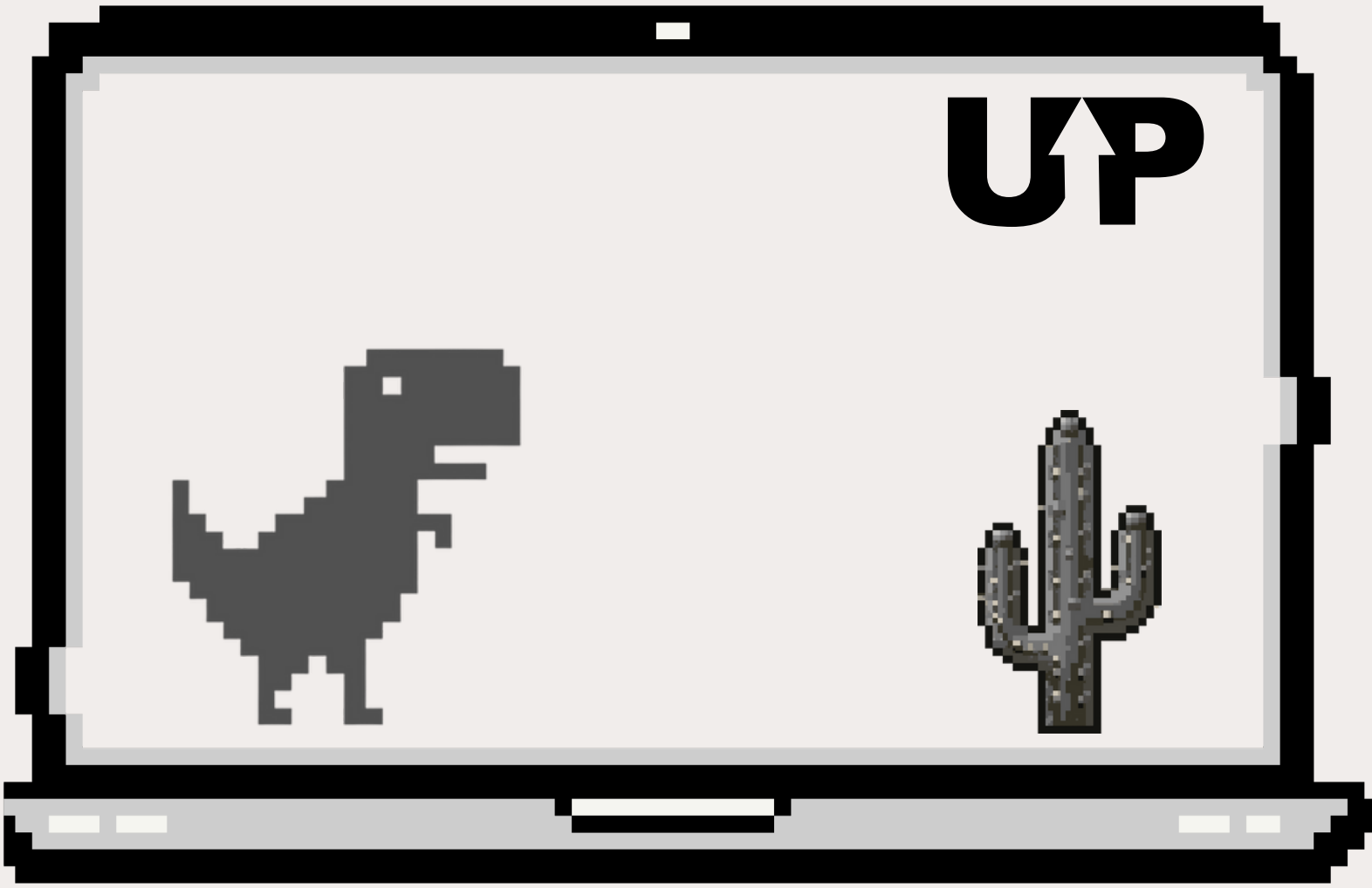


DESARROLLAO:

YAZU

BAUTISTA

DISEÑO Y MAQUETACIO:

RODRIGOALEJANDRO

DINO

Manual Detallado del Juego “DINO RUNNER”

1. Introducción

El juego “DINO RUNNER” es una aplicación interactiva desarrollada con HTML, CSS y JavaScript. En él, el jugador controla un dinosaurio que debe saltar para esquivar cactus que aparecen de manera continua. Cada obstáculo que se evita incrementa el puntaje, mientras que la velocidad de los cactus aumenta progresivamente para ofrecer mayor desafío. Este juego combina estructura HTML, estilos y animaciones CSS y lógica de interacción en JavaScript, ofreciendo una experiencia dinámica y entretenida. Además, permite al jugador mejorar reflejos y coordinación, mientras sirve como ejemplo educativo de desarrollo web interactivo.
uelve a permitir reiniciar.

