al producto final.	27 a la 38

OA 04	Comunicar el diseño, la planificación u otros procesos del desarrollo de un servicio, utilizando herramientas TIC, considerando diferentes tipos de objeti-vos y audiencias y teniendo en cuenta aspectos éticos.	28, 33 y 38
OAA A (actitudes)	Respetar al otro y al medio ambiente, lo que se expresa en los requerimientos del trabajo colaborativo exigido en la producción de soluciones tecnológicas, en la reflexión y el debate sobre el análisis de productos tecnológicos, en la conservación de los recursos y del bien común, entre otros.	22 a la 38
OAA B (actitudes)	Valorar las potencialidades propias y del otro, en relación con el desarrollo de capacidades técnicas y tecnológicas, de los desarrollos tecnológicos en virtud de su aporte al mejoramiento de la calidad de vida, con todo lo que su producción requiere.	22 a la 38
OAA C (actitudes)	Trabajar colaborativamente, lo que se refleja en el compromiso por la prosecución de los objetivos del equipo; en asumir responsabilidades en el grupo y establecer maneras de trabajo eficiente; en aceptar consejos y críticas, escuchando y respetando al otro para llegar a acuerdos; en tomar conciencia y superar las dificultades personales y del trabajo; en aprender de los errores; en solicitar y prestar ayuda a sus pares para el cumplimiento de las metas del trabajo.	27 a la 38
OAA D (actitu-des)	Demostrar disposición hacia la prevención de riesgos y el autocuidado, entendidos como la capacidad progresiva de valorar la vida, del cuerpo, el bienestar y la salud; así como el desarrollo de prácticas y hábitos para mejorar la propia seguridad y la de los demás y, con ello, prevenir riesgos.	27 a la 38

OAH A (habilidades)	Búsqueda y análisis de información: comprendidas como habilidades relacionadas con la identificación de una variedad de tipos de fuentes; el acceso a estas; su examen, para luego aceptarlas o rechazarlas; y el análisis e interpretación de la información que dichas fuentes proveen.	27 a la 38
OAH B (habilidades)	Adaptabilidad y flexibilidad: relacionadas con un grupo de habilidades que permiten asumir cambios personales frente a las exigencias que imponen la dinámica y rapidez de las transformaciones en el ámbito tecnológico. Esto es, capacidades para generar ideas explorando muchas soluciones posibles, y encontrar nuevas maneras de abordar y resolver problemas y situaciones.	27 a la 38
OAH C (habilidades)	Creación: vinculada con habilidades para proponer y diseñar un nuevo objeto, sistema o servicio tecnológico como alternativa de solución frente a problemas personales o colectivos asociados a la tecnología usando lenguajes técnicos.	27 a la 38
OAH D (habilidades)	Emprendimiento: entendido como la capacidad para resolver y superar situaciones en las que la aplicación de la tecnología o la innovación en ella se constituyen en una oportunidad de mejorar la calidad de vida.	27 a la 38
OAH E (habilidades)	Manejo de materiales, recursos energéticos, herramientas, técnicas y tecnología: referidas al conjunto de habilidades asociadas a la capacidad de transformar y adaptar recursos tangibles e intangibles en pos de producir soluciones tecnológicas.	22 a la 38

OAH F (habilidades)	Trabajo en equipo: entendido como la capacidad de centrarse en los objetivos y coordinar acciones con otros, gestionar el tiempo, debatir y escuchar para llegar a acuerdos, solicitar y prestar cooperación para el cumplimiento de tareas habituales o emergentes.	27 a la 38
OAH G (habilidades)	Comunicación: referida a un conjunto de habilidades asociadas a informar diseños, planes y resultados de su trabajo en procesos tecnológicos; contribuir productivamente en su discusión y/o elaboración; escuchar, comprender y responder en forma constructiva; y utilizar una variedad de formatos de comunicación.	28, 33 y 38
OAH H (habilidades)	Reflexión crítica y responsable: referida a un conjunto de habilidades asociadas a la capacidad de reflexionar sobre tecnología, considerando criterios de impacto social y ambiental, de calidad, de efectividad, de respeto y ética.	27 a la 38

Unidad 01

Iniciación al Pensamiento Computacional y el Impacto de las tecnologías en nuestra sociedad

Ideo Digital

Lección 1: del diseño a la implementación de una campaña

Propósito

Al principio de la clase, se realizará una breve introducción acerca de la importancia de comprender que los estudiantes sean ciudadanos digitales, identificando problemas o necesidades y proponiendo soluciones, a través de un proyecto, utilizando herramientas digitales para difundir la información. A continuación, se explicará el significado y relevancia de algunos nuevos conceptos (ciudadanía digital, proyecto comunitario y campaña digital). Luego, verán un video y responderán algunas preguntas. Una vez concluida esta Lección, conformarán grupos de trabajo y crearán un listado de problemas con sus posibles soluciones. Finalmente, elaborarán en el pizarrón un listado común con las mejores propuestas de problemas detectados y posibles soluciones.

Los objetivos de esta Lección son desarrollar habilidades de trabajo en equipo, comunicación, reflexión crítica y responsable y aprendizaje de contenidos como: que es la ciudadanía digital, cómo puedo ejercer la ciudadanía digital en mi comunidad y qué relación existe entre la ciudadanía digital y la creación de proyectos comunitarios. En esta Lección, los estudiantes crearán un listado de problemas con sus respectivas posibles soluciones.

Secuencia para el aprendizaje

- Conocimiento inicial (10 min)
- Ampliación del conocimiento (15 min)
- Transferencia del conocimiento (15 min)
- Evaluación (5 min)

Objetivo

Los estudiantes serán capaces de:

- Discutir cómo las tecnologías computacionales han cambiado el mundo y expresar como esas tecnologías influencian y están influenciadas por prácticas culturales.

Preparación

(Opcional) vea el video Embajadoras Digitales.

Encuentre un espacio que cuente con las herramientas necesarias para su desarrollo. Un laboratorio de computación con conectividad es una opción recomendable.

Asegúrese que cada estudiante registre en su cuaderno de Tecnología el listado común con las mejores propuestas de problemas detectados y posibles soluciones.

Recursos

Para los Profesores:

Video – Embajadoras Digitales

Para los estudiantes:

Video – Embajadoras Digitales

Vocabulario

Ciudadanía Digital: normas de comportamiento que conciernen al uso de la tecnología. Un ciudadano digital es quién sabe aprovechar el acceso a los servicios en línea que ofrecen las instituciones en la actualidad.

Proyecto comunitario: lecciones que se llevan a cabo para solucionar un problema que sufren los integrantes de cierto grupo social.

Campaña digital: estrategia que tiene como propósito impactar con un mensaje a un público determinado, generando una acción que contribuya directamente en la consecución de los objetivos estipulados.

Estrategia de aprendizaje

Conocimiento inicial (10 min)

Introducción del diseño a la implementación de una campaña

En esta Lección, los estudiantes comenzarán con una breve introducción al tema. Luego, se explicará el significado y relevancia de algunos nuevos conceptos (ciudadanía digital, proyecto comunitario y campaña digital). Para terminar, sus estudiantes verán un video relacionado con el tema (conociendo el proyecto "Embajadoras digitales") y responderán algunas preguntas.

Decir: Hoy vamos a aprender la importancia de comprender qué significa ser un ciudadano digital y su relación con la creación de proyectos comunitarios, utilizando medios digitales. La clase comenzará con una introducción acerca del tema y luego aprenderemos vocabulario nuevo.

Análisis: ¿qué es la ciudadanía digital? ¿cómo puedo ejercer la ciudadanía digital en mi comunidad? y ¿qué relación existe entre la ciudadanía digital y la creación de proyectos comunitarios?

Guíe el análisis hacia una conversación sobre la importancia que tiene que los estudiantes sean ciudadanos digitales, identificando problemas y proponiendo soluciones, a través de un proyecto usando tecnologías.

Vocabulario

Esta lección tiene tres nuevas e importantes palabras:

- **Ciudadanía Digital:** La ciudadanía digital, puede definirse como las normas de comportamiento que conciernen al uso de la tecnología o la manera de proceder con medios electrónicos. Un ciudadano digital es quién sabe aprovechar el acceso a los servicios en línea que ofrecen las instituciones públicas en la actualidad. Es quién logra hacer de forma más eficiente cualquier trámite relacionado con empleo, comercio, pensión, salud y educación.
- **Proyecto comunitario:** Un proyecto es un conjunto de ideas, planes y acciones que se desarrollan de manera coordinada con el objetivo de lograr el cumplimiento de un objetivo. Comunitario, es aquello que se asocia a una comunidad (el grupo de seres humanos o países que mantienen intereses en común). La noción de proyecto comunitario, se refiere a las lecciones que se llevan a cabo para solucionar un problema o paliar un déficit que sufren los integrantes de cierto grupo social. Lo habitual es que esta clase de proyecto sea impulsada por las propias personas que tienen que resolver la cuestión.
- **Campaña digital:** Una campaña digital es una estrategia que tiene como propósito impactar con un mensaje o propuesta de valor a un público determinado, generando una acción que contribuya directamente en la consecución de los objetivos estipulados.

Decir: Ahora que ya se realizó la introducción al tema de la clase y la explicación del significado y relevancia de algunos nuevos conceptos. Vamos a ver un video con el propósito de responder algunas preguntas acerca del proyecto "Embajadoras Digitales" de estudiantes del Colegio Arturo Toro Amor, de la Región Metropolitana:

- http://bit.ly/2mPmu99/ (Embajadoras Digitales)

A continuación, comience a escribir en el pizarrón el listado de preguntas:

- ¿Cuál fue el problema o necesidad planteada en el video por las estudiantes?
- ¿Cuál fue su propuesta de solución? ¿De qué se trataba?
- ¿Cuál fue su plan de trabajo?
- ¿De qué manera contribuyeron a la sociedad?

Continúe esta Lección hasta que los estudiantes hayan respondido todas las preguntas, pídales que registren su listado en su cuaderno de Tecnología.

Ampliación del conocimiento (15 min)

Identificación de algún problema y propuesta de una solución

En esta Lección, los estudiantes conformarán grupos de trabajo y conversarán acerca de algún problema o necesidad que afecte a su comunidad. Luego, a partir de lo discutido en clases y, considerando los problemas o necesidades identificadas por los grupos de trabajo, los estudiantes escribirán un listado de problemas con sus posibles soluciones.

Decir: Ya hemos respondido las preguntas, acerca del video del proyecto “Embajadoras Digitales” de las alumnas del Colegio Arturo Toro Amor, de la Región Metropolitana y sabemos que quienes implementan esta iniciativa, tienen como objetivo realizar talleres de uso seguro de internet en su comunidad. Ahora, vamos a trabajar en grupos y elaboraremos un listado de problemas con sus posibles soluciones.

Interacción: Pida a los estudiantes que trabajen en equipo para elaborar el listado de problemas o necesidades con sus posibles soluciones.

Continúe hasta que los estudiantes hagan un listado con al menos 2 ejemplos de problemas o necesidades con sus posibles soluciones.

Recomendaciones para el docente de algunas ideas de problemas o necesidades detectadas en su comunidad, con sus respectivas soluciones:

PROBLEMA COMUNIDAD	PROPUESTA SOLUCIÓN
Separación de residuos	Destinar recursos para comprar y ubicar en espacios estratégicos (por ejemplo: la entrada y el patio del colegio) nuevos contenedores de basura de diferentes colores para facilitar la tarea de la clasificación de residuos por tipo dentro del colegio (por ejemplo: contenedores de basura para plásticos, vidrio, cartones y papeles, latas, tetra pack y desechos orgánicos). Habilitar más espacios, como puntos limpios o verdes además de la entrada y patio dentro del colegio (por ejemplo: salas de clase, sala de fotocopiadoras y secretaría). Crear un proyecto de reciclaje de basura que permita clasificar los desechos por tipo en espacios accesibles dentro del colegio.

PROBLEMA COMUNIDAD	PROPUESTA SOLUCIÓN
Alimentación saludable	Destinar recursos para ofrecer y vender en el kiosco del colegio más alternativas de comida saludable (por ejemplo: sándwich de pan integral, agua embotellada, frutas, ensaladas, frutos secos, entre otros). Realizar un cambio en el menú del almuerzo del colegio que incluya el consumo regular de alimentos que favorezcan una dieta balanceada y saludable (por ejemplo: cereales, verduras y frutas, lácteos, carne, pescado, huevos, legumbres, etc). Crear un proyecto de alimentación saludable que promueva en el colegio una alimentación baja en grasas y rica en nutrientes en todos los horarios establecidos para el consumo de alimentos (por ejemplo: los recreos y el horario de almuerzo).

PROBLEMA COMUNIDAD	PROPUESTA SOLUCIÓN
Utilización de paneles solares	Destinar recursos para invertir en energía solar e instalar sistemas solares fotovoltaicos de autoconsumo sobre la fachada del colegio para concientizar a la comunidad educativa del uso de energías renovables no convencionales, disminuir los costos energéticos y ayudar al medio ambiente, mediante el uso y suministro de energía limpia. Crear un proyecto de eficiencia energética en el colegio para promover el uso de energía solar, que represente no sólo un ahorro económico sustancial para el establecimiento educativo, sino que además le permita generar su propia energía y contribuir con la reducción de la huella de carbono.

PROBLEMA COMUNIDAD	PROPUESTA SOLUCIÓN
Creación de una huerta orgánica	Destinar recursos para la compra de materiales para la construcción de un huerto y establecimiento de una bodega para guardar insumos, herramientas y materiales en el colegio (por ejemplo: semillas, insecticidas y abonos orgánicos, tierra negra, materia orgánica, arena, azadón, palas, mangueras, entre otros), con el objeto de contribuir a una educación autosustentable y saludable. Habilitar nuevos espacios para construir huertas caseras en el colegio y preparar a la comunidad (por ejemplo: padres y apoderados) para cultivar alimentos en sus hogares. Si no hay suficiente terreno se puede hacer una huerta vertical usando material PET. Para ello necesitarán botellas de plástico, semillas, tierra y luz natural. De estas dos maneras, se estará promoviendo el autocuidado, la alimentación saludable y el cuidado del medio ambiente. Diseñar un proyecto de huertas alimentarias sustentables y orgánicas en el colegio para que los profesores enseñen a sus alumnos a producir sus alimentos y manejar sus desechos. La idea consiste en que a partir de esta experiencia de vida los estudiantes tomen conciencia de la importancia de la alimentación sana y el cuidado del medio ambiente.

Transferencia del conocimiento (15 min)

Elaboración de un listado común de un problema y propuesta de una solución

En esta Lección, los estudiantes elaborarán en el pizarrón un listado común con las mejores propuestas de problemas detectados y posibles soluciones. Para terminar, seleccionarán uno o dos problemas para ser resueltos entre todos.

Algunos ejemplos de problemas o necesidades de la comunidad:

- Clasificación de desechos.
- Búsqueda de un hogar para mascotas.
- Vida y alimentación sana.
- Campaña de donación de ropa y juguetes para niños vulnerables.

Decir: Ahora que ya trabajaron en grupo y elaboraron un listado de problemas o necesidades que afecten a su comunidad, vamos a realizar en el pizarrón un listado común con las mejores propuestas de problemas detectados y posibles soluciones. Luego, seleccionarán uno o dos de ellos con el objeto de definir el tema y la propuesta de solución del proyecto.

Continúe esta Lección hasta que los estudiantes hagan un listado común con al menos dos problemas o necesidades que desean resolver en su comunidad y seleccionen uno de ellos. Aquel problema o necesidad que resulte escogido será el proyecto que en grupos de trabajo comenzarán a desarrollar la próxima clase. Pídales que registren su listado en su cuaderno de Tecnología.

Evaluación (5 min)

Pida a la clase que abra su cuaderno de Tecnología y que escriban el título de la evaluación – Del diseño a la implementación de una campaña. A continuación, indíqueles que escriban 2 ejemplos más de problemas o necesidades con sus respectivas posibles soluciones. Luego de haber explicado claramente las instrucciones, permita que los estudiantes realicen la evaluación de forma individual. Esta Lección puede ser utilizada como una evaluación formativa.

Gracias a la Lección previa, esto no debería significar ningún problema para ellos.

Experiencias de aprendizaje de profundización

Use estos Contenidos para motivar a sus estudiantes a detectar problemas o necesidades presentes en su comunidad y proponer una solución, a través de un proyecto.

Cada vez mejor

- Que la clase intente hacer un diagrama de árbol acerca del problema o necesidad detectada y de las propuestas de soluciones.
- ¿Pueden relatar algunas ideas de soluciones para el problema detectado?

Desafío de curso

- Realice un mural colectivo para la sala que reúna todas las ideas acerca del problema o necesidad detectada y propuestas de solución.

Lección 2: del diseño a la implementación de una campaña

Lección sin conexión

Propósito

Esta Lección, comenzará con una breve síntesis del tema (la importancia de comprender que los alumnos sean ciudadanos digitales, identificando problemas o necesidades y proponiendo soluciones, a través, de un proyecto, utilizando herramientas digitales para presentar y transmitir la información) y del vocabulario (ciudadanía digital, proyecto comunitario y campaña digital). A continuación, los estudiantes conformarán grupos de trabajo y responderán algunas preguntas. Finalmente, elaborarán una lista de materiales en el pizarrón.

Los objetivos de esta Lección son desarrollar habilidades de trabajo en equipo, comunicación, reflexión crítica y responsable y aprendizaje de contenidos como: qué es la ciudadanía digital, cómo puedo ejercer la ciudadanía digital en mi comunidad y qué relación existe entre la ciudadanía digital y la creación de proyectos comunitarios. En esta Lección, los estudiantes responderán algunas preguntas que los ayudarán a definir todo lo necesario para la realización del proyecto.

Secuencia para el aprendizaje

- Conocimiento inicial (10 min)
- Ampliación del conocimiento (15 min)
- Transferencia del conocimiento (15 min)
- Evaluación (5 min)

Objetivo

Los estudiantes serán capaces de:

- Discutir como las tecnologías computacionales han cambiado el mundo y expresar como esas tecnologías influencian y están influenciadas por prácticas culturales.

Preparación

Asegúrese que cada estudiante registre en su cuaderno de tecnología el listado de todos los materiales que necesitan traer para la realización de la campaña la clase siguiente.

Vocabulario

- **Ciudadanía Digital:** normas de comportamiento que conciernen al uso de la tecnología. Un ciudadano digital es quién sabe aprovechar el acceso a los servicios en línea que ofrecen las instituciones en la actualidad.
- **Proyecto comunitario:** lecciones que se llevan a cabo para solucionar un problema que sufren los integrantes de cierto grupo social.
- **Campaña digital:** estrategia que tiene como propósito impactar con un mensaje a un público determinado, generando una acción que contribuya directamente en la consecución de los objetivos estipulados.

Estrategia de aprendizaje

Conocimiento inicial (10 min)

Del diseño a la implementación de una campaña digital

En esta Lección, los estudiantes comenzarán con una breve síntesis del tema (la importancia de comprender que los alumnos sean ciudadanos digitales, identificando problemas o necesidades y proponiendo soluciones, a través, de un proyecto, utilizando herramientas digitales para presentar y transmitir la información) y del vocabulario (ciudadanía digital, proyecto comunitario y campaña digital) de la clase anterior.

Decir: Hoy vamos a recordar la importancia de comprender qué significa ser un ciudadano digital y su relación con la creación de proyectos comunitarios, utilizando medios digitales. La clase comenzará con una síntesis acerca del tema y el vocabulario de la clase anterior.

Análisis: La clase anterior aprendimos qué es la ciudadanía digital, cómo puedo ejercerla en mi barrio o comunidad escolar y la relación que existe entre esta y la creación de un proyecto comunitario. También, aprendimos el significado de algunas palabras, vimos un video y respondimos algunas preguntas relacionadas con el proyecto "Embajadoras Digitales". Pero ¿qué relación existe entre todo lo que hemos aprendido con el diseño y la implementación de un proyecto? ¿por qué fue importante que la clase pasada de un listado común seleccionáramos el problema o necesidad que deseábamos resolver en nuestra comunidad? ¿para qué nos sirve detectar este problema o necesidad? y ¿cuáles son los siguientes pasos para realizar nuestra campaña?

Guíe el análisis hacia una conversación sobre la importancia de ser ciudadanos digitales en el territorio, identificando problemas o necesidades y proponiendo soluciones, a través de un proyecto, utilizando herramientas digitales para presentar y transmitir la información.

Ampliación del conocimiento (15 min)

Creación de un plan de trabajo para el proyecto

En esta Lección, los estudiantes conformarán grupos de trabajo para responder algunas preguntas que los ayudarán a definir todo lo necesario para realizar el proyecto.

Decir: Ya hemos aprendido la importancia de que los alumnos sean ciudadanos digitales, identificando problemas o necesidades y proponiendo soluciones, a través de un proyecto, utilizando herramientas digitales para presentar y transmitir la información. Además, aprendimos vocabulario nuevo y vimos un video relacionado con el tema. Ahora que ya hemos seleccionado un problema o necesidad que deseamos resolver en nuestra comunidad, vamos a trabajar en grupos y responderemos las preguntas del cuestionario, que nos ayudarán a definir todo lo necesario para realizar el proyecto.

Interacción: Pida a los estudiantes que trabajen en equipo para responder algunas preguntas que los ayudarán a definir todo lo necesario para realizar el proyecto.

A continuación, comience a escribir en el pizarrón el listado de preguntas:

- ¿Qué se quiere hacer?
- ¿Por qué se quiere hacer?
- ¿Para qué se quiere hacer?
- ¿Qué queremos alcanzar?
- ¿A quiénes queremos ayudar?

- ¿Dónde se quiere hacer?
- ¿Cómo se va a hacer?
- ¿En qué periodo se va a hacer?
- ¿Quiénes lo van a hacer?
- ¿Con qué se va a hacer?
- ¿Quién debe conocer nuestra campaña?
- ¿Qué redes sociales usaremos?
- ¿Videos, imágenes u otros?
- ¿Quiénes pueden ayudar a difundir?

Tómese un momento para reiterar la importancia de responder todas las preguntas de la Lección, ya que este cuestionario los ayudará a definir todo lo necesario para realizar el proyecto.

Continúe hasta que los estudiantes hayan respondido todas las preguntas de esta Lección.

Recomendaciones para el docente de algunas ideas de preguntas para el diseño de una campaña:

PREGUNTAS PROCESOS DE DISEÑO DE LA CAMPAÑA		
CONCEPTOS	PREGUNTAS DEL PROYECTO	POSIBLES RESPUESTAS ALUMNOS
Tema de la campaña	¿Qué se quiere hacer?	
Origen y fundamentación	¿Por qué se quiere hacer?	
Objetivos y propósito	¿Para qué se quiere hacer?	
Metas	¿Qué queremos alcanzar?	
Destinatarios y Beneficiarios	¿A quiénes queremos ayudar?	
Localización o cobertura	¿Dónde se quiere hacer?	
Lecciones y tareas	¿Cómo se va a hacer?	
Calendarización o cronograma	¿En qué periodo se va a hacer?	
Recursos humanos	¿Quiénes lo van a hacer?	
Recursos materiales	¿Con qué se va a hacer?	

Público objetivo	¿Quién debe conocer nuestra campaña?	
Medios digitales	¿Qué redes sociales usaremos?	
Productos	¿Videos, imágenes u otros?	
Grupos de interés	¿Quiénes pueden ayudar a difundir?	

Transferencia del conocimiento (15 min)

Elaboración de un listado de materiales para el diseño de productos de la campaña

En esta Lección, los estudiantes elaborarán en el pizarrón un listado de todos los materiales que necesitan traer para la realización de la campaña la clase siguiente, ya que deberán trabajar en la creación de los productos de la campaña.

Algunos ejemplos de materiales para la realización de la campaña:

- cámara de video o fotográfica.
- computadores conectados a internet.
- dispositivos móviles como tablet o smartphone.

Decir: Ahora que ya trabajaron en grupo y respondieron todas las preguntas de la ficha de trabajo de la Lección anterior, vamos a realizar un listado con todos los materiales que necesitan traer para la realización de la campaña la clase siguiente.

Tómese un momento para reiterar la importancia de recordar a los estudiantes los materiales que necesitan traer para la clase siguiente, ya que deberán trabajar en la creación de los productos de la campaña.

También es importante mencionar y enfatizar el uso y manejo de otro tipo de medios digitales que servirán para desarrollar y difundir la campaña digital.

Algunos ejemplos de recursos digitales para desarrollar y difundir la campaña:

- Cuentas de redes sociales: Youtube, Facebook, Twitter, Instagram, Pinterest, Whatsapp, SnapChat, entre otros.
- Editores de imágenes y videos: Canva, Piktochart o Fotojet.
- Plataformas de creación de blog: Blogger o Wordpress y Software de presentación.

Continúe esta Lección hasta que los estudiantes hayan escrito en el pizarrón todos los materiales que necesitan traer para la clase siguiente. Pídales que registren su listado en su cuaderno de Tecnología.

Evaluación (5 min)

Pida a la clase que abra su cuaderno de Tecnología y que escriban el título de la evaluación – Del diseño a la implementación de una campaña. A continuación, indíqueles que respondan dos preguntas: ¿Por qué es importante definir un plan de trabajo para el proyecto? Y ¿qué relación

existe entre el uso de herramientas digitales y una campaña de difusión? Luego de haber explicado claramente las instrucciones, permita que los estudiantes realicen la evaluación de forma individual. Esta Lección puede ser utilizada como una evaluación formativa.

Gracias a las lecciones realizadas durante la clase, esta tarea no debería significar ningún problema para ellos.

Experiencias de aprendizaje de profundización

Use estos Contenidos para motivar a sus estudiantes a detectar problemas o necesidades presentes en su comunidad y proponer posibles soluciones, a través de un proyecto.

Cada vez mejor

- Que la clase intente hacer un cuadro de doble entrada acerca de los medios digitales y productos que podrían lograr ser más efectivos en sensibilizar o generar acciones ciudadanas para la campaña.
- ¿Pueden explicar con sus propias palabras por qué es más efectivo usar herramientas digitales para difundir la información?

Desafío de curso

- Realice una pequeña encuesta en su curso de las mejores opciones de redes sociales en las que puede ser difundido el mensaje de la campaña.

Lección 3: del diseño a la implementación de la campaña

Propósito

Al principio de la clase, se realizará una breve síntesis de lo visto la clase anterior. Poniendo especial énfasis en el tema y objetivo de las lecciones más importantes realizadas en el transcurso de la última clase (por ejemplo: planificación del proyecto, a través, de la completación de una ficha con preguntas relativas a definir un plan de trabajo y elaboración de listado de materiales para crear productos para la campaña). Una vez que esta Lección haya concluido, los estudiantes conformarán grupos de trabajo y comenzarán a trabajar en el proceso de creación de productos para la campaña. Para terminar, dedicarán todo el resto de la clase a continuar trabajando en grupo en el diseño de los productos para difundir la campaña.

Los objetivos de esta Lección son desarrollar habilidades de trabajo en equipo, comunicación, reflexión crítica y responsable y aprendizaje de contenidos como: qué es la ciudadanía digital, cómo puedo ejercer la ciudadanía digital en mi comunidad y qué relación existe entre la ciudadanía digital y la creación de proyectos comunitarios. En esta Lección los estudiantes, comenzarán a trabajar en proceso de creación de los productos para la campaña.

Secuencia para el aprendizaje

- Conocimiento inicial (10 min)
- Ampliación del conocimiento (15 min)
- Transferencia del conocimiento (15 min)
- Evaluación (5 min

Objetivo

Los estudiantes serán capaces de:

- Discutir cómo las tecnologías computacionales han cambiado el mundo y expresar cómo esas tecnologías influencian y están influenciadas por prácticas culturales.

Preparación

- (Opcional) vea los enlaces recomendados en los recursos para profesores.
- Encuentre un espacio que cuente con las herramientas necesarias para su desarrollo. Un laboratorio de computación con conectividad es una opción recomendable.
- Asegúrese que cada estudiante respalde la información creada o los avances del diseño de los productos de la campaña digital en alguna carpeta de trabajo en el computador o pendrive.

Recursos

Para los Profesores

- https://vine.co (Micro videos)
- http://www.memegenerator.es (Memes)
- https://www.canva.com/es_es/crear/baners/ (Publicidad)
- http://bit.ly/2xb8lon (Archivo de audio gratuito)

Para los estudiantes:

- https://vine.co (Micro videos)
- http://www.memegenerator.es (Memes)
- https://www.canva.com/es_es/crear/baners/ (Publicidad)
- http://bit.ly/2xb8lon (Archivo de audio gratuito)

Vocabulario

- **Ciudadanía Digital:** normas de comportamiento que conciernen al uso de la tecnología. Un ciudadano digital es quién sabe aprovechar el acceso a los servicios en línea que ofrecen las instituciones en la actualidad.
- **Proyecto comunitario:** lecciones que se llevan a cabo para solucionar un problema que sufren los integrantes de cierto grupo social.
- **Campaña digital:** estrategia que tiene como propósito impactar con un mensaje a un público determinado, generando una acción que contribuya directamente en la consecución de los objetivos estipulados.

Estrategia de aprendizaje

Conocimiento inicial (10 min)

Del diseño a la implementación de una campaña

En esta Lección, los estudiantes comenzarán con una breve síntesis de lo visto la clase anterior. Poniendo especial énfasis en el tema y objetivo de las lecciones más importantes realizadas en esa oportunidad (por ejemplo: planificación del proyecto, a través, de la completación de una ficha con preguntas relativas a definir un plan de trabajo y elaboración de un listado de materiales para la creación de productos para la campaña).

Decir: Hoy vamos a recordar lo que hicimos la clase pasada. Completamos una ficha de trabajo, dónde aparecía un cuestionario con preguntas importantes acerca de nuestro proyecto y además, para terminar realizamos una lista en el pizarrón de los materiales que necesitaban traer esta clase para el diseño o preparación de los productos de nuestra campaña digital. La clase comenzará con una breve síntesis acerca del plan de trabajo de nuestro proyecto y listado de materiales para crear productos para la campaña.

Tómese un momento para reiterar la importancia de recordar a los estudiantes los materiales que necesitan utilizar en el transcurso de la clase, ya que deberán trabajar en la creación de los productos de la campaña.

Algunos ejemplos de materiales para la realización de la campaña:

- cámara de video o fotográfica.
- computadores conectados a internet.
- dispositivos móviles como tablet o smartphone.

También, es importante, mencionar y enfatizar el uso de otro tipo de medios digitales que servirán para desarrollar y difundir la campaña digital.

Algunos ejemplos de recursos digitales para desarrollar y difundir la campaña:

- Cuentas de redes sociales: Youtube, Facebook, Twitter, Instagram, Pinterest, Whatsapp, SnapChat, entre otros.
- Editores de imágenes y videos: Canva, Piktochart o Fotojet.
- Plataformas de creación de blog: Blogger o Wordpress y Software de presentación.

Ampliación del conocimiento (15 min)

Diseño de productos para la campaña

En esta Lección, los estudiantes conformarán grupos de trabajo y comenzarán a trabajar en el proceso de creación de productos para la campaña.

Decir: Ya hemos trabajado en la planificación de nuestro proyecto. Ahora, estamos trabajando en el proceso de diseño de nuestra campaña digital. Para poder implementar nuestra campaña, es importante crear los productos que sean necesarios, ya que serán éstos los que nos permitirán difundirla. Ahora, que ya hemos realizado el plan de trabajo, vamos a comenzar a trabajar en grupo en el proceso de creación de productos de la campaña.

Interacción: Pida a los estudiantes que trabajen en equipo para crear el diseño de los productos para difundir la campaña.

A continuación, comience a escribir en el pizarrón la lista de sitios de medios digitales que los ayudarán en la tarea del diseño de los productos para la campaña:

- https://vine.co (Micro videos).

- http://www.memegenerator.es (Memes)
- https://www.canva.com/es_es/crear/baners/ (Publicidad).
- http://bit.ly/2xb8lon (Archivo de audio gratuito).

Tómese un momento para reiterar la importancia de esta parte del proceso, ya que la preparación de los productos de la campaña digital permitirá generar acciones ciudadanas, sobre el problema o necesidad detectada en la comunidad, utilizando tecnologías (por ejemplo: computadores conectados a internet, dispositivos móviles como tablet o smartphone, etc.).

También es importante enfatizar que, considerando el tiempo disponible que tienen los estudiantes para la realización de la Lección, es recomendable comentarles que deben usar una cantidad limitada de imágenes y redes sociales en el diseño de la campaña digital.

Transferencia del conocimiento (15 min)

Continuación de diseño de productos para difundir la campaña

En esta Lección, los estudiantes dedicarán todo el resto de la clase a continuar trabajando en grupo el diseño de los productos para difundir la campaña, ya que necesitarán tiempo para grabar audios, videos, seleccionar las imágenes y todos los materiales que sean necesarios para su realización.

Decir: Hoy comenzamos a trabajar en el proceso de diseño de la campaña digital. En el transcurso de la última parte de la clase, continuaremos trabajando en la preparación de los productos para la campaña digital, ya que necesitarán tiempo para grabar audios, videos, seleccionar las imágenes y todos los materiales que sean necesarios para su realización. Para terminar, si nos queda tiempo podemos trabajar en el proceso de edición de los productos de la campaña, comunicar parte de los avances al curso y evaluar en conjunto los resultados. De esta forma, tendrán tiempo de realizar todas las correcciones de los productos que sean necesarias y con ello mejorar la campaña.

Tómese un momento para reiterar la importancia de esta parte del proceso, ya que los avances registrados por los estudiantes en el diseño de productos para la campaña digital pueden ser utilizados en éste o en futuros proyectos que sean realizados por los alumnos en el colegio.

Continúe esta Lección hasta que los estudiantes hayan avanzado en la tarea de la creación de productos para la campaña digital (por ejemplo: grabar audios, videos, seleccionar las imágenes y todos los materiales que sean necesarios para su realización). Pídales que respalden la información diseñada o los avances de la creación de los productos para la campaña digital en alguna carpeta de trabajo en el computador o pendrive.

Evaluación (5 min)

Pida a la clase que abra su cuaderno de Tecnología y que escriban el título de la evaluación – Del diseño a la implementación de una campaña. A continuación, indíqueles que respondan dos preguntas: ¿Cuál es la diferencia entre productos y medios digitales? Y ¿por qué es importante el diseño de los productos para difundir la campaña? Luego de haber explicado claramente las instrucciones, permita que los estudiantes realicen la evaluación de forma individual. Esta Lección puede ser utilizada como una evaluación formativa.

Gracias a las lecciones realizadas durante la clase, esto no debería significar ningún problema para ellos.

Experiencias de aprendizaje de profundización

Use estos Contenidos para motivar a sus estudiantes a detectar problemas o necesidades presentes en su comunidad y proponer una solución, a través de una campaña de difusión.

Cada vez mejor

- Que la clase intente hacer una lista acerca de los mejores productos que podrían ser más efectivos para sensibilizar o generar acciones ciudadanas para la campaña.
- ¿Pueden comentar algunos ejemplos de mensajes o acciones para ser presentados y transmitidos en la campaña?

Desafío de curso

- Realice una muestra en su curso de las mejores opciones de videos o imágenes para la campaña digital.

Lección 4: Programación con papel cuadriculado

Lección sin conexión

Propósito

Al "programarse" unos a otros para hacer dibujos, los estudiantes tendrán la oportunidad de experimentar los conceptos clave de la programación de una forma divertida y accesible.

Al principio de la clase, los estudiantes usarán símbolos para instruirse entre ellos a pintar cuadrados en un papel cuadriculado, a fin de reproducir una imagen existente. Si dispone de tiempo, la Lección puede terminar con imágenes creadas por los mismos estudiantes.

Los objetivos de esta Lección son desarrollar habilidades de pensamiento crítico, sembrar interés por este curso e introducir algunos conceptos fundamentales de la programación que serán usados a lo largo del curso. Al introducir conceptos básicos, como secuenciación o algoritmos, a través de una Lección sin conexión, incluso los estudiantes que no se sienten familiarizados con un computador, podrán sentar las bases para comprender estos temas. En esta Lección, los estudiantes aprenderán cómo desarrollar un algoritmo y codificarlo en un programa.

Secuencia para el aprendizaje

- Conocimiento inicial (10 min)
- Ampliación del conocimiento (30 min)
- Transferencia del conocimiento (15 min)
- Evaluación (10 min)

Objetivos

Los estudiantes serán capaces de:

- Restructurar una secuencia de pasos en un programa codificado.
- Explicar las limitaciones de traducir problemas desde el lenguaje humano al lenguaje de las máquinas

Preparación

- (Opcional) vea el video Lección en acción en la sección recursos para profesores.
- Imprima una guía de trabajo y una evaluación para cada estudiante.
- Asegúrese de que cada estudiante tenga su Bitácora.

Recursos

¡Atención!
Por favor, haga una copia de cada documento que planee compartir con los estudiantes.

Para los Profesores:

- Video – Lección en acción: programación con papel cuadriculado (no olvide activar los subtítulos al español)
- Guía del profesor – Programación con papel cuadriculado
- Guía del profesor – Evaluación – Programación con papel cuadriculado

Para los estudiantes:

- Guía de trabajo – Programación con papel cuadriculado
- Video – Programación con papel cuadriculado
- Evaluación – Programación con papel cuadriculado

Vocabulario

- **Algoritmo:** lista de pasos para realizar una tarea.
- **Programa:** algoritmo que ha sido codificado de forma que pueda ser ejecutado por una máquina

Estrategia de aprendizaje

Conocimiento inicial (10 min)

Introducción a la programación con papel cuadriculado

En esta Lección, los estudiantes codificarán instrucciones para guiarse unos a otros a hacer dibujos, sin que el resto del grupo vea la imagen original. Esta sección contextualizará el ejercicio para la clase.

Mostrar: vea uno de los videos a continuación para contextualizar a los estudiantes sobre la clase de cosas que puede hacer un robot:

- Robot Honda Asimo (1:51).
- Robot diseñador de huevos (3:15).
- Robot bailarín de Lego (1:35).

Análisis: ¿cómo creen que el robot sabe hacer las cosas que hace? ¿los robots tienen un cerebro similar al nuestro?

Guíe el análisis hacia una conversación sobre cómo las personas programan a los robots para hacer cosas específicas, a través de comandos específicos

Este breve análisis tiene por objetivo hacer énfasis en que los robots, a pesar de que pareciesen comportarse como humanos, realmente sólo responden a su programa. Es probable que los estudiantes hagan referencia a algunos robots de las películas y de la televisión con comportamientos más humanos. Guíelos a considerar robots que hayan visto u oído en la vida real, como el Roombas, o incluso asistentes digitales como Alexa o el asistente de Google.

Ampliación del conocimiento (30 min)

Práctica en conjunto

En esta Lección, los estudiantes tomarán los roles de programador y robot. En una hoja de papel cuadriculado, pintarán cuadrados de acuerdo con los programas que se hayan escrito los unos a los otros.

Distribuir: los estudiantes usarán una cuadrícula de 4x4 (u hojas de papel cuadriculado seccionado en cuadriculas de 4x4). También necesitarán una imagen de modelo.

Mostrar: proyecte estos comandos o escríbalos en la pizarra. No estarán mucho tiempo ahí, sólo lo suficiente para ayudar a los estudiantes a hacer la transición de algoritmos a programas.

- Moverse un cuadrado a la derecha
- Moverse un cuadrado a la izquierda
- Moverse un cuadrado arriba
- Moverse un cuadrado abajo
- Rellenar el cuadrado

Generar una instancia de aprendizaje donde los programadores serán los estudiantes y el docente sigue las instrucciones al pie de la letra (como si fuese un robot). Luego, dividimos en grupos para que todos tengan su turno.

Modelar: muestre la imagen que usará de ejemplo y la cuadrícula en blanco que rellenará con su Sistema de Ejecución Automática (SEA). Asegúrese de que las instrucciones, la cuadrícula y la imagen permanezcan visibles al mismo tiempo.

Presentar un robot imaginario que funciona con un Sistema de Ejecución Automática (SEA). Esto significa que reaccionará de forma automática a sus instrucciones, pero sólo a las que pueda entender.

Empiecen en la esquina superior izquierda. Guíen mi SEA diciéndome las instrucciones en voz alta.

