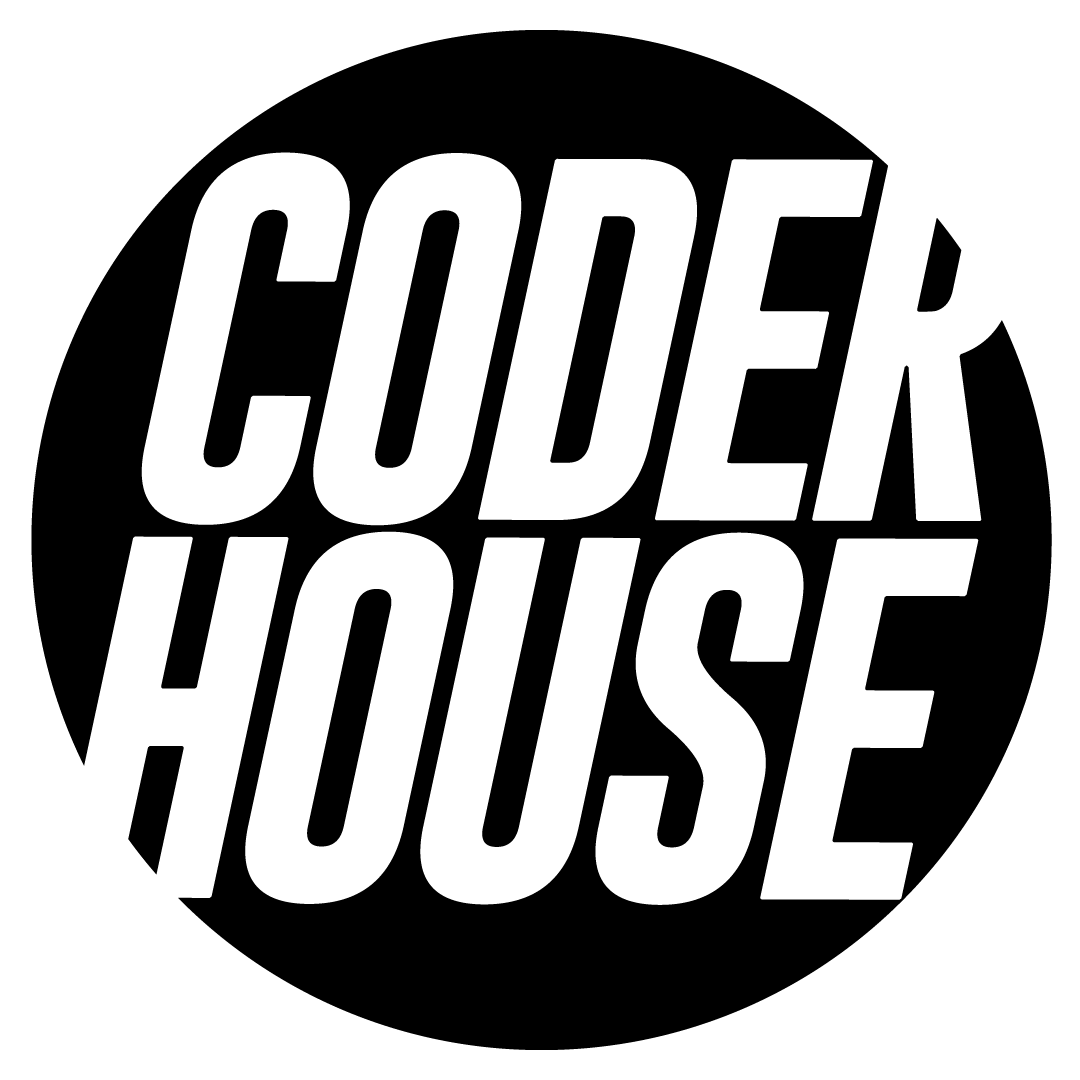
*JavaScript*

*MANUAL DE DESAFÍOS*



| Desafío 1 | *CREAR UN ALGORITMO JS SIMPLE* | |
| --- | --- | --- |
| Clase 1 | **Formato:** código fuente en JavaScript. Debe identificar el apellido del alumno/a en el nombre de archivo por “claseApellido”.  **Sugerencia:** usamos prompt() para solicitar datos al usuario y console.log() o alert() para mostrar el resultado de las operaciones realizadas con esos datos. Si vas a sumar una entrada tene en cuenta que tenes que parsearla antes. Usando parseInt() o parseFloat() | ENTREGABLE |
| **Consigna**:  Crea un script en JS que le solicite al usuario ingresar uno o más datos. Luego, con JavaScript, realiza operaciones matemáticas o de concatenación sobre las entradas teniendo en cuenta el tipo de dato. Al finalizar mostrar el resultados con alert() o console.log() | | |
| **Aspectos a incluir en el entregable:**   * *Archivo HTML con código JavaScript entre etiquetas <script></script>, que incluya la definición de un algoritmo.* | | |
| **Ejemplo:**  Pedir nombre mediante prompt y mostrarlo en consola junto con algún texto de saludo. Ejemplo: ¡Hola, Juan!  Pedir un número mediante prompt, parsearlo, sumarlo a otro que se encuentre almacenado en una variable y luego mostrar el resultado en consola.  Pedir un texto mediante prompt, luego otro, concatenarlos y mostrarlo en un alerta.  Haz [clic aquí](https://drive.google.com/drive/folders/1QFGnVaLCGqNVwBqupiugLhRfS20gBVUm?usp=sharing) para ver más. | | |

| Desafío 2 | *CREAR UN ALGORITMO CON UN CONDICIONAL* | |
| --- | --- | --- |
| Clase 2 | **Formato:** página HTML y código fuente en JavaScript. Debe identificar el apellido del alumno/a en el nombre de archivo comprimido por “claseApellido”.  **Sugerencia:** tener en cuenta que los valores obtenidos por prompt() son string, si se busca operar con números hay que parsearlos antes y si van a usar cadenas recordar tener cuidado con mayúsculas y minúsculas en las comparaciones de igualdad. (Ej. “Hola” y “HOLA” no son iguales) | ENTREGABLE |