2. Archivos Requeridos

El juego utiliza tres tipos de archivos principales:

- **HTML (dos.html) → Define la estructura de la página, los elementos visibles como el dinosaurio, el cactus, el puntaje y los botones de inicio y reinicio.**
- **CSS (dos.css) → Da estilo a todos los elementos: colores, tamaños, posiciones, animaciones del cactus y transición del salto del dinosaurio.**
- **JavaScript (dos.js) → Controla la interactividad del juego: el salto del dinosaurio, el movimiento del cactus, el incremento del puntaje y la detección de colisiones.**

Imágenes utilizadas:

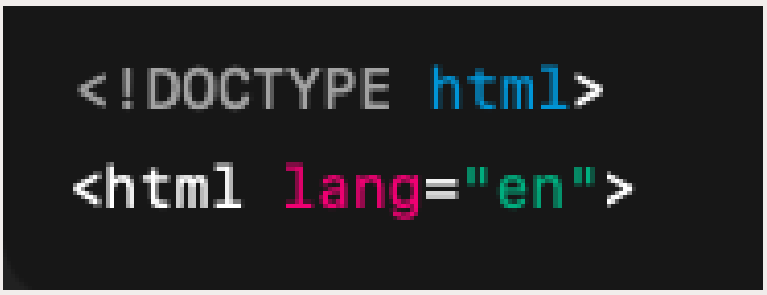
- **dino.svg → Dinosaurio principal.**
- **cactus.svg → Obstáculo que aparece durante el juego.**

Nota: Todos los archivos deben permanecer en la misma carpeta y las imágenes se cargan desde URLs externas para que el juego funcione correctamente.

3. Estructura HTML (dos.html)

El archivo dos.html define la estructura y los elementos visibles del juego DINO RUNNER. A continuación se explica línea por línea:

3.1 Declaración del documento



- **<!DOCTYPE html> → Indica que el documento es HTML5.**
- **lang="en" → Define el idioma como inglés, importante para accesibilidad y motores de búsqueda.**

3.2 Encabezado (<head>)

- **<meta charset="utf-8"> → Permite mostrar caracteres especiales y emojis correctamente.**
- **<meta name="viewport" content="width=device-width, initial-scale=1"> → Hace que la página sea responsiva, ajustándose a cualquier tamaño de pantalla.**
- **<link rel="stylesheet" href="dos.css"> → Conecta el archivo CSS para estilos y animaciones.**
- **<title> → Texto que aparece en la pestaña del navegador.**

3.3 Cuerpo del juego (<body>)

```
<body>
  <div class="container">
```

- **<body> → Contiene todos los elementos visibles.**
- ** → Contenedor principal que centra y organiza todos los elementos del juego.**

3.4 Área de juego

```
<div class="card">
  <img src="https://rerofya.github.io/resources/dino.svg" class="dino" />
  <img src="https://rerofya.github.io/resources/cactus.svg" class="kaktus" />
  <div class="score">Puntos: <span id="score">0</span></div>
</div>
```

- ** → Área de juego donde se mueve el dinosaurio y aparece el cactus.**
- **<img class="dino"> → Dinosaurio que controla el jugador.**
- **<img class="kaktus"> → Cactus que se mueve horizontalmente y representa el obstáculo.**
- ** → Contenedor para mostrar el puntaje en tiempo real.**
    - **0 → Valor numérico inicial del puntaje, que será actualizado mediante JavaScript.**

3.5 Botones de control

```
<div class="buttons">
  <button id="start-button">Iniciar</button>
  <button id="restart-button" style="display: none;">Reiniciar</button>
</div>
```

- ** → Agrupa los botones de acción.**
- **<button id="start-button"> → Botón que inicia el juego.**
- **<button id="restart-button" style="display: none;"> → Botón para reiniciar el juego; se oculta al inicio y aparece al terminar el juego.**

```
  <script src="dos.js"></script>
</body>
```