Modelar: a continuación, la clase podría dar instrucciones como éstas. Cuando escuche una instrucción que pretenda seguir, asegúrese de repetir dicha instrucción en voz alta, de manera que los estudiantes puedan llevar un registro de sus movimientos.

- Moverse un cuadrado a la derecha
- Rellenar el cuadrado
- Moverse un cuadrado a la derecha
- Moverse un cuadrado abajo
- Rellenar el cuadrado

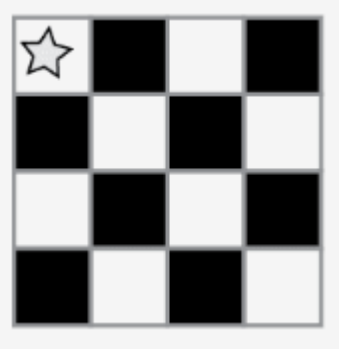
Continúe con la Lección hasta completar la cuadrícula de ejemplo.

- Moverse un cuadrado a la derecha
- Rellenar el cuadrado
- Moverse un cuadrado a la derecha
- Moverse un cuadrado abajo
- Rellenar el cuadrado

Captar: escriba cada uno de los comandos, de manera que los estudiantes puedan ver todos los pasos realizados para dibujar la imagen.

Recordar la definición de algoritmos. Como programadores podemos entenderlos fácilmente. PERO ¿qué pasa si queremos escribir el algoritmo para un dibujo como este?

Mostrar: muestre una imagen más complicada, como ésta.



A continuación, comience a escribir algunas de las instrucciones para replicar la imagen. Con suerte, los estudiantes verán que escribir todo a mano podría fácilmente volverse una pesadilla.

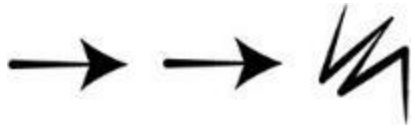
- Moverse un cuadrado a la derecha
- Rellenar el cuadrado
- Moverse un cuadrado a la derecha
- Moverse un cuadrado a la derecha
- Rellenar el cuadrado
- Moverse un cuadrado abajo
- Moverse un cuadrado a la izquierda
- Rellenar el cuadrado
- Moverse un cuadrado a la izquierda
- Moverse un cuadrado a la izquierda
- Rellenar el cuadrado
- ¡¡Y 12 instrucciones más!!

El objetivo de este análisis es llegar a la idea de que los estudiantes pueden usar símbolos para representar frases completas. Una vez que lo comprendan, coménteles que pasar de enlistar pasos detallados a codificarlos, se llama “programación”.

Mostrar: muestre esta lista de símbolos.

Análisis: ¿cómo podemos usar estos símbolos para facilitar nuestras instrucciones?

Continúe explorando ideas que apoyen la transición de instrucciones verbales a símbolos. Una vez que los estudiantes comprendan la idea, indique que el texto:



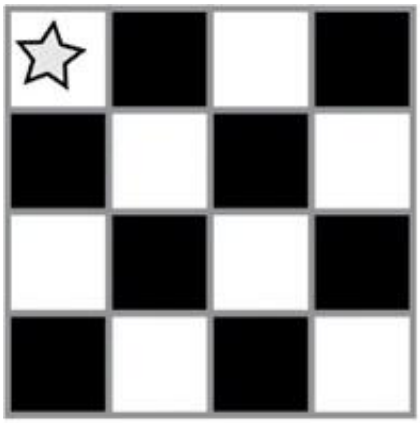
“Moverse un cuadrado a la derecha, moverse un cuadrado a la derecha, rellenar el cuadrado” Ahora corresponde al programa:

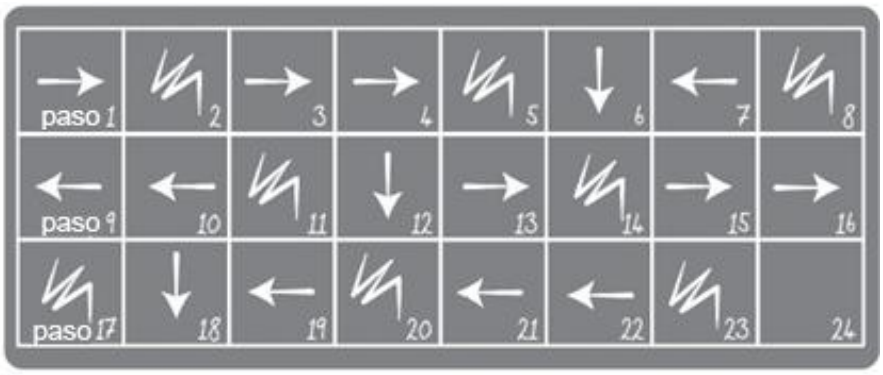
Modelar: ahora, que la clase le ayude a dibujar la imagen grande usando sólo símbolos. Por ahora, no se preocupe si se incluyen pasos innecesarios; si el programa funciona para recrear la imagen, considérelo correcto.

En este punto, los estudiantes podrían estar emocionados y animados dando sugerencias. Si entienden lo esencial del ejercicio, este es un buen momento para analizar otras alternativas para rellenar la misma cuadrícula. Si aún no están listos, guarde esta idea para otro día y realice otro ejemplo.

Note como hemos escrito el programa de izquierda a derecha, como se leería un libro en español. Algunos estudiantes prefieren este método, mientras que otros prefieren empezar cada línea de la cuadrícula en una nueva línea de la hoja. La forma en que escriban el programa no importa mucho, mientras los demás estudiantes puedan leer y seguir el programa.

Vea una solución de ejemplo a continuación:

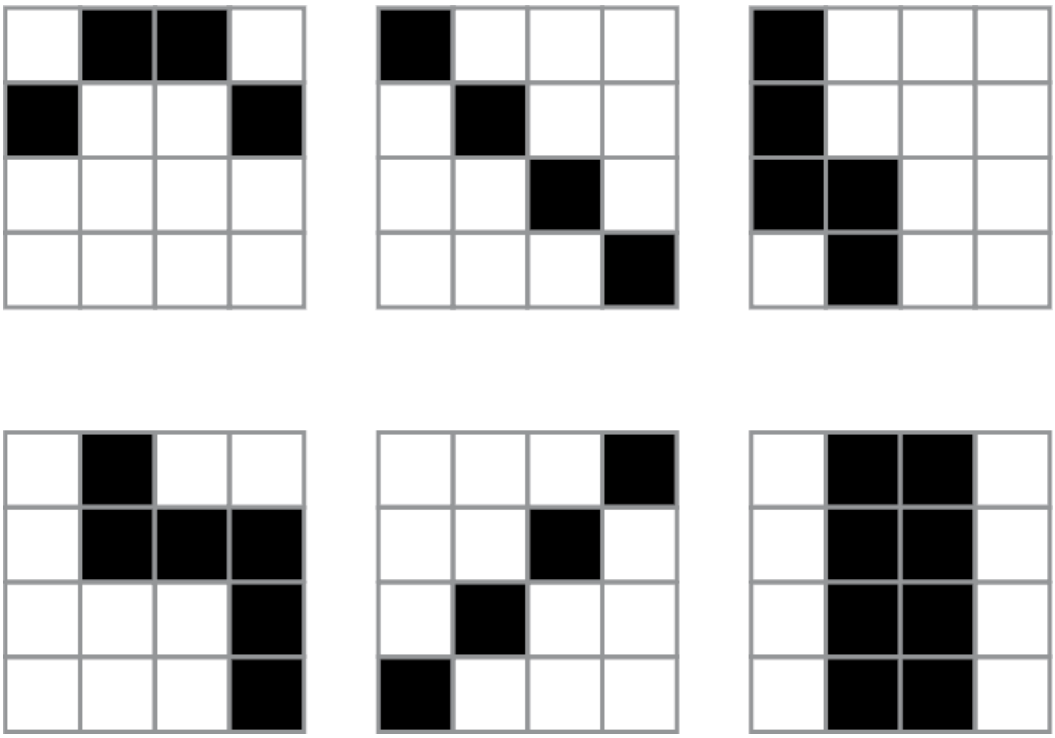




Turno del estudiante

Grupo: divida a los estudiantes en parejas o pequeños grupos.

- Que cada grupo/pareja elija una imagen de la guía de trabajo.
- Discutan con sus compañeros el algoritmo necesario para dibujar la imagen elegida.
- Conviertan el algoritmo en un programa, usando los símbolos.
- Intercambien los programas con otros grupos/parejas, para que dibujen las imágenes de otros.
- Elijan otra imagen y ¡volvamos a empezar!



Transferencia del conocimiento (15 min)

Escribir en la bitácora y charla rápida

El acto de escribir en su bitácora sobre lo aprendido, respecto de si les pareció útil y de lo que sintieron, ayuda a sus estudiantes a fortalecer cualquier conocimiento que hayan obtenido hoy y servir como un resumen al que puedan recurrir en el futuro.

Sugerencias para la bitácora:

- ¿Sobre qué se trataba la Lección de hoy?
- ¿Cómo te sentiste durante la Lección?
- Dibuja otra imagen que podrías codificar. ¿Puede escribir el programa que corresponde con la imagen?
- ¿Qué otro tipo de robots podríamos programar si cambiáramos lo que significan las flechas?

Evaluación (10 min)

Entregue la Evaluación – Programación con papel cuadriculado. Luego de haber explicado claramente las instrucciones, permita que los estudiantes realicen la evaluación de forma individual.

Gracias a la Lección previa, esto no debería significar ningún problema para ellos.

Experiencias de aprendizaje de profundización

Use estos Contenidos para ampliar el aprendizaje de los estudiantes. Se pueden usar como Contenidos extras fuera del aula.

Cada vez mejor

- Que su clase intente hacer sus propias imágenes.
- ¿Pueden descifrar cómo codificar las imágenes que acaban de crear?

Desafío de curso

- Dibuje una imagen en una cuadrícula de 5x5 en lugar de una 4x4.

Lección 5: Introducción a los desafíos en línea

Lección en línea Ver en Code Studio

Propósito

En este conjunto de desafíos, los estudiantes comenzarán con una introducción (o repaso, dependiendo de la experiencia de su clase) del espacio de trabajo en línea de Code.org. Habrá videos indicando las funciones básicas del espacio de trabajo, como los botones ejecutar (run), reiniciar (reset) y paso (step). En estos videos también se aborda cómo arrastrar, borrar y conectar bloques Blockly. Luego, sus estudiantes pondrán en práctica sus habilidades de secuenciación y depuración en un laberinto. A partir de ahí, los estudiantes verán nuevos tipos de desafíos, como la recolectora, el artista o la cosechadora, mientras aprenden lo básico de los bucles.

Comprendemos que en cada sala de clases hay un espectro de comprensión diferente para cada tema. Algunos estudiantes pueden ser muy hábiles con los computadores mientras que otros pueden tener muy poca experiencia con ellos. Con el objetivo de nivelar el área de juego (y de aprendizaje), hemos desarrollado esta "etapa de reforzamiento" para el curso D. Esto puede ser tanto una introducción como un repaso de cómo usar Code.org y conceptos básicos de las Ciencias de la Computación.

Secuencia para el aprendizaje

- Conocimiento inicial (10 min)
- Lección puente: programación (10 min)
- Ampliación del conocimiento (30 min)
- Transferencia del conocimiento (10 min)

Objetivos

Los estudiantes serán capaces de:

- Ordenar comandos de movimientos como pasos secuenciales de un programa.
- Modificar un programa existente para reparar errores.
- Descomponer una secuencia larga de instrucciones en una secuencia de repetición más corta.

Preparación

- Realice los desafíos para encontrar cualquier área potencialmente problemática para su clase.
- Asegúrese de que cada estudiante tenga su Bitácora.

Recursos

¡Atención!
Por favor, haga una copia de cada documento que planee compartir con los estudiantes.

Para los estudiantes:

- Video – Pair programming
- Puntos clave – Programación en parejas

Vocabulario

- **Bug (error)**: parte de un programa que no funciona correctamente.
- **Depurar (debugging)**: encontrar y solucionar los problemas en un algoritmo o programa.
- **Bucle (loop):** la acción de hacer algo una y otra vez.
- **Programa**: algoritmo que ha sido codificado de forma que pueda ser ejecutado por una máquina.
- **Programación**: el arte de crear un programa.

Estrategia de aprendizaje

Conocimiento inicial (10 min)

Introducción

Los estudiantes aprenderán muchos conceptos nuevos o repasarán conceptos básicos. En base a la experiencia de su clase, puede revisar las palabras programa, programación, bug, depurar, ciclo o bucle en el glosario del curso o pasar directamente a la Lección puente. Si las definiciones no están cubiertas explícitamente, recomendamos usar las palabras en oraciones.

Lección puente: programación (10 min)

Esta Lección ayudará a sus estudiantes a traer los conceptos sin conexión de "programación con papel cuadriculado" al mundo digital al cual se están sumergiendo.

Revisión de desafíos en línea en conjunto

Elija un desafío de la Lección. Para esta Lección recomendamos progresar con estos desafíos. Divida a los estudiantes en grupos de tres o cuatro. Pídales que programen a "Red", de Angry Birds, para que llegue al cerdo, usando las flechas de "programación con papel cuadriculado".

La clase no necesitará usar el último símbolo.

Una vez que toda la clase tenga una respuesta, discutan el camino a seguir en conjunto como clase.

Ampliación del conocimiento (30 min)

Desafíos en línea

Los profesores son de vital importancia en la educación de las ciencias de la computación y juegan un rol fundamental para producir un ambiente vibrante y colaborativo en la sala de clases. Durante las lecciones en línea, el rol del profesor es alentar y apoyar. Los desafíos en línea están estructurados para estar centrados en el estudiante, así que los profesores deben evitar involucrarse cuando los alumnos tengan problemas para resolverlos. Algunas ideas de cómo hacerlo son:

- Use la programación en parejas cada vez que sea posible durante la Lección.
- A través de desafíos o preguntas, anime a los estudiantes para que busquen respuestas con sus respectivas parejas.
- Las preguntas sin responder pueden ser delegadas a un grupo cercano, que podría ya tener la respuesta.
- Recuérdeles usar el proceso de depuración antes de que usted se acerque a ayudar.

Enséñeles a los estudiantes la manera correcta de trabajar en parejas:

- No usar el teclado del compañero.
- No tocar el mouse del compañero.
- Asegúrese que tu compañero pueda describirte la solución en voz alta antes de que vayas.

- Pida a los estudiantes que describan el problema que estén viendo. ¿Qué se supone que debe hacer?, ¿qué hace?, ¿qué te dice eso?
- Recuérdeles que la frustración es un paso en el camino del aprendizaje y que la perseverancia dará sus frutos.
- Si un estudiante sigue con problemas para avanzar después de todo esto, haga preguntas clave para que los estudiantes identifiquen el bug (error) por ellos mismos.

Lección en Code Studio (link)

Acceden a la plataforma para realizar los desafíos en línea.

Transferencia del conocimiento (10 min)

Escribir en la bitácora

El acto de escribir en su bitácora sobre lo aprendido, respecto de si les pareció útil y de lo que sintieron, ayuda a sus estudiantes a fortalecer cualquier conocimiento que hayan obtenido hoy y servir como un resumen al que puedan recurrir en el futuro.

Sugerencias para la bitácora:

- ¿Sobre qué se trataba la Lección de hoy?
- ¿Cómo te sentiste durante la Lección?
- Haga una lista de los bugs que encontraste en tus programas hoy.
- ¿Cuál fue tu desafío favorito? Dibuja a tu personaje favorito completando desafíos.

Sugerencias de Evaluación

Se sugiere evaluar formativamente los aprendizajes:

- Resuelven problemas,
- Asignan roles en los grupos
- Dividen un problema en otros menores

Lección 6: Programación de relevos

Lección sin conexión

Propósito

Esta Lección comenzará con una breve Lección de depuración y perseverancia. Luego, rápidamente se transformará en una carrera contra el tiempo, mientras los estudiantes se dividen en grupos y trabajan en equipo para escribir un programa, una instrucción a la vez.

El trabajo en equipo es muy importante en la computación. Los equipos escriben y depuran códigos en conjunto, en lugar de trabajar de forma individual. Los estudiantes aprenderán a trabajar en equipo intentando ser lo más eficientes posible.

Esta Lección también involucra el concepto de urgencia, lo cual enseñará a los estudiantes que deben administrar su tiempo cuidadosamente, evitando cometer errores sin comprometer mucho tiempo. Esta experiencia puede ser estresante (¡y lo será!). Asegúrese de compartir con los estudiantes las herramientas para lidiar con la frustración.

Secuencia para el aprendizaje

- Conocimiento inicial (15 min)
- Ampliación del conocimiento (20 min)
- Transferencia del conocimiento (15 min)

Objetivos

Los estudiantes serán capaces de:

- Definir ideas usando código y símbolos.
- Verificar el trabajo hecho por compañeros.
- Identificar las señales de la frustración.

Preparación

- Vea el Video – Programación de relevos.
- Encuentre un espacio abierto para esta Lección, como el gimnasio o un área verde.
- Imprima un Paquete de lecciones – Programación de relevos para cada grupo.
- Provea a cada grupo hojas y lápices.
- Imprima una Guía de trabajo – Programación de relevos para cada estudiante.
- Asegúrese de que cada estudiante tenga su Bitácora.

Recursos

¡Atención!
Por favor, haga una copia de cada documento que planee compartir con los estudiantes.

Para los profesores:

- Guía del profesor – Programación de relevos
- Guía de depuración para el profesor – Programación de relevos
- Video – Programación de relevos

Para los estudiantes:

- Guía de trabajo – Programación de relevos
- Video – Programación de relevos
- Paquete de lecciones – Programación de relevos

Vocabulario

- **Algoritmo:** lista de pasos para realizar una tarea.
- **Bug (error)**: parte de un programa que no funciona correctamente.
- **Depurar (debugging)**: encontrar y solucionar los problemas en un algoritmo o programa.
- **Frustración**: sentirse molesto o enojado porque algo no funciona como quieres.
- **Perseverancia**: intentarlo una y otra vez, incluso cuando algo es muy difícil.
- **Programa**: algoritmo que ha sido codificado de forma que pueda ser ejecutado por una máquina.

Estrategia de aprendizaje

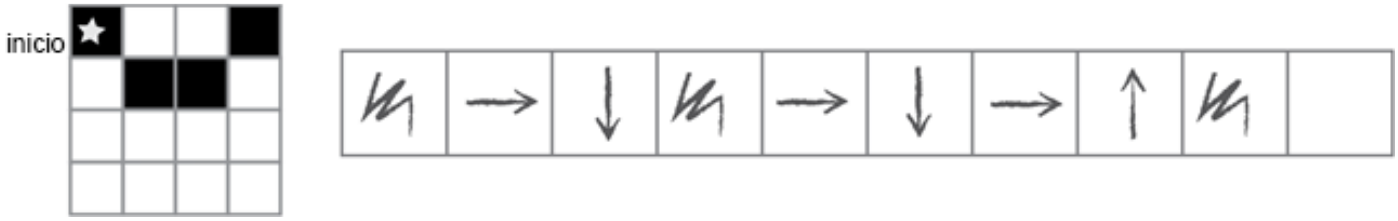
Conocimiento inicial (15 min)

Recordemos que en “programación con papel cuadriculado” guiamos el Sistema de Ejecución Automática (SEA) de nuestros compañeros usando flechas. Esta sección trae de vuelta esas ideas, las cuales serán necesarias en la Ampliación del conocimiento.

¿Dónde me equivoqué?

Objetivo: en esta Lección, queremos ayudar a los estudiantes a identificar y resolver bugs en sus programas. La forma más sencilla de hacerlo es trabajar en un programa que ya contenga bugs, que ya están realizados por alguien que no son sus estudiantes. Una vez que le hayan ayudado a reparar “su” programa, comparta con ellos lo frustrante que puede llegar a ser cometer errores, y ayúdeles a ver que esos sentimientos son absolutamente normales, no deben sentirse avergonzados por ellos.

Mostrar: Muestre la imagen de la Guía de depuración para el profesor – Programación de relevos.



Análisis: Presente la imagen e indique que tiene un bug ¿pueden identificarlo?

Tómese un momento para repasar las reglas: comenzar en la estrella, seguir las instrucciones paso a paso, terminar cuando se hayan ejecutado todos los cuadrados de la derecha.

Reflexión: ¿pueden descifrar por qué el programa no funciona?

Interacción: Pida a los estudiantes que trabajen en equipo para ver si pueden descifrar cómo debería estar escrito el programa.

Exposición: pregunte si alguien encontró una manera de resolver el problema. Cuando le den una respuesta correcta, recuerde la definición de depuración

Discusión: En la programación muchas veces nos vamos a ver enfrentados a la frustración, donde quizás te veas tentado a rendirte; sin embargo, la frustración es un sentimiento normal y es un gran indicio de que estás a punto de aprender algo. En lugar de rendirte, intenta ser perseverante. Sigue intentándolo una y otra vez. Después de algunos intentos, ¡empezarás a entender cómo depurar tus problemas!

Sugerencias para lidiar con la frustración:

- Contar hasta 10
- Respirar hondo
- Escribir acerca del problema
- Hablar con algún amigo sobre el problema
- Pedir ayuda

Sugerencias para ser perseverante:

- Llevar registro de lo que ya has intentado
- Describir qué es lo que pasa
- Describir qué debería hacer
- ¿Qué te dice eso?
- Haga un cambio y vuelve a intentar

Distribuir: para asegurarse de que los estudiantes entienden la idea de encontrar y solucionar problemas (depurar), distribuya la Guía de trabajo – programación de relevos y que trabajen en parejas.

Opcional: Si no quiere invertir mucho tiempo en esta Lección, pueden realizarla en conjunto como clase.

Transición: ¡llegó la hora de la acción!

Ampliación del conocimiento (20 min)

Programación de relevos

Con los conceptos de "programación con papel cuadriculado" en mente, es momento de dividirse en equipos y prepararse para la Lección en relevos.

Preparación: prepárese imprimiendo el Paquete de lecciones – Programación de relevos para cada equipo, de 4 a 5 estudiantes. Corte o doble cada página por la línea punteada.

Revise las reglas del juego con su clase:

- Dividirse en equipos de 4-5 estudiantes.
- Que cada grupo haga una fila, uno detrás de otro.
- Colocar una de las imágenes guía al otro lado de la sala/gimnasio/área para cada equipo (use la misma imagen para todos los equipos).
- El primer estudiante de la fila debe correr hasta la imagen, revisarla, y escribir el primer símbolo del programa para reproducir esa imagen.
- El primer estudiante debe volver y tocar al siguiente de la fila. Luego, debe ponerse al final de la fila.
- La siguiente persona en la fila debe correr hasta la imagen y revisarla. Luego, debe revisar el programa que se ha escrito y realizar sólo una de estas dos opciones: depurar el programa tachando los símbolos incorrectos o agregar un símbolo nuevo. El estudiante corre de vuelta y toca al siguiente de la fila para que sea su turno.
- Repetir el punto anterior hasta que un equipo termine el programa.

Algunas cosas que deben ser aclaradas y recordadas de vez en cuando:

- Sólo una persona de cada grupo puede estar frente a la imagen a la vez.
- Está permitido discutir el algoritmo con el resto del grupo mientras están en la fila, incluso pueden planificar quién y qué van a escribir en el programa.
- Cuando un estudiante depura el programa tachando una instrucción incorrecta (o un conjunto de instrucciones incorrectas), cuenta como su turno. El siguiente estudiante debe descifrar cómo corregir las instrucciones eliminadas.

¡El primer grupo que consiga un programa que calce con la imagen es el ganador! Repita esta Lección varias veces, aumentando la dificultad.

Repita el juego tantas veces como pueda, hasta que se agote el tiempo o hasta que los estudiantes se sientan fatigados.

Transición: una vez que el juego acabe, reúna a todos los estudiantes en un círculo para compartir lo aprendido.

Discusión: ¿qué aprendimos hoy?

- ¿Qué tal si cada persona en la fila pudiese hacer 5 flechas a la vez?
- ¿Qué tan importante puede ser depurar nuestro propio trabajo y el trabajo del programador anterior a nosotros?
- ¿Y si pudiesen hacerse 10 flechas?
- ¿Y si fuesen 10.000? ¿Sería más o menos importante?
- ¿Crees que un programa es mejor o peor cuando más de una persona trabaja en él?
- ¿Crees que las personas comenten más o menos errores cuando están apurados?
- Si encuentras un error ¿debes deshacer todo el programa y comenzar desde cero?

Transferencia del conocimiento (15 min)

Escribir en la bitácora

El acto de escribir en su bitácora sobre lo aprendido, respecto de si les pareció útil y de lo que sintieron, ayuda a sus estudiantes a fortalecer cualquier conocimiento que hayan obtenido hoy y servir como un resumen al que puedan recurrir en el futuro.

Sugerencias para la bitácora:

- ¿Sobre qué se trataba la Lección de hoy?
- ¿Cómo te sentiste durante la Lección?
- ¿Cómo es que el trabajo en equipo influyó en el éxito de escribir el programa de hoy?
- ¿Te sentiste frustrado en algún punto? ¿qué hiciste respecto a eso?

Experiencias de aprendizaje de profundización

Use estos contenidos para ampliar el aprendizaje de los estudiantes. Se pueden usar como Contenidos extras fuera del aula.

Pasa la hoja

Si no tiene tiempo o espacio para una Lección de relevos al aire libre, puede reunir a los estudiantes en grupos y que pasen la hoja con el programa de un puesto a otro. Que cada estudiante haga una flecha antes de pasar la hoja a un compañero.

Llénalo, muévelo

- Cómo profesor, dibuje una imagen con tantos cuadrados pintados como niños haya en cada grupo.
- Pida a los estudiantes que anoten tantas flechas en el programa como sea necesario para llegar a un cuadrado pintado, incluyendo la acción de pintar ese cuadrado, antes de pasar la hoja al siguiente estudiante.

Depurando juntos

Dibuje una imagen en la pizarra. Pídale a cada estudiante crear un programa para esa imagen. Luego, que intercambien sus programas con sus compañeros de al lado y depuren sus códigos.

- Encierra en un círculo el primer paso incorrecto, luego devuelve el programa.
- Deles a los estudiantes otra oportunidad para revisar y depurar sus propios programas.
- Pida que un voluntario comparta su programa.

Pregunte:

- ¿Cuántos tienen el mismo programa?
- ¿Alguien tiene algo diferente?

Sugerencias de evaluación

Se sugiere el siguiente indicador para evaluar formativamente los aprendizajes:

- Identifican las partes que componen una situación.
- Elaboran instrucciones.

Lección 7: Depuración con Laurel

Lección en línea Ver en Code Studio

Propósito

En esta Lección en línea, los estudiantes practicarán cómo depurar en un ambiente de "recolección", al practicar cómo leer y editar códigos para reparar desafíos con algoritmos simples, bucles y bucles anidados.

El propósito de esta Lección es enseñar a los estudiantes que fallar es normal cuando se estamos aprendiendo nuevas habilidades. A los estudiantes se les darán programas que NO funcionan correctamente y se les pedirá repararlos. Este proceso, llamado "depuración", les ayudará a desarrollar habilidades de pensamiento crítico y resolución de problemas, habilidades que los acompañarán mientras avanzan a proyectos de programación cada vez más complejos.

Secuencia para el aprendizaje

- Conocimiento inicial (15 min)
- Lección puente: depuración (15 min)
- Ampliación del conocimiento (30 min)
- Transferencia del conocimiento (15 min)

Objetivos

Los estudiantes serán capaces de:

- Leer y comprender un código dado.
- Identificar un bug y los problemas que causa en un programa.
- Describir e implementar un plan para depurar un programa.

Preparación

- Realice los desafíos para encontrar cualquier área potencialmente problemática para su clase.
- Asegúrese de que cada estudiante tenga su Bitácora.

Recursos

¡Atención!
Por favor, haga una copia de cada documento que planee compartir con los estudiantes.

Para los estudiantes:

- Act. Previa. Paquete de lecciones – Programación de relevos
- Manipulativos – Bloques sin conexión (cursos C - F)

Vocabulario

- **Bug (error)**: parte de un programa que no funciona correctamente.
- **Depurar (debugging)**: encontrar y solucionar los problemas en un algoritmo o programa.

Estrategia de aprendizaje

Conocimiento inicial (15 min)

Introducción

Una de las partes más importantes en el aprendizaje de la programación es aprender a depurar. Pregunte a la clase si alguna vez han aprendido una habilidad nueva y si alguna vez han fallado.

Por ejemplo:

- Aprender a andar en bicicleta y caerse.
- Aprender a hornear y quemar la comida.
- Aprender algún deporte y no ganar ningún juego.

Equivocarse es muy común cuando se aprenden cosas nuevas. Pida a los estudiantes que discutan sobre situaciones en que se hayan equivocado y cómo las superaron.

En la programación, los programadores normalmente encuentran “bugs” en sus códigos.

- Bug (error): parte de un programa que no funciona correctamente.
- Un bug puede realmente arruinar un programa, por lo que es importante aprender a “depurar” el código.
- Depurar (debugging): encontrar y solucionar los problemas en un algoritmo o programa.
- Entretenga la conversación si cree que su clase necesita una mayor introducción, pero no olvide dejar tiempo para una de las lecciones puente.

Lección puente: depuración (15 min)

Estas lecciones ayudarán a sus estudiantes a traer los conceptos sin conexión de “depuración sin conexión: programación de relevos” al mundo digital al cual se están sumergiendo. Elija una de las siguientes lecciones:

Lección sin conexión: bloques de papel

Divida a la clase en equipos de 3 a 5 estudiantes y encuentre un lugar amplio, puede ser un gimnasio o un área verde. Ordene los equipos en fila, similar a como se hizo en la Lección “programación de relevos”. Tome un diseño mediadamente difícil del Paquete de lecciones – Programación de relevos. A una distancia considerable, coloque una imagen por cada equipo. Además, entrégueles suficientes bloques de papel de los Manipulativos – Bloques sin conexión (cursos C - F). Cada equipo necesitará muchos bloques llena 1 y mover____. Los bloques mover____ pueden ser rellenados con anticipación o durante el mismo juego. De cualquier modo, asegúrese de que los bloques estén bien definidos mientras se desarrolle el juego.

Una vez que todos los equipos estén formados, muestre o lea en voz alta las reglas:

- El primer estudiante en la fila debe correr hacia la imagen, revisarla, y colocar el primer bloque de código en el programa para reproducir esa imagen.
- El primer estudiante debe volver y tocar al siguiente de la fila. Luego, debe ponerse al final de la fila.
- La siguiente persona en la fila debe correr hasta la imagen y revisarla. Luego, debe revisar el programa que se ha escrito y hacer una de dos: depurar el programa quitando los bloques de código incorrectos o agregar un bloque nuevo. Luego, el estudiante debe correr de vuelta y toca al siguiente en la fila para que sea su turno.

- Repetir el punto anterior hasta que un equipo termine el programa.

Asegúrese de que los estudiantes usen sólo los bloques llena 1 y mover____ y que sólo coloquen un bloque por turno. ¡El primer equipo que escriba correctamente el código de su imagen, gana!

Revisión de desafíos en línea en conjunto

Agrupe a los estudiantes en grupos de 3. Elija un desafío de la Lección, recomendamos el número 5. Pídales que se sienten frente a un computador con el desafío en pantalla. Cada equipo puede tener sólo un computador y un estudiante mirando la pantalla. Muestre o lea las reglas:

- Sólo un estudiante por equipo puede estar mirando la pantalla.
- Esa persona sólo puede añadir o borrar un bloque a la vez. Cuando esa persona haya añadido o eliminado un bloque, puede tocar el hombro de un compañero para que sea su turno.
- El siguiente estudiante puede ponerse frente a la pantalla y jugar su turno.
- Los turnos no pueden repetirse ni saltarse, todos deben jugar la misma cantidad de veces.

¡El primer equipo en resolver el desafío correctamente, gana!

Ampliación del conocimiento (30 min)

Desafíos en línea

Podría ser muy útil que los integrantes de los equipos de la Lección puente se sienten cerca unos de otros. Cada estudiante debe realizar los desafíos de forma individual o en parejas; sin embargo, contar con un grupo de trabajo conocido para hacer y responder preguntas puede ayudarlos a sentir más confianza y a comprender mejor el tema.

Lección en Code Studio

Acceder a Code Studio para realizar los desafíos de depuración.

Transferencia del conocimiento (15 min)

Escribir en la bitácora

El acto de escribir en su bitácora sobre lo aprendido, respecto de si les pareció útil y de lo que sintieron, ayuda a sus estudiantes a fortalecer cualquier conocimiento que hayan obtenido hoy y servir como un resumen al que puedan recurrir en el futuro.

Sugerencias para la bitácora:

- ¿Sobre qué se trataba la Lección de hoy?
- ¿Cómo te sentiste durante la Lección?
- ¿Qué es un bug? ¿cómo sabes que hay un bug en tu programa?
- ¿Qué significa “depurar” un código? ¿cómo depuras un programa?

Sugerencia para evaluación

Se sugiere el siguiente indicador para evaluar formativamente los aprendizajes:

- Comprenden y analizan un problema.
- Piensan en la solución.

- Buscan alternativas; es decir, preparan los elementos para que desarrollen el programa y resuelvan el problema.

Lección 8: Diseñador de Minecraft

Lección en línea Ver en Code Studio

Propósito

La hora del código con Minecraft ayuda a que todos los estudiantes puedan aprender las valiosas habilidades adquiridas a través de la enseñanza de las Ciencias de la Computación. Los estudiantes participarán una hora donde experimentarán la programación en un ambiente divertido y libre de estrés, aprenderán conceptos clave de la computadora y los videojuegos.

Los estudiantes aprenderán ciencias de la computación a través del taller: La hora del código con Minecraft, dónde diseñarán un juego Minecraft teniendo un acercamiento a conceptos que utilizarán a lo largo del curso.

Secuencia para el aprendizaje

- Ampliación del conocimiento (55 min)
- Transferencia del conocimiento (5 min)

Objetivo

Los estudiantes serán capaces de:

- Entender conceptos básicos que conforman el diseño de un videojuego, de una manera divertida: estructura de código por bloques, comandos, ciclos, comandos aleatorios, eventos, agregar elementos y puntuación.

Preparación

- Computadoras con conectividad
- Realice las lecciones para encontrar cualquier área potencialmente problemática para su clase.

Recursos

¡Atención!
Por favor, haga una copia de cada documento que planee compartir con los estudiantes.

Para los profesores:

- Hora del código Minecraft

Estrategia de aprendizaje

Ampliación del conocimiento (55 min)

Se guiará a los estudiantes al curso, entrando al siguiente link - hora del código Minecraft - e identificando el curso mostrado en las imágenes a

continuación:

Posterior iniciarán el curso dando clic en:

Ahora como clase, podrán explorar el taller.

Diseñador de Minecraft

Grupos: forma equipos de 2-3 personas, según la capacidad del aula.

Explica: se explicará a los estudiantes sobre los elementos que conforman un videojuego, pensarán en sus videojuegos favoritos y se les preguntará ¿qué contienen? ¿qué observan? ¿cómo creen que funcionan?

Como clase observarán el video de introducción:

Los equipos deberán seguir avanzando en las diversas etapas y lecciones del taller, además podrán seleccionar el personaje que más les guste para la Lección.

Conocerán bloques de programación que deberán aprender a utilizar para resolver diversos retos.

Como clase observarán el video que permitirá a los equipos aprender a utilizar ciclos de programación, que harán sus programas más eficientes, lo aprendido lo aplicarán en los ejercicios de programación.

Como clase observarán el video, después los equipos harán uso de eventos que les permitirán incluir eventos dependientes de una acción en su videojuego. Lo aprendido lo aplicarán en los ejercicios de programación.

Como clase observarán el video para generar criaturas, lo aprendido lo aplicarán en los ejercicios de programación.

Como clase observarán el video de felicitaciones, en este momento habrán concluido su primer acercamiento al desarrollo de videojuegos.

Transferencia del conocimiento (5 min)

Reflexión

Pregunta a los estudiantes: ¿qué fue lo que más les gustó? y hagan un listado en sus cuadernos los elementos que utilizaron a lo largo del taller.

Los conceptos aprendidos los utilizarán a lo largo del curso, a mayor profundidad.

Si cuentas con más tiempo o tu clase está muy interesada en este tema pueden realizar más lecciones de la hora del código con Minecraft.

Sugerencias para evaluación

Se sugiere el siguiente indicador para evaluar formativamente los aprendizajes:

- Desarrollan un programa con elementos o fases más pequeñas.
- En una conversación pueden dar un ejemplo de aplicación de elementos como: secuencias, eventos, bucles y condicionales

Lección 9: Trazar formas

Lección en línea

Propósito

Los estudiantes exploran los desafíos de comunicar cómo dibujar con formas y usar una herramienta que presente cómo se aborda este problema en Game Lab. La Lección de preparación demuestra rápidamente los desafíos de la posición de comunicación sin punto de referencia compartido. En la Ampliación del conocimiento, los estudiantes exploran una herramienta de Game Lab que les permite colocar formas interactivamente en la cuadrícula de 4 por 4 de Game Lab. Luego se turnan para dar instrucciones a un compañero sobre cómo dibujar una imagen oculta con esta herramienta, lo que representa muchos desafíos que enfrentarán al programar en Game Lab. Los estudiantes opcionalmente crean su propia imagen para comunicarse antes de una discusión de conclusión.

El objetivo principal de esta Lección es presentar a los estudiantes el sistema de coordenadas que usarán en Game Lab. Los estudiantes pueden tener experiencia limitada con el uso de una cuadrícula de coordenadas o pueden tener dificultades con el eje Y "invertido" en Game Lab. La herramienta de dibujo también obliga a los estudiantes a pensar en otras características de Game Lab que verán cuando comiencen a programar en la próxima Lección. Estos incluyen la necesidad de tener en cuenta el orden al dibujar, la necesidad de especificar el color y el hecho de que los círculos se colocan por su centro y cuadrados por la esquina superior izquierda. Al final de esta Lección, los estudiantes deben estar listos para transferir lo que han aprendido sobre la comunicación de posición a la programación que harán en la próxima Lección.

Secuencia para el aprendizaje

- Conocimiento inicial (10 min)
- Ampliación del conocimiento (35 min)
- Transferencia del conocimiento (5 min)

Objetivos

Los estudiantes serán capaces de:

- Manejar las ubicaciones en la cuadrícula de coordenadas de Game Lab.
- Comunicar cómo dibujar una imagen en Game Lab, teniendo en cuenta la posición de la forma, el color y el orden.

Recursos

¡Atención!
Por favor, haga una copia de cada documento que planee compartir con los estudiantes.

Para los estudiantes:

- Trazar formas- Guía de lecciones
- Trazar formas - Enlace code.org

Estrategia de aprendizaje

Conocimiento inicial (10 min)

Comunicar información de dibujo

Vimos muchos programas diferentes y comenzaron a pensar qué querrían crear. Cuando creamos un programa, una de las cosas que debemos hacer es dibujar todo en la pantalla. Vamos a probar eso con una computadora "estudiantil".

Pida a uno o dos voluntarios que pasen al frente del salón y actúen como "computadoras" para la Lección. Deben sentarse de espaldas para que no puedan ver lo que se proyecta. Entregue a cada voluntario una hoja de papel en blanco.

Visualización: proyecte un dibujo de muestra que pueda ser visto por la clase.

Observaciones: Deberá explicarle a nuestra "computadora" cómo dibujar la imagen. Al final, compararemos el dibujo con la imagen real. De un minuto o dos para describir el dibujo, ya que los estudiantes al frente del salón tratan de dibujarlo. Después de un minuto, deténgase y permita que los estudiantes comparen ambas imágenes.

Preguntar: ¿cuáles son los diferentes "desafíos" o problemas que vamos a necesitar resolver para comunicar con éxito este tipo de dibujos?

Discuta:

- Los estudiantes deben escribir silenciosamente sus respuestas en sus cuadernos.
- Luego deben hablar con un compañero y finalmente con toda la clase.

Meta: Esta discusión tiene la intenciónde mencionar algunos desafíos que los estudiantes necesitarán abordar en las próximas lecciones, tales como especificar la posición, el orden y el color. Los estudiantes no necesitan decidir cómo especificarán tales cosas, pero reconocen que necesitarán un método para hacerlo.