| **Consigna**:  Archivo HTML y Archivo JS, referenciado en el HTML por etiqueta <script src="js/miarchivo.js"></script>, que incluya la definición de un algoritmo en JavaScript que emplee instrucciones condicionales. | | |
| **Aspectos a incluir en el entregable:**   * *Pedir número mediante prompt y si es mayor a 1000 mostrar un alert.* * *Pedir un texto mediante prompt, y si es igual a "Hola" mostrar un alerta por consola.* * *Pedir un número por prompt y evaluar si está entre 10 y 50. En caso positivo mostrar un alert.* | | |
| **Ejemplo:**  Haz [clic aquí](https://drive.google.com/drive/folders/1db9DIBuayzOcyqMylJy1AqOYGnJSMe2S?usp=sharing) para verlo. | | |

| Desafío 3 | *CREAR UN ALGORITMO UTILIZANDO UN CICLO* | |
| --- | --- | --- |
| Clase 3 | **Formato:** página HTML y código fuente en JavaScript. Debe identificar el apellido del alumno/a en el nombre de archivo comprimido por “claseApellido”.  **Sugerencia:** usamos la instrucción for para repetir un número fijo de veces. Mientras que usamos while cuando queremos repetir algo hasta que se deje de cumplir una condición. | ENTREGABLE |
| **Consigna**:  Tomando como base los ejemplos anteriores de la estructura for y while, crear un algoritmo que repita un bloque de instrucciones. En cada repetición es necesario efectuar una operación o comparación para obtener una salida por alerta o consola. | | |
| **Aspectos a incluir en el entregable:**   * *Archivo HTML y Archivo JS, referenciado en el HTML por etiqueta <script src="js/miarchivo.js"></script>, que incluya la definición de un algoritmo en JavaScript que emplee bucles e instrucciones condicionales.* | | |
| **Ejemplo:**  Pedir número mediante prompt y sumarle otro número en cada repetición,realizando una salida por cada resultado  Pedir un texto mediante prompt, concatenar un valor en cada repetición, realizando una salida por cada resultado, hasta que se ingresa “ESC”.  Pedir un número por prompt, repetir la salida del mensaje “Hola” la cantidad de veces ingresada.  Haz [clic aquí](https://drive.google.com/drive/folders/1hpiuoZ8ENiru-ULj7zXDt6Q6Xays6VN1?usp=sharing) para ver más. | | |

