**Core-Code:**

console.log("hoi script.js");

/////////////////////Animaton bzw. Event CTA-Buttons/////////////////////////

//////////////////Fetch and Store APIs und aktuelles Datum//////////////////

// Was wir brauchen aus den Daten:

//Link zu Quotes South Park, Game of Thrones und Advice Slip API

// Aktuelles Datum

//Aktuelles Datum (Aktueller Tag)

const today = new Date();

const formattedDate = today.toLocaleDateString("de-DE"); // deutscher Stil

console.log(formattedDate);

// Speichere das Datum des letzten Speicherns im Local Storage

localStorage.setItem("quotes\_saved\_date", formattedDate);

///Hilfsfunktion

async function loadQuote(url) {

try {

const response = await fetch(url);

return await response.json();

} catch (error) {

console.error(error);

return false;

}

///Hauptfunktion Fetch and Store APIs

}

async function loadandStoreQuotes() {

const quote\_1 = await loadQuote(

"https://api.gameofthronesquotes.xyz/v1/random"

); //GameofThrones

const quote\_2 = await loadQuote("https://api.adviceslip.com/advice"); //AdviceSlip

const quotes\_3 = await loadQuote(

"https://southparkquotes.onrender.com/v1/quotes/3"

); //SouthPark

const quote\_3 = await quotes\_3[0];

if (quote\_1) {

localStorage.setItem("quote\_GoT", JSON.stringify(quote\_1));

console.log("Game of Thrones gespeichert:", quote\_1);

}

if (quote\_2) {

localStorage.setItem("quote\_advice", JSON.stringify(quote\_2));

console.log("Advice gespeichert:", quote\_2);

}

if (quote\_3) {

const quote\_3 = quotes\_3[0];

localStorage.setItem("quote\_southpark", JSON.stringify(quote\_3));

console.log("South Park gespeichert:", quote\_3);

}

}

//Hole aktuelles Datum aus Local Storage

const savedDate = localStorage.getItem("quotes\_saved\_date");

// Prüfen, ob Quotes schon für heute gespeichert wurden

if (savedDate !== formattedDate) {

loadandStoreQuotes();

} else {

// Quotes wurden heute bereits gespeichert – lade sie aus localStorage

const quote1 = localStorage.getItem("quote\_GoT");

const quote2 = localStorage.getItem("quote\_advice");

const quote3 = localStorage.getItem("quote\_southpark");

console.log("Quote GoT:", quote1);

console.log("Quote Advice:", quote2);

console.log("Quote South Park:", quote3);

}

///////////////////////////////Cookie Breaks ////////////////////////////////

document.addEventListener("DOMContentLoaded", () => {

console.log("Script läuft – Fortune Cookie Animation Etappe 1");

const cookieIds = ["leftcookie", "middlecookie", "rightcookie"];

const container = document.getElementById("cookie-container");

cookieIds.forEach((id) => {

const cookie = document.getElementById(id);

cookie.addEventListener("click", async () => {

console.log(Geklickt: ${id});

// Alle anderen Cookies ausblenden

cookieIds.forEach((otherId) => {

if (otherId !== id) {

const otherCookie = document.getElementById(otherId);

if (otherCookie) otherCookie.style.visibility = "hidden";

}

});

// Klone das Cookie-Bild als neues Element zur Animation

const animatedCookie = cookie.cloneNode(true);

animatedCookie.style.position = "absolute";

animatedCookie.style.top = "50%";

animatedCookie.style.left = "50%";

animatedCookie.style.transform = "translate(-50%, -50%)";

animatedCookie.style.zIndex = "10";

animatedCookie.style.width = "30%"; // gleiche Größe wie Original

container.appendChild(animatedCookie);

// Animation je nach Cookie-ID zuweisen

if (id === "leftcookie") {

animatedCookie.style.animation = "leftToCenter 0.7s ease forwards";

} else if (id === "rightcookie") {

animatedCookie.style.animation = "rightToCenter 0.7s ease forwards";

} else if (id === "middlecookie") {

animatedCookie.style.animation = "middlePop 0.6s ease forwards";

}

// Originale Cookies entfernen/verstecken

cookie.style.visibility = "hidden";

// Warte auf Animationsende

await new Promise((resolve) => setTimeout(resolve, 700));

// Entferne das animierte Cookie-Bild

animatedCookie.remove();

// Zeige die zerbrochenen Hälften

/\*const leftHalf = document.createElement("img");

leftHalf.src = "img/FortuneCookieLeft.svg";

leftHalf.alt = "Linke Hälfte";

leftHalf.className = "cookie-half animate-left";

const rightHalf = document.createElement("img");

rightHalf.src = "img/FortuneCookieRight.svg";

rightHalf.alt = "Rechte Hälfte";

rightHalf.className = "cookie-half animate-right"; \*/

const leftHalf = document.createElement("img");

leftHalf.src = "img/FortuneCookieLeft.svg";

leftHalf.alt = "Linke Hälfte";

leftHalf.className = "cookie-half";

const rightHalf = document.createElement("img");

rightHalf.src = "img/FortuneCookieRight.svg";

rightHalf.alt = "Rechte Hälfte";

rightHalf.className = "cookie-half";

container.appendChild(leftHalf);

container.appendChild(rightHalf);

console.log("Cookie zerbrochen!");

// Schritt 1: Hole das zugehörige Quote aus dem LocalStorage

let quoteText = "";

if (id === "leftcookie") {

const quote = JSON.parse(localStorage.getItem("quote\_GoT"));

quoteText = quote?.sentence || quote?.character?.name || "Zitat fehlt.";