Observaciones: hubo varios desafíos que necesitábamos resolver en esta Lección. Necesitamos poder comunicar claramente la posición, el color y el orden de las formas. Vamos a comenzar a explorar cómo resolver este problema.

Ampliación del conocimiento (35 min)

Meta: En lugar de hacer una demostración en vivo de la herramienta, use esta estrategia para que los estudiantes exploren la herramienta ellos mismos. Luego use la información para asegurarse de que todos los estudiantes conozcan las características clave de la herramienta. Los componentes más importantes son la cuadrícula y el hecho de que las coordenadas del mouse se muestran debajo del espacio de dibujo.

Dibujando con una computadora

Grupo: Agrupe a los estudiantes en parejas.

Transición: haga que un miembro de cada grupo tome una computadora y vaya a los contenidos de esta Lección. Hay un solo nivel con una herramienta de Game Lab.

Indicación: trabajar con tu pareja toma dos o tres minutos para descubrir cómo funciona esta herramienta. Luego, prepárese para compartir como clase.

Discuta: después de que las parejas hayan tenido la oportunidad de trabajar con la herramienta, promueva una discusión donde los estudiantes hablen sobre las características que anotan.

Dibujando formas

Distribuya Trazar formas- Guía de lecciones a cada par, asegurándose de que un alumno (Estudiante A) reciba las dos primeras páginas y que el otro alumno (Estudiante B) reciba las dos páginas siguientes. Los estudiantes no deben mirar los papeles de los demás.

Ubicación del origen: El origen de esta cuadrícula, así como el origen en Game Lab, se encuentra en la esquina superior izquierda. Esto refleja el hecho de que los documentos tienden a comenzar en la parte superior izquierda y aseguran que cada punto del avión tenga coordenadas positivas.

Configuración:

En esta Lección, los estudiantes intentarán recrear las imágenes en función de las indicaciones de un compañero. El estudiante que está ilustrando usará la herramienta de dibujo de formas en Game Lab para realizar las formas. Los estudiantes deben mantener sus dibujos ocultos el uno del otro durante toda la Lección. Al completar un dibujo, el instructor tampoco debería poder ver la pantalla de la computadora.

Dibujo 1:

Cada miembro del par debe completar su primer sorteo, turnarse para dar instrucciones y usar la herramienta. Estos dibujos no presentan formas superpuestas, pero los estudiantes pueden tener que lidiar con el hecho de que los círculos se dibujan desde la mitad y los cuadrados desde la esquina superior izquierda. Además, los estudiantes pueden simplemente luchar con la dirección del eje Y.

Cuando continuar: determine si esta última Lección vale la pena en su planificación. Darles a los estudiantes la oportunidad de crear y comunicar su propio dibujo puede ayudar a reforzar su conocimiento, pero si los estudiantes obviamente están logrando los objetivos de aprendizaje de la Lección sin él, también puede pasar al resumen para sintetizar su aprendizaje.

Discuta: da un par de minutos a las parejas para discutir cualquier problema común que estén notando al tratar de completar sus dibujos.

Dibujo 2:

Cada miembro del par debe describir su segundo sorteo a su compañero. Estos dibujos tienen formas superpuestas y los estudiantes tendrán que considerar el orden en que se colocan las formas y cuándo deben cambiar el color del lápiz.

Dibuja el tuyo:

Si el tiempo lo permite, brinde a los estudiantes la oportunidad de crear su propio dibujo para comunicarse con su compañero.

Cuando hacemos imágenes, necesitamos una forma de comunicar exactamente dónde va cada forma. El plano de coordenadas nos ayuda a hacer eso. Nuestro plano de coordenadas tiene dos coordenadas, “x” e “y”. La coordenada “x” nos dice qué tan lejos está nuestra forma de la izquierda de la cuadrícula. La coordenada “y” nos dice cuán lejos está nuestra forma desde la parte superior de la cuadrícula. Los puntos negros en las formas lo ayudan a ser muy específico sobre cómo se coloca la forma en la cuadrícula.

Transferencia del conocimiento (5 min)

Reflexión

Haga que los estudiantes reflexionen sobre cada uno de los siguientes mensajes

- ¿Qué problema está ayudando a resolver la cuadrícula en Game Lab?
- ¿Has visto formas diferentes de resolver este problema en el pasado? ¿Qué son?

Discuta: Haga que las parejas compartan sus respuestas entre sí. Luego abre la discusión a toda la clase.

Al comienzo de la clase vimos que comunicar cómo dibujar formas simples puede ser bastante desafiante. Lo que aprendimos hoy es una solución a este problema, pero hay muchas otras que podrían haber funcionado. De hecho, muchos de ustedes probablemente se dieron cuenta de que la cuadrícula en Game Lab está “volteada”. Las pantallas de las computadoras tienen diferentes formas y tamaños, al igual que el contenido que mostramos en ellas. Tenemos que acordar un punto desde el cual pueda construir el contenido. Como leemos desde la esquina superior izquierda, la cuadrícula en la pantalla de una computadora también comienza en la esquina superior izquierda. También existe el beneficio de no tener que usar números negativos para hablar de ubicaciones en la pantalla. No se preocupe si esta cuadrícula volteada es un poco complicada. Tendremos mucho más tiempo para trabajar en las próximas lecciones.

Lección 10: Dibujar en Game Lab

Lección en línea Ver en Code Studio

Propósito

Se les presenta a los estudiantes Game Lab, el entorno de programación y comienzan a usarlo para colocar las formas en la pantalla. Aprenden los conceptos básicos de secuenciación y depuración, así como algunos comandos simples. Al final de la Lección, los estudiantes podrán programar imágenes como las que hicieron con la herramienta de dibujo en la Lección anterior.

El objetivo principal de esta Lección es dar a los estudiantes la oportunidad de acostumbrarse al entorno de programación, así como la secuencia básica y la depuración**.** Los estudiantes comienzan con una introducción al entorno de desarrollo interactivo (IDE) Gamelab, luego aprenden las tres instrucciones (rect, ellipse y fill) que van a necesitar para codificar los mismos tipos de imágenes que han creado en el papel en la Lección anterior. Los niveles de desafío ofrecen una oportunidad para que los estudiantes que tienen más experiencia en programación exploren más en Game Lab.

Secuencia para el aprendizaje

- Conocimiento inicial (5 min)
- Ampliación del conocimiento (30 min)
- Transferencia del conocimiento (10 min)

Objetivos

Los estudiantes serán capaces de:

- Usar el IDE de Game Lab para trazar diferentes formas de colores en la pantalla.
- Aprender la secuencia de código correctamente para superponer formas.
- Conocer el código de depuración escrito por otros.

Preparación

- Prepara el proyector u otros medios para mostrar videos si desea verlos como una clase.
- Dibujar en Game Lab - Code.org.

Vocabulario

- **Error (bug)**: Parte de un programa que no funciona correctamente.
- **Depuración (debugging)**: Encontrar y solucionar problemas en un algoritmo o programa.
- **Programa**: Algoritmo que ha sido codificado en algo que puede ser ejecutado por una máquina.

Código

- fill(color)
- ellipse(x, y, w h)
- rect(x, y, w, h)

Estrategia de aprendizaje

Conocimiento inicial (5 min)

Programación de imágenes

En la última Lección, creamos imágenes en la computadora organizando cuadrados y círculos en una cuadrícula. Para cada imagen que quería crear, tenía que dibujar esas imágenes manualmente, y si quería recrear una imagen era mucho trabajo. Hoy, vamos a programar la computadora para dibujar esas imágenes para nosotros. Según lo que sabe sobre las computadoras, ¿qué cree que será diferente entre decirle a una persona sobre su imagen y contarle a la computadora su imagen?

De tiempo a los estudiantes para que piensen individualmente y hablen con un compañero, luego junten la clase y escriban sus ideas en el pizarrón.

Para dar instrucciones a una computadora, necesitamos usar un lenguaje que la computadora entienda. Utilizamos HTML, que es ideal para hacer páginas web. Para hacer nuestras animaciones y juegos, usaremos una versión de Javascript que usa bloques. El entorno en el que programaremos se llama Game Lab.

Ampliación del conocimiento (30 min)

Dibujo simple en Game Lab

Grupo: Coloque a los estudiantes en parejas para programar juntos.

Los estudiantes que son nuevos en la programación a menudo tienen algunos conceptos erróneos comunes con los que se topan. Con el fin de evitarque sigan recordando a los estudiantessobre las siguientes cosas

- Un comando por línea
- Los comandos se ejecutan en orden de arriba abajo
- El orden de las entradas en loscomandos de forma importa
- Cada comando de entrada en forma está separado por comas
- (0,0) está en la esquina superiorizquierda de la pantalla
- Todos los valores x - y en la pantalla son positivos

Transición:

Envía a los estudiantes a Code Studio - dibujando en Game Lab.

Apoyo:

A medida que los estudiantes trabajan en los niveles, puede ayudarlos, pero aliéntelos a intentar dedicar algo de tiempo a resolver primero los problemas. Si necesita ayuda para apoyar a los estudiantes, consulte los ejemplos en el visor de respuestas del docente. Cuando los estudiantes alcanzan los niveles de desafío, pueden elegir perseguir uno o más de los desafíos, regresar para mejorar los niveles anteriores o ayudar a un compañero de clase.

Tour de Game Lab

Dependiendo de la edad y el nivel de comodidad de sus estudiantes, puede optar por utilizar este nivel para recorrer el entorno como una clase completa. Asegúrese de que los estudiantes puedan encontrar las instrucciones de nivel, el área de codificación, el área de visualización y los cajones bloqueados. Esta es también una buena oportunidad para señalar algunos de los recursos útiles, como la documentación y el botón Bloquear texto.

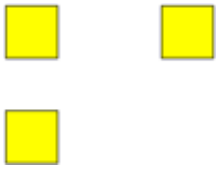
Coloque cuadrados en las esquinas

Lo primero que se debe comprender al usar Game Lab es entender la posición. Recuerde que siempre puede activar la cuadrícula o desplazarse con el mouse para ayudar a encontrar laposición x-y que desea.

Color de relleno:

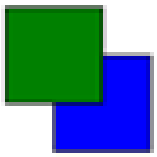
También puede hacer que los rectángulos tengan diferentes colores con fill (rellenar). Establecerá el color para cada forma que viene después en el código.

Haga esto



- Mire el código que establece el color.
- Cambie el color de azul a amarillo.
- Agregue un nuevo cuadrado arrastrando un bloque rect al área de código debajo del comando de relleno. El nuevo cuadrado puede ir a cualquier ubicación en la pantalla que desees.

El orden importa



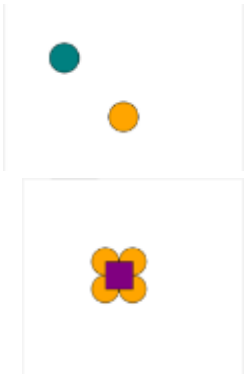
En Game Lab, importa en qué orden se encuentre el código. Se dibujan nuevas formas encima de las anteriores, cubriendo las formas que se dibujan primero. Puede ver la diferencia cuando usa más de un color en el código

Elipse

Puede usar ellipse para hacer un círculo.

Depuración

A menudo, el código no funciona la primera vez que se ejecuta y los programadores tienen que depurarlo. Se supone que el siguiente código hace que la imagen aparezca a la derecha, pero el programador se confundió en qué orden debería estar el código, y cómo colocar un cuadrado en el lugar correcto de la pantalla.



1. rect(175, 175);
2. fill("orange");
3. ellipse(150,150);
4. ellipse(200,150);
5. ellipse(150,200);
6. ellipse(200,200);
7. fill("purple");

Formas simplificadas

Los comandos rect y ellipse que se presentaron a los estudiantes en estas Lecciones son versiones simplificadas de los comandos completos que verán más adelante. Esto permite que la clase se centre únicamente en la ubicación de las formas en el plano de coordenadas antes de preocuparse también por el ancho y la altura de esas formas.

Transferencia del conocimiento (10 min)

Compartir dibujos

Objetivo: Los estudiantes verán la variedad de cosas diferentes que pueden crear con simples dibujos de formas.

Compartir: Una vez que los estudiantes hayan completado sus dibujos, pida que compartan sus dibujos con la clase.

Ticket de salida

Meta: Los estudiantes comparten los trucos que aprendieron a medida que avancen de niveles.

Indicación: Hoy aprendieron a dibujar en Game Lab por primera vez. ¿Qué tipo de consejo compartirán con sus amigos que iban a aprender a dibujar en Game Lab para que sea más fácil para ellos? Escríbanlo.

Recopilar: Tome las respuestas de los estudiantes y seleccione algunas que puedan ser útiles para que los escuchen todos los estudiantes. Comparta estas preguntas al comienzo de la próxima clase.

Este es un buen acertijo para usar como una evaluación de si los estudiantes entienden o no los fundamentos de la secuencia y el posicionamiento de formas en Game Lab.

Lección 11: Formas y aleatorio

Lección en línea Ver en Code Studio

Propósito

En esta Lección, los estudiantes continúan desarrollando su familiaridad con Game Lab al manipular el ancho y la altura de las formas que usan para dibujar. La Lección comienza con una discusión que conecta la funcionalidad de bloque expandido (por ejemplo, formas de diferentes tamaños) con la necesidad de más entradas de bloque, o "parámetros". Los estudiantes aprenden a dibujar con versiones de ellipse() y rect() que incluyen parámetros de ancho y alto. También aprenden a usar el bloque background(). Al final del progreso, se les presenta el bloque randomNumber(). Combinando todas estas habilidades, los estudiantes dibujarán una serpiente arcoíris al azar al final de la Lección.

Esta Lección les da a los estudiantes la oportunidad de expandir ligeramente sus habilidades de dibujo mientras continúan desarrollando habilidades de programación de propósito general. Tendrán que razonar sobre el plano de coordenadas x-y, considerar el orden de su código y aumentar ligeramente la complejidad de sus programas. El bloque randomNumber() es importante para la próxima clase, donde los estudiantes aprenden a almacenar valores utilizando variables. Esta Lección debe enfocarse principalmente en la construcción de habilidades. Si los estudiantes pueden completar la serpiente arco iris de forma independiente, entonces tienen las habilidades que necesitarán para las próximas lecciones.

Secuencia para el aprendizaje

- Conocimiento inicial (5 min)
- Ampliación del conocimiento (40 min)
- Transferencia del conocimiento (5 min)

Objetivos

Los estudiantes serán capaces de:

- Usar y razonar sobre el dibujo de comandos con múltiples parámetros.
- Generar y usar números aleatorios en un programa.

Preparación

- Revisa la secuencia de niveles en Code Studio - formas y aleatorio.

Vocabulario

- **Parámetro:** Dato que se considera como imprescindible y orientativo para lograr evaluar o valorar una determinada situación.

Código

- background(color)
- ellipse(x, y, w, h)
- rect(x, y, w, h)
- randomNumber()

Estrategia de aprendizaje

Conocimiento inicial (5 min)

- ellipse
- randomNumber(5, 10)
- rect

Formas de diferentes tamaños

Indicación:

Nuestros bloques ellipse y rect cada uno tienen dos entradas que controlan dónde están dibujadas: la posición x-y. Si quisiera que estos comandos dibujaran una variedad más amplia de rectángulos y elipses, ¿qué entradas adicionales podría necesitar para proporcionar estos bloques? ¿Qué controlaría cada entrada adicional?

Discuta:

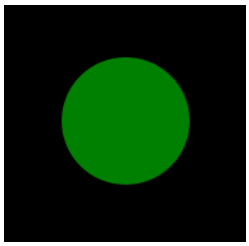
Los estudiantes deben reflexionar sus ideas en silencio, luego compartir con un compañero, después compartir con toda la clase. Registre en el pizarrón las ideas mientras los estudiantes las comparten.

Meta: Esta discusión introduce la palabra de vocabulario "parámetro" y también ayuda a motivar su uso. Los estudiantes verán que las versiones del bloque ellipse() y rect() en esta Lección tienen parámetros adicionales, así como el bloque randomNumber() que tiene dos parámetros. Los estudiantes pueden decir que quieren insumos para el tamaño de las formas, su color, etc. Durante esta conversación, vincule las conductas que los estudiantes desean con las entradas que el bloque necesitaría. Por ejemplo, si desea que los círculos tengan un tamaño diferente, el bloque necesitará una entrada que le permitaal programador decidir qué tan grande debe ser.

Si queremos que nuestros bloques dibujen formas de diferentes maneras necesitarán más insumos que nos permitan decirles cómo dibujar. Las entradas o aperturas en nuestros bloques tienen un nombre formal, parámetros, y hoy vamos a aprender más sobre cómo usarlas.

Ampliación del conocimiento (40 min)

Programación de imágenes



Transición:

Mover a los estudiantes a Code Studio y realiza las lecciones:

- Resumen de la Lección
- Formas y parámetros
- Números al azar

Compartir:

Si algunos estudiantes se toman más tiempo para trabajar en sus proyectos, deles la oportunidad de compartir sus serpientes arcoíris más complejas. Enfoque la conversación sobre qué parámetros están manipulando los estudiantes o aleatorizando para crear sus dibujos.

Fondo:

A veces querrá llenar toda la pantalla con un color. Para eso, puede usar el bloque background. Cubra todo en la pantalla con el color que elijas.

El bloque de fondo se dibujará encima de todo lo que ya está en tu dibujo, por lo que el orden en el código es importante.

Números al azar:

randomNumber() elija un número aleatorio entre un valor mínimo y máximo. Puede usar este bloque en lugar de escribir en el número específico. Si realiza sus dibujos con números aleatorios, se verá un poco diferente cada vez que ejecutes el programa.

Transferencia del conocimiento (5 min)

Reflexión

Preguntar:

Pida a los estudiantes que reflexionen sobre el desarrollo de las cinco prácticas de Descubrimientos CS (Resolución de problemas, Persistencia, Creatividad, Colaboración, Comunicación). Elija una de las siguientes indicaciones según lo considere apropiado.

- Elija una de las cinco prácticas en las que crea que demostró aprendizaje en esta Lección. Escriba algo que hizo que ejemplifica esta práctica.
- Elija una práctica en la que piense que puede seguir creciendo. ¿Qué le gustaría mejorar?
- Elija una práctica que pensó que era especialmente importante para la Lección que completamos hoy. ¿Qué lo hizo tan importante?

Lección 12: Variables

Lección en línea Ver en Code Studio

Propósito

En esta Lección, los estudiantes aprenden a usar variables para etiquetar un número en su programa o guardar un valor generado aleatoriamente. Los estudiantes comienzan la Lección con una descripción muy básica del propósito de una variable. Luego completan un progreso de nivel que refuerza el modelo de una variable como una forma de etiquetar o nombrar un número. Los estudiantes usan variables para guardar un número aleatorio para ver que las variables realmente almacenan o guardan sus valores, lo que les permite usar el mismo número aleatorio varias veces en sus programas.

Esta Lección es la primera vez que los estudiantes verán variables en el curso, y no se espera que entiendan completamente cómo funcionan las variables para su conclusión. Por lo tanto, los estudiantes deben dejar esta Lección sabiendo que las variables son una forma de etiquetar un valor en sus programas para que puedan ser reutilizados o referenciados más adelante. En la siguiente Lección los estudiantes serán presentados a los sprites, que deben ser referenciados por una variable.

Usar variables para manipular dibujos es una habilidad sorprendentemente desafiante que requiere una gran cantidad de previsión y planificación. Si bien los estudiantes usarán o modificarán muchos programas en esta Lección, no se espera que compongan programas que usan variables para modificar las características de un dibujo. En lecciones posteriores, los estudiantes ampliarán su comprensión de las variables y formas más avanzadas en que se pueden usar.

Secuencia para el aprendizaje

- Conocimiento inicial (10 min)
- Ampliación del conocimiento (30 min)
- Transferencia del conocimiento (5 min)

Objetivos

Los estudiantes serán capaces de:

- Identificar una variable como una forma de etiquetar y referenciar un valor en un programa.
- Usar variables en un programa para almacenar una información que se usa varias veces.
- Identificar el motivo y solución de errores comunes encontrados al programar con variables

Preparación

- Revisa el progreso de nivel en Code Studio - Variables.

Recursos

¡Atención!
Por favor, haga una copia de cada documento que planee compartir con los estudiantes.

Para los estudiantes:

- Introducción a las variables - Video **(configure los** subtítulos **en español)**

Vocabulario

- **Variable:** Un marcador de posición para una información que puede cambiar.

Código

- Declara variable

Estrategia de aprendizaje

Conocimiento inicial (10 min)

Etiquetas y valores

Video:

Como clase, miren el video que presenta las variables.

Revisión:

Los estudiantes pueden revisar los puntos más importantes del video en el nivel 4. Los estudiantes no necesitan comprender todas estas ideas ahora mismo, pero pueden usar este nivel como referencia durante toda la Lección.

Hay mucho que aprender sobre las variables. Hoy vamos a usarlas para ayudarnos a hacer dibujos. Sin embargo, lo más importante es simplemente ver cómo dar a una etiqueta un valor nos ayuda a escribir programas.

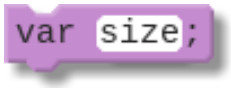
Evitar la carga frontal: si bien esta Lección comienza con dos recursos que explican cómo funcionan las variables, es probable que sean más significativas para los estudiantes una vez que hayan utilizado variables en los programas de dibujo. Asegúrese de que los estudiantes sepan que estos recursos están disponibles, y luego revíselos -si lo desea- al final de la Lección para ayudarlos a crear sentido.

Ampliación del conocimiento (30 min)

Programación con variables

Dirija a los estudiantes a los Code Studio y aborden las siguientes lecciones:

- Introducción a las variables
- Variables
- Nombrar variables
- Desafío: dibujar una imagen
- Desafío: actualización de variables

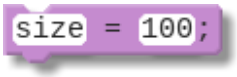


Variables

Una variable le permite almacenar un solo valor en la memoria de su computadora con un nombre descriptivo. El uso de variables le permite consultar fácilmente el mismo valor muchas veces en su programa o guardar un número al que le gustaría consultar más adelante.

Creación de variables

El comando var creará una nueva variable con la etiqueta que le da. Esta variable tiene la etiqueta size.



Asignación de valores

El operador de asignación = asignará un nuevo valor a su variable. Este comando asignó 100 al tamaño variable. La variable siempre debe estar en el lado izquierdo. Usted leería este comando como “obtiene tamaño 100”, ya que el tamaño obtiene un nuevo valor de 100. Cualquier valor antiguo que podría haber sido asignado se pierde para siempre. A las variables también se les puede asignar un número aleatorio. Esto le permite guardar un único valor aleatorio para que pueda usarlo tantas veces como desee en su programa.

Transferencia del conocimiento (5 min)

Reflexión

Indicación:

De a los estudiantes las siguientes instrucciones

- ¿Cuál es tu propia definición de variable?
- ¿Por qué las variables son útiles en los programas?

Discuta:

Haga que los estudiantes escriban sus ideas de forma independiente, posteriormente, invítalos a compartirlas con los compañeros del grupo.

En sus cuadernos:

¿qué conexiones observas entre las variables y el modelo entrada - almacenamiento - procesamiento - salida de una computadora?

Meta: Utilizar esta discusión para evaluar los modelos mentales de una variable de los estudiantes. Es posible que desee que los estudiantes escriban susrespuestas para que pueda recopilarlas y revisarlas más tarde. Debería verprincipalmente que entienden que las variables pueden etiquetar o nombrar un número para que pueda ser utilizado más adelante en sus programas. Si bien hay otras propiedades de una variable que los estudiantes pueden haber aprendido, esta es la más importante antes de pasar a la siguiente Lección.

Sugerencias de evaluación

Se sugiere el siguiente indicador para evaluar formativamente los aprendizajes:

- Nombran las variables en relación con su significado (nemotécnico).
- Utilizan las variables para hacer operaciones.

Lección 13: Sprites

Lección en línea Ver en Code Studio

Propósito

Para crear imágenes más interesantes y detalladas, se les presenta a los estudiantes el objeto sprite. A cada sprite se le puede asignar una imagen para mostrar, y los sprites también hacen un seguimiento de múltiples valores acerca de ellos mismos, lo que resultará útil en el camino al hacer animaciones.

Hacer un seguimiento de muchas formas y las diferentes variables que controlan aspectos de esas formas puede ser muy complejo. Habrá muchas variables con diferentes nombres. En cambio, los científicos informáticos crearon algo llamado objeto que permite un nombre de variable para controlar tanto la forma como todos sus aspectos. En Game Lab usamos un cierto tipo de objeto llamado sprite. Un sprite es solo un rectángulo con propiedades para controlar su aspecto. Las propiedades son las variables que están unidas a un sprite. Puede acceder a ellos mediante notación de puntos.

Usando la pestaña Animación, los estudiantes pueden crear o importar imágenes para ser usadas con sus sprites. Más adelante, estos sprites se convertirán en una herramienta útil para crear animaciones, ya que sus propiedades se pueden cambiar y actualizar a lo largo del curso de un programa.

Secuencia para el aprendizaje

- Conocimiento inicial (5 min)
- Ampliación del conocimiento
- Transferencia del conocimiento (5-10 min)
- Evaluación

Objetivos

Los estudiantes serán capaces de:

- Asignar un sprite a una variable.
- Usar notación de puntos para actualizar las propiedades de un Sprite.
- Crear una escena estática combinando sprites, formas y texto.

Preparación

- Sprite planificación de escena - Guía de lecciones

Recursos

¡Atención!
Por favor, haga una copia de cada documento que planee compartir con los estudiantes.

Para los profesores:

- lecciones Sprite enlace code.org

Para los estudiantes:

- Sprite planificación de escena - Guía de lecciones

Vocabulario

- **Propiedad**: Atributos que describen las características de un objeto.
- **Sprite:** Un personaje gráfico en la pantalla con propiedades que describen su ubicación, movimiento y apariencia.

Código

- drawSprites(group)
- var sprite = createSprite(x, y, width, height)
- sprite.scale

Estrategia de aprendizaje

Conocimiento inicial (5 min)

¿Cuánta información?

Piensa, empareja, comparte:

Hasta ahora solo hemos escrito programas que ponen formas simples en la pantalla. Haga una lista de todas las diferentes piezas de información que has utilizado para controlar cómo se dibujan estas formas.

Preguntar:

¿Qué pasaría si quisiéramos crear programas con imágenes más detalladas, tal vez incluso con personajes con los que pudiéramos interactuar? ¿Qué otras piezas de información pueden necesitar en su código?

Hoy aprenderemos cómo crear personajes en nuestras animaciones llamados sprites. Estos sprites se almacenarán en variables, al igual que almacenaron números en el pasado, pero los sprites pueden contener muchos datos, lo que le permitirá crear programas mucho más interesantes (¡y eventualmente animados!).

El objetivo aquí es hacer que los estudiantes piensen en todos los diferentes valores que forman una única forma en la pantalla, y cuántos valores más pueden necesitar para controlar un carácter más detallado en un programa. Si los estudiantes están luchando para encontrar ideas, puede usar algunas de las siguientes indicaciones: *¿Cómo le dice a una figura a dónde ir a la pantalla?*

¿Cómo le dices a una forma qué tamaño necesita ser? ¿Cómo le dices a una forma de qué color debería ser? ¿Qué hay de su esquema? ¿Qué sucede si quiere cambiar alguno de esos valores durante su programa o controlar otros elementos, como la rotación?

Ampliación del conocimiento

Introducción a Sprites

Distribuir:

Sprite planificación de escena - Guía de lecciones. Los estudiantes pueden usar esta hoja para planear la escena de Sprite que crean al final de esta Lección, pero la planificación también se puede completar en papel borrador.

Transición:

Enviar estudiantes a Code Studio y que realicen los Niveles de Code Studio. Sprite.

- Creando Sprites
- La pestaña de animación
- Escenas de sprite
- Extiende tu escena

El sprite es un tipo de datos llamado **objeto**. Si bien todavía no estamos introduciendo explícitamente el concepto de objetos, los estudiantes deben comprender que un sprite es un tipo de valor diferente de los que hemos visto anteriormente, uno que puede contener referencias a muchos más valores. Para los estudiantes que sienten curiosidad sobre si hay otros objetos en nuestros programas, pídales que vean si hay más bloques en la caja de herramientas que siguen la misma notación de *puntos* (como World.widthy World.height)

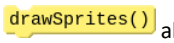
Sprites

Crear Sprites:

El bloque 

crea un nuevo sprite y lo asigna a una variable. El nombre predeterminado es sprite, por lo que querrán cambiarlo a algo más significativo.

Dibujo de Sprites:

Los Sprites solo aparecen en la pantalla cuando los dibuja 

allí. Llamar al comando dibuja todos los sprites creados en la pantalla.

Imágenes:

Una vez que haya creado un sprite, puede usar el sprite. setAnimation() comando para cambiar el aspecto de su sprite de un rectángulo a una imagen. Todas las imágenes que ha cargado en la pestaña Animación aparecen en el sprite.setAnimation() menú desplegable.

Suba su propia imagen:

También puede usar la pestaña Animación para cargar o dibujar tu propia imagen.

- Haga clic para cargar una imagen.
- Seleccione el archivo de tu computadora.
- Cambie el nombre de su imagen para que sea fácil de recordar. Para cambiarle el nombre, Haga clic en el texto debajo de la imagen.
- De vuelta en el modo de código, agregue un bloque sprite.setAnimation para que su sprite use su nueva animación.

Cambiar el tamaño con escala

En el cajón de Sprites de la caja de herramientas, verá un nuevo bloque llamado sprite.scale. Este comando permite cambiar el tamaño de un sprite en relación con su tamaño original. sprite.scale = 1 es el tamaño normal. sprite.scale = 0.5 hace que su sprite sea la mitad de grande, mientras sprite.scale = 2 que lo hace dos veces más grande.

Agregar texto

El bloque text le permite colocar texto en cualquier lugar que desee en la pantalla. Cambie este texto en el bloque provisto a otra cosa y agregue un segundo text bloque para escribir en una parte diferente de la pantalla.

Consejo:

El tamaño de texto predeterminado es bastante pequeño, pero puede usar el bloque textSize para cambiar eso. También puede usar el bloque fill para cambiar el color de tu texto.

Transferencia del conocimiento (5-10 min)

Compartir

Permita a los estudiantes compartir sus escenas de Sprite. Aliente a los estudiantes a reflexionar sobre sus escenas e identificar las formas en que les gustaría mejorar.

Evaluación

Evaluar escenas de sprite

Para evaluar las escenas de Sprite, pida a los estudiantes que hablen de su código. Verifique para asegurarse de que los estudiantes sepan por qué secuenciaron su código de la manera en que lo hicieron y, en particular, busquen “código muerto” o código que no afecte a la escena final. En este punto, es probable que los estudiantes sigan dibujando formas antes de dibujar el fondo (que luego no se verá) o que estén llamando drawSprites varias veces (solo se debe llamar una vez).

Lección 14: Ciclo de dibujo aleatorio

Lección en línea Ver en Code Studio

Propósito

En esta Lección, se presenta a los estudiantes el ciclo de sorteo, uno de los paradigmas de programación principales en Game Lab. Para comenzar la Lección, los estudiantes miran algunos flipbooks físicos para ver que tener muchos marcos con diferentes imágenes crea la impresión de movimiento. Luego, los estudiantes ven un video que explica cómo el ciclo de sorteo en Game Lab ayuda a crear la misma impresión en sus programas. Los estudiantes combinan el ciclo de dibujo con números aleatorios para manipular algunas animaciones simples con puntos y luego con sprites. Al final de la Lección, los estudiantes usan lo que aprendieron para actualizar su escena de sprites de la Lección anterior.

El ciclo de sorteo es un componente central de Game Lab. El hecho de que el entorno de Game Lab llama repetidamente a esta función muchas veces por segundo (por defecto 30 veces) es lo que permite a la herramienta crear animaciones. Esta Lección tiene dos objetivos, el primero es que los estudiantes vean cómo la animación consiste en mostrar muchas imágenes ligeramente diferentes en una secuencia. Para ayudar a los estudiantes a tener flipbooks físicos pueden usar un video. El segundo objetivo es que los estudiantes comprendan cómo el ciclo de dibujo les permite crear este comportamiento en Game Lab. Los estudiantes deben dejar la Lección entendiendo que los comandos del ciclo de sorteo se invocan después de todos los demás códigos, pero luego se los llama repetidamente a una velocidad de fotogramas. Los estudiantes tendrán la oportunidad de continuar desarrollando una comprensión de este comportamiento en las próximas dos lecciones.

Secuencia para el aprendizaje

- Conocimiento inicial (5 min)
- Ampliación del conocimiento (60 min)
- Transferencia del conocimiento (5 min)

Objetivos

Los estudiantes serán capaces de:

- Explicar qué es una animación y cómo crea la ilusión de un movimiento suave.
- Explicar cómo el ciclo de sorteo permite la creación de animaciones en Game Lab.
- Utilizar el ciclo de sorteo en combinación con el comando randomNumber, las formas y los sprites para realizar animaciones simples.

Preparación

- Imprime Flipbook no aleatorio - Flipbook Manipulativo y Aleatorio - Manipulador

Recursos

¡Atención!
Por favor, haga una copia de cada documento que planee compartir con los estudiantes.

Para los profesores:

- Aleatorio - Manipulador
- Ejemplo de Flipbook - Video

Vocabulario

- **Animación:** Una serie de imágenes que crean la ilusión de movimiento al mostrarse rápidamente una tras otra.
- **Marco:** Una sola imagen dentro de una animación.
- **Frecuencia de fotogramas**: La velocidad a la que se muestran los fotogramas de una animación, normalmente medida en fotogramas por segundo.

Código

- function draw() { }
- World.frameRate

Estrategia de aprendizaje

Conocimiento inicial (5 min)

Video:

Mostrar ejemplo de Flipbook - Video

Indicación:

Este video muestra un flipbook para hacer animaciones. Con sus propias palabras, explique: ¿cómo está funcionando? ¿Por qué "engaña a nuestros ojos" para que se piense que algo se está moviendo?

Discuta:

Haga que los estudiantes escriban sus ideas de forma independiente, luego compártanlo con sus compañeros, luego en grupo completo.

Objetivo: Esta discusión debe presentaralgunas ideas clave sobre la animación. Los estudiantes deben entender que la clave es ver muchas imágenes seguidas que son ligeramente diferentes. Introduzca la palabra de vocabulario "marco" como una de esas imágenes. Luego haga la transición al hecho de que pronto los estudiantes crearán sus propias animaciones.

Vamos a empezar a aprender cómo hacer animaciones, no solo imágenes fijas. Para hacer esto, necesitamos una forma de que nuestros programas dibujen muchas imágenes por segundo. Para hacer esto, tendremos que aprender una nueva herramienta importante.

Ampliación del conocimiento (45 min)

Video:

Miren el video de introducción a ciclos de dibujo,

Dirija a los estudiantes a CodeStudio y realice los Niveles de Code Studio.

- Formas y el lazo
- Sprites y Draw Loop
- Sprite Propiedades
- Anima tu escena

Meta: Use estos avisos para evaluar si los estudiantes han entendido los principales objetivos de aprendizaje de la Lección.

Conceptos clave: Existen muchos conceptos erróneos comunes con el ciclo de extracción. Asegúrese de que los estudiantes entiendan lo siguiente

- El ciclo de extracción se ejecutadespués de todos los demás códigos en su programa. En realidad, no importa donde se encuentre en su programa
- The Game Draw es ejecutado porGame Lab a una tasa de cuadros constante de 30 fotogramas por segundo. En realidad, no necesitallamar a la función usted mismo
- Los "marcos" en Game Lab se pueden considerar hojas de transparencia, A menos que dibujesun fondo, todas tus nuevas formas o sprites simplemente aparecerán encima de tus antiguos
- Solo debe de tener un ciclo deextracción en su programa

Transferencia del conocimiento (10 min)

Compartir:

Los estudiantes sólo necesitan hacer pequeños cambios en sus proyectos (p. Ej., Agitando un solo sprite), pero pida que compartan con un compañero o como clase completa.

Preguntar:

Haga que los estudiantes respondan a las siguientes instrucciones:

- ¿Qué es una animación?
- ¿Por qué el ciclo de sorteo nos ayuda a hacer animaciones?
- ¿Cuáles son algunos errores comunes que debemos tener en cuenta a medida que seguimos programando con el ciclo de sorteo?

Revisión:

Regrese a los recursos que los estudiantes vieron al comienzo de la Lección y aborde los conceptos erróneos que han surgido en la Lección.

Sugerencias para la evaluación

Se sugiere el siguiente indicador para evaluar formativamente los aprendizajes:

- Ejecutan ciclos (loops), a partir de un patrón que se repite en una secuencia.
- Evalúan su trabajo de manera colaborativa.
- Utilizan procesos de depuración
- Aprovechan recursos disponibles en las áreas de trabajo.

Lección 15: Patrón de contador desconectado

Lección sin conexión

Propósito

Los estudiantes exploran el comportamiento subyacente de las variables a través de una Lección desconectada. Usando tarjetas de notas y cadenas para simular variables dentro de un programa, los estudiantes implementan algunos programas cortos. Una vez que se sienten cómodos con esta sintaxis, los estudiantes usan el mismo proceso con las propiedades de los sprites, haciendo un seguimiento del progreso de un sprite en la pantalla.

El razonamiento sobre las variables puede ser complicado, especialmente para los nuevos programadores. En esta Lección, los estudiantes completan una Lección desenchufada utilizando manipulativos físicos (tarjetas y cuerdas) para construir un modelo mental de cómo la información puede almacenarse en una variable y ser manipulada por un programa. Este modelo luego se extiende a las propiedades de los sprites, que mantienen los valores de una manera similar. Esta Lección presenta la sintaxis y los conceptos que los estudiantes podrán "conectar" en la siguiente Lección.

Secuencia para el aprendizaje

- Conocimiento inicial (15 min)
- Ampliación del conocimiento (20 min)
- Ampliación del conocimiento 2 (30 min)
- Transferencia del conocimiento (10 min)

Objetivos

Los estudiantes serán capaces de:

- Describir la conexión entre la actualización de las propiedades de ubicación de un sprite y el movimiento de los sprites en la pantalla.
- Leer y seguir los pasos de un breve programa escrito en pseudocódigo que manipula los valores de las variables.

Preparación

- Prepara materiales para etiquetas y valores: Fichas, post-its o trozos de papel, etc.
- Preparar materiales para conectores: piezas de cuerda, plumas o limpiapipas, etc.
- Tablero de variables desconectado - Manipulador para cada grupo o reúne papel para que los estudiantes lo utilicen para hacer sus tableros.
- Revisa las reglas de la Lección Desconectada de Variables para asegurarte de que las comprenda y prepárate para responder preguntas, especialmente si las demostrarás tú mismo.
- Sacar copias de Variables desconectada- Guía de lecciones para cada alumno.

Recursos

¡Atención!
Por favor, haga una copia de cada documento que planee compartir con los estudiantes.

Para los estudiantes:

- Variables desconectada- Guía de lecciones

Vocabulario

- **Expresión:** Cualquier unidad de código válida que resulta en un valor.
- **Variable:** Un marcador de posición para una información que puede cambiar.

Estrategia de aprendizaje

Conocimiento inicial (15 min)

Pregunta:

En la última Lección, usamos el ciclo de dibujo para hacer que nuestros sprites se muevan en la pantalla. ¿Cuáles son otras formas en que podríamos hacer que nuestros sprites se muevan?

Ponga las ideas de los estudiantes en el pizarrón.

En las próximas Lecciones, vamos a ver muchas formas de mover nuestros sprites. Para hacer eso, necesitamos aprender un poco más sobre las variables y cómo funcionan. Hoy vamos a hacer una Lección con variables y propiedades de sprites que nos ayudarán a hacer estos tipos de movimiento. A medida que avanzamos en la Lección, piense cómo, lo que está aprendiendo, puede ayudarlo a hacer que sus sprites se muevan de la manera que desee.

Esta Lección está diseñada para abordar muchos **conceptos erróneos comunes** con variables y memoria.