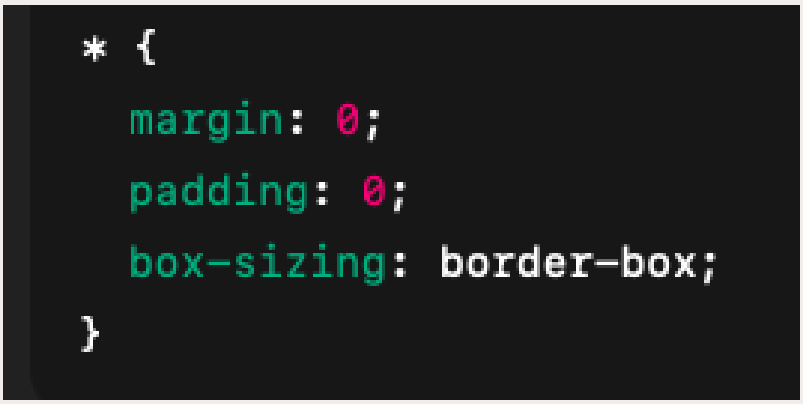
3.6 Vinculación del JavaScript

- **<script src="dos.js"></script> → Conecta el archivo JS que controla la lógica, el movimiento, las animaciones y la puntuación del juego.**
- **Al final del <body> para que los elementos HTML estén cargados antes de que JS intente manipularlos.**

4. Estilos CSS (dos.css)

El archivo dos.css define la apariencia visual y las animaciones del juego. A continuación se explica en detalle:

4.1 Reset de estilos



```
* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
}
```

- **margin: 0 → Elimina márgenes predeterminados del navegador para todos los elementos.**
- **padding: 0 → Elimina relleno predeterminado, asegurando consistencia en tamaños.**
- **box-sizing: border-box → Incluye el borde y padding dentro de ancho y alto de los elementos, facilitando su alineación y dimensionamiento.**

4.2 Estilo general del body

```css
body {
  width: 100%;
  height: 100vh;
  background-color: #f0f0f0;
  font-family: Arial, Helvetica, sans-serif;
  color: #FF0000;
  display: flex;
  justify-content: center;
  align-items: center;
}
```

- **width: 100%; height: 100vh → Ocupa toda la ventana del navegador.**
- **background-color: #f0f0f0 → Fondo gris claro, para que los elementos del juego resalten.**
- **font-family → Fuente legible y estándar.**
- **color: #FF0000 → Color de texto por defecto (puede afectar puntuación u otros textos).**
- **display: flex; justify-content: center; align-items: center → Centra vertical y horizontalmente el contenedor del juego.**

4.3 Contenedor principal

```css
.container {
  width: 100%;
  height: 100%;
  position: relative;
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
}
```

- **position: relative → Permite que los elementos hijos .card y botones se posicionen de forma absoluta dentro de él.**
- **flex-direction: column → Los elementos se colocan verticalmente.**
- **align-items y justify-content → Centran los elementos horizontal y verticalmente.**

4.4 Área de juego (.card)

```css
.card {
  width: 600px;
  height: 300px;
  position: relative;
  background: #fff;
  border: 2px solid #ccc;
  overflow: hidden;
}
```

- **width y height → Tamaño fijo del área de juego.**
- **background: #fff → Fondo blanco para contrastar con cactus y dinosaurio.**
- **border: 2px solid #ccc → Borde gris que define visualmente el área de juego.**

overflow: hidden → Evita que elementos animados como el cactus se vean fuera de la tarjeta.

4.5 Dinosaurio (.dino)

```css
.dino {
  width: 80px;
  height: 80px;
  position: absolute;
  left: 60px;
  bottom: 10px;
  transition: bottom 0.2s ease;
}
```

- **width y height → Tamaño visible del dinosaurio.**
- **position: absolute → Permite mover el dinosaurio dentro de .card mediante JS.**
- **left: 60px; bottom: 10px → Posición inicial, cerca de borde inferior izquierdo.**
- **transition: bottom 0.2s ease → Hace que el salto sea suave y no instantáneo.**

4.6 Cactus (.kaktus)

```css
.kaktus {
  width: 60px;
  height: 60px;
  position: absolute;
  right: -10%;
  bottom: 10px;
  animation: run 3s linear infinite;
  animation-play-state: paused;
}
```

- **width y height → Tamaño del cactus, proporcional al dinosaurio.**
- **right: -10% → Empieza fuera de la pantalla a la derecha.**
- **bottom: 10px → Misma altura del dinosaurio, para colisiones correctas.**
- **animation: run 3s linear infinite → Se mueve continuamente de derecha a izquierda.**
- **animation-play-state: paused → Inicialmente pausado hasta que se presiona el botón Iniciar.**

Animación del cactus

```
@keyframes run {
  0% { right: -10%; }
  100% { right: 110%; }
}
```

- **Define movimiento horizontal del cactus desde fue de la pantalla derecha hasta fuera de la izquierda.**
- **0% → Posición inicial.**
- **100% → Posición final.**
- **linear → Movimiento constante sin aceleración.**