| Desafío 4 | *SIMULADOR INTERACTIVO* | |
| --- | --- | --- |
| Clase 4 | **Formato:** página HTML y código fuente en JavaScript. Debe identificar el apellido del alumno/a en el nombre de archivo comprimido por “claseApellido”.  **Sugerencia:** algunos criterios a tener en cuenta para seleccionar un proceso a simular por primera vez son:  “ELEGIR UN PROCESO BIEN CONOCIDO” : Si conozco una situación que implique adquirir cierta información y estoy bien familiarizado en “cómo se hace” es más fácil traducir la solución a un lenguaje de programación.  “ELEGIR UN PROCESO QUE ME RESULTE INTERESANTE” : Si me siento motivado sobre el tema, es más llevadero enfrentar los retos de desarrollo e interpretación: Antes de programar existe la etapa de relevamiento y análisis que me permite identificar cómo solucionar el proceso. | ENTREGABLE |
| **Consigna**:  Con los conocimientos vistos hasta el momento, empezarás a armar la estructura inicial de tu proyecto integrador. A partir de los ejemplos mostrados la primera clase, deberás:  Pensar el alcance de tu proyecto: ¿usarás un cotizador de seguros? ¿un simulador de créditos? ¿un simulador personalizado?  Armar la estructura HTML del proyecto.  Incorporar al menos un prompt para pedir un dato y luego mostrarlo mediante alert realizando alguna operación.  Utilizar funciones para realizar esas operaciones. | | |
| **Aspectos a incluir en el entregable:**   * *Archivo HTML y Archivo JS, referenciado en el HTML por etiqueta <script src="js/miarchivo.js"></script>, que incluya la definición de un algoritmo en JavaScript que emplee funciones para resolver el procesamiento principal del simulador* | | |
| **Ejemplo:**  Calcular costo total de productos y/o servicios seleccionados por el usuario.  Calcular pagos en cuotas sobre un monto determinado.  Calcular valor final de un producto seleccionado en función de impuestos y descuentos.  Calcular tiempo de espera promedio en relación a la cantidad de turnos registrados.  Calcular edad promedio de personas registradas.  Calcular nota final de alumnos ingresados.  Haz [clic aquí](https://drive.google.com/drive/folders/1EL5aSU-fVvJb5xM6TZ_febr6dWg_kTsn?usp=sharing) para ver más. | | |

| Desafío complementario | *FUNCIONES RELACIONADAS* | |
| --- | --- | --- |
| Clase 4 | **Formato:** código fuente en JavaScript  [Sublime Text](https://www.sublimetext.com/3) o [VisualStudio](https://visualstudio.microsoft.com/es/downloads/?rr=https%3A%2F%2Fwww.google.com%2F).  **Sugerencia:** para llevar adelante esta tarea, te sugerimos pensar un proceso complejo, descomponerlo al menos en tres partes, y realizar una función que se encargue de cada una de ellas. | COMPLEMENTARIO |
| **Consigna**:  Codifica al menos tres funciones cuyas instrucciones permitan resolver un problema particular, segmentado en tareas. La información a procesar debe ser ingresada por el usuario, y el resultado del procesamiento visualizado en una salida. | | |
| **Aspectos a incluir en el entregable:**   * *Archivo HTML y archivo JavaScript referenciados, que incluyan la definición y llamada de al menos tres funciones.* | | |
| **Ejemplo:**   * Ejemplo de procesamiento: cálculo de IVA   1) Ingresar precio de costo - 2) Sumar IVA - 3) Mostrar precio calculado.   * Ejemplo de procesamiento: determinar si un número es múltiplo   1) Ingresar números a verificar - 2) ¿Es múltiplo? - 3) Mostrar resultado.  Haz [clic aquí](https://drive.google.com/drive/folders/1EL5aSU-fVvJb5xM6TZ_febr6dWg_kTsn?usp=sharing) para ver más. | | |