Conceptos erróneos comunes

- Las variables pueden tener valores múltiples (no pueden, las variables tienen como máximo un valor)
- Las variables "recuerdan" los valores antiguos (no, por lo tanto, los valores antiguos se eliminan enla Papelera)
- Las variables se conectan después de un comando como "x=y" (Esto puede surgir más tarde en el curso cuando los estudiantes usan sprites y se abordarán en gran detalle.
- Por ahora, los estudiantes se ven obligados a crear nuevas tarjetas de valor cada vez porque incluso para una declaración como x=y nohay "conexión" entre las variablesformadas)
- Las variables tienen expresiones (por ejemplo, 1+5). Las variables solocontienen valores, las expresiones se calculan de antemano. Es por eso que las tarjetas de valores solo se crean una vez que los estudiantes tienen un solo valor. Aplicar esta regla de cerca

Ampliación del conocimiento (20 min)

Variables Lección desconectada

Grupo:

Agrupar a los estudiantes en parejas.

Entregar a cada par:

- Un conjunto de etiquetas / valores y conectores
- Una sola hoja de papel para crear su pizarrón
- 2 copias de las Variables desconectada - Guía de lecciones.

Pantalla:

Muestre las reglas de la página principal de la guía de lecciones y escriba los primeros dos programas de la segunda página.

Hoy vamos a trabajar en un mundo de etiquetas, valores y conectores entre ellos. Para simular este mundo, usarás los trozos de papel y cuerda que les he dado. Para empezar, tendremos que configurar nuestros tableros, y luego repasaremos cómo funcionan los comandos para este mundo.

Demuestre:

Muestre a la clase cómo dividir sus tablas en 3 secciones y etiquételas en secuencia, como se muestra en la primera página de la guía de lecciones.

Apoyo:

Los estudiantes deben trabajar en los dos primeros programas como un grupo, mientras hace referencia a los pasos en la guía de lecciones. A medida que los equipos se sienten cada vez más cómodos al completar los comandos, alientan a los equipos a ejecutar cada comando de forma independiente antes de comparar sus tableros con un compañero. El objetivo es solo asegurarnos de que todos tengan la oportunidad de comprender los pasos de la Lección.

Ahora es su turno de intentar ejecutar algunos de estos programas por su cuenta. En la parte inferior de la página hay dos programas más que puede ejecutar. Para cada uno, debe ejecutar el programa para averiguar el estado final del programa. En otras palabras, debe saber qué etiquetas están conectadas a qué valores. Una vez que llegue al final de cada programa, puede comparar sus resultados con un compañero. Si no está de acuerdo, vuelve para ver si puede encontrar dónde perdiste la pista.

Apoyo:

Los estudiantes deben trabajar en parejas a través de los dos programas en la guía de lecciones. Haga un chequeo para asegurarte de que todos estén de acuerdo con el estado final del programa (qué etiquetas están conectadas con qué valores). Si los estudiantes no están de acuerdo, refuerza la necesidad de depurar al leer el código retrocediendo y rastreando cada paso.

Ampliación del conocimiento 2 (30 min)

Sprite propiedades

Objetivo: Los estudiantes deben ver que los comandos, como x=x+1 mover un sprite de una manera deliberadaa través de la pantalla, en posición al movimiento aleatorio que vieron en laLección anterior

Distribuir:

Propiedades de Sprite en Variables desconectada- Guía de lecciones si no lo has incluido en la guía de lecciones original.

Demuestra:

Muestre a los estudiantes las reglas de Lección en la guía de lecciones. Trabaje en el programa 5 como grupo y muestra cómo crear una nueva tarjeta Sprite y conectarla a su tarjeta de etiqueta variable y a las tarjetas de propiedad de sprite. Asegúrese de que los estudiantes comprendan que deberían crear una nueva tarjeta sprite cada vez que vean el comando createSprite y que dibujen sus sprites en la cuadrícula cada vez que vean el comando drawSprites.

Apoyo:

Los estudiantes deben trabajar en los últimos dos programas usando sus materiales manipulables.

Pregunta:

¿Cómo se movió el sprite a través de la red en el Programa 3? ¿Cómo se movió el sprite a través de la red en el Programa 4 de la guía de lecciones?

Después de que los estudiantes hayan completado las preguntas de reflexión, deben comparar con un compañero, luego discutirlo como clase.

Transferencia del conocimiento (10 min)

Discusión

El objetivo de la discusión no es hacer que los estudiantes piensen en soluciones a todos los problemas, si no que los identifiquen, preparándolos para la Lección del patrón contrario. Algunos programas que los estudiantes puedan ver son que las caras sonrientes permanecieron en la pantalla, en lugar de moverse, que solo podían trasladarse la izquierda y la derechao hacia arriba o hacia abajo, o que lasimágenes se detienen después de un cierto periodo de tiempo.

Indicación:

Vimos algunas pistas hoy de cómo podríamos programar los tipos de movimiento que queremos para nuestros sprites. ¿Cuáles son algunos problemas que aún tenemos que resolver para que el sprite parezca moverse en la forma que deseo?

Permita que los estudiantes hagan una lluvia de ideas sobre problemas y anótalos en el pizarrón.

Preguntar:

Elija uno o dos de estos problemas y comienza a pensar en algunas formas en que podría ser resuelto.

De tiempo a los estudiantes para que hagan una lluvia de ideas individualmente antes de compartir sus soluciones.

Estas son buenas ideas. En la próxima Lección, vamos a ver cómo podemos usar algunas de las cosas que hemos aprendido hoy para hacer que nuestros sprites se muevan de muchas maneras diferentes.

Sugerencias para evaluación

Se sugiere el siguiente indicador para evaluar formativamente los aprendizajes:

- Describen de manera explícita, cada paso que compone un proceso secuencial.

Lección 16: Movimiento de sprite

Lección en línea Ver en Code Studio

Propósito

Al combinar Dibujo de repetición (Draw loop) y patrón de contador, los estudiantes escriben programas que mueven los sprites a través de la pantalla, así como también animan otras propiedades de los sprites.

Esta Lección combina el dibujo con ciclos que los estudiantes conocieron anteriormente, de igual manera que el patrón de contador que aprendieron para crear programas con un movimiento determinado. Al aumentar o disminuir las propiedades de los sprites, como sprite.x, puedes escribir programas que muevan los sprites en los patrones esperados, en lugar de uno aleatorio que usamos en el pasado. Las animaciones que los estudiantes aprenden a crear en esta Lección sientan las bases para todas las animaciones y juegos que harán en el resto del curso.

Secuencia para el aprendizaje

- Conocimiento inicial (5 min)
- Ampliación del conocimiento (40 min)
- Transferencia del conocimiento (5 min)

Objetivos

Los estudiantes serán capaces de:

- Usar el patrón de contador para incrementar o disminuir las propiedades de los sprites.
- Identificar qué propiedades de los sprites deben cambiarse, y de qué manera, para lograr un movimiento específico.

Estrategia de aprendizaje

Conocimiento inicial (5 min)

Revisando las propiedades de Sprites

Pregunte:

En una hoja de papel, enumere todas las propiedades de los sprites que pueda imaginar y el aspecto de un sprite que afecten.

Comente:

¿Qué tipo de animaciones podrías hacer al combinar las propiedades de los sprites con el patrón contrario? Considere agregar y restar propiedades, o incluso actualizar varias propiedades al mismo tiempo. Registre ideas mientras los estudiantes las comparten en el pizarrón.

El propósito de esta discusión es motivar a los estudiantes a pensar cómo podrían usar las diversas propiedades de sprites que han visto hasta ahora para hacer animaciones con un movimiento determinado. Si los estudiantes luchan para encontrar ideas, pueden limitar la pregunta a propiedades específicas. Por ejemplo:

- ¿Qué le pasaría a un sprite si aumentara constantemente su x propiedad?
- ¿Qué le pasaría a un sprite si aumentara constantemente su y propiedad?

¿Qué pasa con otras propiedades o combinando propiedades múltiples?

Ampliación del conocimiento (40 min)

Niveles: Sprites e imágenes

Transición:

Envía estudiantes a Code Studio - Movimiento Sprite.

Apoyo:

Los estudiantes deben progresar a través de los niveles sin detenerse en esta clase. Cuando hayan terminado los niveles de desarrollo de habilidades, se les dará la opción de qué proyecto desean extender.

Niveles de Code Studio:

- El patrón de contador
- Movimiento con el patrón de contador
- Depuración con vigilantes
- Animar a los Sprites
- Crea tu propia animación
- Desafíos
- Juego gratis

Transferencia del conocimiento (5 min)

Preguntar

Pida a los estudiantes que reflexionen sobre el desarrollo de las cinco prácticas del curso DISCOVERIES CS (Resolución de problemas, Persistencia, Creatividad, Colaboración, Comunicación). Elije una de las siguientes indicaciones según lo consideres apropiado.

- Elija una de las cinco prácticas en las que crees que demostraste crecimiento en esta Lección. Escribe algo que hiciste que ejemplifica esta práctica.
- Elija una práctica en la que pienses que puedes seguir creciendo. ¿Qué te gustaría mejorar?
- Elija una práctica que pensaste que era especialmente importante para la Lección que completamos hoy. ¿Qué lo hizo tan importante?

Lección 17: Booleanos desconectados

Lección sin conexión

Propósito

En esta Lección, se les presenta a los estudiantes los valores booleanos y la lógica, así como las declaraciones condicionales. La clase comienza jugando un juego simple de levantarse (Stand Up), sentarse (Sit Down) en el que las afirmaciones booleanas (verdadero / falso) describen propiedades personales (color de pelo u ojos, tipo de ropa, edad, etc.). Esto hace que los estudiantes piensen cómo pueden enmarcar una propiedad con múltiples valores potenciales (como la edad) con una pregunta binaria.

A partir de ahí, los estudiantes reciben un grupo de objetos con propiedades físicas similares pero variables. Con un compañero, agrupan esos objetos basándose en declaraciones booleanas (declaraciones lógicas) cada vez más complejas, incluidos booleanos compuestos con AND (y) y OR (o).

Finalmente, revelamos qué Condicionales es una herramienta para tomar decisiones o afectar el flujo de un programa que utiliza declaraciones booleanas como entrada.

Secuencia para el aprendizaje

- Conocimiento inicial (10 min)
- Ampliación del conocimiento (30 min)
- Transferencia del conocimiento (5 min)

Objetivos

Los estudiantes serán capaces de:

- Organizar objetos basados en declaraciones booleanas simples y compuestas.
- Describir las propiedades de un objeto usando declaraciones booleanas.

Preparación

- Propiedades booleanas - Guía de lecciones para cada estudiante.
- (Opcional) Reúne objetos con características similares pero variables para usar en lugar de la hoja de trabajo (los ladrillos LEGO funcionan bien, una bolsa de dulces mixta también puede ser divertida).

Recursos

¡Atención!
Por favor, haga una copia de cada documento que planee compartir con los estudiantes.

Para los estudiantes:

- Propiedades booleanas - Guía de lecciones.

Vocabulario

- **Booleano:** Un valor único de VERDADERO o FALSO.
- **Condicionales:** Declaraciones que solo se ejecutan bajo ciertas condiciones.
- **Expresión:** Cualquier unidad de código válida que se resuelve en un valor.

Estrategia de aprendizaje

Conocimiento inicial (10 min)

Párate, siéntate

Distribuir:

Entregue a cada alumno una tarjeta y pida que respondan las siguientes preguntas (puedes agregar algunas propias).

1. ¿Cuál es tu color de cabello?
2. ¿Usas lentes o lentes de contacto?
3. ¿Cuál es tu número favorito?
4. ¿Cuál es tu color favorito?
5. ¿En qué mes naciste?
6. ¿Tienes hermanos?
7. ¿Cuál es el último dígito de tu número de teléfono?
8. Escribe algo acerca de ti que las personas no saben y no pueden deducir con solo mirarte.

Luego recoja las cartas. Para jugar, sigue estos pasos:

Para cada tarjeta:

1. Selecciona una tarjeta
2. Diga: Voy a leer la respuesta al # 7, pero si eres tú, no digas nada.
3. Lee la respuesta al # 7
4. Diga: Ahora todos se ponen de pie y vamos a hacer algunas preguntas con respuestas booleanas para ayudar a determinar quién es esta persona. Voy a decir un montón de declaraciones. Si son ciertos, mantente en pie. Si son falsos, siéntate.
5. Traduce las respuestas del # 1 al # 7 en enunciados que pueden ser verdaderos o falsos, consulta a continuación, en voz alta. La persona que queda en pie es aquella que respondió verdadero a todas las afirmaciones y, por lo tanto, es responsable de la tarjeta.
6. Según cómo se preguntaron los números 3, 4, 5 y 7, es probable que más de una persona siga en pie. Si es el caso, vuelve a preguntarlas, esta vez restringiendo más la información, por ejemplo, “Mi color favorito es el violeta” para la pregunta 4.

Traducción de ejemplo

Pregunta	Responder	Declaración verdadera / falsa
1) ¿Cuál es tu color de pelo?	marrón	"El color de mi cabello es marrón".
2) ¿Usa lentes o lentes de contacto?	sí	"Uso gafas o lentes de contacto".
3) ¿Cuál es tu número favorito?	12	"Mi número favorito es mayor que 10 y menor que 20."
4) ¿Cuál es tu color favorito?	púrpura	"Mi color favorito aparece en una puesta de sol".
5) ¿En qué mes naciste?	Mayo	"Nací en la primavera".
6) ¿Tienes hermanos?	sí	"Tengo hermanos".
7) ¿Cuál es el último dígito de su número de teléfono?	5	"El último dígito de mi número de teléfono es excelente".

Juega esto varias veces cambiando los enunciados “verdadero / falso” que usas. Sé creativo con el uso y recuerda a los estudiantes que la OR (o) significa que una parte de la declaración es verdadera dará como resultado que toda la declaración sea verdadera.

Discuta:

El juego Pararse/Sentarse con los estudiantes:

- ¿Qué tipo de cuestionamientos realizamos?
- ¿Alguna vez te confundiste sobre si debieses estar parado o sentado? ¿Por qué?
- En cualquier punto del juego, ¿en cuántos estados diferentes podrías estar?

Introduzca el vocabulario booleano como una descripción para los tipos de preguntas que estábamos haciendo. La característica que define a un booleano es que puede tener solo dos estados: en nuestro juego, esos estados eran Verdadero y Falso, o de pie y sentado.

Ampliación del conocimiento (30 min)

Hacer las preguntas correctas

Idea Genial

Preguntar:

Haga una lluvia de ideas sobre lugares donde hayan visto valores booleanos anteriormente, ya sea en la clase o en el mundo.

Discuta:

Haga que los estudiantes compartan sus respuestas. Las posibles respuestas podrían **incluya**:

- Binario
- Diagramas de flujo
- Interruptores de luz (y otros dispositivos que pueden estar encendidos o apagados)

Ordenando con booleanos

En el juego anterior, las preguntas booleanas se basaron en sus propiedades. Sus propiedades no tienen que existir solo en dos estados (¿cuántos colores de cabello diferentes hay en el salón?), Pero las preguntas que hizo tenían que dividirlas en dos estados (¿cuántas personas en el salón tienen el pelo rojo?). Vamos a hacer una clasificación similar usando las propiedades de varias imágenes.

Los estudiantes a menudo luchan con la idea de hacer *mayor que* o *igual a* sersiempre FALSO - tienden a pensar en esos como declaraciones de verdad enlugar de preguntas de relación.

En esta Lección usamos el lenguaje *algo* es igual a *otra cosa* para reflejar el código que verán más adelante (porejemplo something == something_else), pero para los estudiantes que estan luchando por ver esto como una pregunta en lugar de una afirmación, puede alentarlos a que reformulen la afirmación moviendo el 'es' al principio, entonces:

algo es igual a otra cosa

se convierte

¿es algo igual a otra cosa?

Al combinar booleanos con " O", estamos utilizando lo que se llama *lógica o* - lo que significa que estamospreguntando si alguno (o ambos) o los booleanos son VERDADEROS. A menudo, los estudiantes piensan en O como *exclusivo o* - lo que significa que estamos preguntando si solo uno (pero no el otro) es VERDADERO.

Asegúrese de hacer preguntas o preguntarles a los estudiantes donde losdos booleanos son VERDADEROS para asegurarse de que pueda obtener estaidea equivocada temprano.

Grupo:

Organice a los estudiantes en parejas.

Configuración:

Asigne a cada miembro del par un verdadero y un falso.

Distribuir:

Distribuya las imágenes recortadas de la hoja de trabajo o proporcione a los estudiantes algunos objetos para ordenar.

Voy a leer un montón de enunciados binarios en forma de "la figura equivale a un cuadrada o tiene más de 4 lados", y ustedes van a clasificar sus objetos para organizarlos en pilas VERDADERAS y FALSAS.

Si los estudiantes no están de acuerdo con respecto a qué pila debe colocar un objeto, primero deben analizar cuál es la propiedad, cuáles son los dos resultados de la pregunta binaria y luego, si aún no pueden ponerse de acuerdo, deben llevarla a la clase para votar.

Si está utilizando los recortes proporcionados, puede comenzar con las siguientes preguntas:

- Lados es igual a 3
- El relleno es igual a negro
- Las esquinas son menos de 1
- El ancho es igual a la altura
- El relleno es igual a gris y lados son más que 4
- Lados es mayor que 4 y menos de 7
- Lados es mayor que 4 o menor que 7
- Lados es más que o igual a 5

Condicionales

Meta:

Después de acostumbrarnos a ordenar objetos en VERDADERO y FALSO, debemos presentarles a los estudiantes el concepto de que los Booleanos también se pueden usar para controlar el flujo de un programa.

Una condicional nos permite tomar una decisión basada en el resultado de una pregunta (o condición) booleana. De hecho, estábamos usando implícitamente condicionales en la Lección de Pararse/Sentarse porque había una acción relacionada con cada resultado potencial del booleano. Podríamos haber reformulado las instrucciones como si la declaración es verdadera: permanece de pie de lo contrario siéntate.

Preguntar:

Seleccione un objeto de tu montón y sosténgalo.

Voy a hacerte una pregunta booleana sobre tu objeto y darte una acción relacionada con el resultado del booleano. Averigua cuál debería ser tu respuesta para tu forma y Haga la respuesta correcta.

Preguntar:

Haga a los estudiantes algunas preguntas booleanas sobre ese único objeto y diga a los estudiantes algo que hacer si esa pregunta es verdadera. Por ejemplo:

- Si los lados son iguales a 4, Haga un baile.
- Si el patrón es igual a rayado, siéntate.
- Si el ancho es igual a la altura, salta sobre un pie.

Transferencia del conocimiento (5 min)

Condición explícita:

Los booleanos y condicionales son en realidad algo que utilizamos en nuestra vida cotidiana, simplemente no solemos ser explícitos al respecto.

Como una forma de practicar el pensamiento de forma explícita sobre condicionales, considere despedirse de sus estudiantes utilizando compuestos booleanos y condicionales. Por ejemplo.

- Si te sientas en la mesa cuatro y tu cabello es marrón, puedes irte.
- Si tu primer nombre comienza con A o B, puedes irte.
- Si tus zapatos son negros, puedes irte.

Sugerencias para evaluación

Se sugiere el siguiente indicador para evaluar formativamente los aprendizajes:

- Organizan objetos en base a criterios simples
- Describen el proceso de trabajo para la organización de los elementos con los que trabajan.

Lección 18: Booleanos y condicionales

Lección en línea Ver en Code Studio

Propósito

Los estudiantes comienzan usando booleanos para comparar el valor actual de una propiedad de sprite con un valor objetivo, usando esa comparación para determinar cuándo un sprite ha alcanzado un punto en la pantalla, agrandado a un tamaño determinado o alcanzado un valor usando el patrón contrario. Después de usar booleanos directamente para investigar los valores o las propiedades de sprite, los estudiantes agregan sentencias condicionales "if" (si) para escribir código que respondan a esas comparaciones booleanas.

Esta Lección sigue de cerca el modelo booleano que los estudiantes experimentaron por primera vez en la Lección Booleanos desconectados. Como antes, comenzamos con el uso de booleanos directamente antes de usar booleanos para desencadenar sentencias if. En la siguiente Lección, presentaremos algunos bloques de producción booleana, como keyDown(), que se pueden usar en lugar de comparaciones booleanas simples para escribir programas que respondan a la entrada del usuario.

Secuencia para el aprendizaje

- Conocimiento inicial (5 min)
- Ampliación del conocimiento (40 min)
- Transferencia del conocimiento (5 min)

Objetivos

Los estudiantes serán capaces de:

- Predecir el resultado de declaraciones booleanas simples.
- Usar condicionales para reaccionar a los cambios en las variables y las propiedades de los sprites.

Vocabulario

- **Expresión Booleana:** En la programación, una expresión que se evalúa como Verdadera o Falsa.
- **If-Statement:** La estructura de programación común que implementa "declaraciones condicionales".

Código

- If statement
- Equality operator
- Inequality operator
- Greater than operator
- Greater than or equal operator
- Less than operator
- Less than or equal operator

Estrategia de aprendizaje

Conocimiento inicial (5 min)

Contestando preguntas Booleanas

Al final del juego preguntas booleanas de la Lección anterior, los estudiantes comenzaron a agregar condiciones a sus preguntas booleanas, lo que significa que, si la respuesta a la pregunta es verdadera, debería suceder algo. Antes de programar con condicionales, queremos asegurarnos de que los estudiantes tengan un conocimiento sólido de lo que realmente son los booleanos.

Rápido:

- ¿Cuántos números diferentes hay en el mundo?
- ¿Cuántas palabras diferentes o combinación de letras y otros personajes hay?
- ¿Cuántos valores booleanos diferentes hay?

Discutir:

Los estudiantes deben darse cuenta de que las dos primeras preguntas (números y cadenas) son esencialmente infinitas, pero que los booleanos están limitados a dos estados.

A medida que comience a programar hoy, estará usando booleanos para hacer programas que cambien su comportamiento dependiendo de la respuesta a esas preguntas booleanas.

Aunque aparentemente es simple, entender como una declaración booleana evaluará puede ser difícil dado que los diferentes lenguajes de programación tienen opiniones diferentes sobre "veracidad" y "falsedad". De hecho, JavaScript (el lenguaje utilizado en este curso) tiene dos operadores diferentes para probar la igualdad booleana == y ===.

El operador doble igual (==) es bastante generoso para determinar la veracidad, por ejemplo, cada uno de los siguientes se considera true (verdadero) en JavaScript cuando se usa el == operador, pero se usaría el === operador: para false

1=="1" verdadero

1==="1" falso

1 == verdadero;

"1" == verdadero;

5 == "5";

null == undefined; " " == falso;

Ampliación del conocimiento (40 min)

Booleanos

Transición:

Envíe estudiantes a Code Studio y realiza los Niveles de Code Studio - Condicionales.

- Booleanos y operadores de comparación
- Comparación Booleana
- Declaraciones "si"
- Condicionales Básicos
- Juego gratis

Transferencia del conocimiento (5 min)

Agregar condicionales

Reflexión:

Piense en todos los programas que ha escrito hasta ahora;

¿Cómo podría usar condicionales para mejorar uno de sus programas de lecciones pasadas? ¿Qué condición verificaría y cómo respondería a ella?

Sugerencias de evaluación

Se sugiere el siguiente indicador para evaluar formativamente los aprendizajes:

- Representan con ejemplos simples la lógica booleana
- Combinan diversos criterios de búsqueda utilizando los operadores booleanos
- Incorporan preguntas condicionales en la construcción de sus soluciones

Lección 19: Condicionales y entrada del usuario

Lección en línea Ver en Code Studio

Propósito

Después de la introducción a las declaraciones booleanas y "si" en la Lección anterior, se introduce a los estudiantes en un nuevo bloque llamado keyDown() que devuelve un booleano y se puede usar en sentencias de condicionales para mover sprites alrededor de la pantalla. Al final de esta Lección, los estudiantes tendrán programas escritos que tomarán la entrada de teclado del usuario para controlar los sprites en la pantalla.

Una forma común de usar condicionales es verificar los diferentes tipos de entrada del usuario, especialmente las pulsaciones de teclas. Tener una forma para que un usuario interactúe con un programa lo hace más interesante y dinámico. Sin interacción del usuario, es muy difícil crear un juego. Por lo tanto, la introducción de condicionales y aportes de los usuarios para la toma de decisiones es el primer gran paso hacia la creación de juegos.

Secuencia para el aprendizaje

- Conocimiento inicial (5 min)
- Ampliación del conocimiento (40 min)
- Transferencia del conocimiento (5 min)

Objetivos

Los estudiantes serán capaces de:

- Usar condicionales para reaccionar a la entrada del teclado
- Mover los sprites en respuesta a la entrada del teclado.

Recursos

¡Atención!
Por favor, haga una copia de cada documento que planee compartir con los estudiantes.

Para los profesores:

- Expresiones booleanas - video
- Contenido de Lección

Código

- keyDown(code)

Estrategia de aprendizaje

Conocimiento inicial (5 min)

Tomando entrada

Debate:

Hasta ahora, todos los programas que ha escrito se ejecutan sin intervención del usuario. ¿De qué manera la adición de la interacción del usuario puede hacer que sus programas sean más útiles, efectivos o entretenidos? ¿Cómo podría un usuario proporcionar información sobre su programa?

El objetivo aquí no es entrar en las especificaciones técnicas de cómo los programas pueden recibir información (los estudiantes entenderán eso en la parte en línea de la Lección), sino más bien hacer que los estudiantes piensen en cómo permitir que los usuarios que ingresen puedan cambiar los programas que ellos 'han hecho'. Anime a los estudiantes a pensar en las diversas entradas y salidas informáticas que existen ¿Qué entradas serían más útiles para los tipos de programas que han estado haciendo?

Ampliación del conocimiento (40 min)

Entrada de teclado

Transición:

Envíe a los estudiantes a Code Studio, contenido de Lección

- Entrada de teclado
- Edición de imágenes
- Desafío: Agregar más entradas

Transferencia del conocimiento (5 min)

Considerando las condiciones

Indicación:

Para que los estudiantes continúen pensando en cómo se pueden usar los condicionales en la programación, pida que ideen escenarios en juegos o programas que usan regularmente que pueden ser activados por los condicionales.

Discuta:

Haga que los estudiantes compartan las respuestas. Las respuestas de los estudiantes pueden incluir:

- Si mi nombre de usuario y contraseña son correctos, conéctate a Facebook
- Si Pacman ha reunido todas las bolas, comienza el siguiente nivel
- Si mi teclado o mouse no se movió en 10 minutos, encienda el protector de pantalla.

Sugerencias de evaluación

Se sugiere el siguiente indicador para evaluar formativamente los aprendizajes:

- Usan sentencias condicionales en un proyecto de programación,
- Permitir que los usuarios que interactúen con el proyecto puedan describir la lógica programada.

Lección 20: Otras formas de entrada

Lección en línea Ver en Code Studio

Propósito

En esta Lección, los estudiantes continúan explorando maneras de usar declaraciones condicionales para tomar la opinión del usuario. Además del comando keyDown() simple aprendido ayer, los estudiantes aprenderán sobre varios otros comandos de entrada de teclado, así como formas de tomar la entrada del mouse.

Los estudiantes han aprendido cómo tomar decisiones simples con condicionales. A veces, sin embargo, queremos tomar una decisión en función de si la condición sobre la que preguntamos originalmente era falsa o si queremos tomar una decisión basada en que múltiples condiciones sean ciertas. Ahí es donde aparecen enunciados y condicionales más complejos. Estas declaraciones son una segunda declaración adjunta a una instrucción if (si). Otras declaraciones se ejecutan cuando la sentencia if (si) a la que está asociada es falsa. Puede pensarlo como “si algo es verdadero Haga cosa 1 sino (else) Haga cosa 2”.

Este concepto se presenta junto con varios comandos nuevos de ingreso de teclas y mouse, lo que permite a los estudiantes crear gradualmente programas que ingresan de diferentes maneras.

Secuencia para el aprendizaje

- Conocimiento inicial (5 min)
- Ampliación del conocimiento (40 min)
- Transferencia del conocimiento (5 min)

Objetivos

Los estudiantes serán capaces de:

- Usar una instrucción else (sino) como el caso de respaldo a una sentencia if (si).
- Diferenciar entre las condiciones que son verdaderas una vez por interacción y aquellas que permanecen verdaderas a lo largo de la duración de una interacción.

Vocabulario

- **Condicionales:** declaraciones que solo se ejecutan bajo ciertas condiciones.

Código

- keyWentDown(code)
- keyWentUp(code)
- mouseDidMove()
- mouseDown(button)
- mouseWentDown(button)
- MouseWentUp(button)
- sprite.visible
- If/else statement

Estrategia de aprendizaje

Conocimiento inicial (5 min)

Verificar la comprensión

Retomaremos donde lo dejamos antes, usando condicionales para escribir programas que respondan a los comentarios de los usuarios. Actualicemos lo que aprendimos.

Rápido:

- ¿Qué es un booleano? (respuesta; un valor verdadero / falso)
- ¿Cuál es la relación entre un booleano y un condicional? (resp; un condicional hace una pregunta booleana y ejecuta el código si la respuesta es verdadera)
- ¿Cuáles son algunos ejemplos de operadores de comparación que dan como resultado un booleano? (resp.,>, <, ==)
- ¿Cuál es la diferencia entre = y ==? (resp., =se usa para asignar un valor, == se usa para verificar si dos valores son iguales).

Ampliación del conocimiento (40 min)

Sí / más y más entradas

Video:

Miren el video de condicionales (conditional) juntos como una clase. Será una revisión de las declaraciones if (si) e introducirá también algunos conceptos nuevos.

Transición:

Envíe a los estudiantes a Code Studio Contenido de la Lección

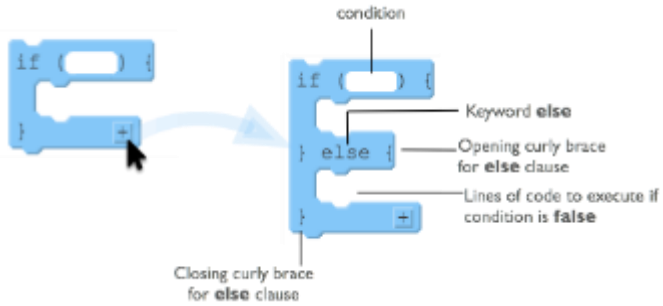
- Sigue al mouse
- Video: Condicionales
- Declaraciones de If-Else (Si-Sino)
- Entrada con If-Else (Si-Sino)
- Desafío: verificar múltiples condiciones

If-Else (Si-Sino) Declaraciones

Cómo funcionan las declaraciones If-Else (Si-Sino)

Con una instrucción if-else, usted está dando un comando cualquiera-or (o): en el cual se ejecutarán las líneas de código dentro de if o las líneas dentro de else. Esas son las opciones.

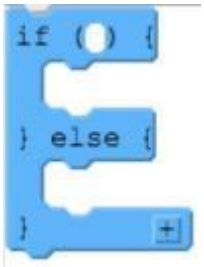
En el video (**recuerde activar los subtítulos**) se muestra cómo agregar una cláusula else a una instrucción if: pulsa el pequeño símbolo + en la cola de la declaración if.



Dentro de las llaves de la cláusula else, coloque las líneas de código que desea ejecutar si la condición booleana de la declaración if es falsa. Algunas notas importantes sobre la cláusula else:

- El else debe venir inmediatamente después de la llave de cierre de una declaración if.
- El otro else también tiene su propio conjunto de llaves de apertura y cierre para encapsular líneas de código.
- ¿Qué pasa si mi condición no es verdadera?

A veces queremos decirle a nuestro programa qué hacer si se trata de una condición true, pero también qué hacer si es así false. Al presionar el botón más en la parte inferior de su bloque condicional le dará otra sección llamada else. Esta sección else es una alternativa: se llamará siempre que exista la condición if anterior que sea falsa.



Clics del mouse

Las pulsaciones de teclas son geniales, pero a veces desea que los usuarios interactúen a través de los clics del mouse. Hay un nuevo bloque llamado mouseDown() que, similar a keyDown(), compruebe si se presionan los botones izquierdo o derecho del mouse. Si está usando una computadora con un mouse o un panel táctil que solo tiene un botón, querrá usar siempre mouseDown("left").

mouseDidMove (el mouse se movió)

También podemos usar expresiones booleanas para verificar si el mouse se movió o no. El bloque mouseDidMove devolverá false si el mouse está quieto, pero true si el mouse se ha movido.

Transferencia del conocimiento (5 min)

Compartir

El último programa de esta Lección consiste en pedir a los estudiantes que amplíen un programa a usos condicionales para la participación del usuario de maneras nuevas e interesantes. Haga que sus estudiantes compartan sus proyectos con la clase, lo que les pedirá que expliquen cómo usaron los condicionales en sus programas.

Sugerencias de evaluación

Se sugieren los siguientes indicadores para evaluar formativamente los aprendizajes:

- Crean variables con nombres claros que expliquen tipos de datos
- Incorporan condicionales y explican la funcionalidad de forma predictiva
- Prueban y perfeccionan sistemáticamente los programas utilizando una serie de casos de prueba

Lección 21: Proyecto – Tarjeta interactiva

Lección en línea Ver en Code Studio

Propósito

En este proyecto, los estudiantes planifican y desarrollan una tarjeta de felicitación interactiva usando todas las técnicas de programación que han aprendido hasta este momento.

Esta evaluación es un buen lugar para que los estudiantes junten todas las piezas que aprendieron (dibujo, variables, sprites, imágenes, condicionales, entrada del usuario) en un solo lugar. Los estudiantes todavía deberían estar trabajando con un código que sea fácil de leer y que no sea difícil de entender para ellos. Darles a los estudiantes la oportunidad de ser realmente creativos después de aprender todos estos nuevos conceptos los ayudará a involucrarse más en el contenido.

Secuencia para el aprendizaje

- Conocimiento inicial (10 min)
- Ampliación del conocimiento (2 días)
- Transferencia del conocimiento (10 min)
- Evaluación

Objetivos

Los estudiantes serán capaces de:

- Usar condicionales para reaccionar a la entrada del teclado o cambios en las variables / propiedades.
- Usar Comandos de secuencia para dibujar en el orden correcto.
- Aplicar un patrón de iterador a las variables o propiedades en un ciclo.

Preparación

- Tarjeta Interactiva - Guía de lecciones

Recursos

¡Atención!
Por favor, haga una copia de cada documento que planee compartir con los estudiantes.

Para los profesores:

- Tarjeta interactiva
- Contenido de la Lección- Code.org

Para los estudiantes:

- Tarjeta Interactiva - Guía de lecciones
- Tarjeta interactiva - Rúbrica
- Tarjeta interactiva - Revisión por pares

Estrategia de aprendizaje

Conocimiento inicial (10 min)

Meta: Los estudiantes ven un ejemplo de un proyecto final y discuten los diferentes elementos que se utilizaron para hacerlo.

Demo Ejemplo de proyecto (Nivel 2)

Pantalla:

En el proyector, ejecute el ejemplo para los estudiantes. Dado que esto está en el nivel 2 del progreso, si es más fácil para los estudiantes hacerlo en sus propias computadoras, también podría hacer eso.

Niveles de Code Studio

- Ejemplo de tarjeta interactiva CSD U3

Proyecto de ejemplo

Ejecute el programa varias veces y haga las siguientes preguntas:

1. ¿Qué elementos parecen usar comandos de dibujo?
2. ¿Qué elementos parecen ser Sprites?
3. Para cada Sprite, ¿qué propiedades se están actualizando?
4. ¿Dónde ves condicionales que se utilizan?
5. ¿Hay elementos que no entiendes?

Discuta:

Haga que los estudiantes compartan sus observaciones y análisis del ejemplar.

Aliente a la clase a considerar que existen múltiples enfoques para programar cualquier cosa, pero que puede haber pistas sobre cómo se creó algo. En particular, cuando comparten sus ideas, pida que especifiquen lo siguiente:

- Pistas que sugieren que se utilizó un Sprite
- Se usaron pistas que sugieren un condicional
- Se usaron pistas que sugieren un patrón de iterador

Pantalla:

Muestre a los estudiantes la tarjeta interactiva - rúbrica. Revise los diferentes componentes de la rúbrica con ellos para asegurarse de que entienden los componentes del proyecto.

Ampliación del conocimiento (2 días)

Meta: Los estudiantes deben planear lo que quieren crear antes de dirigirse a la computadora para que, una vez que lleguen a ella, simplemente estén ejecutando el plan.

Desconectado: planificación interactiva de tarjetas

Distribuir:

Entregue la Tarjeta Interactiva - Guía de lecciones para los estudiantes. Ésta es la herramienta que los estudiantes usarán para determinar sus proyectos antes de ingresar a las computadoras. De tiempo a los estudiantes para que hagan una lluvia de ideas sobre el tipo de tarjeta que quieren crear y quién será el destinatario.

Pasos

1. La primera capa de la tarjeta interactiva es un fondo dibujado sólo con los comandos en el cajón de Dibujo. El anverso de la Guía de lecciones proporciona una cuadrícula para que los estudiantes expongan sus fondos, una tabla de referencia de comandos de dibujo y un área para que los estudiantes tomen notas y escriban pseudocódigos.
2. Los siguientes estudiantes piensan en los Sprites que necesitarán, completando una tabla con la etiqueta, las imágenes y las propiedades de cada Sprite.

3. Finalmente, los estudiantes consideran los condicionales que necesitarán para hacer su tarjeta interactiva.

Niveles: implementación de la tarjeta interactiva (nivel 3 - 7)

Transición:

Una vez que los estudiantes hayan completado su hoja de planificación, es hora de dirigirse a las lecciones de Code studio. La secuencia de nivel corto les pide a los estudiantes completar cada elemento de su proyecto.

Revisión por pares

Distribuir:

Entregue a cada alumno una copia de la Tarjeta interactiva - Revisión por pares.

Los estudiantes deben pasar 15 minutos revisando la tarjeta del otro alumno y completando la guía de revisión por pares.

Iterar - Actualizar código

Recorre el salón:

Los estudiantes deben completar la guía de la revisión por pares y decidir cómo responder a los comentarios que se les dieron. Luego deberían usar ese comentario para mejorar sus cartas.

Reflexionar

Usando la Tarjeta interactiva - Rúbrica, los estudiantes deben evaluar su propio proyecto antes de enviarlo.

Envía a los estudiantes a Code Studio para completar su reflexión sobre sus actitudes hacia la informática. Aunque sus respuestas son anónimas, los datos agregados estarán disponibles una vez que al menos cinco estudiantes hayan completado la encuesta.

Transferencia del conocimiento (10 min)

Compartir tarjetas

Meta: Los estudiantes comparten sus creaciones con la clase.

Compartir:

Encuentre una forma para que los estudiantes compartan sus tarjetas entre ellos y con el destinatario deseado. Es probable que sea útil utilizar el enlace compartido para el proyecto, de modo que los estudiantes puedan compartir el proyecto con otros estudiantes.

Evaluación

Se proporciona una rúbrica para evaluar los proyectos de los estudiantes en los recursos.

Unidad 2

Desarrollo e Implementación de soluciones con TIC

Ideo Digital

Lección 22: Velocidad

Lección en línea Ver en Code Studio

Propósito

Después de una breve revisión de cómo utilizaron el patrón de contador para mover los sprites en las lecciones anteriores, los estudiantes reciben las propiedades que establecen la velocidad y rotación. A medida que utilizan estas nuevas propiedades de diferentes maneras, desarrollan las habilidades necesarias para crear un juego básico de desplazamiento lateral.

En esta Lección, se les enseña a usar los bloques de velocidad para simplificar el código para mover un sprite a través de la pantalla. Esto marca un cambio en cómo se introducen los nuevos bloques. Mientras que los bloques anteriores se presentaron como habilitantes de comportamientos completamente nuevos, ahora se presentan como un código simplificador que los estudiantes podrían haber escrito con los bloques disponibles anteriormente. En las siguientes lecciones, los estudiantes verán cómo este método les permite producir comportamientos de sprites más interesantes.

Secuencia para el aprendizaje

- Conocimiento inicial (15 min)
- Ampliación del conocimiento (75 min)
- Transferencia del conocimiento (5 min)

Objetivos

Los estudiantes serán capaces de:

- Usar los bloques velocidad y rotación para crear y cambiar los movimientos de los sprites.
- Describir las ventajas de simplificar el código mediante el uso de bloques de mayor nivel.