4.7 Puntaje

```
.score {
  position: absolute;
  top: 10px;
  left: 10px;
  font-size: 18px;
  color: black;
}
```

- **position: absolute → Permite ubicarlo sobre el área de juego.**
- **top y left → Esquina superior izquierda.**
- **font-size → Tamaño legible del puntaje.**
- **color → Texto negro para contraste.**

4.8 Botones

```css
button {
  padding: 10px 20px;
  margin: 0 10px;
  font-size: 18px;
  background-color: #28a745;
  color: white;
  border: none;
  border-radius: 6px;
  cursor: pointer;
}

button#restart-button {
  background-color: #dc3545;
}
```

- **padding → Espacio interno para que el botón sea cómodo de presionar.**
- **margin → Separación entre botones.**
- **border-radius → Bordes redondeados para un diseño más amigable.**
- **cursor: pointer → Cambia el cursor al pasar sobre el botón.**
- **Botón de reinicio en rojo para diferenciarlo visualmente del de inicio.**

5. Lógica JavaScript (dos.js)

El archivo dos.js controla la interactividad, la mecánica del juego y la puntuación. Se desglosa en secciones pequeñas para entender cada línea.

5.1 Selección de elementos del DOM

```
const dino = document.querySelector('.dino');
const kaktus = document.querySelector('.kaktus');
const scoreDisplay = document.getElementById('score');
const startBtn = document.getElementById('start-button');
const restartBtn = document.getElementById('restart-button');
```

- **document.querySelector y document.getElementById permiten acceder a los elementos HTML desde JavaScript.**
- **dino → dinosaurio que se mueve al saltar.**
- **kaktus → cactus que se mueve hacia el dinosaurio.**
- **scoreDisplay → contenedor donde se actualiza el puntaje en tiempo real.**
- **startBtn y restartBtn → botones de inicio y reinicio del juego.**

5.2 Variables de control

```
let score = 0;
let gameOver = false;
let gameStarted = false;
let speed = 3; // duración en segundos del movimiento del cactus
let scoreInterval;
let speedInterval;
```

- **score → almacena los puntos del jugador.**
- **gameOver → indica si el juego terminó.**
- **gameStarted → indica si el juego ha comenzado.**
- **speed → define la duración de la animación del cactus (en segundos). Valores menores = cactus más rápido.**
- **scoreInterval y speedInterval → manejan intervalos de tiempo para actualizar puntaje y velocidad del juego.**

5.3 Función salto del dinosaurio

```
function jump() {
  if (!gameStarted || gameOver || dino.classList.contains('jumping')) return;

  dino.classList.add('jumping');
  dino.style.bottom = '150px';

  setTimeout(() => {
    dino.style.bottom = '10px';
    dino.classList.remove('jumping');
  }, 600);
}
```

- **Evita saltar si el juego no comenzó, ya terminó o si el dino ya está saltando.**
- **dino.style.bottom = '150px' → altura máxima del salto.**
- **setTimeout → regresa al dino a su posición original después de 0.6 segundos, simulando la caída natural.**
- **dino.classList.add('jumping') → se puede usar para animaciones CSS opcionales o prevenir saltos dobles.**

5.4 Eventos de teclado

```
document.addEventListener('keydown', (e) => {
  if (e.code === 'Space') {
    e.preventDefault();
    jump();
  }
});
```

- **Detecta cuando el jugador presiona la barra espaciadora.**
- **e.preventDefault() → evita que la página se desplace al presionar espacio.**
- **Llama a la función jump() para ejecutar el salto del dinosaurio.**

5.5 Botones de inicio y reinicio

```
startBtn.addEventListener('click', () => {
  gameStarted = true;
  startGame();
});

restartBtn.addEventListener('click', () => {
  location.reload();
});
```

- **startBtn → activa animación del cactus y empieza la puntuación.**
- **restartBtn → recarga la página para reiniciar el juego completamente.**

5.6 Función iniciar juego

```
function startGame() {
  kaktus.style.animation = `run ${speed}s linear infinite`;
  kaktus.style.animationPlayState = 'running';
  startBtn.style.display = 'none';
  restartBtn.style.display = 'inline-block';

  scoreInterval = setInterval(() => {
    if (!gameOver) {
      score++;
      scoreDisplay.textContent = score;
    }
  }, 500);

  speedInterval = setInterval(() => {
    if (!gameOver && speed > 1) {
      speed -= 0.2;
      kaktus.style.animation = `run ${speed}s linear infinite`;
    }
  }, 5000);
}
```