| Desafío 5 | *INCORPORAR OBJETOS* | |
| --- | --- | --- |
| Clase 5 | **Formato:** página HTML y código fuente en JavaScript. Debe identificar el apellido del alumno/a en el nombre de archivo comprimido por “claseApellido”.  **Sugerencia:** reconocer elementos en el simulador cuya información está compuesta por más de un valor y existen operaciones comunes (funciones) para todos los elementos de este tipo y sus propiedades. | ENTREGABLE |
| **Consigna**:  A partir de los ejemplos mostrados la primera clase, y en función del tipo de simulador que hayas elegido, deberás:   * Crear al menos un objeto para controlar el funcionamiento de tu simulador. * Incorporarle sus propiedades y su constructor. * Invocar a ese objeto en algún momento donde el usuario realice alguna acción. * Utilizar sus mètodos. | | |
| **Aspectos a tener en cuenta en el entregable:**   * Archivo HTML y Archivo JS, referenciado en el HTML por etiqueta <script src="js/miarchivo.js"></script>, que incluya la definición de un algoritmo en JavaScript que emplee objetos para elementos con propiedades y mètodos comunes. | | |
| **Ejemplo:**  Algunos objetos a identificar que forman parte del simulador pueden ser: producto, persona, libro, auto, comida, bebida, tarea, etc.  Haz [clic aquí](https://drive.google.com/drive/folders/12JIg3HcdLOrqCcDBJzbR0_PCQpdryrYp?usp=sharing) para ver más. | | |

| Desafío 6 | *INCORPORAR ARRAYS* | |
| --- | --- | --- |
| Clase 6 | **Formato:** página HTML y código fuente en JavaScript. Debe identificar el apellido del estudiante en el nombre de archivo comprimido por “claseApellido”.  **Sugerencia:** los Array cumplen el papel de listas en el programa. Principalmente, los usamos para agrupar elementos de un mismo tipo. Siempre que sea posible emplear los métodos disponibles para trabajar con ellos | ENTREGABLE |
| **Consigna**:  Traslada al proyecto integrador el concepto de objetos, visto en la clase de hoy. A partir de los ejemplos mostrados la primera clase, y en función del tipo de simulador que hayas elegido, deberás:   * Incorporar al menos un Array en tu proyecto. * Utilizar algunos de los métodos o propiedades vistos en la clase. | | |
| **Aspectos a incluir en el entregable:**   * Archivo HTML y Archivo JS, referenciado en el HTML por etiqueta <script src="js/miarchivo.js"></script>, que incluya la definición de un algoritmo en JavaScript que emplee array para agrupar elementos similares. | | |
| **Ejemplo:**  Podemos crear arrays para los objetos identificados en el simulador la clase anterior, Ejemplo: Array de Productos, Array de Personas,Array de Libros, Array de Autos, Array de Comidas, Array de Bebidas, Array de Tareas, etc.  Haz clic aquí para ver más. | | |

| Desafío complementario | *ORDENAR UN ARRAY DE OBJETOS* | |
| --- | --- | --- |
| Clase 6 | **Formato:** código fuente en JavaScript  [Sublime Text](https://www.sublimetext.com/3) o [VisualStudio](https://visualstudio.microsoft.com/es/downloads/?rr=https%3A%2F%2Fwww.google.com%2F).  **Sugerencia:** recuerda que para ordenar una estructura de datos, los elementos deben ser del mismo tipo. Puedes emplear la función sort() para armar el algoritmo. <https://developer.mozilla.org/es/docs/Web/JavaScript/Referencia/Objetos_globales/Array/sort> | COMPLEMENTARIO |
| **Consigna**:  Codifica una función cuyas instrucciones permitan ordenar una colección (array). Preferentemente, recibir el criterio de ordenamiento por parámetro, y mostrar el resultado del procesamiento en una salida. | | |
| **Aspectos a incluir en el entregable:**   * *Archivo HTML y archivo JavaScript referenciado, que incluya la definición un array de objetos, la declaración y llamada de una función que ordene la colección.* | | |
| **Ejemplo de criterio de ordenamiento:**  1) Array de objetos “Productos”. Ordenar por menor precio.  2) Array de objetos “Personas”. Ordenar por mayor edad.  3) Array de objetos “Date”. Ordenar por menor fecha.  Haz [clic aquí](https://drive.google.com/drive/folders/1gKoGXL2tlpzeMYVYxM6MOItFbRT2rDAT?usp=sharing) para ver más. | | |