Recursos

¡Atención!
Por favor, haga una copia de cada documento que planee compartir con los estudiantes.

Para los profesores:

- Contenido de la Lección - Code.org

Código

- sprite.rotationSpeed
- sprite.velocityX
- sprite.velocityY

Estrategia de aprendizaje

Conocimiento inicial (15 min)

Demuestre:

Pida que un voluntario pase al frente de la clase y actúe como sprite. Digamos que le dará instrucciones al sprite como si fuera un programa de Game Lab.

Cuando su estudiante esté listo, pídale que se pare frente a la clase. Dé un poco de espacio frente a ellos y pídale que “avance 1”. El voluntario debe dar un paso adelante. Luego repita el comando varias veces, cada vez que espera que el estudiante avance un paso. Debería apuntar a que la repetitividad de estas instrucciones sea clara. Después de que el estudiante haya completado esta Lección, pida que vuelvan a donde comenzaron. Esta vez repita la demostración, pero solicite al alumno que “Avance 2” y haga que el alumno tome 2 pasos cada vez. Una vez que el alumno haya hecho esto varias veces, pida a la clase que dé una ronda de aplausos e invite al estudiante a volver a su asiento.

Preguntar:

Sólo estaba dando instrucciones a mi Sprite, pero parecían ser bastante repetitivas. ¿Cómo podría haber simplificado mis instrucciones?

Discuta:

Da a los estudiantes un minuto para escribir sus pensamientos antes de invitarlos a compartir con un compañero. Luego Haga que la clase comparta sus pensamientos. Puede escribir sus ideas en el pizarrón.

Meta: la demostración anterior debería haber reforzado el hecho de que repetir las mismas instrucciones es algo que nunca harías en la vida real. En su lugar, se le ocurriría una forma de capturar quelas instrucciones deberían repetirse, como “seguir avanzando por 1”

Una forma de simplificar estas instrucciones es decirle a nuestro sprite que siga moviéndose por 1 o 2, o por todos los pasos que queramos. Esto haría las instrucciones más fáciles de entender para los humanos, y como estamos a punto de ver, también hay una manera similar de simplificar nuestro código.

Muestre:

Video de Sprite Velocity

Pantalla:

En el pizarrón, escribe las siguientes piezas de código.

- sprite.velocityX = 4;
- sprite.velocityY = -1;
- sprite.rotationSpeed = 2;

Indicación:

Acabamos de ver cómo estos nuevos bloques nos ayudan a simplificar el patrón de contador que usamos para mover los sprites. En una hoja de papel, anota el patrón de contador que está reemplazando cada una de estas líneas de código.

Comprobar la comprensión: esto es solo una comprobación rápida para entender después del video. Se les pide rápidamente a los estudiantes que vean si entendieron el punto principal del video y, si no, tienen la oportunidad de reforzarlo antes de pasar a los niveles de Code Studio. Deje la traducción delos bloques de velocidad a si patrón decontador asociado en el tablero para hacer referencia a lo largo de la Lección

Discuta:

Haga que los estudiantes escriban sus ideas en una hoja de papel antes de hablar sobre sus respuestas como clase. Finalmente, escriba las respuestas correctas en el tablero al lado de cada una, como se muestra a continuación.

- sprite.x = sprite.x + 4;
- sprite.y = sprite.y - 1;
- sprite.rotation = sprite.rotation + 2;

Estos nuevos bloques de "nivel superior" nos ayudan a escribir código que ya sabíamos cómo escribir. Están simplificando nuestro código para nosotros al ocultar algunos de los detalles innecesarios. Sin embargo, como estamos a punto de ver, estos nuevos bloques cambiarán el tipo de programas y juegos que podemos escribir.

Ampliación del conocimiento (75 min)

Aprendiendo a usar los bloques de velocidad

Transición:

Mover a los estudiantes a Code Studio, Contenido de la Lección -Code.org.

¿Dentro o fuera del lazo de dibujo?: Para los primeros rompecabezas, asegúrese de que los estudiantes estén configurando las velocidades y velocidades de rotación *fuera* del círculo de dibujo, inmediatamente después de que creen sus sprites. El código funcionará para los primeros rompecabezas, incluso si establecenlas velocidades dentro del bucle de extracción, pero causará problemas más adelante.

Recorre el salón:

Estos niveles introducen las propiedades de velocidad X, velocidad Y, velocidad de rotación que acaba de analizar con los estudiantes. Verifique con los estudiantes para ver cómo están y mantenga un registro de cuándo han llegado al final del nivel 10.

VelocidadX

Una forma de mover los sprites en Game Lab es con el patrón de contador. Por ejemplo, sprite1.x = sprite1.x + 1 mueve un sprite por 1 píxel en cada fotograma del ciclo de sorteo. Este patrón es tan común que los sprites tienen una velocityX propiedad que hace esto por ti.

Velocidad de rotación

Ya han aprendido cómo hacer que su sprite gire usando el rotation bloque. Por ejemplo, cuando querían que su sprite girara dos grados cada vez que se dibujaba, colocaban sprite.rotation = sprite.rotation + 2 dentro del círculo de extracción.

Ahora pueden usar los rotationSpeed para hacer que sus sprites giren una cierta cantidad cada vez que se dibujan. Si desea que su sol gire dos grados cada vez que se dibuja, puede usar sun.rotationSpeed = 2 antes del ciclo de extracción, después de crear su sprite.

Control de velocidad

Usaron rotatationSpeed fuera del lazo de extracción para hacer que el sprite girara cuando el programa comienza. También pueden usar rotationSpeed dentro del ciclo de sorteo para cambiar la velocidad del sprite durante el juego. Por ejemplo, un Sprite puede comenzar a girar cuando el usuario presiona la barra espaciadora, y seguirá girando hasta que se le indique que se detenga.

Cambiando la velocidad con la posición

Una ventaja del uso de bloques de velocidad dentro de condicionales (if bloques) es que el sprite se mantiene en movimiento, incluso después de que la condición deje de ser cierta.

Si los estudiantes terminan temprano, anímalos a experimentar con diferentes aspectos del juego de desplazamiento que crearon. Pueden cambiar las animaciones, crear un fondo, etc.

Transferencia del conocimiento (5 min)

Indicación:

Hoy aprendieron algunos bloques nuevos. A primera vista, estos bloques hicieron el mismo tipo de cosas que ya habíamos hecho con el patrón de contador, pero nos simplificaron hacerlas. Sin embargo, al pasar por los rompecabezas, comenzaron a hacer algunos movimientos interesantes que no habíamos podido hacer antes.

- Describa uno de esos movimientos y cómo lo hiciste.
- Describa otro movimiento que te gustaría hacer, pero aún no sabes cómo.
- Describa otro bloque que te gustaría tener.
- ¿Tomaría algún argumento?
- ¿Qué haría?
- ¿Qué código escondería dentro?

Todos los movimientos que hicimos hoy son posibles sin los nuevos bloques, pero sería muy complicado codificarlos. Uno de los beneficios de bloques como la velocidad es que cuando no tenemos que preocuparnos por los detalles de los movimientos y acciones simples, podemos usar esa potencia cerebral adicional para resolver problemas más complicados. A medida que desarrollen su juego de desplazamiento lateral, seguiremos buscando nuevos bloques que simplifiquen las cosas, para que podamos construir juegos cada vez más complicados.

Sugerencias para evaluación

Se sugieren los siguientes indicadores para evaluar formativamente los aprendizajes:

- Utilizan bloques de velocidad y rotación, cambiando el movimiento de los sprites.
- Incorporan el código, los medios y las bibliotecas existentes en los programas originales, y le dan atribución.
- Prueban y perfeccionan sistemáticamente los programas.
- Documentan aquellos programas para que sean más fáciles de seguir, probar y depurar.

Lección 23: Detección de colisión

Lección en línea Ver en Code Studio

Propósito

Los estudiantes aprenden sobre la detección de colisión en la computadora. Trabajando en parejas, exploran cómo una computadora puede usar la ubicación de sprites y propiedades de tamaño y matemática para detectar si dos sprites se están tocando. Luego usan el bloque isTouching para crear diferentes efectos cuando los sprites colisionan, incluidos los sonidos de reproducción. Por último, usan sus nuevas habilidades para mejorar el juego de desplazamiento lateral que comenzaron en la última Lección.

Esta Lección introduce formalmente el uso de abstracciones, formas simples de representar la complejidad subyacente.

En la última Lección, los estudiantes fueron expuestos a la idea de usar un bloque para representar código complejo. Los estudiantes exploran más a fondo esta idea en el contexto del desafío matemático complejo e intencional de determinar si dos sprites se están tocando. Al usar un solo bloque para representar esta complejidad, en este caso, el bloque isTouching, se vuelve mucho más fácil escribir y razonar sobre el código, y los estudiantes pueden apreciar el valor de usar abstracciones. En lecciones posteriores, los estudiantes continuarán construyendo sobre la abstracción isTouching() para crear interacciones de sprites más complejas.

Secuencia para el aprendizaje

- Conocimiento inicial (10 min)
- Ampliación del conocimiento
- Transferencia del conocimiento (5 min)

Objetivos

Los estudiantes serán capaces de:

- Utilizar el bloque isTouching para determinar cuándo se tocan dos sprites.
- Describir cómo las abstracciones ayudan a manejar la complejidad del código.

Preparación

- Detección de colisiones - Guía de lecciones de modo que cada par de estudiantes tenga una parte A y una parte B.

Recursos

¡Atención!
Por favor, haga una copia de cada documento que planee compartir con los estudiantes.

Para los estudiantes:

- Detección de colisiones - contenido de la Lección
- Detección de colisiones - Guía de lecciones

Vocabulario

- **Abstracción:** una representación simplificada de algo más complejo. Las abstracciones le permiten ocultar detalles para ayudarlo a manejar la complejidad, enfocarse en conceptos relevantes y razonar sobre problemas en un nivel superior.
- **Depuración:** encontrar y solucionar problemas en un algoritmo o programa.
- **If-Statement**: la estructura de programación común que implementa “declaraciones condicionales”.

Código

- sprite.isTouching(target)
- sprite.debug

Estrategia de aprendizaje

Conocimiento inicial (10 min)

Revisión:

Recuerde brevemente a los estudiantes el juego de desplazamiento lateral que hicieron en la última Lección y pida que compartan cualquier idea que tengan para mejorar su juego en el futuro.

Mostrar el juego: en este punto,sólo muestre el juego. Tienen una actvidad desconectada bastante importante inmediatamente después y importante superarla antes de interactuar con el juego.

Pantalla:

En un proyector o simplemente en una pantalla de computadora, demuestre el juego que aparece en el primer nivel en Code Studio para esta Lección. El maestro o un alumno puede controlar a la rana para saltar sobre un hongo. Asegúrese de que al menos algunos saltos sean exitosos y que algunos no vean que la detección de colisiones está funcionando.

Inducción:

Una mejora interesante en este juego es que el hongo se mueve cuando la rana lo toca. ¿Pueden pensar en alguna forma en que la computadora pueda usar las propiedades de los sprites para averiguar si se están tocando entre sí?

Discuta:

Permita que los estudiantes hagan una lluvia de ideas sobre cómo la computadora podría determinar si los dos sprites se están tocando. Haga una lista de sus ideas en el pizarrón y diga que tendrán la oportunidad de probar sus teorías en un momento.

Meta: El propósito de esta discusión es solo para obtener algunas ideas en la pizarra para que los estudiantes las utilicen en la próxima Lección. No es necesario evaluarlos o probarlos, porque los estudiantes trabajarán juntos para hacerlo inmediatamente después de la discusión

Ampliación del conocimiento

Colisiones desconectadas

Grupo:

Agrupe a los estudiantes en parejas.

Ahora van a tener la oportunidad de probar las estrategias que surgieron como grupo. Cada guía de lecciones tiene cuatro hojas de papel. Un compañero debe tomar los papeles con la “A” en la parte superior, y el otro debe tomar los papeles con la “B” en la parte superior. Cada uno dibujará dos sprites secretos en el gráfico, y su compañero tratará de averiguar si están tocando o no según la misma información que la computadora tendrá sobre las propiedades de cada uno de los sprites, así que no permitas que el compañero mire lo que el otro está dibujando.

Distribuir:

La Detección de colisiones - Guía de Lecciones para cada conjunto de socios. Asegúrese de que un compañero haya tomado las páginas con “A” en la parte superior y el otro haya tomado las páginas con “B” en la parte superior.

Cada alumno tendrá una línea para dibujar dos cuadrados. El estudiante elige la ubicación y el tamaño de cada uno de los cuadrados, y luego registra la información sobre los cuadrados en una tabla. A continuación, cambian las tablas (no los dibujos) y tratan de determinar si los dos sprites se están tocando en función del ancho de cada sprite y la distancia entre ellos.

La matemática para determinar si los sprites se están tocando es la siguiente:

- Resta las posiciones x (o y) de los sprites para encontrar la distancia entre sus centros.
- Divide el ancho (o la altura) de cada cuadrado por 2 para obtener la distancia desde el centro hasta el borde.
- Si la distancia entre los centros de los sprites es mayor que la suma de las distancias desde sus centros hasta sus bordes, los sprites no se tocan.
- Si la distancia entre los centros de los sprites es igual a la suma de las distancias desde sus centros hasta sus bordes, los sprites apenas se tocan.
- Si la distancia entre los centros de los sprites es mayor que la suma de las distancias desde sus centros hasta sus bordes, los sprites se superponen.

Recorra el aula:

Apoye a los estudiantes mientras completan la hoja de trabajo. Si los estudiantes no están seguros de cómo determinar si los sprites se están tocando, aliéntalos a usar una de las ideas en el pizarrón. Recuerde que no están siendo calificados sobre si tienen razón o no, sino sobre su capacidad para usar el proceso de resolución de problemas. Si alguno de los estudiantes termina antes de tiempo, motívelos a que encuentren un método que funcione para sprites en cualquier lugar de la cuadrícula, no solo en la misma línea.

Compartir:

Después de que todos los estudiantes hayan tenido la oportunidad de probar sus soluciones, pida que compartan lo que descubrieron.

La gente puede usar muchas estrategias diferentes para resolver un problema como éste. Debido a que las computadoras no pueden “ver” los dibujos de la misma manera que las personas, necesitan usar la matemática para determinar si dos cosas se están tocando. Veamos cómo esto puede funcionar a lo largo de una línea. Podemos combinar estos métodos para trabajar en cualquier parte de la pantalla del juego.

Meta: El objetivo de esta discusiónes que los estudiantes vean que las matemáticas en el código son las mismas matemáticas descritas en la Lección desenchufada. El códigocombina las pruebas para x - y en declaraciones “si” anidadas.

IsTouching

Transición:

Haga que la clase inicie sesión en la Unidad 3 - Niveles de Code Studio para la Lección 16.

Pantalla:

Muestre la burbuja 2 y permita que los estudiantes observen y analicen por parejas cómo podrían codificar el comportamiento que ven en el programa.

Niveles de Code Studio

El propósito de este nivel es que los estudiantes vean que pueden probar si dos sprites se están tocando con los bloques que ya están disponibles para ellos. El código es complicado, pero solo usa bloques que los estudiantes ya han aprendido a detectar cuando dos sprites se están tocando.

Discuta:

Pregunte a los estudiantes cómo se relaciona el código a continuación con lo que hicieron en la Lección desconectada.

Aunque este código funciona, puede ser difícil de leer, y sería fácil cometer un error al escribirlo. También tomaría algo de tiempo escribir cada vez, por lo que un programador guardó el código en un bloque que podemos usar para determinar si dos sprites se están tocando, y no necesitamos escribir todo este código. Debido a que esto es algo que un programador quiere hacer una y otra vez, alguien ha creado un bloque

llamado isTouching que ya tiene todas la matemática dentro de él. Ocultar los detalles de cómo se hace algo para facilitar la programación se llama abstracción.

IsTouching

Escribir la matemática cada vez que desees comprobar si dos sprites se están tocando puede llevar un tiempo, por lo que un programador creó el bloque isTouching, que puede comprobar si un sprite está tocando otro sprite (el objetivo). La computadora sigue haciendo los mismos cálculos que en el programa anterior, pero no tiene que preocuparse porque otro programador ya hizo ese trabajo.

Recorre el aula:

Apoye a los estudiantes mientras trabajan en los niveles 3-8. El nivel 8 permite a los estudiantes trabajar en el desplazamiento lateral que crearon en la última Lección. Es muy abierto, así que anime a los estudiantes a ser creativos y tratar de codificar las interacciones de los sprites que no han visto antes. Los estudiantes que completan el nivel temprano pueden continuar agregando nuevas interacciones o mejorando sus juegos de otras maneras.

Transferencia del conocimiento (5 min)

Preguntar:

¿Cuáles fueron algunas de las cosas diferentes que tus sprites hicieron cuando interactuaron?

¿Cuál es un tipo de interacción que te gustaría, pero que aún no has visto?

¿Tienes alguna idea de cómo podrías escribir el código para ese tipo de interacción?

Compartir:

Permita a los estudiantes compartir el diferente tipo de interacciones que les gustaría ver. Hágales saber que aprenderán otros métodos para que los sprites interactúen más adelante.

Sugerencia para evaluación

Se sugiere los siguientes indicadores para evaluar formativamente los aprendizajes:

- Construyen y evalúan estrategias de manera colaborativa al resolver problemas no rutinarios.
- Incorporan el código, los medios y las bibliotecas existentes en los programas originales, para darles atribución.
- Prueban y perfeccionan sistemáticamente los programas.
- Documentan aquellos programas para que sean más fáciles de seguir, probar y depurar.

Lección 24: Movimiento complejo de Sprite

Lección en línea Ver en Code Studio

Propósito

Los estudiantes aprenden a combinar las propiedades de velocidad de los sprites con el patrón de contador para crear movimientos de sprites más complejos. En particular, los estudiantes aprenderán cómo simular la gravedad, hacer un sprite y permitir que un sprite flote hacia la izquierda o hacia la derecha. En los niveles finales de Code Studio, los estudiantes combinan estos movimientos para animar y controlar un solo sprite y crear un juego simple en el que un personaje vuela y recoge una moneda. Se anima a los estudiantes a hacer sus propias adiciones al juego en el nivel final.

Esta Lección muestra la combinación de herramientas, en particular las abstracciones aprendidas en lecciones anteriores, les permite construir nuevos comportamientos para sus sprites. Esto resalta el punto más amplio de que las abstracciones no sólo simplifican el código, sino que también pueden usarse como componentes básicos de un comportamiento aún más complejo.

La Lección presenta algunas de las programaciones más desafiantes. Los estudiantes combinarán múltiples construcciones de programación, incluyendo las propiedades de velocidad, el patrón del contador, la interacción del usuario y la detección de colisiones. Los patrones que los estudiantes usan en esta Lección generalmente son útiles para construir juegos y los estudiantes pueden reutilizarlos más adelante.

Secuencia para el aprendizaje

- Conocimiento inicial (10 min)
- Ampliación del conocimiento (60 min)
- Transferencia del conocimiento (10 min)

Objetivos

Los estudiantes serán capaces de:

- Usar velocidad de sprite con el patrón de contador para crear diferentes tipos de movimiento de sprite.
- Explicar cómo las construcciones de programación individual pueden combinarse para crear un comportamiento más complejo.

Recursos

¡Atención!
Por favor, haga una copia de cada documento que planee compartir con los estudiantes.

Para los profesores:

- Contenido de la Lección - code.org

Estrategia de aprendizaje

Conocimiento inicial (10 min)

Los cuatro conceptos principales que los estudiantes aprenderán en la Lección de hoy son isTouching, keyDown, sprite.velocityX/ sprite.velocityY. Pida a los estudiantes que hablen con un compañero y recuerden mutuamente para qué se usa cada una de estas cuatro construcciones y cómo funcionan. Como clase, defina cada uno según lo conversado.

En las últimas lecciones aprendimos mucho de construcciones de programación poderosas que nos permitieron crear juegos mucho más interesantes. Hoy vamos a explorar cómo la combinación de estas nos dará aún más control sobre los tipos de juegos que podemos hacer.

Ampliación del conocimiento (60 min)

Transición:

Mueva a los estudiantes a Code Studio, Contenido de la Lección - code.org. Con la excepción de la discusión posterior hasta el primer nivel, los estudiantes trabajarán en Game Lab hasta el final de la Lección.

Resumen

La clase aprende a combinar las propiedades de velocidad de los sprites con el patrón de contador para crear un movimiento de sprite más complejo, como simular la gravedad, hacer que un sprite salte y permitir que un sprite flote hacia la izquierda o hacia la derecha. En los niveles finales, la clase combina estos movimientos para animar y controlar un solo sprite y construir un juego simple en el que un personaje vuela y recoge monedas.

Este nivel introduce el nuevo patrón de programación principal de esta Lección, combinando el patrón de contador con las propiedades de velocidad de los sprites. Anime a los estudiantes a tomar en serio sus predicciones antes de ejecutar realmente el código.

Discuta:

Después de que los estudiantes hayan hecho predicciones sobre cómo se ejecutó rápidamente el código en el primer nivel, analicen por qué vieron que el automóvil comenzó a moverse más rápido. Tengan en cuenta que mientras que antes usaban el patrón de contador para aumentar la posición de un automóvil, ahora se usa para aumentar la velocidad del automóvil.

Niveles 2-4:

Estos niveles les dan a los estudiantes la práctica de usar la velocidad dentro del patrón del contador.

Velocidad y el patrón contador

Usar una sprite.velocityX propiedad con el patrón de contador cambiará la velocidad de un sprite durante el programa. Esto hace que el sprite se acelere. Practica un poco usando este patrón tú mismo.

Niveles 5 - 6:

Estos niveles introducen el uso de la velocidad y el patrón del contador para ralentizar un sprite, eventualmente moviéndolo en la dirección opuesta. Esto lleva a la introducción de cómo este patrón podría usarse para simular la gravedad.

Niveles 7 a 9:

Los estudiantes comienzan a trabajar en un juego de volante. En estos niveles usan las construcciones de programación que han aprendido para hacer que su personaje principal se mueva. El personaje responde a gravedad simulada, saltos y flotantes a izquierda y derecha.

Saltando

Aumentar la velocidad de un sprite dentro del patrón de contador puede simular la gravedad. Al agregar interacciones del usuario, puedes hacer que tu sprite parezca que también salta.

Flotante a la derecha

Ahora están usando el patrón de contador con la velocidad Y del sprite para simular la gravedad y saltar. Si usan la velocidad X del sprite en el patrón contrario, pueden hacer que tu sprite flote de lado a lado también.

Niveles 10 a 12:

Los estudiantes agregan una moneda al juego para que su personaje la coleccione. En el último nivel, se les anima a actualizar el juego ellos mismos. Aproveche esta oportunidad en particular para alentar a los estudiantes a usar otros patrones de programación que hayan aprendido, por ejemplo, crear un marcador.

Transferencia del conocimiento (10 min)

Compartir:

Pida a los estudiantes que compartan con sus compañeros de clase qué adiciones hicieron a su último juego. Haga que los estudiantes se centren no sólo en cómo funciona el juego, sino también en cómo se ve el código para crear ese tipo de funcionalidad.

Preguntar:

En su papel, hagan dos listas. Primero una lista de cosas nuevas que puedan programar después de la Lección de hoy. En la segunda lista anoten todos los nuevos bloques que aprendieron hoy.

Discuta:

Haga que los estudiantes compartan sus listas con sus compañeros de clase. Deberían haber enumerado muchos nuevos movimientos de sprites pero los estudiantes no han aprendido ningún bloque nuevo en esta Lección.

Objetivo: Esta conversación debe resaltar que los estudiantes no aprendieron ningún bloque nuevo en la Lección de hoy, sino que aprendieron nuevas formas de combinar bloques y patrones que habían aprendido previamente. El punto más amplio aquí es que la programación no siempre se trata de aprender nuevos bloques, sino de ser creativos al combinar las herramientas que ya sabes usar en el lenguaje.

Meta: Reforzar el hecho de que aprender a programar no es solo memorizar bloques. Ser creativo con laprogramación a menudo significa idear maneras inteligentes de combinar los comandos y patrones que ya sabe usar.

Después de comentar que todos los movimientos nuevos que crearon hoy se hicieron combinando bloques y patrones que ya aprendieron, pregúnteles a los estudiantes qué piensan que les dice esto respecto de cómo los programadores desarrollan el código.

Indicación:

Hoy hemos construido muchos movimientos de sprites nuevos, como la gravedad y el salto, pero nada de esto nos obligó a aprender nuevos bloques. ¿Creen que aprender a programar siempre significa aprender nuevos comandos?

Discuta:

Lidere un seguimiento rápido de su discusión inicial sobre este punto.

Vamos a seguir aprendiendo algunas herramientas más en Game Lab. Para crear nuevos tipos de programas, no siempre es necesario aprender nuevos bloques. La mayoría de las veces, la creatividad de la programación proviene de aprender a combinar cosas que ya conoces de maneras nuevas y creativas.

Sugerencias para evaluación

Se sugieren los siguientes indicadores para evaluar formativamente los aprendizajes:

- Crean nuevos tipos de programas, combinando elementos que ya conocen de manera nueva y creativa.
- Perfeccionan el trabajo involucrando modificaciones y testeo.

Lección 25: Colisiones

Lección en línea ver en Code Studio

Propósito

Los estudiantes programan sus sprites para interactuar de nuevas maneras. Después de una breve revisión de cómo usaron el bloque isTouching, los estudiantes intercambian ideas sobre otras formas en que dos sprites podrían interactuar. A continuación, utilizan isTouching para hacer que un sprite empuje al otro a través de la pantalla antes de practicar con los cuatro bloques de colisión (collide, displace, bounce, y bounceOff).

Esta Lección introduce colisiones, otra abstracción útil que permitirá a los estudiantes manipular sus sprites de maneras completamente nuevas. Si bien los estudiantes teóricamente pueden haber escrito sus propios comandos de desplazamiento, colisión o rebote, la capacidad de ignorar los detalles de este código les permite centrar su atención en la estructura de alto nivel de los juegos que desean construir.

Esta Lección también pretende que los estudiantes practiquen utilizando los nuevos comandos que han aprendido. En realidad, es la última vez que aprenderán un nuevo comportamiento de sprite, y después de esta Lección, los estudiantes pasarán a centrarse más en cómo organizan su código cada vez más complejo.

Secuencia para el aprendizaje

- Conocimiento inicial (5 min)
- Ampliación del conocimiento (40 min)
- Transferencia del conocimiento (10 min)

Objetivos

Los estudiantes serán capaces de:

- Utilizar los bloques “displace”, “collide”, “rebunce”, y “bounceOff” para producir interacciones de sprites
- Describir cómo se pueden construir las abstracciones para desarrollar aún más abstracciones.

Vocabulario

Abstracción: una representación simplificada de algo más complejo. Las abstracciones le permiten ocultar detalles para ayudarlo a manejar la complejidad, enfocarse en conceptos relevantes y razonar sobre problemas en un nivel superior.

Código

- sprite.bounce(target)
- sprite.bounceOff(target)
- sprite.collide(target)
- sprite.displace(target)
- setCollider(type, xOffset, yOffset, width/radius, height, rotationOffset)
- sprite.bounciness

Estrategia de aprendizaje

Conocimiento inicial (5 min)

Pantalla:

Si tiene la capacidad, proyecte la animación en el primer nivel en Code Studio para esta Lección. De lo contrario, pida a las parejas de estudiantes que lo vean juntos. Muestre dos sprites, uno moviéndose a través de la pantalla hacia el otro y finalmente empujando uno cuando colisionan.

Resumen:

La clase programa sus sprites para interactuar de nuevas maneras. Después de una breve revisión de cómo usaron el bloque isTouching, la clase intercambia ideas sobre otras formas en que dos sprites podrían interactuar. Luego usan isTouching para hacer que un sprite presione otro en la pantalla, antes de practicar con los cuatro bloques de colisión (colisionar, desplazar, rebotar y rebotar).

Indicación:

Usando los bloques que ya sabemos cómo usar, ¿cómo podríamos crear la interacción de sprites que podemos ver en este programa?

Meta: El objetivo de esta discusión es que los estudiantes piensen en formasde resolver el problema de tener un elemento sprite empujando otro enla pantalla. No es necesario que los estudiantes lleguen a un consenso, ya que cada uno de ellos tendrá la oportunidad de probar una solución en el siguiente nivel en Code Studio. Los estudiantes deben entender quees posible usar bloques para producir el movimiento deseado solo con los bloques que ya han aprendido.

Tenemos muchas ideas geniales sobre cómo podríamos hacer que un sprite empuje otro en la pantalla. Ahora que ya se han preparado, puedes probar sus ideas en Code Studio. Una gran parte del problema es averiguar cuándo se tocan los dos sprites, pero como ya hemos descubierto cómo hacerlo y ahora podemos usar el bloque isTouching, ya no necesitamos pensarlo. Podemos enfocarnos en la nueva parte del problema.

Ampliación del conocimiento

Niveles de Code Studio

Niveles 2-4:

En estos niveles, se muestra a los estudiantes una interacción de sprites. Luego implementan sus ideas para crear la interacción de los sprites que observaron. La primera vez pueden implementar las ideas del grupo. La segunda vez desafíalos a implementar ese comportamiento de forma independiente.

Interacciones de Sprite

Hasta ahora han sido capaces de crear interacciones de sprites simples usando el sprite. bloque isTouching(). Por ejemplo, han reestablecido una moneda a una ubicación diferente en la pantalla cuando un personaje la toca. Ahora es el momento de comenzar a hacer que los sprites tengan interacciones más complejas.

Hacer esto

- Ejecuten el programa y observen la interacción entre los dos sprites.
- Discutan con un compañero: usando sólo los comandos que ya conocen, ¿cómo podrían crear este tipo de interacción? Hay muchas maneras de hacerlo, pero aquí hay algunos bloques a considerar:
- sprite.isTouching
- sprite.velocityX
- sprite.velocityY
- sprite.x
- sprite.y

Prepárense para compartir ideas con los compañeros de clase.

Inducción:

Éste fue un problema desafiante, pero pudimos resolverlo.

¿Qué nos ayudó a resolver este problema?

Todas estas cosas son muy importantes y surgen mucho en Informática. Una cosa que fue particularmente útil fue el isTouching bloque, que ocultó el código complicado que nos dice si los dos sprites se están tocando. También hay un bloque (displace) que oculta el código que acabamos de escribir, y algunos otros bloques que ocultan el código para otros tipos de interacciones de sprites. Tendrás la oportunidad de probar estos bloques en los siguientes niveles y utilizarlos para mejorar tu juego de vuelo.

Niveles 5-10:

Estos niveles introducen cómo usar los 4 nuevos bloques de colisión que los estudiantes aprenderán en esta Lección.

Desplazar

sprite.displace hará que un sprite empuje al otro cuando se toquen.

Tipos de colisión

Hay cuatro tipos de colisiones que usamos en Game Lab. Estos bloques causarán un cierto tipo de interacción entre el sprite y su objetivo.

Desplazar

El bloque displace hace que el sprite empuje al objetivo siempre que se toquen entre sí.

El sprite se mueve normalmente.

Chocar

El bloque collide hace que el sprite se detenga cuando se encuentra con el objetivo. Si el objetivo se está moviendo, empujará al objeto con él. El objetivo sigue moviéndose normalmente.

Este es un buen momento para decir cuánto han progresado los estudiantes en sus habilidades desde el comienzo de la unidad. Este problema habría parecido casi imposible a principios de año. Algunas cosas que hicieron que elproblema fuera más fácil de resolver fueron:

- Preparación: Los estudiantes intercambiaron ideas y pensaron en soluciones antes de probar sucódgo
- Cooperación: Los estudiantes trabajaron en grupos para llegar auna solución
- Abstracción: los estudiantes pudieron usar los bloques **isTouching** y **velocityY** para ocultar parte de la complejidad de la solución

Rebotar

El bloque bounce hace que el sprite y el objetivo reboten cuando se tocan entre sí. Tanto el sprite como el objetivo cambian cómo se están moviendo. El bloque bounceOff hace que el sprite rebote en el objetivo. El objetivo sigue moviéndose normalmente.

Depurar

Hay una sprite.debug propiedad especial que puede usar para comprender mejor por qué los sprites interactúan de la manera en que lo hacen.

SetCollider

Los Sprites interactúan en función del tamaño y la forma de su colisionador, no de las imágenes que se les asignan. Solo se puede ver el colisionador cuando se activa el modo de depuración. Puede cambiar la forma del colisionador usando el sprite.setCollider() bloque, que le permite elegir entre un “rectángulo” o un “círculo”. Por defecto, todos los colisionadores son “rectangulares”.

Niveles 11-15:

Estos niveles guían a los estudiantes al agregar colisiones al juego que comenzaron en la Lección anterior, y eventualmente invitan a los estudiantes a agregar sus propias modificaciones al juego.

Bounciness

Hasta el momento, bounceOff ha hecho que los sprites se alejen de otros objetos tan rápido como rebotaban en ellos. En el mundo real, casi todo se ralentiza un poco cuando rebota en otra cosa. Puede usar el bounciness bloque para decirle a tu sprite cuánto desacelerar o acelerar cuando rebota en otra cosa.

Transferencia del conocimiento (10 min)

Compartir y publicar 3-2-1

De a los estudiantes tiempo para jugar los juegos de los demás. Pida que se centren no sólo en el nuevo comportamiento que agregaron sino también en el código que usaron para crearlo.

Revisar:

Pida a los estudiantes que escriban y reflexionen sobre las siguientes indicaciones.

- ¿Tres cosas que viste en el juego de otra persona que realmente te gustaron?
- ¿Cuáles son dos mejoras que harías en tu juego si tuvieras más tiempo?
- ¿Cuál es un bloque que te gustaría tener en Game Lab y cómo funcionaría?

Sugerencias para evaluación

Se sugiere el siguiente indicador para evaluar formativamente los aprendizajes:

- Utilizan bloques para programar en base un diseño predictivo
- Explican la forma en que diseñan su programa.

Lección 26: Funciones

Lección en línea Ver en Code Studio

Propósito

Los estudiantes aprenden cómo crear funciones para organizar su código, hacerlo más legible y eliminar bloques repetidos de código. Un calentamiento desconectado explora cómo las direcciones en diferentes niveles de detalle pueden ser útiles dependiendo del contexto. Los estudiantes aprenden que los pasos de mayor nivel o más abstractos facilitan la comprensión y razonamiento de los pasos. Luego, los estudiantes aprenden a crear funciones en Game Lab. Usarán funciones para eliminar bloques largos de código y para reemplazar piezas repetidas de código con una sola función. Al final de la Lección, los estudiantes usan estas habilidades para organizar y agregar funcionalidad a la versión final de su juego de desplazamiento lateral.

En las primeras cuatro lecciones, los estudiantes han aprendido a utilizar una serie de abstracciones en sus programas, que incluyen las propiedades de velocidad, el toque y las colisiones. Estas abstracciones les han permitido crear programas mucho más complejos sin tener en cuenta los detalles de cómo se crea ese comportamiento. En esta Lección, los estudiantes aprenden a crear abstracciones propias mediante la creación de funciones.

Los estudiantes utilizarán principalmente las funciones para dividir el código en fragmentos lógicos que son más fáciles de razonar. Esto forma parte de la transición de la construcción de habilidades técnicas a los procesos organizacionales utilizados para desarrollar software.

Secuencia para el aprendizaje

- Conocimiento inicial (10 min)
- Ampliación del conocimiento (60 min)
- Transferencia del conocimiento (10 min)

Objetivos

Los estudiantes serán capaces de:

- Crear y usar funciones para bloques de código que realizan una sola tarea de alto nivel dentro de un programa
- Crear y usar funciones para eliminar bloques repetidos de código de sus programas
- Crear y usar funciones para mejorar la legibilidad de sus programas
- Explicar cómo las abstracciones permiten a los programadores razonar sobre un programa en un nivel superior.

Vocabulario

Función: un grupo nombrado de instrucciones de programación. Las funciones son abstracciones reutilizables que reducen la complejidad de escribir y mantener programas.

Código

- Define a function
- Call a function

Estrategia de aprendizaje

Conocimiento inicial (10 min)

Objetivo: Esta conversación demuestra a los estudiantes que a menudo usan un nombre o descripción de alto nivel para un comportamiento mucho más complejo. Está motivando el valor de agrupar o combinar muchos pasos más pequeños con un nombre más grande y proporciona alguna de las justificaciones para crear funciones dentro de los programas. A saber, los pasos de alto nivel hacen que sea másfácil de entender y razonar sobre un proceso.

Instrucciones de alto nivel vs. Nivel bajo

Preguntar:

Imaginen que necesitan escribir en un conjunto de instrucciones de 5 pasos para pasar la mañana. ¿Qué serían? Escriban sus pasos y prepárense para compartir.

Discuta:

El alumno debe escribir las instrucciones de 5 pasos para la mañana y compartir con un compañero. Pida a un par de estudiantes que compartan con la clase.

Inducción:

Este conjunto de instrucciones es bastante fácil de seguir y comprender. Están en el nivel en el que podrían pensar en describir su día a un amigo. Ahora vamos a un nivel más profundo. Elijan uno de tus 5 pasos y divídelo en 5 pasos más pequeños que necesitan para completar esa tarea más grande. Prepárense para compartir las ideas de nuevo.

Discuta:

Los estudiantes deben compartir sus pasos más pequeños. Nuevamente, solicite a algunos voluntarios que luego compartan su paso original y cómo lo separaron.

Esto se está poniendo interesante. Parece que la primera vez que dimos nuestros pasos, estábamos “ocultando” algunos de los detalles necesarios para completar la tarea. Probemos esto una vez más. Tomen uno de sus 5 pasos más pequeños y divídanlo de nuevo en 5 pasos aún más pequeños.

Pida a los estudiantes que compartan sus pasos una vez más. Pida nuevamente a algunos voluntarios que compartan cómo dividieron uno de sus pasos en pasos aún más pequeños.

Indicación:

Imaginen que dividimos cada uno de los primeros 5 pasos en 5, y luego dividimos todos esos pasos nuevamente. Esto significa que tendríamos un conjunto de instrucciones de alto nivel, y en la parte inferior un conjunto de instrucciones realmente bajo o detallado. Prepárense para responder a las siguientes preguntas.

- ¿Cuándo sería más apropiado el conjunto de pasos de alto nivel que acabas de escribir?
- ¿Cuándo sería más apropiado el conjunto de pasos de nivel más bajo?
- ¿Cuál es más fácil de razonar o entender?

Discutir:

Pida a los estudiantes que compartan sus pensamientos y opiniones. Después de un par de minutos mueva la conversación a la transición.

Proceso de desglose: El orden en el que están desglosando una tarea más grande aquí también reflejacómo se les pedirá que escribancódigo con funciones. A menudo, los programadores primero escriben el nombre de la función que pretenden escribir en función de lo que debe de hacer antes de entrar y escribir losdetalles.

A veces los detalles son importantes, pero a menudo los pasos de alto nivel son mucho más fáciles de razonar y dejan claro lo que está sucediendo. Hemos aprendido que bloques como velocityX o isTouching en realidad solo contienen código que podríamos haber escrito nosotros mismos. Usar estos comandos, o abstracciones, es realmente útil ya que podemos pensar en el código a un alto nivel. Hoy aprenderemos cómo agrupar muchos comandos para crear un bloque nuevo. En la programación cuando creamos un nuevo bloque como este, lo llamamos función.

Ampliación del conocimiento (60 min)

Transición:

Mueva a los estudiantes a Code Studio, contenido de la Lección donde aprenderán a crear funciones.

Esta Lección cubre las funciones como una forma de organizar su código, hacerlo más legible y eliminar bloques repetidos de código. La clase aprende que los pasos de mayor nivel o más abstractos facilitan la comprensión y razonamiento de los pasos, luego comienza a crear funciones en Game Lab. Al final de la Lección, la clase usa estas habilidades para organizar y agregar funcionalidad a la versión final de su juego de desplazamiento lateral.