- **Activa animación del cactus (animationPlayState = 'running').**
- **Oculta el botón de Iniciar y muestra Reiniciar.**
- **scoreInterval → suma puntos cada 0.5 segundos mientras el juego siga.**
- **speedInterval → aumenta la dificultad reduciendo la duración de la animación (speed) cada 5 segundos, haciendo al cactus más rápido.**

5.7 Detección de colisión

```
setInterval(() => {
  if (!gameStarted || gameOver) return;

  const dinoBound = dino.getBoundingClientRect();
  const kaktusBound = kaktus.getBoundingClientRect();

  const isCollision =
    dinoBound.right - 10 >= kaktusBound.left &&
    dinoBound.left + 20 <= kaktusBound.right &&
    dinoBound.bottom >= kaktusBound.top;

  if (isCollision) {
    gameOver = true;
    kaktus.style.animationPlayState = 'paused';
    clearInterval(scoreInterval);
    clearInterval(speedInterval);
    document.querySelector('.card').innerHTML = `
      <h1 style="text-align:center; padding-top:80px;">💀 Game Over 💀</h1>
      <p style="text-align:center;">Puntaje final: ${score}</p>
    `;
  }
}, 100);
```

- **Cada 100ms revisa si el dinosaurio y el cactus se superponen, indicando colisión.**
- **getBoundingClientRect() → obtiene posición y tamaño de los elementos en la pantalla.**
- **Si hay colisión:**
  - **Detiene animación del cactus.**
  - **Para los intervalos de puntuación y velocidad.**

Muestra Game Over y puntaje final dentro del área de juego.

6. Instrucciones de uso

6.1 Preparación de archivos

- Debes tener tres archivos principales en la misma carpeta:
  - dos.html → Contiene la estructura y los elementos visibles del juego.
  - dos.css → Define la apariencia visual y las animaciones.
  - dos.js → Controla la lógica del juego: movimientos, colisiones y puntaje.
- Mantener todos los archivos juntos evita errores de referencia y asegura que el juego funcione correctamente.

6.2 Abrir y ejecutar el juego

1. Abre el archivo dos.html en un navegador moderno (Chrome, Edge, Firefox o Safari).
2. Se mostrarán:

- Dinosaurio a la izquierda.
- Cactus en movimiento (al iniciar).
- Panel de puntaje (comienza en 0).
- Botones Iniciar y Reiniciar (el de reinicio oculto al inicio).

6.3 Controles del juego

- **Barra espaciadora → Salto del dinosaurio.**
- **Botón Iniciar → Comienza la animación del cactus y el conteo de puntaje.**
- **Botón Reiniciar → Reinicia el juego recargando la página.**

6.4 Objetivo

- **Evitar al cactus saltando correctamente.**
- **Acumular puntos mientras sobrevives.**
- **La velocidad del cactus aumenta progresivamente, aumentando la dificultad.**
- **El juego termina al chocar con el cactus, mostrando el puntaje final.**

7. Recomendaciones para jugar y personalizar

7.1 Recomendaciones de juego

- **Usar un navegador actualizado para compatibilidad con animaciones y JS.**
- **Jugar en pantalla amplia para mejor visibilidad.**
- **Evitar abrir muchas pestañas simultáneamente.**
- **Practicar el tiempo del salto y anticipar la velocidad creciente del cactus.**

7.2 Personalización

- **Cambiar altura del salto → Modificar dino.style.bottom en dos.js.**
- **Ajustar velocidad del cactus → Cambiar la variable speed y su decremento en speedInterval.**
- **Reemplazar imágenes del dinosaurio o cactus, respetando tamaños proporcionales (80x80 px dino, 60x60 px cactus).**
- **Modificar colores, fuentes y bordes → directamente en dos.css.**
- **Aumentar dificultad → Reducir speed inicial o disminuir tiempo de incremento en speedInterval.**

8. Precauciones

- **No eliminar enlaces a dos.js y dos.css, el juego depende de ellos.**
- **Mantener la estructura HTML (.container, .card, .dino, .kaktus) para que JavaScript funcione correctamente.**
- **No cambiar nombres de ID o clases (score, start-button, restart-button) sin actualizar referencias en dos.js.**

9. Conclusión

- **Este juego es un ejemplo completo de cómo combinar HTML, CSS y JavaScript:**
- **HTML → Define la estructura de la página y los elementos visibles.**
- **CSS → Da estilo, colores, posiciones y animaciones.**
- **JavaScript → Controla interactividad, movimiento, detección de colisiones y puntaje.**
- **Permite jugar, aprender cómo funciona cada elemento y modificarlo para personalizar la experiencia de manera segura y clara.**