| Proyecto final | *PRIMERA ENTREGA DEL PROYECTO FINAL* | |
| --- | --- | --- |
| Clase 6 | **Formato:** página HTML y código fuente en JavaScript. Debe identificar el apellido del alumno/a en el nombre de archivo comprimido por “claseApellido”.  **Sugerencia:** si bien por el momento sólo podemos hacer entradas con prompt(), y salidas con alert() o console.log(), es suficiente para empezar a pensar el proceso a simular en términos de entradas, variables, estructuras, funciones, métodos y salidas. Verificar Rúbrica | PROYECTO FINAL |
| **>>Objetivos generales:**   1. Codificar la funcionalidad inicial del simulador. 2. Identificar el flujo de trabajo del script en términos de captura de entradas ingresadas por el usuario, procesamiento esencial del simulador y notificación de resultados en forma de salida.   **>>Objetivos específicos:**   1. Capturar entradas mediante prompt(). 2. Declarar variables y objetos necesarios para simular el proceso seleccionado. 3. Crear funciones y/o métodos para realizar operaciones (suma, resta, concatenación, división, porcentaje, etc). 4. Efectuar una salida, que es el resultado de los datos procesados, la cual puede hacerse por alert() o console.log().   **>>Se debe entregar:**   * Estructura HTML del proyecto. * Variables de JS necesarias. * Funciones esenciales del proceso a simular. * Objetos de JS. | | |
| **Ejemplo:**  Haz [clic aquí](https://drive.google.com/drive/folders/1gKoGXL2tlpzeMYVYxM6MOItFbRT2rDAT?usp=sharing) para verlo. | | |

| Desafío genérico | *EJERCITAR JSON Y STORAGE* | |
| --- | --- | --- |
| Clase 7 | **Cuentas con 40 minutos para realizar la actividad.** | GENÉRICO (NO ENTREGABLE) |
| **Consigna**:  Realiza un algoritmo que almacene información en Storage y guarde un array de objetos en formato JSON. | | |
| **Aspectos a tener en cuenta:**  *Deberá cumplir los siguientes requisitos:*   * *Almacenar en Storage información ingresada por el usuario. Puede ser un texto, números, o combinación. Luego mostrarla mediante alert o console.* * *Declarar un array de objetos (literales, con funciòn constructora o con clases) y almacenar el array en formato JSON en el storage.* | | |

| Desafío 7 | *INTERACTUAR CON HTML* | |
| --- | --- | --- |
| Clase 8 | **Formato:** página HTML y código fuente en JavaScript. Debe identificar el apellido del estudiante en el nombre de archivo comprimido por “claseApellido”.  **Sugerencia:** generalmente, identificamos a un único elemento del DOM con el atributo id y a un conjunto asociado por class. | ENTREGABLE |
| **Consigna**:  Traslada al proyecto integrador el concepto de objetos, visto en la clase de hoy. En función del tipo de simulador que hayas elegido, deberás:  Crear elementos manipulando el DOM a partir de la informaciòn de tus objetos.  Modificar etiquetas existentes en función del resultado de operaciones. | | |
| **Aspectos a incluir en el entregable:**   * Archivo HTML y Archivo JS, referenciado en el HTML por etiqueta <script src="js/miarchivo.js"></script>, que incluya la definición de un algoritmo en JavaScript que opere sobre el DOM, modificando, agregando o eliminando elementos. | | |
| **Ejemplo:**  Podemos crear elementos HTML en función del listado de nuestros objetos identificados en la clase 6.  Establecer un mensaje de bienvenida aleatorio usando un array de mensajes.  Capturar una o màs entradas por promp() y mostrarlas en el HTML, modificando el DOM  Haz [clic aquí](https://drive.google.com/drive/folders/1CN6IPbKfGrvQB4AX7LhYO8HNmmITEDQr?usp=sharing) para ver más. | | |