Introduciendo funciones

En estas primeras lecciones a los estudiantes simplemente se les muestra la sintaxis de las funciones, aún no se les pide que escriban o creen las suyas propias. Puede ser útil explicar la creación de una función como básicamente "crea un nuevo bloque" al igual que otro programador creó losbloques "isTouching" o "velocity" quehan visto realmente contienen otrocódigo más complejo.

Vocabulario

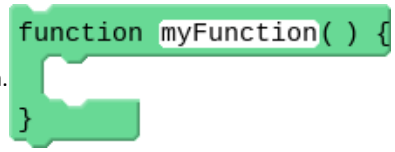
- **Función**: una pieza de código que se puede llamar fácilmente una y otra vez.

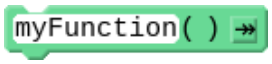
Funciones de llamada

Las funciones permiten construir sus propios bloques y decidir qué código va dentro de ellos. Este es el comando que le permite crear una nueva función.

Usa o "llama" su función como cualquier otro bloque usando el nombre que le dio.

Los bloques para crear y llamar a funciones se pueden encontrar en la pestaña "Funciones" de la paleta.





Reordenando el código

Colocar el código dentro de las funciones facilita la lectura y la realización de cambios. Pueden llamar a las funciones en un orden diferente para realizar rápidamente cambios significativos en la ejecución de su programa.

Funciones de llamada en Draw Loop

Pueden llamar a una función dentro del ciclo de extracción, tal como lo harían en cualquier otro lugar de su código.

Llamar a las funciones varias veces

A menudo querrán usar el mismo código en muchos lugares de su programa. Una vez que hayas creado una función, puede llamarla tantas veces como desee.

Por qué usar las funciones

Este y los siguientes dos nivelespresentan tres usos de funciones, a saber, eliminar la repetición en losprogramas, permitir que el código se cambie rápidamente en varios puntos yproporcionar organización en el código. Los estudiantes necesitarán escribir más de sus propias funciones en estos niveles.

Hacer cambios en las funciones

Una buena ventaja de utilizar funciones para eliminar código repetido es que ahora puedes realizar fácilmente cambios en múltiples lugares en su código. Simplemente cambie la forma en que creas las funciones, y su programa ahora usará el nuevo código en todas partes donde se llame su función.

Transferencia del conocimiento (10 min)

Pregunte:

¿Por qué diríamos que las funciones nos permiten “crear nuestros propios bloques”?

¿Por qué es esto algo que quisiéramos hacer?

Discuta: Haga que los estudiantes discutan en su mesa antes de hablar como clase.

Preguntar: Escriba su propia definición de abstracción ¿Por qué una función cuenta como una abstracción?

Discuta: Haga que los estudiantes discutan en su mesa antes de hablar como clase.

Preguntar: Escribe su propia definición de abstracción ¿Por qué una función cuenta como una abstracción?

Discuta: Haga que los estudiantes discutan en su mesa antes de hablar como clase.

Las funciones son una herramienta útil para ayudarnos a escribir y organizar fragmentos de código más complejos. Poder mantener su código organizado será una habilidad importante.

Meta: Use este primer mensaje para realizar lo que los estudiantes aprendieron hoy. Cuando crean una función, crean su propio bloque que pueden llamar o usar cuando lo deseen. Vieron al menos dos motivaciones principales para crear funciones hoy en día, incluidas.

Simplificando el código dividiendo en fragmentos lógicamente nombrados
Evitando el código repetido haciendo un bloque, puedes usarlo varias veces

Sugerencias para evaluación

Se sugiere el siguiente indicador para evaluar formativamente los aprendizajes:

- Agrupan la información en unidades más pequeñas
- Explican cómo las funciones hacen el trabajo más eficiente

Lección 27: El proceso de diseño de juego

Lección en línea Ver en Code Studio

Propósito

Esta Lección presenta a los estudiantes el proceso que usarán para diseñar juegos. Este proceso se centra en una guía de proyecto que les pide a los estudiantes que definan sus sprites, variables y funciones antes de comenzar a programar su juego. En esta Lección, los estudiantes comienzan jugando un juego en Game Lab donde el código está oculto. Discutan qué creen que deberían hacer los sprites, las variables y las funciones para hacer el juego. Luego reciben una guía de proyecto completa que muestra una forma de implementar el juego. A continuación, los estudiantes pasan por este proceso a través de una serie de niveles. Como parte de esta Lección, los estudiantes también aprenden brevemente a utilizar animaciones de cuadros múltiples en Game Lab. Al final de la Lección, los estudiantes tienen la oportunidad de hacer mejoras en el juego para hacerlo suyo.

Esta Lección presenta animaciones de cuadros múltiples y es la primera de una secuencia centrada en el proceso de creación de software.

Si bien las lecciones anteriores se centraron en el uso de abstracciones para gestionar la complejidad del código, esta Lección se centra en la gestión de la complejidad del proceso de desarrollo de software. La Lección entrega en gran medida el proceso de desarrollo de software al proporcionar a los estudiantes una guía de proyecto completa, proporcionando código de inicio, y encaminando a los estudiantes a través de su implementación. En las lecciones posteriores, los estudiantes deberán completar una mayor parte de esta guía de forma independiente, y para el proyecto final, seguirán este proceso.

Secuencia para el aprendizaje

- Conocimiento inicial (15 min)
- Ampliación del conocimiento (60 min)
- Transferencia del conocimiento (20 min)

Objetivos

Los estudiantes serán capaces de:

- Identificar las construcciones de programación central necesarias para construir diferentes componentes de un juego.
- Crear y usar animaciones de multitarea en un programa.
- Implementar diferentes características de un programa siguiendo una guía de proyecto estructurado.

Preparación

- Contenido de la Lección - code.org

Recursos

¡Atención!
Por favor, haga una copia de cada documento que planee compartir con los estudiantes.

Para los estudiantes:

- Defender Game - Guía de proyectos

Estrategia de aprendizaje

Conocimiento inicial (15 min)

Jugar Cake Defender

Esta Lección comienza inmediatamente en Code Studio Contenido de la Lección - code.org. En el primer nivel, encontrarán un juego, pero no podrán ver el código. Deben jugar el juego y seguir las instrucciones que les piden que enumeren las variables, los sprites y las funciones que creen que son necesarios para crear este juego.

¡Defiende tu Torta!

Es un ejemplo de un juego que construirás al final de esta Lección.

Detener: revisar la guía de proyectos

Los estudiantes deberán crear individualmente una lista de variables, sprites y funciones que crean que se utilizan en el juego. Pida a los estudiantes que compartan sus listas con un compañero antes de hablar como clase.

Distribuir:

Entregue a cada alumno o pareja de estudiantes una copia del Juego Defender - Guía de proyectos

Preguntar:

Compare los componentes del juego que creían que se incluirían con los de esta guía de proyecto. ¿Notan alguna diferencia?

Discuta:

Como clase, compara la lista que tenías en el pizarrón con la lista de variables, sprites y funciones en la guía del proyecto. Tenga en cuenta las similitudes. Donde hay diferencias que intenten entender por qué. Evite “correcto” o “incorrecto”.

Por lo general, hay muchas maneras de estructurar un programa para que funcione de la manera que desea. Lo importante al escribir programas complejos o grandes, es que comienzan con un plan. Hoy vamos a ver cómo podemos implementar este plan para construir nuestro propio juego. Al final de la Lección, no sólo habrán construido un juego, sino que sabrán cómo cambiarlo y hacerlo propio. ¡Vamos!

Mantener el enfoque: los estudiantespueden distraerse fácilmente con la diversión de jugar el juego. Déjelos jugar por un tiempo, pero eventualmente anímelos a seguir las instrucciones en pantalla y haga una lista de las variables, los sprites y las funciones quesería necesarias para jugar el juego.

Ejecución de la conversación: puede escribir “variables”, “sprites” y “funciones” en la pizarra y registrar sus ideas debajode cada una. Pídales a los estudiantesque justifiquen sus decisiones, pero nosientan la necesidad de conformarse con una sola respuesta correcta

Guía del proyecto: la guía del proyecto se completa de manera exhaustiva para los estudiantes, de modo que pueda experimentar su uso como referencia cuando programe. Esto debería darlesmás contexto al completar su propia guía de proyectos en las próximas dos lecciones.

Puede dar a cada uno su propia copia como referencia, pero también puede optar por imprimir una copia por par, compartir copias digitales o simplemente mostrar la guía en el proyector. Siempre que esté disponible para referencia, cualquier enfoque funcionará bien.

Ampliación del conocimiento (60 min)

Animaciones de multitrama

Transición:

Los estudiantes deben regresar a Code Studio. Antes de implementar realmente el plan, los estudiantes deberán revisar rápidamente una nueva habilidad, cómo usar animaciones de multitrama. Los estudiantes aprenderán rápidamente a crear, modificar y cambiar el nombre de las animaciones, así como a elegirlas de la biblioteca.

Detener

Antes de seguir adelante, deberán consultar la Guía del proyecto.

Implementar la guía del proyecto

Los estudiantes reciben una gran cantidad de código de inicio en este proyecto. Los sprites, las variables y las funciones ya se les han asignado. El trabajo de este proyecto es escribir el código para las funciones individuales. Estos niveles guían a los estudiantes a través de cómo implementar esas funciones. A medida que los estudiantes avanzan por los niveles, señalan cómo se está utilizando la guía del proyecto.

Las habilidades más desafiantes que usan los estudiantes en estos niveles es reconocer la necesidad de crear nuevas funciones para reemplazar el código repetido. Los estudiantes necesitan desarrollar esta habilidad por su cuenta, pero estos niveles demuestran una instancia donde esto podría suceder.

Realizar los niveles de Code Studio correspondiente a la Lección.

Transferencia del conocimiento (20 min)

Hagalo propio

Este último nivel alienta a los estudiantes a hacer suyo el juego. Si los estudiantes han llegado a este punto, tienen todas las habilidades que necesitan para progresar a través del plan de estudios, por lo que no hay presión para completar ninguna de las modificaciones sugeridas en este nivel. Sin embargo, si tienen tiempo, obtener una práctica de planificación e implementar nuevas características será una habilidad útil. Incluso modificar las animaciones del juego es una forma sencilla de que los estudiantes puedan apropiarse del juego.

Compartir:

Una vez que los estudiantes han completado el proyecto, pueden compartir su trabajo con sus compañeros de clase. Alienta a los estudiantes a exhibir el código adicional que escribieron y explica cómo ha cambiado la forma en que funciona el juego.

Sugerencias para evaluar

Se sugiere los siguientes indicadores para evaluar formativamente los aprendizajes:

- Elaboran una guía de programa
- Explican la finalidad de las diferentes estructuras utilizadas
- Colaboran de forma grupal en el desarrollo del diseño del juego

Lección 28: Uso del proceso de diseño de juego

Lección en línea Ver en Code Studio

Propósito

En esta Lección de varios días, los estudiantes usan el proceso de resolución de problemas para crear un juego de salto de plataforma. Comienzan por mirar un ejemplo de un saltador de plataforma y luego definen cómo se verán sus juegos. A continuación, utilizan un proceso estructurado para planificar los fondos, las variables, los sprites y las funciones que necesitarán para implementar su juego. Después de escribir el código del juego, los estudiantes reflexionarán sobre cómo se puede mejorar el juego e implementarán esos cambios.

Los estudiantes ya han aprendido todos los constructos de programación que necesitan para hacer un juego. Esta Lección repasa muchos de esos conceptos, al tiempo que los presenta a un proceso estructurado que los ayudará a administrar el trabajo. Se basa en el uso de la Guía del proyecto en la Lección anterior al hacer que los estudiantes completen más de esta guía de proyecto de forma independiente antes de usarla para construir un juego. Esta Lección prepara a los estudiantes a escribir su propio juego desde cero para el proyecto final.

Secuencia para el aprendizaje

- Conocimiento inicial (10 min)
- Ampliación del conocimiento (50 min)
- Transferencia del conocimiento (5 min)

Objetivos

Los estudiantes serán capaces de:

- Identificar las construcciones de programación central necesarias para construir diferentes componentes de un juego.
- Implementar diferentes características de un programa siguiendo una guía de proyecto estructurado.

Preparación

- Planificación de su plataforma de juego

Recursos

¡Atención!
Por favor, haga una copia de cada documento que planee compartir con los estudiantes.

Para los profesores:

- Planificación de su plataforma de juego - contenido code.org
- Para los estudiantes:
- Planificación de su plataforma de juego - Guía de proyectos (anexo)

Estrategia de aprendizaje

Conocimiento inicial (10 min)

Introducción

El proceso de resolución de problemas nos ayuda a resolver todo tipo de problemas. Piensa en el problema de construir una pieza de software más grande, como el juego que construimos en la última Lección. ¿Cómo se veía cada uno de los 4 pasos? ¿Por qué fueron importantes?

Discuta:

Los estudiantes deben intercambiar ideas en voz baja y anotar qué puede ser cada paso. Después, dirige una discusión de compartir. Pueden grabar ideas en el tablero. Las posibles partes de cada paso incluyen:

- Definir: Averiguar cómo quieren que se vea el juego, cómo debería funcionar y quién lo jugará.
- Preparar: Planifiquen cómo se verá su código. Decidan una estructura para el juego.
- Intentar: Escribir el código que sigue al plan.
- Reflexionar: probar el código, jueguen el juego para asegurarse de que funciona, reciban comentarios de otras personas para mejorar el juego.

Meta: Los estudiantes deben compartir sus pensamientos, pero si no surgen naturalmente, sugiérales los ejemplos proporcionados. Esta discusión motivará el uso de la guía del proyectopara construir un juego más adelante en la Lección.

Cuando construyen un software, el proceso de resolución de problemas puede ser una guía útil. Obviamente, necesitamos escribir el código, pero tener cuidado de definir lo que queremos construir, hacer un buen plan para construirlo y reflexionar luego sobre cómo mejorarlo son parte de la creación de un buen software. Hoy vamos a usar este proceso para hacer un nuevo juego.

Ampliación del conocimiento

Jugar Alien Jumper

Distribuir:

Entregue a cada alumno una copia de la planificación de su **Guía de plataforma de juego**

Vamos a construir un juego de saltos hoy. Tendrán la oportunidad de jugar un juego de muestra y luego planificar cómo desean crear el juego en la Guía de proyectos.

Mueva a los estudiantes a Code Studio. En el primer nivel, encontrarán un juego, pero no podrán ver el código. Deben jugar el juego y seguir las instrucciones que les piden que enumeren las variables, los sprites y las funciones que creen que son necesarios para crear este juego.

Discutir la guía de proyectos

Recorra el salón:

Los estudiantes deben completar la guía del proyecto en el estilo de la que vieron en la Lección anterior. Es probable que quieran mantener el juego mientras intentan determinar el comportamiento que tendrán cada uno de los sprites.

Compartir:

Los estudiantes comparten sus planes para hacer el juego. Asegúrese que hay muchas maneras correctas de crear la misma pieza de software, y que tendrán la oportunidad de probar sus ideas en Code Studio.

- Los estudiantes trabajan en parejas para crear el juego.
- Los estudiantes tienen la oportunidad de mejorar su juego después de estar expuestos a otras dos versiones de un saltador de plataforma.
- Los estudiantes comparten sus juegos con sus compañeros de clase.

Hacer el juego tomará al menos dos periodos de clase. Si no hay suficientetiempo para que todos los estudiantesterminen la clase, los grupos de estudiantes pueden trabajar para codificar diferentes aspectos y luego compartir el código entre ellos. Por ejemplo, un grupo podría trabajar en las plataformas, uno en las estrellas y otro en el jugador. Los estudiantes que terminaron temprano pueden elegir más desafíos de los niveles posteriores.

Reflexión

Preguntar:

Pida a los estudiantes que reflexionen sobre el desarrollo de las cinco prácticas de Descubrimientos CS (Resolución de problemas, Persistencia, Creatividad, Colaboración, Comunicación). Elija una de las siguientes indicaciones según lo considere apropiado.

- Elija una de las cinco prácticas en las que cree que demostró crecimiento en esta Lección. Escribe algo que hiciste que ejemplifica esta práctica.
- Elija una práctica en la que piense que puede seguir creciendo. ¿Qué le gustaría mejorar?
- Elija una práctica que pensó que era especialmente importante para la Lección que completamos hoy. ¿Qué lo hizo tan importante?

Sugerencias para evaluación

Se sugiere los siguientes indicadores para evaluar formativamente los aprendizajes:

- Implementan un diseño siguiendo la guía del proyecto
- Explican la finalidad de las diferentes estructuras utilizadas
- Colaboran de forma grupal en el desarrollo del diseño del juego

Lección 29: IA con Google Teachable Machine I

Lección con conexión

Propósito

En esta Lección básica los estudiantes aprenderán los principios de la inteligencia artificial a partir del entrenamiento de modelos en Google Teachable Machine. La inteligencia artificial tiene 3 componentes claves: datos de entrenamiento, el modelo y las predicciones.

En las dos sesiones que dura este proyecto introductorio, los estudiantes experimentarán con Google Teachable machine, desarrollarán su propio modelo de inteligencia artificial y documentarán el proceso, experimentando de primera mano y de forma práctica el funcionamiento de los sistemas de inteligencia artificial y su uso.

En la primera sesión los estudiantes se familiarizarán con los conceptos básicos de la inteligencia artificial y con la plataforma Teachable Machine, de forma que, en la segunda sesión, puedan desarrollar un proyecto por cuenta propia y documentarlo.

Secuencia para el aprendizaje

- Conocimiento inicial (5 min)
- Ampliación del conocimiento (30 min)
- Transferencia del conocimiento (10 min)
- Evaluación (- min)

Objetivos

Los estudiantes serán capaces de:

- Discutir cómo las tecnologías computacionales han cambiado el mundo y expresar cómo esas tecnologías influencian y están influenciadas por prácticas culturales.
- Diseñar proyectos que combinen hardware y software para recolectar e intercambiar datos.

Preparación

Acceder a sala de computación con cámara web y conexión a internet.

Vea los video tutoriales de teachable machine y cree algunos modelos con imágenes para experimentar.

Practique entrenando algunos modelos en la plataforma.

Recursos

Para los Profesores:

- Video: ¿Cómo entrenar un modelo con Teachable Machine?

Para los estudiantes:

- Enlace: Google Teachable Machine

Vocabulario

- **Datos de entrenamiento:** Datos utilizados para entrenar un modelo de inteligencia artificial para que pueda discriminar y tomar decisiones.
- **Conjunto de datos:** Un conjunto de datos (conocido también por el anglicismo dataset, comúnmente utilizado en algunos países hispanohablantes) es una colección de datos habitualmente tabulados.
- **Modelo:** Proceso paso a paso mediante el cual se propone una entrada, salida y un resultado esperado.
- **Predicciones:** Resultado que el sistema de inteligencia artificial genera a partir del análisis de un input nuevo a partir del estudio de ejemplos en un conjunto de datos.

Estrategia de aprendizaje

Conocimiento inicial (5 min)

¿Qué es la inteligencia artificial?

En esta Lección inicial, se enseñará a los estudiantes lo que es la inteligencia artificial, cuáles son los conceptos claves para que luego puedan verlos en funcionamiento con la aplicación Google Teachable Machine.

Consultar a los estudiantes en qué piensan cuando escuchan el término inteligencia artificial. Anote sus aportes en la pizarra.

El objetivo es guiar a los estudiantes hacia la siguiente definición:

La inteligencia artificial es esa rama de la informática y las ciencias de la computación que se encarga en crear programas que sean capaces de aprender y tomar decisiones por cuenta propia en base a información que se le entrega previamente.

Algunas ideas de modelos con Teachable Machine:

Identificar si un estudiante tiene un gorro puesto o no.

Identificar tipos de útiles escolares.

Identificar sonidos.

Identificar voces de diferentes personas diciendo una misma palabra.

Consulte si es que conocen instancias en la vida real donde se utilice la inteligencia artificial.

La inteligencia artificial ya no es un sueño de la ciencia ficción en la que se mostraban robot humanoides que se comportaban como personas. En la vida real la inteligencia artificial se encuentra presente en todo. Desde las recomendaciones de Google al buscar algo en internet, pasando por las páginas que te recomiendan juegos, series o música basados en tus gustos

Algunos ejemplos de inteligencia artificial se encuentran en YouTube, Amazon, MercadoLibre, Google (buscador), los chatbots de tiendas entre otros.

Indíqueles a los estudiantes que van a trabajar con un sistema de inteligencia artificial que les permitirá crear su propio modelo.

Organice a los estudiantes en equipos de 2 a 4 personas y asígneles al menos un computador para la siguiente Lección.

Ampliación del conocimiento (30 min)

Experimentando con Teachable Machine

Anote los siguientes términos en la pizarra.

Datos de entrenamiento.

Modelo de inteligencia artificial.

Predicción.

Explicar que un sistema de inteligencia artificial es un sistema que toma decisiones, o sea revisa los datos que tiene a su disposición y en base a eso decide qué hacer.

Todos los modelos de inteligencia artificial cuentan con al menos tres componentes:

- Los datos de entrenamiento que le permite tener una base para tomar decisiones.
- Un modelo que es la programación de la máquina.
- Y las predicciones que hace la máquina al recibir datos que no son los de entrenamiento.

A continuación, muestre a los estudiantes la página de Google Teachable Machine y muéstreles un ejemplo de cómo entrenar un modelo o el siguiente fragmento del video donde se muestra como entrenar un modelo de forma simple: ¿Cómo entrenar un modelo con Teachable Machine?

Deje que los estudiantes experimenten con el software por 20 minutos y las diferentes maneras de entrenar modelos (con sonido, fotos y capturas de cámara web). Preste apoyo en caso de ser necesario. El objetivo de esta Lección es que el curso se familiarice con la plataforma para que la próxima clase puedan diseñar un modelo y documentar sus lecciones.

Transferencia del conocimiento (10 min)

¿Qué modelo crearon?

Los estudiantes presentan a sus compañeros los modelos que han entrenado.

Explicando lo siguiente:

- Objetivo del modelo (¿Qué se quería identificar?)
- Datos de entrenamiento (¿Qué datos se le entregó a la máquina?)
- Luego los estudiantes escriben el resumen de su modelo en sus bitácoras de aprendizaje.
- Indíqueles que la próxima sesión van a entrenar otro modelo y van a generar un reporte de sus lecciones.

Evaluación (- min)

En la segunda sesión existe una evaluación que puede ser utilizada de manera formativa.

Experiencias de aprendizaje de profundización

Use estos Contenidos para ampliar el aprendizaje de los estudiantes. Se pueden usar como Contenidos extras fuera del aula.

Cada vez mejor

- Los estudiantes pueden desarrollar otros tipos de modelo en la casa y probando diferentes combinaciones

Desafío de curso

- Piensen en conjunto qué utilidad se le podría dar en la vida real los modelos de inteligencia artificial que están entrenando.

Lección 30: IA con Google Teachable Machine II

Lección con conexión

Propósito

En esta Lección básica los estudiantes aprenderán los principios de la inteligencia artificial a partir del entrenamiento de modelos en Google Teachable Machine. La inteligencia artificial tiene 3 componentes claves: datos de entrenamiento, el modelo y las predicciones.

En las dos sesiones que dura este proyecto introductorio, los estudiantes experimentarán con Google Teachable machine, desarrollarán su propio modelo de inteligencia artificial y documentarán el proceso, experimentando de primera mano y de forma práctica el funcionamiento de los sistemas de inteligencia artificial y su uso.

En esta segunda sesión utilizarán lo aprendido para poder crear un proyecto de inteligencia artificial con Teachable Machine y documentarlo.

Secuencia para el aprendizaje

- Conocimiento inicial (10 min)
- Ampliación del conocimiento (25 min)
- Transferencia del conocimiento (10 min)
- Evaluación (- min)

Objetivos

Los estudiantes serán capaces de:

- Discutir como las tecnologías computacionales han cambiado el mundo y expresar como esas tecnologías influencian y están influenciadas por prácticas culturales.
- Diseñar proyectos que combinen hardware y software para recolectar e intercambiar datos.

Preparación

Acceder a sala de computación con cámara web y conexión a internet.

Vea los video tutoriales de teachable machine y crea algunos modelos con imágenes para experimentar.

Practique entrenando algunos modelos en la plataforma.

Imprimir una hoja de reporte por grupo.

Recursos

Para los Profesores:

- Video: ¿Cómo entrenar un modelo con Teachable Machine?

Para los estudiantes:

- Enlace: Google Teachable Machine

Vocabulario

- **Datos de entrenamiento:** Datos utilizados para entrenar un modelo de inteligencia artificial para que pueda discriminar y tomar decisiones.
- **Conjunto de datos:** Un conjunto de datos (conocido también por el anglicismo dataset, comúnmente utilizado en algunos países hispanohablantes) es una colección de datos habitualmente tabulados.
- **Modelo:** Proceso paso a paso mediante el cual se propone una entrada, salida y un resultado esperado.
- **Predicciones:** Resultado que el sistema de inteligencia artificial genera a partir del análisis de un input nuevo a partir del estudio nuevo de ejemplos en un conjunto de datos.
- **Documentación**: Material escrito creado para dejar constancia del funcionamiento y la lógica de diseño de un programa informático.

Estrategia de aprendizaje

Conocimiento inicial (10 min)

Instrucciones del proyecto

Consulte a los estudiantes si recuerdan la Lección realizada la sesión anterior. Completen en conjunto en la pizarra un modelo que indique los componentes de un sistema de inteligencia artificial.

Datos de entrenamiento

Modelo

Predicciones

Solicite a algunos grupos que cuenten el proyecto que realizaron.

Explique que en esta sesión van a crear un sistema de inteligencia artificial con Teachable Machine, y a la vez documentar los pasos de su creación.

Escriba lo siguiente en la pizarra.

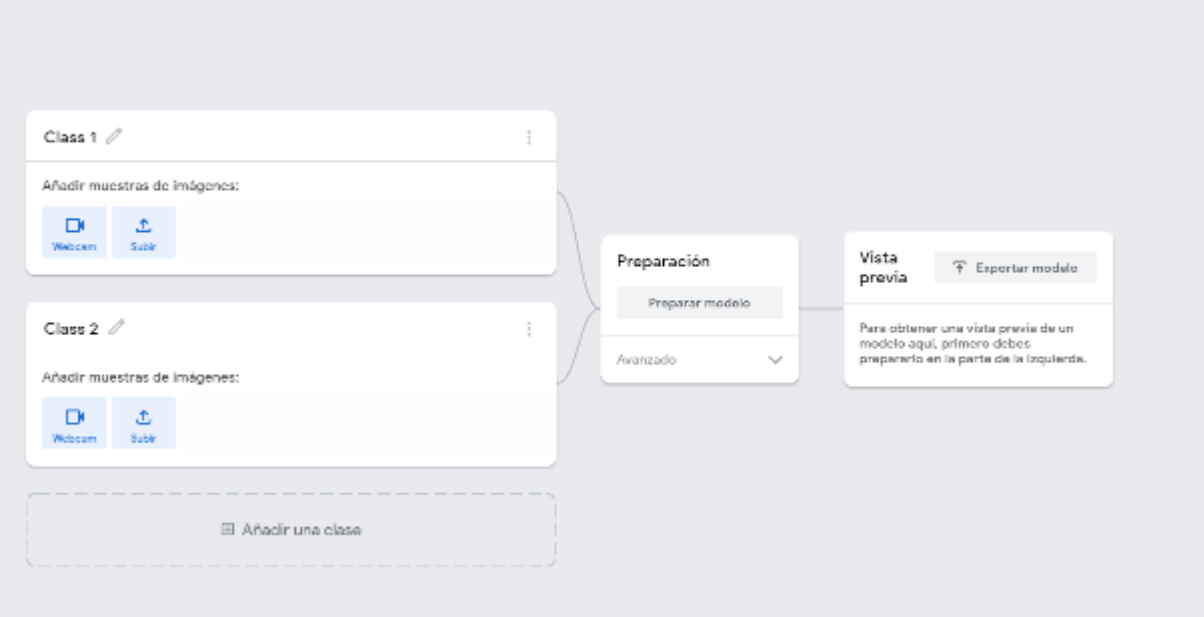
- Nombre del Proyecto:
- Fecha de creación del proyecto:
- Tipo de proyecto.
- Objetivo del Proyecto:
- Datos de entrenamiento:
- Reporte de confiabilidad:

Explique que el documento que deben crear para entregar con su modelo de inteligencia artificial debe contener esa información.

El nombre y fecha de creación del proyecto corresponde al nombre que le asignen a su proyecto y la fecha en que lo crearon. El tipo de proyecto corresponde a la categoría que tenga en Google:

El objetivo del proyecto corresponde a lo que se quiere hacer con el modelo de inteligencia artificial, debe comenzar con un verbo, ejemplo "Distinguir entre diferentes útiles escolares".

Los datos de entrenamiento son aquellos que se utilizan para crear clases. Los estudiantes deben completar a su vez, el nombre de cada clase y los datos que utilizaron.



El reporte de confiabilidad indica cuál fue el porcentaje aproximado de confiabilidad que tenía el sistema al identificar una imagen.

Al testear los sistemas de inteligencia artificial, los estudiantes deben registrar el porcentaje aproximado de confianza con al menos tres objetos.

Ampliación del conocimiento (30 min)

Experimentando con Teachable Machine

Se pide a los estudiantes que anoten en sus cuadernos los puntos indicados al inicio de la clase y que comiencen a desarrollar un proyecto con Google Teachable Machine y que tengan en cuenta los puntos solicitados en los reportes.

Transferencia del conocimiento (10 min)

¿Qué modelo crearon?

Los estudiantes completan los datos del reporte en su cuaderno y describen en sus propias palabras los tres componentes de un sistema de inteligencia artificial.

Evaluación (5 min)

- Los estudiantes describen en su cuaderno cuáles son los tres componentes básicos de un sistema de inteligencia artificial.
- Alternativamente, pueden utilizar la clase para completar este reporte.

Experiencias de aprendizaje de profundización

Use estos Contenidos para ampliar el aprendizaje de los estudiantes. Se pueden usar como Contenidos extras fuera del aula.

Cada vez mejor

- Los estudiantes pueden desarrollar otros tipos de modelo en la casa y probando diferentes combinaciones

Desafío de curso

- Piensen en conjunto qué utilidad se le podría dar en la vida real los modelos de inteligencia artificial que están entrenando.

Lección 31: Chatbots e IA I

Lección con conexión Ver en ISTE

Propósito

Si bien las personas alguna vez asociaron la IA con robots o jugar al ajedrez, los alumnos de hoy en día a menudo interactúan con chatbots de IA todos los días. Los chatbots de IA como Siri, Alexa, Cortana y el Asistente de Google se encuentran comúnmente en los teléfonos inteligentes, las computadoras y los dispositivos domésticos de los alumnos, pero ¿cuánto saben ellos acerca de cómo trabajan y cómo se pueden usar?

En este proyecto basado en la indagación, los alumnos aprenderán cómo la IA usa el procesamiento del lenguaje natural para conversar de una manera similar a la humana, luego aplicarán este conocimiento para desarrollar un modelo de cómo un asistente virtual de IA o agente conversacional podrían realizar una tarea relacionada con el tema de su curso.

En la primera sesión los estudiantes activarán el aprendizaje acerca de los chatbots y aprenderán cómo se relacionan estos con la inteligencia artificial y como son una herramienta que se usa día a día. Además, experimentarán con un chatbot creado por la organización mundial de la salud y que se utiliza a través de WhatsApp.

Secuencia para el aprendizaje

- Conocimiento inicial (15 min)
- Ampliación del conocimiento (20 min)
- Transferencia del conocimiento (5 min)
- Evaluación (5 min)

Objetivos

Los estudiantes serán capaces de:

- Discutir como las tecnologías computacionales han cambiado el mundo y expresar como esas tecnologías influencian y están influenciadas por prácticas culturales.
- Pensar en maneras de mejorar la accesibilidad y usabilidad de productos tecnológicos para diversas necesidades y requerimientos de los usuarios.

Preparación

Solicitar acceso a sala de computación o computadores para sus estudiantes.

Contar teléfonos con acceso a internet (o WhatsApp en este caso) para poder utilizar un chatbot de la organización mundial de la salud.

Familiarizarse con los conceptos de chatbots y procesamiento de lenguaje natural.

Utilizar el chatbot de la organización mundial de la salud con datos sobre el COVID19

Recursos

Para los Profesores:

- Enlace: ¿Qué son los chatbot?
- Enlace: ¿Qué es el procesamiento de lenguaje natural?
- Video: HYPERLINK "https://www.youtube.com/watch?v=l-zfWYoghhs" Chatbots en retail.

Para los estudiantes:

- Enlace: Números para diferentes países del chatbot de la OMS.

Vocabulario

- **Chatbot:** son aplicaciones software que surgen en los años 60, y que simulan mantener una conversación con una persona al proveer respuestas automáticas
- **Procesamiento de lenguaje natural:** es un campo de las ciencias de la computación, de la inteligencia artificial y de la lingüística que estudia las interacciones entre las computadoras y el lenguaje humano. Se ocupa de la formulación e investigación de mecanismos eficaces computacionalmente para la comunicación entre personas y máquinas por medio del lenguaje natural, es decir, de las lenguas del mundo.

Estrategia de aprendizaje

Conocimiento inicial (10 min)

Activación de conocimientos previos

Dé a los alumnos la definición de chatbot: un programa de cómputo que simula conversaciones con seres humanos. Los chatbots simples pueden interactuar con los seres humanos usando preguntas, respuestas y enunciados predeterminados que están codificados en su programación. Los chatbots más avanzados usan IA para funciones adicionales como voz a texto, texto a voz, procesamiento de vocabulario amplio o bases de conocimientos y aprendizaje de conversaciones pasadas.

Presente el siguiente video para demostrar lo que hacen los chatbot en la vida real: Chatbots en retail.

Pregunte a los alumnos: ¿Alguna vez han utilizado un chatbot de IA como Siri de Apple, Alexa de Amazon, Cortana de Microsoft o el Asistente de Google? ¿Para qué lo utilizaron? ¿Qué tareas realizó el chatbot? ¿Qué tan parecido fue a hablar con el agente de IA a hablar con un ser humano? ¿De qué forma fue diferente? ¿En algún momento sentiste que hablabas con un ser humano y no con un chatbot?

Ampliación del conocimiento (20 min)

Experimentando con un chatbot

Indíqueles que van a utilizar un chatbot de la OMS para obtener consejos e información de la OMS. Para poder activarlo deben seguir los siguientes pasos:

- Agregar el siguiente teléfono como contacto al celular: +41 79 893 18 92
- Revisar whatsapp y escribir “Hola”
- Solicitar información al bot.

Los estudiantes responden en sus bitácoras las siguientes preguntas.

- ¿Qué función cumple el bot?
- ¿Cómo sabe lo que se le quiere preguntar?
- ¿Qué ocurre si uno escribe otra cosa?
- ¿Qué información entrega y qué cosas se pueden hacer con él?

Transferencia del conocimiento (15 min)

¿Cuál es el estado actual de los chatbot?

Muestre el siguiente video donde se presenta un asistente de Google que imita a un humano.

Los estudiantes reflexionan y responden en sus bitácoras las siguientes preguntas:

¿Es un chatbot un buen reemplazo para los humanos?

¿Para qué pueden ser mejores los chatbots?

¿En cuánto tiempo más creen que esa tecnología esté disponible para toda la gente?

Evaluación

- Se puede definir las preguntas de la transferencia del conocimiento como evaluación formativa para los estudiantes.

Experiencias de aprendizaje de profundización

Use estos Contenidos para ampliar el aprendizaje de los estudiantes. Se pueden usar como Contenidos extras fuera del aula.

Cada vez mejor

- Los estudiantes pueden investigar acerca de Eliza el primer chatbot de la historia.

Desafío de curso

- Los estudiantes reflexionan sobre qué tipos de chatbot existen y cuáles pueden ser las funciones que son capaces de cumplir.

Objetivo del análisis: En las siguientes clases los estudiantes descubrirán que hay dos tipos de chatbot e incluso ellos van a diseñar un prototipo de uno. Pueden anotar las ideas para contrastar a futuro e incluso descubrir nuevas funciones que aún no se han inventado.

Lección 32: Chatbots e IA II

Lección con conexión Ver en ISTE

Propósito

Si bien las personas alguna vez asociaron la IA con robots o jugar al ajedrez, los alumnos de hoy en día a menudo interactúan con chatbots de IA todos los días. Los chatbots de IA como Siri, Alexa, Cortana y el Asistente de Google se encuentran comúnmente en los teléfonos inteligentes, las computadoras y los dispositivos domésticos de los alumnos, pero ¿cuánto saben ellos acerca de cómo trabajan y cómo se pueden usar?

En este proyecto basado en la indagación, los alumnos aprenderán cómo la IA usa el procesamiento del lenguaje natural para conversar de una manera similar a la humana, luego aplicarán este conocimiento para desarrollar un modelo de cómo un asistente virtual de IA o agente conversacional podrían realizar una tarea relacionada con el tema de su curso.

En esta sesión los estudiantes tendrán experiencias prácticas utilizando chatbots e identificarán las diferencias entre los asistentes virtuales y agentes conversacionales.

Secuencia para el aprendizaje

- Conocimiento inicial (20 min)
- Ampliación del conocimiento (15 min)
- Transferencia del conocimiento (5 min)
- Evaluación (5 min)

Objetivos

Los estudiantes serán capaces de:

- Discutir como las tecnologías computacionales han cambiado el mundo y expresar como esas tecnologías influencian y están influenciadas por prácticas culturales.
- Pensar en maneras de mejorar la accesibilidad y usabilidad de productos tecnológicos para diversas necesidades y requerimientos de los usuarios.

Preparación

Solicitar acceso a sala de computación o computadores para sus estudiantes.

Familiarizarse con los conceptos de chatbots y procesamiento de lenguaje natural.

Recursos

Para los Profesores:

- Enlace: HYPERLINK "http://deixilabs.com/index.html" Chatbots Alicia y Eliza
- Enlace: Chatbots de Chatcompose

Para los estudiantes:

- Llevar su bitácora de notas.

Vocabulario

- **Chatbot:** son aplicaciones software que surgen en los años 60, y que simulan mantener una conversación con una persona al proveer respuestas automáticas
- **Asistente virtual:** un agente de programas de cómputo de IA que realiza tareas específicas basadas en los comandos o preguntas de un usuario.
- **Agente conversacional:** un agente de programas de cómputo de IA diseñado para sostener diálogos naturales con los usuarios

Estrategia de aprendizaje

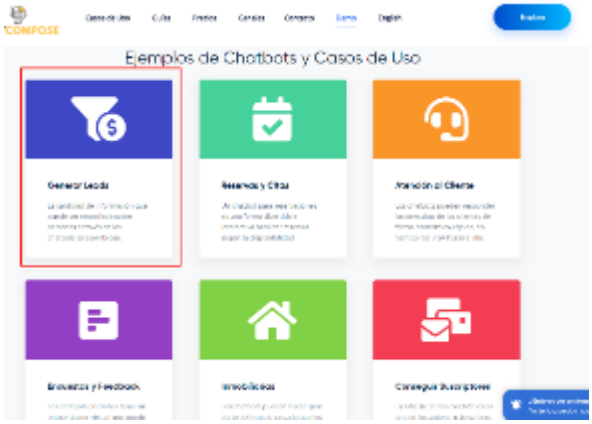
Conocimiento inicial (20 min)

Activación de conocimientos previos

En esta Lección basada en la indagación, los alumnos interactuarán con dos tipos de chatbots de IA: asistentes virtuales y agentes conversacionales. Analizarán las fortalezas y debilidades de cada forma de chatbot.

Informe a los alumnos que en esta Lección aprenderán sobre dos tipos de chatbots de IA: asistentes virtuales y agentes conversacionales.

Haga que los alumnos interactúen y vean las demostraciones de varios asistentes virtuales y agentes conversacionales. Pueden usar los que se enumeran a continuación o complementar esta lista con ejemplos de chatbots relacionados con su área temática específica. A medida que interactúan con cada chatbot, los alumnos deben pedirle que complete varias tareas básicas, como reservar una cita, programar un temporizador o calcular un problema de matemáticas; y también tratar de mantener una conversación con él sobre temas cotidianos como pasatiempos o pensamientos acerca del mundo.



Chatbot conversacional: Chatbots Alicia y Eliza

Chatbots de asistentes virtuales: Chatbots de Chatcompose

Cómo utilizar los chatbot de Chatcompose.

Los estudiantes deben seleccionar los diferentes ejemplos y hacer click en la esquina inferior derecha para activar los chatbots.

Ampliación del conocimiento (15 min)

Definiendo los tipos de chatbot

Muestre un diagrama de Venn en blanco. Escriba "asistente virtual" en un círculo y "agente conversacional" en el otro. Pídales a los alumnos que reflexionen sobre sus interacciones con los chatbots e identifiquen las características que los asistentes virtuales y los agentes conversacionales tienen en común, y las características que los diferencian. Las posibles respuestas incluyen:

- **Asistentes virtuales:** Siguen instrucciones para realizar tareas discretas, como administrar listas de tareas por hacer, tomar notas, establecer temporizadores o citas y buscar información. Tienen acceso a una base de conocimientos y solo pueden responder preguntas acerca de esa información. Son utilizados con frecuencia para servicios de atención al cliente. Cumplen un propósito específico.
- **Agentes conversacionales**: Interacción natural, similar a la humana. Hacen preguntas y hablan acerca de una amplia variedad de temas, incluso personales. Pueden sostener conversaciones triviales.
- **Ambos**: Pueden reconocer el lenguaje humano. Pueden sostener una conversación. Conocimiento limitado o inexistente de temas más allá de su base de conocimientos. Hacen preguntas al usuario. No siempre pueden recordar lo que han dicho. No siempre tienen sentido.

Transferencia del conocimiento (5 min)

Chatbots a mi manera

Solicite a los estudiantes que elaboren una definición de cada tipo de chatbot estableciendo similitudes y diferencias entre cada uno.

Evaluación (5 min)

Se pueden utilizar las definiciones de chatbot como evaluación formativa para evaluar la comprensión de los estudiantes.

Experiencias de aprendizaje de profundización

Use estos Contenidos para ampliar el aprendizaje de los estudiantes. Se pueden usar como Contenidos extras fuera del aula.

Cada vez mejor

- Los estudiantes pueden consultar con sus familiares si han tenido que usar un chatbot en tiendas o servicios online y cuáles han sido sus experiencias de uso. En futuras clases se puede abrir con esas consultas.

Desafío de curso

- Los estudiantes reflexionan sobre qué tipos de chatbot existen y cuáles pueden ser las funciones que son capaces de cumplir.

Lección 33: Chatbots e IA III

Lección con conexión Ver en ISTE

Propósito

Si bien las personas alguna vez asociaron la IA con robots o jugar al ajedrez, los alumnos de hoy en día a menudo interactúan con chatbots de IA todos los días. Los chatbots de IA como Siri, Alexa, Cortana y el Asistente de Google se encuentran comúnmente en los teléfonos inteligentes, las computadoras y los dispositivos domésticos de los alumnos, pero ¿cuánto saben ellos acerca de cómo trabajan y cómo se pueden usar?

En esta Lección, los alumnos verán más a fondo cómo funcionan los chatbots. Los alumnos considerarán cómo la calidad del procesamiento del lenguaje natural (PLN) de una IA afecta su capacidad de interactuar naturalmente con los seres humanos.

Además, esta clase sienta las bases para el siguiente proyecto de design thinking y chatbots, en la que los estudiantes van a generar una solución para su comunidad a partir de un chatbot.

Secuencia para el aprendizaje

- Conocimiento inicial (15 min)
- Ampliación del conocimiento (20 min)
- Transferencia del conocimiento (5 min)
- Evaluación (5 min)

Objetivos

Los estudiantes serán capaces de:

- Discutir como las tecnologías computacionales han cambiado el mundo y expresar como esas tecnologías influencian y están influenciadas por prácticas culturales.
- Pensar en maneras de mejorar la accesibilidad y usabilidad de productos tecnológicos para diversas necesidades y requerimientos de los usuarios.

Preparación

Solicitar acceso a sala de computación o computadores para sus estudiantes.

Recursos

Para los Profesores:

- Enlace: ¿Qué son los Chatbot?
- Enlace: ¿Qué es el procesamiento de lenguaje natural?
- Video: ¿Qué son los Chatbot?
- Video: ¿Qué es el procesamiento de lenguaje natural?
- Video: Test de Turing

Para los estudiantes:

- Enlace: Números para diferentes países del chatbot de la OMS.

Vocabulario

Chatbot: son aplicaciones software que surgen en los años 60, y que simulan mantener una conversación con una persona al proveer respuestas automáticas

Procesamiento de lenguaje natural: es un campo de las ciencias de la computación, de la inteligencia artificial y de la lingüística que estudia las interacciones entre las computadoras y el lenguaje humano. Se ocupa de la formulación e investigación de mecanismos eficaces computacionalmente para la comunicación entre personas y máquinas por medio del lenguaje natural, es decir, de las lenguas del mundo.

Estrategia de aprendizaje

Conocimiento inicial (20 min)

Chatbots y procesamiento de lenguaje natural

En esta Lección, los alumnos verán más a fondo cómo funcionan los chatbots. Los alumnos considerarán cómo la calidad del procesamiento del lenguaje natural (PLN) de una IA afecta su capacidad de interactuar naturalmente con los seres humanos.

Consulte a los estudiantes qué recuerdan de los chatbots y qué tipos existen

(Ver definiciones en vocabulario). Comentarles que en esta lección aprenderán sobre el procesamiento de lenguaje natural y cuál es el mayor problema que tienen los chatbot.

Haga que los estudiantes vean estos tres videos.

¿Qué son los Chatbot?

¿Qué es el procesamiento de lenguaje natural?

Test de Turing

Refuerce los siguientes conceptos, con base en el contenido de los videos:

- Los chatbots con IA funcionan con procesamiento del lenguaje natural.
- El procesamiento del lenguaje natural es la tecnología de IA utilizada para entender e interactuar con el lenguaje humano.
- El procesamiento del lenguaje natural potencia tecnologías como las experiencias de voz y asistentes, predictores de texto, verificaciones gramaticales y traductores de idiomas.
- Para percibir y comprender lo que dicen las personas, la IA utiliza sensores para recibir información.
- La IA debe descifrar sonidos, sintaxis, semántica y contexto para extraer un significado. Para responder adecuadamente, la IA debe ser capaz de generar oraciones coherentes.

Ampliación del conocimiento (15 min)

Chatbots y humanos

Discuta con sus estudiantes:

- En base a sus interacciones y observaciones de los chatbots de IA, ¿qué tan bien cree que están simulando una conversación humana?
- A medida que los chatbots se parecen más a los seres humanos, ¿cuáles podrían ser las consideraciones éticas de decirle o no a las personas que se están comunicando con una IA?
- ¿Por qué creen que algunos chatbots hacen un mejor trabajo que otros?
- ¿Cómo podrían beneficiarse los chatbots al combinar las habilidades de los asistentes virtuales y los agentes conversacionales en una sola entidad de IA?
- ¿Cuáles podrían ser los beneficios de usar chatbots en hogares, escuelas o negocios? ¿Cuáles podrían ser los desafíos o riesgos?
- ¿Creen que es importante que todos tengan un conocimiento básico acerca de cómo funcionan los chatbots de IA? ¿Por qué?

Transferencia del conocimiento (10 min)

¿Cuál es el estado actual de los chatbot?

Los estudiantes registran en sus cuadernos una idea en la que los chatbot podrían ayudar a su comunidad cercana.

Evaluación

Se puede definir las preguntas de la transferencia del conocimiento como evaluación formativa para los estudiantes.

Experiencias de aprendizaje de profundización

Use estos Contenidos para ampliar el aprendizaje de los estudiantes. Se pueden usar como Contenidos extras fuera del aula.

Cada vez mejor

- Como trabajo de investigación por cuenta propia, los estudiantes pueden averiguar qué chatbots están disponibles en sitios que ellos o su familia utilizan a menudo. ¿Son asistentes virtuales o agentes conversacionales?

Desafío de curso

- Lluvia de ideas para definir qué tipo de chatbot se podría diseñar para apoyar las lecciones del curso. (Uno que recuerde los trabajos y tareas, por ejemplo.

Lección 34: Design thinking y chatbots I

Lección con conexión

Propósito

En este proyecto los alumnos utilizarán un proceso de pensamiento de diseño para identificar una forma en que la IA podría usarse para resolver un problema que afecte a su comunidad. Los alumnos aprenderán a implementar de forma creativa una solución a una problemática de la vida real a través de la IA y los chatbots.

Este proyecto que consta de 5 sesiones permitirá a los estudiantes crear un prototipo o concepto para resolver un problema de su comunidad. La metodología y resultados pueden permitir que estudiantes diseñen a futuro una solución en software concreta para resolver el problema.

En esta primera sesión los estudiantes aprenderán los pasos del *design thinking*, y seleccionarán una problemática para orientar sus proyectos hacia su resolución.

Secuencia para el aprendizaje

- Conocimiento inicial (10 min)
- Ampliación del conocimiento (30 min)
- Transferencia del conocimiento (5 min)
- Evaluación (- min)

Objetivos

Los estudiantes serán capaces de:

- Buscar diversas perspectivas para mejorar artefactos computacionales.
- Usar un proceso iterativo para planificar el desarrollo de un programa incluyendo las perspectivas de otros y las preferencias de los usuarios.

Preparación

Solicitar acceso a sala de computación o computadores para sus estudiantes.

Recursos

Para los Profesores:

- Video: ¿Qué es el design thinking?
- Video: Metodología ¿Qué es design thinking?
- Enlace IT Madrid: ¿Qué es y para qué sirve el design thinking?

Para los estudiantes:

- Enlace: 17 Objetivos para transformar nuestro mundo ONU.

Vocabulario

Design Thinking: en español, pensamiento de diseño, es una metodología o proceso que permite o facilita la solución de problemas, el diseño y desarrollo de productos y servicios de todo tipo y sectores económicos, utilizando para ello equipos altamente motivados, y la innovación y creatividad como motores o mantras. (fuente: https://www.itmadrid.com/que-es-y-para-que-sirve-design-thinking/)

Estrategia de aprendizaje

Conocimiento inicial (10 min)

¿Qué es y para qué sirve el design thinking?

Este es el inicio de un nuevo proyecto para los estudiantes en el que utilizarán sus experiencias previas con otros sistemas de inteligencia artificial para poder proponer una solución a una problemática que ellos logren identificar en sus comunidades.

Exponga lo siguiente a sus estudiantes:

“En este proyecto vamos a aplicar design thinking para poder ofrecer una solución a una problemática de nuestra comunidad.

El design thinking es una técnica para resolución de problemas que tiene 5 pasos:

1. Sentir empatía: Poniéndose en el lugar de otras personas de nuestra comunidad y viendo qué dificultades enfrentan.
2. Definir un problema: Al descubrir una dificultad o problema, debemos pensar en cómo explicarlo y cómo debería funcionar una solución.
3. Idear: Generar varias ideas de diferentes tipos, sin importar lo descabelladas que parezcan y a partir de estas ideas aproximarse a una solución.
4. Crear un prototipo: Generar un producto que resuelva el problema.
5. Probar: Ver si el prototipo funciona, evaluar en qué puede mejorar y seguir repitiendo el proceso hasta que el prototipo funcione bien.

Ampliación del conocimiento (15 min)

Chatbots y humanos

Divida al curso en equipos de 2 a 4 estudiantes, los cuales trabajarán hasta el final del proyecto.

Exponga lo siguiente:

“En esta sesión vamos a aplicar el primer paso: sentir empatía. Por lo tanto, vamos a preguntarnos ¿Cuáles son los problemas ambientales o de sostenibilidad que noto en mi comunidad? ¿Cómo afectan estos problemas a las personas que me rodean?”

Los alumnos pueden encontrar inspiración en la lista de las Naciones Unidas de los 17 Objetivos para transformar nuestro mundo. Cada equipo deberá identificar un desafío ambiental o de sostenibilidad local específico para explorarlo más a fondo.

A partir de lo anterior deben generar un documento en línea o de forma local respondiendo las dos preguntas de las problemáticas identificadas.

El objetivo es que los estudiantes tengan una noción del problema sobre el que quieren trabajar para continuar desarrollándolo en la próxima sesión.

Transferencia del conocimiento (5 min)

Definiendo los primeros pasos e investigando más a fondo

Anote en la pizarra cada equipo y la problemática sobre la que estén investigando.

“En la próxima clase vamos a continuar definiendo el problema y expondrán lo que han descubierto al resto de sus compañeros y compañeras.”

Evaluación (- min)

Al final de este proyecto, se puede evaluar el prototipo que presenten de forma sumativa.

Experiencias de aprendizaje de profundización

Use estos Contenidos para ampliar el aprendizaje de los estudiantes. Se pueden usar como Contenidos extras fuera del aula.

Cada vez mejor

- Sugiera a los estudiantes entrevistar a las personas de la comunidad que se ven afectadas por la situación para comprender mejor el problema.

Desafío de curso

- Al final de este proyecto, los estudiantes pueden presentar sus prototipos en una feria escolar en conjunto con una ficha que resuma la problemática investigada.

Lección 35: Design thinking y chatbots II

Lección con conexión

Propósito

En este proyecto los alumnos utilizarán un proceso de pensamiento de diseño para identificar una forma en que la IA podría usarse para resolver un problema que afecte a su comunidad. Los alumnos aprenderán a implementar de forma creativa una solución a una problemática de la vida real a través de la IA y los chatbots.

Este proyecto que consta de 5 sesiones permitirá a los estudiantes crear un prototipo o concepto para resolver un problema de su comunidad. La metodología y resultados pueden permitir que estudiantes diseñen a futuro una solución en software concreta para resolver el problema.

En esta segunda sesión los estudiantes definirán su problema y realizarán una lluvia de ideas para poder generar soluciones creativas a la problemática.

Secuencia para el aprendizaje

- Conocimiento inicial (15 min)
- Ampliación del conocimiento (25 min)
- Transferencia del conocimiento (5 min)
- Evaluación (- min)

Objetivos

Los estudiantes serán capaces de:

- Buscar diversas perspectivas para mejorar artefactos computacionales.
- Usar un proceso iterativo para planificar el desarrollo de un programa incluyendo las perspectivas de otros y las preferencias de los usuarios.

Preparación

Solicitar acceso a sala de computación o computadores para sus estudiantes.

Revisar material de referencia sobre el design thinking en la sección de recursos.

Recursos

Para los Profesores:

- Video: ¿Qué es el design thinking?
- Video: Metodología ¿Qué es design thinking?
- Enlace IT Madrid: ¿Qué es y para qué sirve el design thinking?

Para los estudiantes:

- Enlace: 17 Objetivos para transformar nuestro mundo ONU.

Vocabulario

- **Design Thinking**: en español, pensamiento de diseño, es una metodología o proceso que permite o facilita la solución de problemas, el diseño y desarrollo de productos y servicios de todo tipo y sectores económicos, utilizando para ello equipos altamente motivados, y la innovación y creatividad como motores o mantras.
- **Lluvia de ideas:** Es un método donde no se pone límite a la factibilidad de las ideas, aceptando incluso aquellas que sean imposibles de realizar o descabelladas. El objetivo de este método es llegar a soluciones factibles a partir de proposiciones imposibles.

Estrategia de aprendizaje

Conocimiento inicial (15 min)

Identificando problemáticas

Escriba en el pizarrón la frase “problemáticas locales”.

Los estudiantes se reúnen en los grupos que habían conformado la sesión pasada.

Comente que, en esta segunda Lección, van a seguir trabajando con las problemáticas locales que identificaron. Cada grupo va a contar la problemática que quieren abordar.

Solicite a sus estudiantes que cuenten brevemente la problemática que quieren abordar y anote las ideas en la pizarra.

Ampliación del conocimiento (15 min)

Lluvia de ideas para soluciones sin límites

En este momento a los estudiantes se les dibujará una hoja de ruta para que sepan en qué Lección se encuentran y hacia cual van.

Comente:

“¿Recuerdan los pasos del design thinking? Estos eran 1- Empatizar. 2- Definir un problema. 3-Idear soluciones. 4- Crear un prototipo. 5- Probar el prototipo.

Ya hemos empatizado y definido un problema. Hoy vamos a idear soluciones para este problema. O sea, ya estamos en el paso 3.”

A continuación, se les presentará a los estudiantes la metodología de lluvia de ideas para poder ofrecer soluciones a los problemas y posteriormente comenzar a diseñar soluciones.

Comente:

“Vamos a hacer una lluvia de ideas, vamos a anotar todo lo que se nos ocurra para poder solucionar el problema. No importa que la idea sea loca o requiera muchos recursos. Vamos a anotar en 5 minutos absolutamente todo lo que se nos ocurra para solucionar el problema, una vez pasen los 5 minutos se acaba el tiempo y no anotamos nada más.”

De 5 minutos a los estudiantes para que puedan escribir todas las soluciones que se les ocurran. Fomente que sean creativos y que no discutan las ideas. En esta etapa sólo importa tener varias ideas para trabajar.

A los 5 minutos pídales que se detengan y levanten las manos (para que tengan una señal kinestésica de que no se puede seguir trabajando en ideas).

Solicite que les comenten varios tipos de idea por grupos. Desde las más conservadoras hasta las más descabelladas. Las ideas descabelladas pueden servir para llegar a soluciones creativas.

Comente

"Muy bien, a partir de estas ideas que comentaron, quiero que armen una solución para el problema utilizando IA o chatbots. Por ejemplo, una página web para que los vecinos puedan conocerse o un sistema de reparto de ropa para los vecinos que necesiten o una planta de reciclaje comunitaria que se utilice con un chatbot."

Deje el resto de la clase para que los estudiantes definan un concepto de solución.

Transferencia del conocimiento (5 min)

Compartiendo ideas

Consulte en voz alta al curso que soluciones están proponiendo. Anote algunas ideas en la pizarra como ejemplo. Los estudiantes pueden utilizar ideas de otros grupos para mejorar las suyas.

Solicite que escriban en el cuaderno ideas que les llamaron la atención de otros grupos.

Comente:

"Las próximas dos sesiones vamos a trabajar en prototipos de las soluciones que crearon."

Evaluación

Al final de este proyecto, se puede evaluar el prototipo que presenten de forma sumativa.

Experiencias de aprendizaje de profundización

Use estos Contenidos para ampliar el aprendizaje de los estudiantes. Se pueden usar como Contenidos extras fuera del aula.

Cada vez mejor

- Existe otro tipo de ideas o soluciones que se llaman "Spaced out" en el proceso de lluvia de ideas. Puede solicitar a los estudiantes que investiguen al respecto.

Desafío de curso

- Al final de este proyecto, los estudiantes pueden presentar sus prototipos en una feria escolar en conjunto con una ficha que resuma la problemática investigada.

Lección 36: Design thinking y chatbots III

Lección con conexión

Propósito

En este proyecto los alumnos utilizarán un proceso de pensamiento de diseño para identificar una forma en que la IA podría usarse para resolver un problema que afecte a su comunidad. Los alumnos aprenderán a implementar de forma creativa una solución a una problemática de la vida real a través de la IA y los chatbots.

Este proyecto que consta de 5 sesiones permitirá a los estudiantes crear un prototipo o concepto para resolver un problema de su comunidad. La metodología y resultados pueden permitir que estudiantes diseñen a futuro una solución en software concreta para resolver el problema.

En esta tercera sesión los estudiantes trabajarán en el diseño de un prototipo de su chatbot o solución implementada con AI.

Secuencia para el aprendizaje

- Conocimiento inicial (10 min)
- Ampliación del conocimiento (30 min)
- Transferencia del conocimiento (5 min)
- Evaluación (- min)

Objetivos

Los estudiantes serán capaces de:

- Buscar diversas perspectivas para mejorar artefactos computacionales.
- Usar un proceso iterativo para planificar el desarrollo de un programa incluyendo las perspectivas de otros y las preferencias de los usuarios.

Preparación

Solicitar acceso a sala de computación o computadores para sus estudiantes.

Revisar material de referencia sobre prototipado.

Familiarícese con el uso de la herramienta botframe para crear maquetas de chatbots.

Recursos

Para los Profesores:

- Video: ¿Qué es el design thinking?
- Video: Metodología ¿Qué es design thinking?
- Video: Prototipar, 5 ejemplos de design thinking.
- Video: Consejos para design thinking prototipos.

Para los estudiantes:

- **Enlace:** Botframe, herramienta para diseño de chatbots.

Vocabulario

- **Design Thinking**: en español, pensamiento de diseño, es una metodología o proceso que permite o facilita la solución de problemas, el diseño y desarrollo de productos y servicios de todo tipo y sectores económicos, utilizando para ello equipos altamente motivados, y la innovación y creatividad como motores o mantras.
- **Prototipo:** En el contexto del design thinking, los prototipos permiten testar el objeto antes de que entre en producción, detectar errores, deficiencias, etc. Cuando el prototipo está suficientemente perfeccionado en todos los sentidos requeridos y alcanza las metas para las que fue pensado, el objeto puede empezar a producirse.

Estrategia de aprendizaje

Conocimiento inicial (10 min)

¿Qué es el prototipado?

Los estudiantes se reúnen en los grupos que habían conformado la sesión pasada.

Recordar con los estudiantes la hoja de ruta y el punto en el que se encuentran.

Mencione:

"¿Cuáles eran los pasos del design thinking? 1- Empatizar. 2- Definir un problema. 3-Idear soluciones. 4- Crear un prototipo. 5- Probar el prototipo. En esta sesión vamos a trabajar en el prototipo de una solución. Un prototipo es una muestra de lo que nuestra solución con inteligencia artificial va a hacer."

Muestre el video con ejemplos de prototipos en design thinking.

Prototipar, 5 ejemplos de design thinking.

Ampliación del conocimiento (15 min)

Diseñando prototipos de chatbots

Haga que los alumnos creen un elemento multimedia, como una infografía, un video o una presentación de diapositivas, que incluya diagramas o descripciones sobre cómo funcionaría la IA en la solución propuesta.

Los estudiantes pueden utilizar Botframe (disponible en los links) para crear su prototipo de chatbot.

Mencione:

"Ustedes pueden hacer el prototipo como una presentación con diapositivas donde se muestren las partes que van a componer su sistema, pueden dibujar en sus cuadernos el aspecto que va a tener el programa que creen o pueden preparar un video de ejemplo explicando como funcionaría su solución."

Toda la clase consiste en la elaboración de prototipos. Monitoree a los estudiantes para identificar necesidades o regular su proceso de diseño enfatizando en que los prototipos no deben ser perfectos y que la idea es que sean fáciles de modificar.

Transferencia del conocimiento (5 min)

Compartiendo ideas

Recuerde a los estudiantes que la próxima clase tendrán una sesión extra para trabajar en sus prototipos, pero que debe completarse la siguiente sesión. Enfatice en el hecho de que los prototipos deben ser realizados de forma rápida y económica, puesto que luego hay que modificarlos en base a las observaciones de los usuarios.

También recuérdeles que pueden traer material extra para completarlo la próxima sesión, como lápices de colores, cartulina u otro elemento físico que quieran presentar para publicitar su idea.

Evaluación

Al final de este proyecto, se puede evaluar el prototipo que presenten de forma sumativa.

Experiencias de aprendizaje de profundización

Use estos Contenidos para ampliar el aprendizaje de los estudiantes. Se pueden usar como Contenidos extras fuera del aula.

Cada vez mejor

- Los estudiantes pueden utilizar código para crear un prototipo funcional en Python, Code, Scratch u otro sistema.

Desafío de curso

- Al final de este proyecto, los estudiantes pueden presentar sus prototipos en una feria escolar en conjunto con una ficha que resuma la problemática investigada.

Lección 37: Design thinking y chatbots IV

Lección con conexión

Propósito

En este proyecto los alumnos utilizarán un proceso de pensamiento de diseño para identificar una forma en que la IA podría usarse para resolver un problema que afecte a su comunidad. Los alumnos aprenderán a implementar de forma creativa una solución a una problemática de la vida real a través de la IA y los chatbots.

Este proyecto que consta de 5 sesiones permitirá a los estudiantes crear un prototipo o concepto para resolver un problema de su comunidad. La metodología y resultados pueden permitir que estudiantes diseñen a futuro una solución en software concreta para resolver el problema.

En la tercera y cuarta sesión los estudiantes definirán un prototipo y comenzarán a trabajar en él para poder ofrecer una solución a la problemática que definieron originalmente.

Secuencia para el aprendizaje

- Conocimiento inicial (10 min)
- Ampliación del conocimiento (30 min)
- Transferencia del conocimiento (5 min)
- Evaluación (- min)

Objetivos

Los estudiantes serán capaces de:

- Buscar diversas perspectivas para mejorar artefactos computacionales.
- Usar un proceso iterativo para planificar el desarrollo de un programa incluyendo las perspectivas de otros y las preferencias de los usuarios.

Preparación

Solicitar acceso a sala de computación o computadores para sus estudiantes.

Revisar material de referencia sobre prototipado.

Familiarícese con el uso de la herramienta botframe para crear maquetas de chatbots.

Recursos

Para los Profesores:

- Video: ¿Qué es el design thinking?
- Video: Metodología ¿Qué es design thinking?
- Video: Prototipar, 5 ejemplos de design thinking.
- Video: Consejos para design thinking prototipos.

Para los estudiantes:

- **Enlace:** Botframe, herramienta para diseño de chatbots.
- **Video:** Tips para prototipado.

Vocabulario

- **Design Thinking**: en español, pensamiento de diseño, es una metodología o proceso que permite o facilita la solución de problemas, el diseño y desarrollo de productos y servicios de todo tipo y sectores económicos, utilizando para ello equipos altamente motivados, y la innovación y creatividad como motores o mantras.
- **Prototipo:** En el contexto del design thinking, los prototipos permiten testar el objeto antes de que entre en producción, detectar errores, deficiencias, etc. Cuando el prototipo está suficientemente perfeccionado en todos los sentidos requeridos y alcanza las metas para las que fue pensado, el objeto puede empezar a producirse.

Estrategia de aprendizaje

Conocimiento inicial (10 min)

Consejos para prototipar

Los estudiantes se reúnen en los grupos que habían conformado la sesión pasada.

Recordar con los estudiantes la hoja de ruta y el punto en el que se encuentran.

Mencione:

"¿Cuáles son los pasos del design thinking? 1- Empatizar. 2- Definir un problema. 3-Idear soluciones. 4- Crear un prototipo. 5- Probar el prototipo. En esta sesión vamos a trabajar en el prototipo de una solución. Un prototipo es una muestra de lo que nuestra solución con inteligencia artificial va a hacer."

Muestre el siguiente video que habla de prototipos, ya que los estudiantes tienen experiencia trabajando en prototipos, los consejos de la autora del video sobre prototipos les van a ser útiles. Active los subtítulos en español.

Consejos para design thinking prototipos.

Ampliación del conocimiento (30 min)

Diseñando prototipos de chatbots

Toda la clase consiste en la elaboración de prototipos. Monitoree a los estudiantes para identificar necesidades o regular su proceso de diseño enfatizando en que los prototipos no deben ser perfectos y que la idea es que sean fáciles de modificar.

Mencione:

"Ahora van a continuar trabajando en sus prototipos. Recuerden que debe ser una idea general que permita mostrar lo que quieren hacer y que se preste para conversar al respecto."

Transferencia del conocimiento (5 min)

Recordando los objetivos

Recuerde a los estudiantes que en la próxima sesión deben presentar sus prototipos ya sea con modelos físicos, dibujos, o una herramienta para presentaciones (como prezzi, google presentations, powerpoint u otra).

Condiciones de la presentación para la próxima sesión. Aparte de presentar el prototipo, la presentación de los estudiantes debe responder las siguientes preguntas:

1. ¿Cuál fue la problemática identificada?
2. ¿Qué hace que el problema sea difícil de resolver para las personas sin la ayuda de la IA?
3. ¿Qué solución propone el grupo y cómo funcionaría?
    a. ¿El prototipo recibe datos de entrada?
    b. ¿El prototipo entrega datos de salida?

c. ¿Presenta alguna mejora respecto a un humano? (tiempo, precisión, horario disponible, etc.)

Evaluación (- min)

Al final de este proyecto, se puede evaluar el prototipo que presenten de forma sumativa.

Experiencias de aprendizaje de profundización

Use estos Contenidos para ampliar el aprendizaje de los estudiantes. Se pueden usar como Contenidos extras fuera del aula.

Cada vez mejor

- Los estudiantes pueden utilizar código para crear un prototipo funcional en Python, Code, Scratch u otro sistema.

Desafío de curso

- Al final de este proyecto, los estudiantes pueden presentar sus prototipos en una feria escolar en conjunto con una ficha que resuma la problemática investigada.

Lección 38: Design thinking y chatbots V

Lección con conexión

Propósito

En este proyecto los alumnos utilizarán un proceso de pensamiento de diseño para identificar una forma en que la IA podría usarse para resolver un problema que afecte a su comunidad. Los alumnos aprenderán a implementar de forma creativa una solución a una problemática de la vida real a través de la IA y los chatbots.

Este proyecto que consta de 5 sesiones permitirá a los estudiantes crear un prototipo o concepto para resolver un problema de su comunidad. La metodología y resultados pueden permitir que estudiantes diseñen a futuro una solución en software concreta para resolver el problema.

En la última sesión los estudiantes presentarán sus prototipos y recibirán retroalimentación de su curso y docente para poder realizar mejoras. Dentro de un proceso de design thinking, los estudiantes deberían iterar y mejorar los prototipos. Sin embargo, el objetivo de este proyecto es que los estudiantes circulen por los diferentes pasos del proceso de design thinking.

Secuencia para el aprendizaje

- Conocimiento inicial (5 min)
- Ampliación del conocimiento (30 min)
- Transferencia del conocimiento (10 min)
- Evaluación (- min)

Objetivos

Los estudiantes serán capaces de:

- Buscar diversas perspectivas para mejorar artefactos computacionales.
- Usar un proceso iterativo para planificar el desarrollo de un programa incluyendo las perspectivas de otros y las preferencias de los usuarios.

Preparación

Solicitar acceso a sala de computación o computadores para sus estudiantes.

Revisar material de referencia sobre prototipado.

Recursos

Para los Profesores:

- Video: ¿Qué es el design thinking?
- Video: Metodología ¿Qué es design thinking?
- Video: Prototipar, 5 ejemplos de design thinking.
- Video: Consejos para design thinking prototipos.

Para los estudiantes:

- **Enlace:** Botframe, herramienta para diseño de chatbots.
- **Video:** Tips para prototipado.

Vocabulario

- **Design Thinking**: en español, pensamiento de diseño, es una metodología o proceso que permite o facilita la solución de problemas, el diseño y desarrollo de productos y servicios de todo tipo y sectores económicos, utilizando para ello equipos altamente motivados, y la innovación y creatividad como motores o mantras.
- **Prototipo:** En el contexto del design thinking, los prototipos permiten testar el objeto antes de que entre en producción, detectar errores, deficiencias, etc. Cuando el prototipo está suficientemente perfeccionado en todos los sentidos requeridos y alcanza las metas para las que fue pensado, el objeto puede empezar a producirse.

Estrategia de aprendizaje

Conocimiento inicial (5 min)

Antes de la presentación

Los estudiantes se reúnen en los grupos que habían conformado la sesión pasada.

Recordar con los estudiantes la hoja de ruta y el punto en el que se encuentran.

Mencione:

"¿Cuáles son los pasos del design thinking? 1- Empatizar. 2- Definir un problema. 3-Idear soluciones. 4- Crear un prototipo. 5- Probar el prototipo. En esta sesión vamos a realizar una prueba del prototipo. El objetivo es que lo presenten a su público y reciban sugerencias de cómo mejorarlo y qué cosas se pueden cambiar en futuras versiones."

Ampliación del conocimiento (30 min)

Compartiendo prototipos y recibiendo retroalimentación

En este paso, los estudiantes compartirán su solución, obtendrán comentarios e identificarán posibles mejoras. Si bien los productos multimedia se compartirán con los compañeros de clase, opcionalmente los estudiantes pueden compartir su idea con los miembros de la comunidad que se verían más afectados por el problema y la posible solución. Los alumnos deben pedirles comentarios y consejos sobre la solución propuesta.

1. ¿Cuál fue la problemática identificada?
2. ¿Qué hace que el problema sea difícil de resolver para las personas sin la ayuda de la IA?
3. ¿Qué solución propone el grupo y cómo funcionaría?
    a. ¿El prototipo recibe datos de entrada?
    b. ¿El prototipo entrega datos de salida?
    c. ¿Presenta alguna mejora respecto a un humano? (tiempo, precisión, horario disponible, etc.)

Transferencia del conocimiento (5 min)

¿Cómo mejorar el prototipo?

Antes de entregar su producto final, los alumnos deben tener tiempo para considerar los comentarios, hacer revisiones a su prototipo y pensar si creen que valdría la pena llevar su proyecto más lejos (a través del trabajo en el curso actual o por otros medios).

Evaluación (- min)

Al final de este proyecto, se puede evaluar el prototipo que presenten de forma sumativa. Utilice esta rúbrica.

Experiencias de aprendizaje de profundización

Use estos Contenidos para ampliar el aprendizaje de los estudiantes. Se pueden usar como Contenidos extras fuera del aula.

Cada vez mejor

- Los estudiantes pueden utilizar las recomendaciones y evaluaciones del profesor y compañeros de curso para seguir desarrollando su idea.

Desafío de curso

- Al final de este proyecto, los estudiantes pueden presentar sus prototipos en una feria escolar en conjunto con una ficha que resuma la problemática investigada.

ANEXOS – RECURSOS DE ESTUDIANTES

Trazar formas - Recursos del estudiante – Lección 9

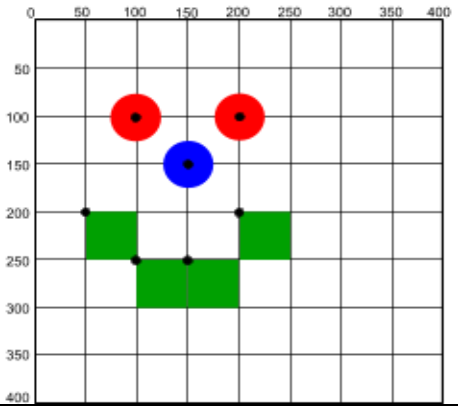
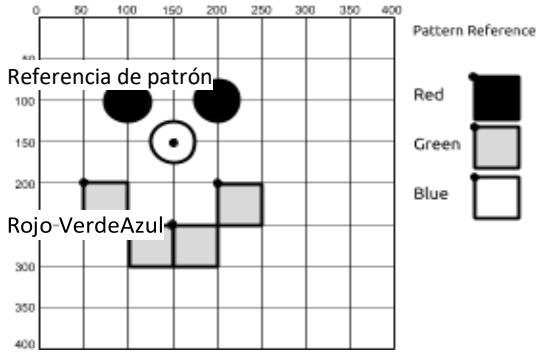
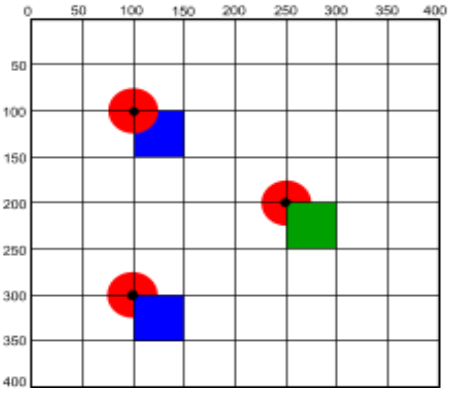
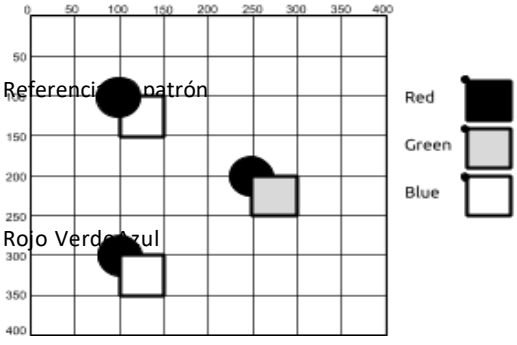
(reproducir e imprimir copia para cada estudiante)

Nombre(s)_____Periodo__Fecha___

Trazar formas A

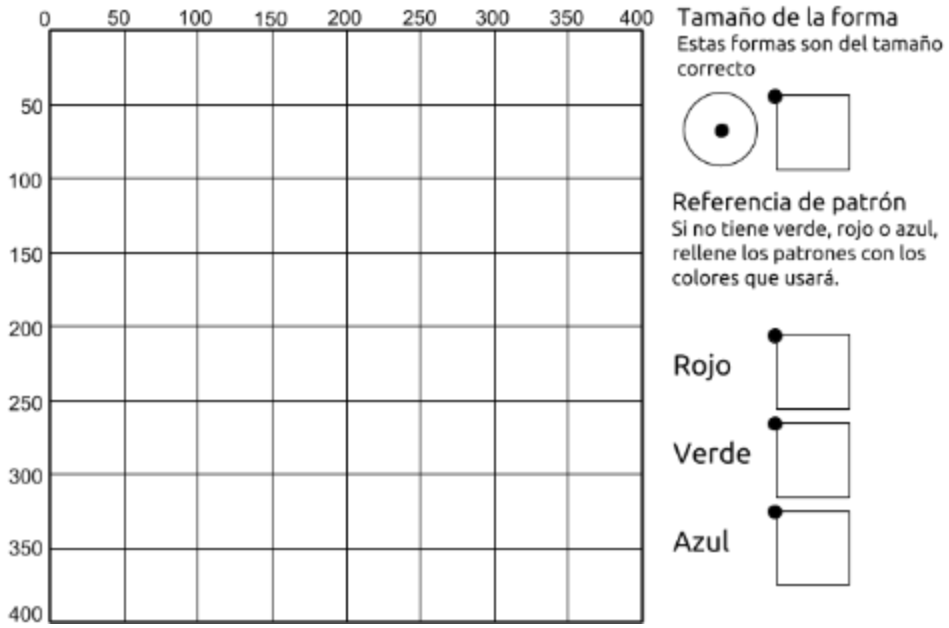
Resumen

Tu compañero debe tener la herramienta de dibujo abierta en una computadora donde no puedas verla. Trata de explicar cómo dibujar tu imagen. Luego revisa su trabajo, peroasegúrate de mantener tus dibujos ocultos.

Dibujo 1A (Color)	Dibujo 1A (Negro y blanco)
	
Dibujo 2A (Color)	**Dibujo 2A (Negro y blanco)**
	

Dibuja el tuyo

Usa el espacio a continuación para dibujar tu propia imagen con las formas. Luego ve si puedes comunicarlo a tu compañero para dibujar usando la herramienta de dibujo de formas en Game Lab. También puedes dar tu dibujo a otro grupo para usarlo como desafío.