| Desafío 8 | *INCORPORAR EVENTOS* | |
| --- | --- | --- |
| Clase 9 | **Formato:** página HTML y código fuente en JavaScript. Debe identificar el apellido del estudiante en el nombre de archivo comprimido por “claseApellido”  **Sugerencia:** es posible asociar más de un evento a un elemento y se pueden emplear funciòn comunes, anónimas y arrow para los manejadores de eventos. | ENTREGABLE |
| **Consigna**:  Con lo que vimos sobre DOM, ahora puedes sumarlo a tu proyecto, para interactuar entre los elementos HTML y JS. Es decir, asociar eventos que buscamos controlar sobre los elementos de la interfaz de nuestro simulador | | |
| **Aspectos a tener en cuenta en el entregable:**   * Archivo HTML y Archivo JS, referenciado en el HTML por etiqueta <script src="js/miarchivo.js"></script>, que incluya la definición de un algoritmo en JavaScript que opere sobre el DOM manejando eventos. | | |
| **Ejemplo:**   * Cuando el usuario completa algún dato, por ejemplo cantidad de cuotas, se captura ese dato y se agregan elementos al DOM mediante JS. * Capturar la tecla ENTER para confirmar alguna acción.   Haz [clic aquí](https://drive.google.com/drive/folders/1mo9KBm_fC5WgJvd1lqcrGWVt3uNoW4oo?usp=sharing) para ver más. | | |

| Desafío complementario | *GENERAR HTML* | |
| --- | --- | --- |
| Clase 9 | **Formato:** código fuente en JavaScript  [Sublime Text](https://www.sublimetext.com/3) o [VisualStudio](https://visualstudio.microsoft.com/es/downloads/?rr=https%3A%2F%2Fwww.google.com%2F).  **Sugerencia:** en esta actividad, puedes generar los nuevos elementos del DOM mientras recorres un array de objetos, creando en cada iteración (con createElement) uno o más elemento html, agregándolos como hijos al body u otro nodo (con appendChild). | COMPLEMENTARIO |
| **Consigna**:  Codifica un script cuyas instrucciones permitan generar de forma dinámica una sección del HTML. Los valores que alimentan este proceso comprenden una array de objetos, cuyos datos deberán incluirse empleando métodos del DOM y elementos apropiados para su representación. | | |
| **Aspectos a tener en cuenta:**   * *Archivo HTML y archivo JavaScript referenciado, que incluya la definición un array de objetos, la declaración y llamada de una función que genere proceduralmente una sección del HTML.* | | |
| **Ejemplo de estructura HTML resultante:**   * 1) Generar títulos y párrafos a partir de un array de “Publicaciones”. * 2) Generar cards y botones a partir de un array de “Productos”. * 3) Generar imágenes y badges a partir de un array de “Personas”.   Haz [clic aquí](https://drive.google.com/drive/folders/1mo9KBm_fC5WgJvd1lqcrGWVt3uNoW4oo?usp=sharing) para ver más. | | |

| Proyecto final | *SEGUNDA ENTREGA DEL PROYECTO FINAL* | |
| --- | --- | --- |
| Clase 10 | **Formato:** página HTML y código fuente en JavaScript. Debe identificar el apellido del alumno/a en el nombre de archivo comprimido por “claseApellido”.  **Sugerencia:** en la segunda entrega buscamos programar el código esencial para garantizar dinamismo en el HTML con JavaScript. En relación a la primera entrega, ya no usamos alert() como salida y promt() como entrada, ahora modificamos el DOM para las salidas y capturamos los eventos del usuario sobre inputs y botones para las entradas. | PROYECTO FINAL |
| **>>Objetivos generales:**   1. Codificar funciones de procesos esenciales y notificación de resultados por HTML, añadiendo interacción al simulador. 2. Ampliar y refinar el flujo de trabajo del script en términos de captura de eventos, procesamiento del simulador y notificación de resultados en forma de salidas por HTML, modificando el DOM.   **>>Objetivos específicos:**   1. Definir eventos a manejar y su funciòn de respuesta. 2. Declarar una estructura de datos de tipo JSON, para definir datos iniciales a consumir por el simulador. 3. Modificar el DOM, ya sea para definir elementos al cargar la página o para realizar salidas de un procesamiento. 4. Almacenar datos (clave-valor) en el Storage y recuperarlos.   **>>Se debe entregar:**  Implementación con uso de JSON y Storage.  Modificación del DOM y detección de eventos de usuario. | | |
| **Ejemplo:**  Haz [clic aquí](https://drive.google.com/drive/folders/1yT67a7d3Os06lTnrUd0dGybjOwHqZUkt?usp=sharing) para verlo. | | |