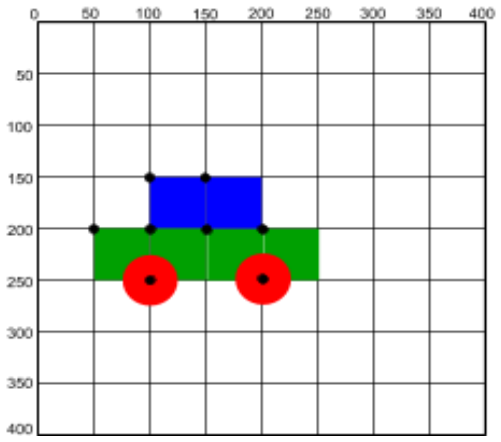
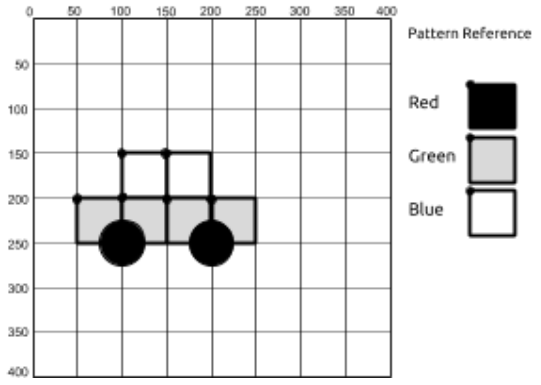
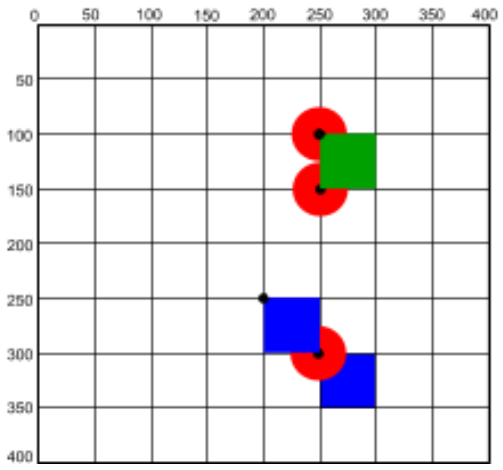
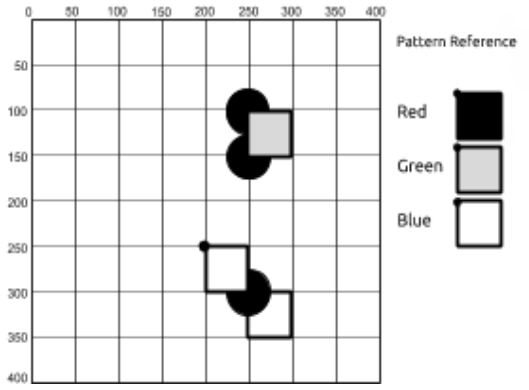
Reflexión

- ¿Qué problema está ayudando a resolver la cuadrícula en Game Lab?
- ¿Has visto formas diferentes de resolver este problema en el pasado? ¿Qué son?

Trazar formas B

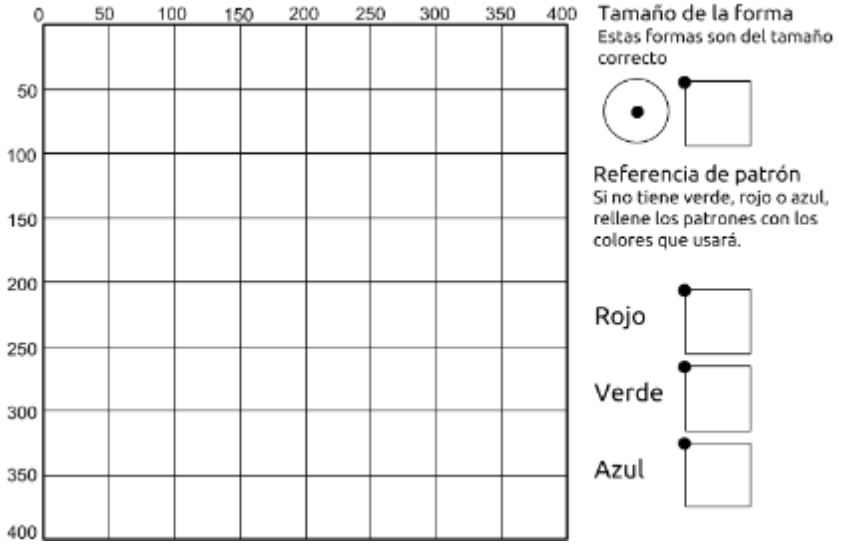
Resumen

Tu compañero debe tener la herramienta de dibujo abierta en una computadora donde no puedas verla. Trata de explicar cómo dibujar su imagen. Luego revisa su trabajo, peroasegúrate de mantener tus dibujos ocultos.

Dibujo 1B (Color)	Dibujo 1B (negro y blanco)
	
Dibujo 2B (Color)	**Dibujo 2B (negro y blanco)**
	

Dibuja el tuyo

Usa el espacio a continuación para dibujar tu propia imagen con las formas. Luego veasi puede comunicarlo a su compañero para dibujar usando la herramienta de dibujode formas en Game Lab. También puede dar su dibujo a otro grupo para usarlo comodesafío.



Reflexión

- ¿Qué problema está ayudando a resolver la cuadrícula en Game Lab?
- ¿Has visto formas diferentes de resolver este problema en el pasado? ¿Qué son?

Sprites - Recursos del estudiante – Lección 13

Nombre(s)_____Periodo__Fecha___

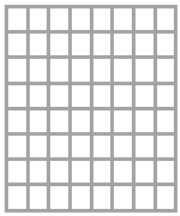
Sprite planificación de escena

Usando formas, sprites y texto, vas a crear una escena simple. Puede pensar en estocomo una página en una historia, un panel en un cómic o simplemente una escena independiente.

Dibuja tu escena

Lo primero que debe considerar al diseñar su escena es cómo se verá su fondo. Puedeusar los comandos de dibujo que has usado en el pasado para diseñar un fondo simplesobre el cual colocarás tus sprites.

El espacio de abajo es 50 por 50. Eso significa que cada cuadrado en el papel se asignaráa 50 píxeles en la computadora. Dibuje su fondo usando solo los comandos de dibujo(referencia proporcionada a la derecha). Enumera los sprites que usarás a continuación.

	formas:
	background(color)
	rect(x, y, width, height)
	ellipse(x, y, width, height)
	line(x1, y1, x2, y2)
	shape(x1, y1, x2, y2…)
	text(string, x, y, width, height)
	textSize(pixels)
	Color and Style:
	fill('color')
	noFill()
	stroke('color')
	noStroke()
	strokeWeight()

Etiqueta sprite	Descripción

Patrón de contador desconectado - Recursos del Estudiante – Lección 15

Tablero variables desconectada

Etiquetas	Valores
Espacio de trabajo	

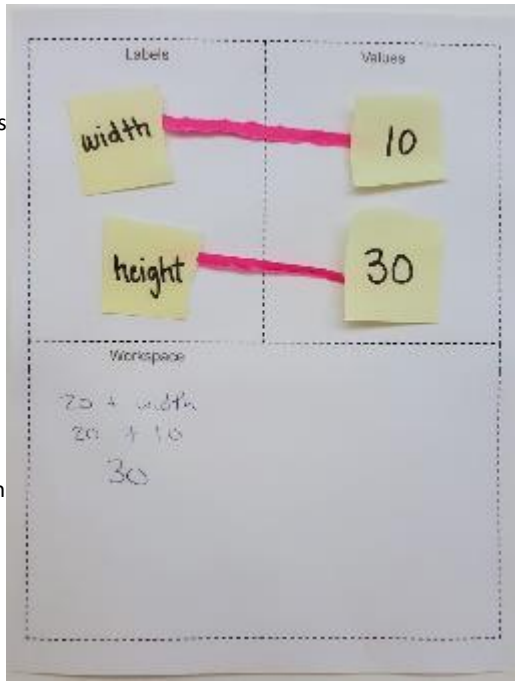
Variables desconectadas

La Lección

Trabajarás con tu compañero para conectar etiquetas de variables y sus valores en función de diferentes programas.

Preparar

- Tablero: un pedazo de papel dividido en 3 secciones:
- Etiquetas, valores y espacio de trabajo.
- Etiquetas de variables y tarjetas de valor (Variable Label y Value Cards): pequeños trozos de papel (~ 50).
- Conectores: algo para conectar las etiquetas y los valores.



Reglas

1. Las tarjetas de etiqueta siempre deben tener un solo conector.
2. No puedes crear la misma tarjeta de etiqueta dos veces.
3. Debes crear una nueva tarjeta de valor para cada comando "=".
4. Las tarjetas de valor sólo pueden tener un número escrito en ellas.
5. Cualquier tarjeta de valor no conectada a una tarjeta de etiqueta al final de un comando se descarta.

Comandos

Ejecuta los programas en orden siguiendo los pasos para cada comando.

Paso 1:

Encuentra o crea la tarjeta de etiqueta y el conector

A. Si el comando comienza con "var", crea una nueva tarjeta de etiqueta con la palabra que viene antes del "=". También debes darle un conector, que usarás más tarde.

B. Si el comando no comienza con "var", busca la etiqueta con la palabra que aparece antes del "=".

Paso 2:

Calcula el número de tu tarjeta de valor

A. Copia lo que está a la derecha de "=" en tu área de trabajo.

B. Tacha todas las etiquetas de variables y reemplázalas con los valores (números) conectados a ellas.

C. Haz cualquier cálculo (suma, resta) en los números.

D. Escribe el número final en una nueva tarjeta de valor.

Paso 3:

Conecta tu tarjeta de etiqueta y tu tarjeta de valor

A. Coloca tu nueva tarjeta de valor en la sección "Valores".

B. Mueve el conector de la tarjeta de etiqueta del Paso 1 para conectarse a esta nueva tarjeta de valor.

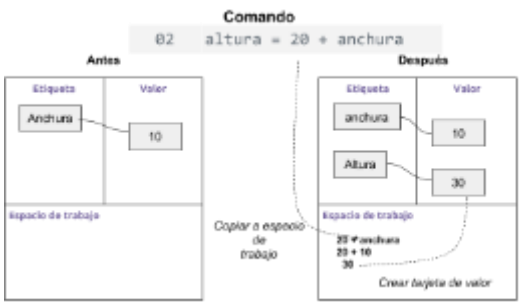
Paso 4:

Eliminar los valores sobrantes

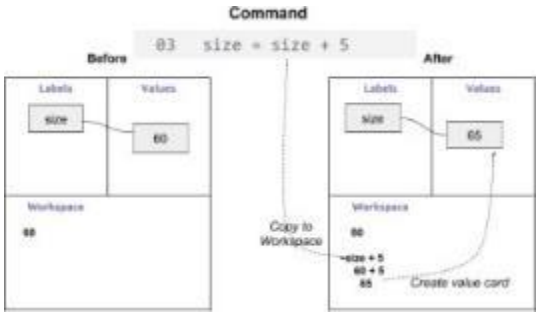
A. Toma todas las tarjetas de valor que no estén conectadas a las tarjetas de etiquetas y deséchalas.

Pruébalo

Ejecuta los programas a continuación. Para ayudarte en los dos primeros programas, consulta los ejemplos anteriores y posteriores a la derecha que muestran cómo se ejecutan los diferentes comandos.



Programa 1	
01	var anchura = 10
02	var altura = 20 + width
Estado final	
anchura:	
altura:	



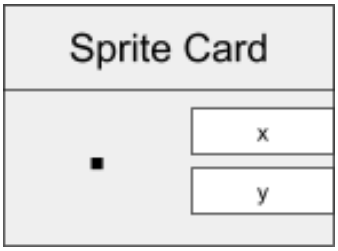
Programa 2	
01	var size = 30
02	size = 60
03	size = size + 5
Estado final	
tamaño:	

Programa 3	
01	var age = 11;
02	var height = 60;
03	age = age + 1;
04	height = height + 5;
Estado final	
edad:	
altura:	

Programa 4	
01	var xPosition = 100
02	var yPosition = xPosition
03	xPosition = yPosition + 30
04	yPosition = yPosition + 50
Estado final	
xPosición:	
yPosición:	

2020 [Imagen]. Recuperada de http://www.code.org

Sprite propiedades



Puedes cambiar las propiedades de tus sprites de la misma manera que cambias tus variables, lo que te permite controlar cómo se mueve el sprite en la pantalla.

Para hacer un seguimiento de las propiedades de un sprite, necesitarás una carta de sprite. La tarjeta Sprite va en la sección de

valores, conectada a una etiqueta variable. En la parte inferior de la tarjeta, enumera las propiedades de las que desea realizar un seguimiento. Utilizarás un conector para cada propiedad.

En el ejemplo de la derecha, la tarjeta realiza un seguimiento de las propiedades “x” y “y” del sprite, que le indican a la computadora dónde colocarlo en la pantalla.

Creando una tarjeta Sprite

Cada vez que veas el comando `createSprite`, tendrás que crear una nueva tarjeta sprite con las propiedades que deseas controlar.

Usa las propiedades sprite

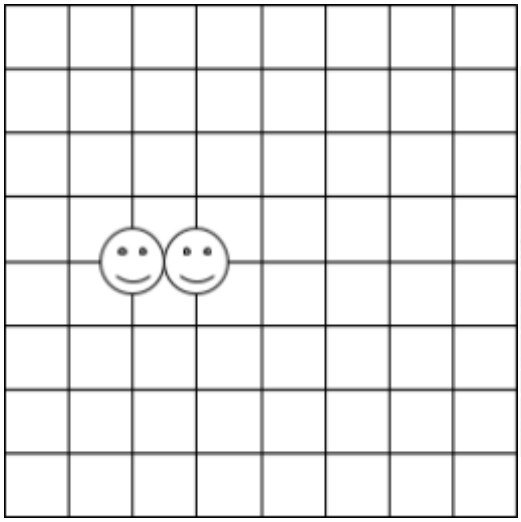
Puede usar los valores de propiedad de tu sprite de la misma manera que usaste los valores de variable.

Dibuja tu sprite

Cada vez que veas el comando `drawSprites`, dibuja tu sprite en la cuadrícula según tus valores de coordenadas x-y.

Puede usar una carita sonriente para la animación de tu sprite.

Puede hacer tus propias tarjetas de sprites.



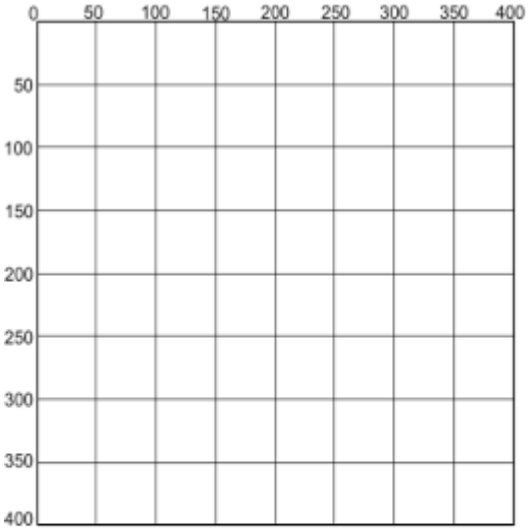
Programa 5

```
01  var smiley = createSprite(100,200);
02  smiley.setAnimation("smileyFace");
03  drawSprites();
05  smiley.x = smiley.x + 50;
06  drawSprites();
```

Pruébalo

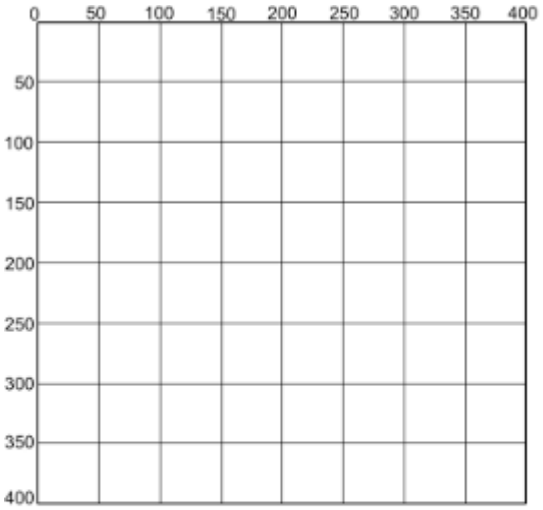
Ejecute estos programas con su compañero. No olvides dibujar tus sprites cuando veas el comando `drawSprites`.

Program 6	
01	`var smiley = createSprite();`
02	`smiley.setAnimation("smiley");`
03	`smiley.x = 50;`
04	`smiley.y = 100;`
05	`drawSprites();`
06	`smiley.x = smiley.x + 50;`
07	`drawSprites();`
08	`smiley.x = smiley.x + 50;`
09	`drawSprites();`
10	`smiley.x = smiley.x + 50;`
11	`drawSprites();`
Estado final	
smiley.x	
smiley.y	



¿Cómo se movió el sprite a través de la red en el Programa 6?

Programa 7	
01	`var smiley = createSprite();`
02	`smiley.setAnimation(“smiley”);`
03	`smiley.x = 200;`
04	`smiley.y = 300;`
05	`drawSprites();`
06	`smiley.y = smiley.y - 30;`
07	`drawSprites();`
08	`smiley.y = smiley.y - 30;`
09	`drawSprites();`
10	`smiley.y = smiley.y - 30;`
11	`drawSprites();`
Estado final	
smiley.x	
smiley.y	



¿Cómo se movió el sprite a través de la red en el Programa 7?

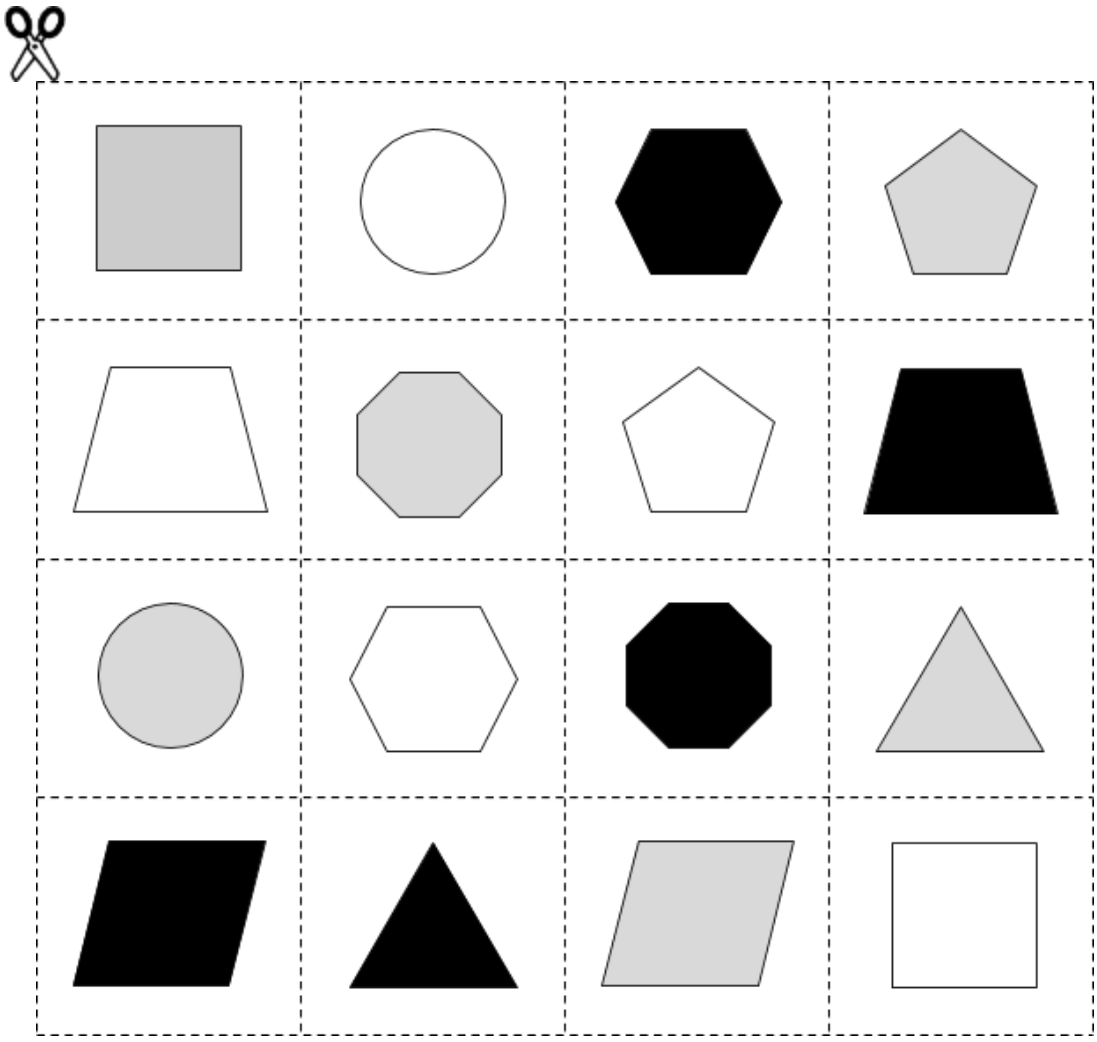
Booleanos desconectados - Recursos del estudiante – Lección 17

Propiedades booleanas

Prepara tus formas

Recorta la siguiente tabla de formas. Estas serán las tarjetas que usarás para la Lección de clasificación de propiedades booleanas.

2020 [Imagen]. Recuperada de http://www.code.org



Tarjeta interactiva - Recursos del estudiante – Lección 21

Guía de proyecto - tarjeta interactiva

Resumen

Vas a desarrollar una tarjeta digital interactiva para compartir con alguien que te importa, pero tendrás que planificar un poco antes de comenzar a programar.

Planificando tus Sprites

Usa la tabla a continuación para planificar sus sprites. (¡Necesitas al menos tres, pero puedes usar tantos como quieras!) Junto a cada elemento, diseña la imagen que usará y qué propiedades cambiarán.

Sprite etiqueta	Imagen(es)	Propiedades

Desarrollando interacciones

El elemento final de su tarjeta a considerar es cómo el usuario interactuará con él, y cómo los sprites pueden interactuar entre sí. Deberá incluir los condicionales que responden a la entrada del teclado (como keyDown ()), así como los condicionales que responden a variables cambiantes o propiedades de sprites (como sprite.y> 300). Usa la tabla a continuación para planificar todos sus condicionales y la acción correspondiente.

If / Else if / Else	Condición	Acción

Desarrolla tu tarjeta

Una vez que el profesor haya aprobado tu diseño, anda a Code Studio para programar tu tarjeta.

Verifique su tarjeta

Verifica tu tarjeta para asegurarte de que tenga todo lo que necesitas.

Reflexión

¿De qué parte de tu proyecto estás más orgulloso?

¿Por qué?

Si tuvieras más tiempo, ¿qué mejorarías de tu tarjeta?

Rúbrica - tarjeta interactiva

Rúbrica de tarjeta digital

Evalúa tu tarjeta digital de acuerdo con los siguientes criterios. Explica en qué parte de tu código se pueden encontrar los criterios en la columna de comentarios.

Criterio	sí/no	comentarios
Utiliza al menos 3 sprites		
Al menos un sprite responde a la entrada del usuario (ejemplo `keyDown`, `mouseDidMove`)		
Actualiza al menor 3 propiedades de sprites diferentes en el ciclo de extracción (ejemplo `sprite.x`, `sprite.scale`, `sprite.visible`)		
Utiliza al menos 1 condicional que se desencadena por una propiedad de variable o sprite (ejemplo `sprite.y > 300`)		
Incrementa a disminuye una variable o propiedad de sprite (eg `score = score + 1`)		
La tarjeta incluye texto que le dice al usuario cómo usar la tarjeta		

Prácticas de reflexión

Práctica	Cosas para celebrar	Cosas para trabajar
Problema resuelto		
Persistencia		
Creatividad		
Colaboración		
Comunicación		

Revisión por pares - tarjeta interactiva

Pre-Revisión

Nombre de los creadores:

Una cosa sobre la que quiero recibir comentarios es ...

Revisar sección

Nombre del revisor:

Preguntas	Clasificación	Comentarios
Hay al menos un sprite que responde a la entrada del usuario (ejemplo. `keyDown`, `mouseDidMove`)	✔ ✖	
Actualiza al menor 3 propiedades de sprites diferentes en el ciclo de extracción (ejemplo. `sprite.x`, `sprite.scale`, `sprite.visible`)	✔ ✖	
Utiliza al menos 1 condicional que se desencadena por una propiedad de variable o sprite (ejemplo. `sprite.y > 300`)	✔ ✖	
Incrementa a disminuye una variable o propiedad de sprite (eg `score = score + 1`)	✔ ✖	

Retroalimentación de respuesta

Me gustó

Deseo

Reflexión del creador

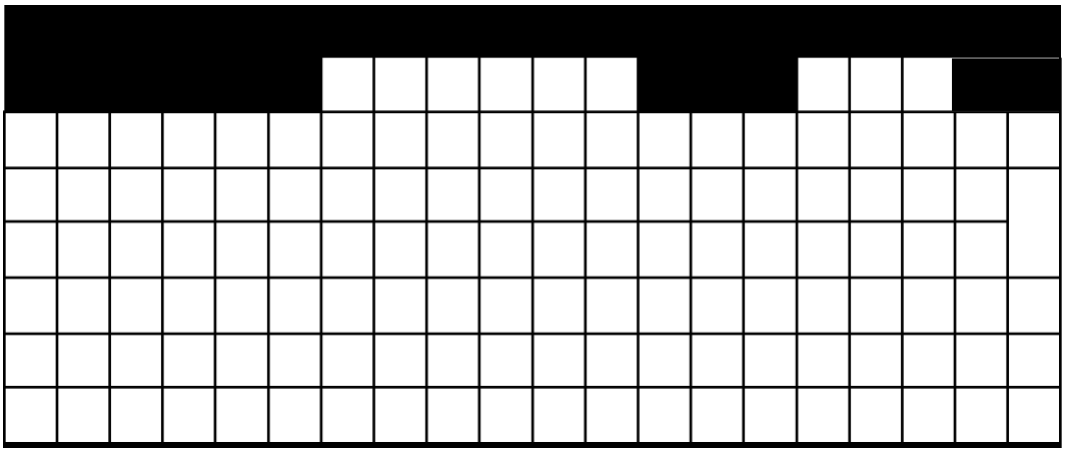
1. ¿Qué comentario te fue más útil? ¿Por qué?
2. ¿Qué comentario te sorprendió más? ¿Por qué?
3. En función de los comentarios, ¿qué cambios harías en tu tarjeta interactiva?

Detección de colisión - Recursos del estudiante – Lección 23

Colisiones A

Dibujar con Sprites

En la cuadrícula de abajo, dibuja dos sprites cuadrados en la línea inferior. Los sprites pueden estar tocándose o no. No dejes que tu pareja vea tus sprites.



Sprites propiedades

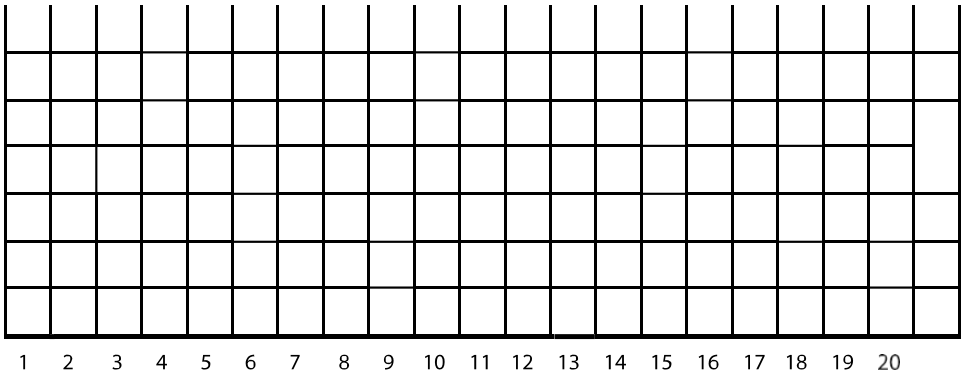
Usando la cuadrícula, encuentra la posición x del sprite (justo en el medio del cuadrado) y el ancho del sprite. Esta es la única información que debe decirle a su compañero sobre los sprites.

	x	anchura
sprite 1		
sprite 2		

Colisiones B

Dibuja tus Sprites

En la cuadrícula de abajo, dibuja dos sprites cuadrados en la línea de la izquierda. Los sprites pueden estar tocándose o no. No dejes que tu pareja vea tus sprites.



Sprites propiedades

Usando la cuadrícula, encuentra la posición del sprite (justo en el medio del cuadrado) y la altura del sprite.

	y	height
sprite 1		
sprite 2		

Colisiones A

Detección de colisión

Copia la información de la primera página en este cuadro, luego gira la imagen para que tu compañero no pueda echar un vistazo. El siguiente cuadro es la única información que tu compañero debería ver sobre tus sprites. Una vez que haya completado el primer cuadro, canjee esta hoja de trabajo con tu compañero para que ambos puedan probar sus detectores de colisión.

	x	anchura
sprite1		
sprite2		

¡Detente!

Cambia las hojas de trabajo con tu compañero antes de continuar.

Mira la información en el cuadro de arriba. ¿Puedes pensar en alguna forma en que puedas usar estos números para averiguar si los dos sprites se están tocando? Haz una lluvia de ideas a continuación.

¿Crees que los sprites se están tocando? ¿Por qué sí o por qué no?

Verifica el dibujo de su compañero para ver si estaba en lo correcto. ¿Crees que tu estrategia fue efectiva?

Colisiones B

Detector de colisiones

Copia la información de la primera página en este cuadro, luego gira la imagen para que tu compañero no pueda echar un vistazo. El siguiente cuadro es la única información que tu compañero debería ver sobre sus sprites. Una vez que hayas completado el primer cuadro, canjea esta hoja de trabajo con tu compañero para que ambos puedan probar sus detectores de colisión.

¡Detente!

	y	altura
sprite1		
sprite2		

Cambia las hojas de trabajo con tu compañero antes de continuar.

Mira la información en el cuadro de arriba. ¿Puedes pensar en alguna forma en que puedas usar estos números para averiguar si los dos sprites se están tocando? Haz una lluvia de ideas a continuación.

¿Crees que los sprites se están tocando? ¿Por qué?

Verifica el dibujo de tu compañero para ver si estaba en lo correcto. ¿Crees que tu estrategia fue efectiva?

El proceso de diseño de juego - Recursos del estudiante – Lección 27

Defender game

Visión en conjunto

Crear una pieza de software más grande, como un juego, puede volverse complejo rápidamente. Comenzar con un plan puede ayudarlo a mantenerse organizado e identificar problemas con anticipación. Mucho del trabajo que haces aquí hará que sea mucho más fácil hacer un seguimiento de lo que debe hacer una vez que comience a escribir su código real.

Jugabilidad y visuales

Comienza pensando en lo que realmente hace tu juego.

¿Cómo se ve?

¿Cómo lo juegas?

¿Qué lo hará divertido, interesante o relevante para el jugador?

Describe tu juego

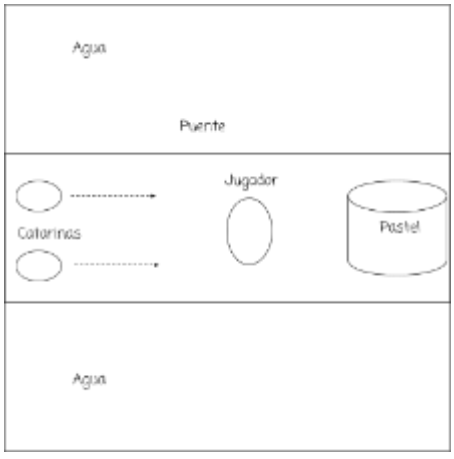
En un par de oraciones, describe el juego que vas a construir y cómo funcionará.

Eres un extraterrestre que defiende su pastel de las malvadas catarinas. Las catarinas marchan a través de un puente hacia su pastel. Tendrás que empujar a las catarinas fuera del puente hacia el agua. Obtenga puntos por detener a las catarinas y pierda puntos por dejar que las catarinas terminen.

Dibuja tu juego

Dibuja un boceto rápido de cómo funcionará tu juego ¿Quiénes son los personajes?

¿Cómo se ve el fondo? ¿Cómo se mueven las cosas? Etiqueta las cosas para que sean más claras.



Sprites y variables

Usando la descripción de tu juego anterior, averigua qué información y personajes necesitarás seguir durante tu juego. Complete una descripción para cada uno en el espacio a continuación.

Sprites

En la siguiente tabla, enumera información sobre los diferentes sprites en tu juego.

¿Dónde están ubicados? ¿Cómo se mueven? ¿Cómo interactúan con otros sprites?

Nombre (Etiqueta) y Apariencia	**Al inicio del juego** (Animación, posición, rotación, velocidad, velocidad de rotación)	**Usuario y Sprite e Interacciones** (¿El usuario controla este sprite? ¿Cómo se mueve? ¿Alguna vez necesita restablecer su posición? ¿Interactúa con otros sprites? ¿Cómo?)
Jugador - alien	x: 300 y: 200 Sin rotación o velocidad	Las teclas de flecha mueven el sprite en 4 direcciones Desplaza las catarinas
pastel - pastel	x: 350 y: 200 Sin rotación o velocidad	Nunca se mueve Cuando las mariquitas te tocan pierdes puntos
enemigo 1 - catarina	x: 0 y: aleatorio entre 150 y 250 velocidadX: 2 (puede necesitar hacer más rápido)	Se mueve a través de la pantalla de izquierda a derecha El jugador sprite puede desplazar a las catarinas Si presionan el restablecimiento del agua y le dan al jugador un punto. Si alcanzan el reinicio del pastel, quita 5 puntos
enemigo 2 - catarinas	x: 0 y: aleatorio entre 150 y 250 velocidadX: 2 (puede necesitar hacer más rápido)	Igual que el otro

Variables

Piensa en la información que tu juego necesita para realizar un seguimiento. ¿Hay un puntaje? ¿Una cantidad de vidas? Describe cada variable en el espacio a continuación.

Nombre	**Lo que hace un seguimiento de**	**¿Cómo cambia durante el juego?** (¿Cuál es el valor inicial, cuándo cambiará?)
Puntuación	Puntuación de jugadores	Comienza en 0. Cuando las mariquitas empujadas en el agua obtienen 1 punto. Cuando las mariquitas golpean el pastel pierden 5 puntos.

Funciones

Tu ciclo de extracción no debería tener un código complejo. En cambio, divide tu programa en los principales pasos que necesitarás para que tu juego funcione. Los diferentes comportamientos que describiste para sus sprites y variables deberían ayudarte a decidir cuáles deberían ser estos pasos. Luego, describe qué debería hacer el código para esa función

Nombre de la función	Que sucede en esta función ¿Qué comportamientos detallaste para tus sprites incluye esta función? ¿Se puede usar esta función en varios lugares de tu programa?
gameBackground()	*Dibuja el fondo del juego*
enemiesTouchCake()	*Restablece a los enemigos cuando tocan el pastel. Aumenta el puntaje del jugador.*
movePlayer()	*Mueve al jugador con las teclas de flecha y cambia las animaciones*
displaceEnemies()	*jugador empuja a las catarinas*
enemiesTouchWater()	*Si alguna catarina toca el agua, se restablece y la puntuación del jugador aumenta*
showScore()	*Muestra el puntaje en las pantalla*

Uso del proceso de diseño de juego - Recursos del estudiante – Lección 28

Planificación de su plataforma de juego

Visión en conjunto

Crear una pieza de software más grande, como un juego, puede volverse complejo rápidamente. Comenzar con un plan puede ayudar a mantenerse organizado e identificar problemas con anticipación. Mucho del trabajo que hace aquí hará que sea mucho más fácil hacer un seguimiento de lo que debe hacer una vez que comience a escribir su código real.

Jugabilidad y visuales

Comienza pensando en lo que realmente hace tu juego ¿Cómo se ve? ¿Cómo lo juegas? ¿Qué lo hará divertido, interesante o relevante para el jugador?

Describe tu juego

En un par de oraciones, describe el juego que vas a construir y cómo funcionará.

Antecedentes

Dibuja un boceto rápido de cómo quieres que se vea el fondo (s) de tu juego.

Sprites y variables

Usando la descripción de tu juego anterior, averigua qué información y personajes necesitarás seguir durante tu juego. Complete una descripción para cada uno en el espacio a continuación.

Sprites

En la siguiente tabla, enumera información sobre los diferentes sprites en tu juego.

¿Dónde están ubicados? ¿Cómo se mueven? ¿Cómo interactúan con otros sprites?

Variables

Nombre (Etiqueta) y Apariencia	**Al inicio del juego** (Animación, posición, rotación, velocidad, velocidad de rotación)	**Usuario, Sprite e Interacciones** (¿El usuario controla este sprite? ¿Cómo se mueve? ¿Alguna vez necesita restablecer su posición? ¿Interactúa con otros sprites? ¿Cómo?)

Piensa en la información que tu juego necesita para realizar un seguimiento. ¿Hay un puntaje? ¿Una cantidad de vidas? Describe cada variable

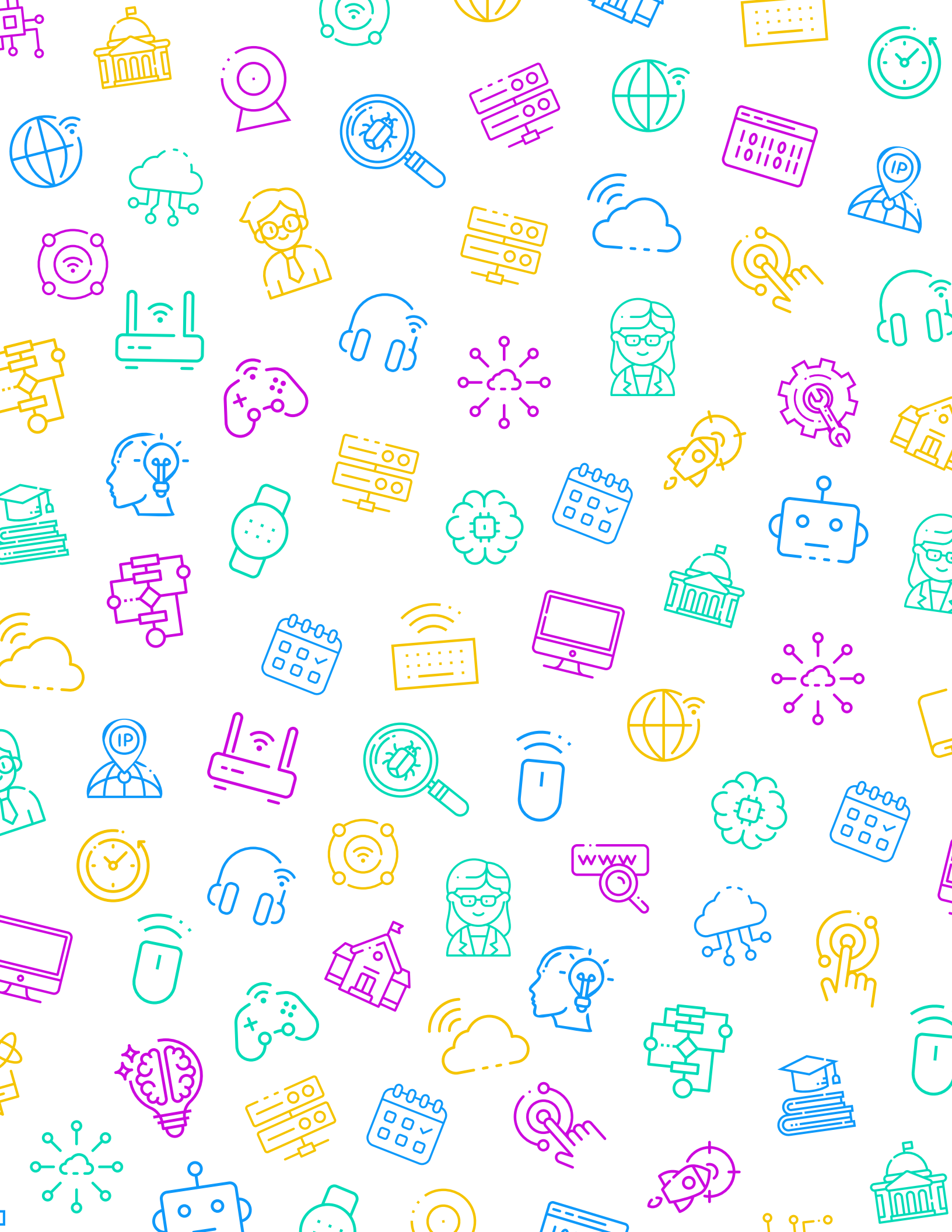
Nombre (Etiqueta)	**Lo que hace un seguimiento de**	**¿Cómo cambia durante el juego?** (¿Cuál es el valor inicial, cuándo cambiará?)

en el espacio a continuación.

Funciones

Tu ciclo de extracción no debería tener un código complejo. En cambio, divide tu programa en los principales pasos que necesitarás para que tu juego funcione. Los diferentes comportamientos que describiste para tus sprites y variables deberían ayudarte a decidir cuáles deberían ser estos pasos. Luego, describe qué debería hacer el código para esa función.

Nombre de la función	**Qué sucede en esta función** ¿Qué comportamientos detallaste para tus sprites incluye esta función? ¿Se puede usar esta función en varios lugares de tu programa?


IP
www

Ideo Digital

Ciencias de la Computación en el aula