| Desafío genérico | *JQUERY* | |
| --- | --- | --- |
| Clase 11 | **Cuentas con 20 minutos para realizar la actividad.** | GENÉRICO (NO ENTREGABLE) |
| **Consigna**:  Desarrolla un algoritmo utilizando HTML, JS y JQuery. | | |
| **Aspectos a tener en cuenta:**   * *Crea una pantalla HTML con diferentes etiquetas y cargar JQuery utilizando un CDN. Utilizar el método append() para incorporar algún elemento a la pantalla modificando el DOM.* * *Prueba también cargando jQuery de manera local.* | | |

| Desafío 9 | *INCORPORAR JQUERY AL PROYECTO* | |
| --- | --- | --- |
| Clase 12 | **Formato:** página HTML y código fuente en JavaScript. Debe identificar el apellido del alumno/a en el nombre de archivo comprimido por “claseApellido”  **Sugerencia:** recuerda que jQuery es una librería que simplifica la notación JS. Es posible reemplazar con selectores todos los métodos nativos de acceso al DOM. Así como reemplazar toda definición de eventos de vanilla JS por on o métodos shortcut. | ENTREGABLE |
| **Consigna**:  Sumar al proyecto integrador los conceptos de jQuery que vimos en las últimas dos clases:  -Utilizar métodos jQuery para incorporar elementos al DOM.  -Utilizar métodos jQuery para determinar respuesta a ciertos eventos. | | |
| **Aspectos a incluir en el entregable:**   * Archivo HTML y Archivo JS, referenciado en el HTML por etiqueta <script src="js/miarchivo.js"></script>, que incluya la definición de un algoritmo en JavaScript con mètodos jQuery para seleccionar, agregar y definir eventos. | | |
| **Ejemplo:**  Manejo de eventos del proyecto: clicks del usuario, cambios en inputs, selectores, etc  Cualquier modificación que necesites hacer sobre el DOM con la página ya cargada: por ejemplo, al seleccionar una opción de un selector aparece una alerta en HTML dando cierta información.  Capturar el evento asociado a presionar ENTER para confirmar el envío de los datos.  Haz [clic aquí](https://drive.google.com/drive/folders/1w48MvFSg8rD-8pbTzC64suILdMsCjIs4?usp=sharing) para ver más. | | |

| Desafío genérico | *ANIMACIONES EN JQUERY* | |
| --- | --- | --- |
| Clase 13 | **Cuentas con 20 minutos para realizar la actividad.** | GENÉRICO (NO ENTREGABLE) |
| **Consigna**:  Desarrolla un algoritmo utilizando HTML, JS y JQuery con animaciones. | | |
| **Aspectos a tener en cuenta:**  *Elimina un elemento usando una animación de jQuery:*  *El método removeElementWithAnimation() tiene que recibir como parámetro un Id y eliminar el elemento, usando en conjunto una animación de jQuery para que el cambio no sea tan brusco.* | | |

| Desafío complementario | *ANIMACIONES ENCADENADAS* | |
| --- | --- | --- |
| Clase 13 | **Formato:** código fuente en JavaScript  Sublime Text o VisualStudio.  **Sugerencia:** ten en cuenta que los métodos de jQuery show, hide, fadeIn, fadeOut, slideUp, slideDown y toggle ofrecen animaciones con comportamiento predefinido. Si quieres armar tus propias animaciones, deberás usar animate.  También recuerda que los métodos jQuery pueden encadenarse:  http://www.w3big.com/es/jquery/jquery-chaining.html | COMPLEMENTARIO |
| **Consigna**:  Codifica animaciones concatenadas sobre uno o más elementos. Es decir que luego de finalizar una animación en su función callback, se especifica la llamada a otra animación. | | |
| **Aspectos a incluir en el entregable:**   * *Archivo HTML y archivo JavaScript referenciado, que incluya la definición de dos o más animaciones y sus respectivas funciones callback.* | | |
| **Ejemplo de secuencia de animación:**   * 1) Mostrar un elemento con fadeIn() y al concluir su transición, ocultar otro elemento con fadeOut(). * 2) Aumentar el margen de un elemento con animate() y al concluir su transición, disminuir el margen del mismo elemento con animate(). * 3) Disminuir la altura de un elemento con animate() y al concluir su transición, esperar con delay() unos ms, y volver al tamaño original.   Haz [clic aquí](https://drive.google.com/drive/folders/1_n6blKkQCgYf1hKsaeqSsgu2Iq9KDoFY?usp=sharing) para ver más. | | |

| Desafío 10 | *AJAX EN TU PROYECTO* | |
| --- | --- | --- |
| Clase 14 | **Formato:** página HTML y código fuente en JavaScript. Debe identificar el apellido del alumno/a en el nombre de archivo comprimido por “claseApellido”  **Sugerencia:** usamos GET para obtener información del servidor y POST para enviar. getJSON es la forma simplificada de obtener la información de un archivo JSON. | ENTREGABLE |
| **Consigna**:  Tomando como ejemplo la llamada AJAX con JSON, incorpora al proyecto del simulador al menos una llamada AJAX, que mediante la interacción del usuario (un botón o envío de formulario), se realice la llamada y la misma responda un JSON estático. incluir esa respuesta en el HTML. | | |
| **Aspectos a tener en cuenta en el entregable:**   * Archivo HTML y Archivo JS, referenciado en el HTML por etiqueta <script src="js/miarchivo.js"></script>, que incluya la definición de un algoritmo en JavaScript con mètodos jQuery con al menos una peticiòn ajax. | | |
| **Ejemplo:**   * Al confirmar el envío del formulario,sin refrescar la página, mostrar alerta de "envío exitoso" * Consultar usuarios desde [JSONPlaceholder](https://jsonplaceholder.typicode.com/) y mostrarlos en el HTML. * Consultar personajes desde la [API de Harry Potter](https://hp-api.herokuapp.com/) y mostrarlos en el HTML.   Haz [clic aquí](https://drive.google.com/drive/folders/1UzdLwDAdzvIhZSdN6hV9fm-07acIuh4i?usp=sharing) para ver más. | | |

| Proyecto final | *TERCERA ENTREGA DEL PROYECTO FINAL* | |
| --- | --- | --- |
| Clase 14 | **Formato:** página HTML y código fuente en JavaScript. Debe identificar el apellido del alumno/a en el nombre de archivo comprimido por “claseApellido”.  **Sugerencia:** en la tercera entrega buscamos que puedan definir el alcance del simulador y sus interacciones esenciales, refinando algunos aspectos trabajados en las entregas anteriores, empleando las herramientas ofrecidas por la librería jQuery. | PROYECTO FINAL |
| **>>Objetivos generales:**   1. Refinar la codificación del simulador y su interacción con el usuario. 2. Construir estructura y estilo final de la aplicación.   **>>Objetivos específicos:**   1. Definir estructura completa del simulador e interfaces con HTML. 2. Declarar reglas de estilo y/o utilizar un framework CSS para estilar la aplicación. 3. Emplear jQuery para definir modifcaciòn del DOM, efectos y animaciones.   **>>Se debe entregar:**   * Maquetación del HTML y eventos a manejar en él. * Incorporación de jQuery para acceder y agregar elementos al DOM. * Implementación de efectos y animaciones. * Estilos base CSS del simulador | | |
| **Ejemplo:**  Haz [clic aquí](https://drive.google.com/drive/folders/1UzdLwDAdzvIhZSdN6hV9fm-07acIuh4i?usp=sharing) para verlo. | | |

| Proyecto final | *ENTREGA FINAL DEL PROYECTO* | |
| --- | --- | --- |
| Clase 16 | **Recuerda que cuentas con 20 días (de corrido) para subir tu Proyecto Final. Pasado este plazo, el botón de entrega se inhabilitará.** | PROYECTO FINAL |
| * Deberás entregar el simulador final funcionando, en un archivo HTML con sus archivos JS complementarios. | | |
| **Ejemplo:**  Haz [clic aquí](https://drive.google.com/drive/folders/13dvmun5UkTT5Gx4ZAMLAxUjSXg_EFCQH?usp=sharing) para verlo. | | |