} else if (id === "middlecookie") {

const quote = JSON.parse(localStorage.getItem("quote\_advice"));

quoteText = quote?.slip?.advice || "Zitat fehlt.";

} else if (id === "rightcookie") {

const quote = JSON.parse(localStorage.getItem("quote\_southpark"));

quoteText = quote?.quote || "Zitat fehlt.";

}

// Schritt 2: Erstelle den Zettel (fortune-paper)

const paper = document.createElement("div");

paper.className = "fortune-paper";

paper.textContent = ""; // zuerst leer

container.appendChild(paper);

// Schritt 3: Setze Breite dynamisch basierend auf Textlänge

// (nur grobe Schätzung, du kannst statisch lassen, wenn gewünscht)

const paperWidth = Math.min(

quoteText.length \* 8 + 100,

window.innerWidth \* 0.6

);

paper.style.width = ${paperWidth}px;

// Schritt 4: Positioniere Cookie-Hälften passend weiter auseinander

/\*const cookieHalfOffset = leftHalf.offsetWidth / 2 || 100;

const cookieHalfrightOffset = 115;

leftHalf.style.left = calc(50% - ${

paperWidth / 2 + cookieHalfOffset

}px);

rightHalf.style.right = calc(50% - ${

paperWidth / 2 + cookieHalfrightOffset

}px); \*/

// Schritt 4: Warte auf Rendering, um tatsächliche Breite zu bekommen

requestAnimationFrame(() => {

const paperActualWidth = paper.offsetWidth;

// Linke Hälfte links vom Zettel positionieren

leftHalf.style.left = calc(50% - ${paperActualWidth / 2}px);

leftHalf.style.transform = translate(-100%, -50%) rotate(-20deg);

// Rechte Hälfte rechts vom Zettel positionieren

rightHalf.style.left = calc(50% + ${paperActualWidth / 2}px);

rightHalf.style.transform = translate(0%, -50%) rotate(20deg);

/\* // Schritt 4: Setze CSS-Variable --offset dynamisch für Animation

const offset = Math.min(paperWidth / 2 + 290, 390); // max 120px

const offsetr = Math.min(paperWidth / 2 + 50);

leftHalf.style.setProperty("--offset", ${offset}px);

rightHalf.style.setProperty("--offset", ${offsetr}px); \*/

// Start Animation

leftHalf.classList.add("animate-left");

rightHalf.classList.add("animate-right");

// Schritt 5: Quote erscheint 1.5s nach Zettel-Erscheinen

setTimeout(() => {

const quoteTextElem = document.createElement("div");

quoteTextElem.className = "fortune-text";

quoteTextElem.textContent = quoteText;

paper.appendChild(quoteTextElem);

}, 50);

});

});

});

});

**Zwischencode mit richtiger Abspeicherung:**

console.log("hoi script.js");

/////////////////////Animaton bzw. Event CTA-Buttons/////////////////////////

//////////////////Fetch and Store APIs und aktuelles Datum//////////////////

// Was wir brauchen aus den Daten:

//Link zu Quotes South Park, Game of Thrones und Advice Slip API

// Aktuelles Datum

//Aktuelles Datum (Aktueller Tag)

const today = new Date();

const formattedDate = today.toLocaleDateString("de-DE"); // deutscher Stil

console.log(formattedDate);

 // Speichere das Datum des letzten Speicherns im Local Storage

localStorage.setItem("quotes\_saved\_date", formattedDate);

//Überprüfung, ob an diesem entsprechenden Datum bereits Quotes abgespeichert

function quotesAreValidForToday() {

  const savedDate = localStorage.getItem("quotes\_saved\_date");

  if (savedDate !== formattedDate) return false;

  try {

    const quote1 = JSON.parse(localStorage.getItem("quote\_GoT"));

    const quote2 = JSON.parse(localStorage.getItem("quote\_advice"));

    const quote3 = JSON.parse(localStorage.getItem("quote\_southpark"));

    if (!quote1 || !quote2 || !quote3) return false;

    // einfache Gültigkeitschecks

    if (!quote1.sentence) return false;

    if (!quote2.slip?.advice) return false;

    if (!quote3.quote) return false;

    return true; // alle vorhanden & valide

  } catch (e) {

    return false; // Parsing-Fehler => ungültig

  }

}

///Hilfsfunktion

async function loadQuote(url) {

  try {

    const response = await fetch(url);

    return await response.json();

  } catch (error) {

    console.error(error);

    return false;

  }

  ///Hauptfunktion Fetch and Store APIs

}

async function loadandStoreQuotes() {

  const quote\_1 = await loadQuote(

    "https://api.gameofthronesquotes.xyz/v1/random"

  ); //GameofThrones

  const quote\_2 = await loadQuote("https://api.adviceslip.com/advice"); //AdviceSlip

  const quotes\_3 = await loadQuote(

    "https://southparkquotes.onrender.com/v1/quotes/3"

  ); //SouthPark

  const quote\_3 = await quotes\_3[0];

  if (quote\_1) {

    localStorage.setItem("quote\_GoT", JSON.stringify(quote\_1));

    console.log("Game of Thrones gespeichert:", quote\_1);

  }

  if (quote\_2) {

    localStorage.setItem("quote\_advice", JSON.stringify(quote\_2));

    console.log("Advice gespeichert:", quote\_2);

  }

  if (quote\_3) {

    const quote\_3 = quotes\_3[0];

    localStorage.setItem("quote\_southpark", JSON.stringify(quote\_3));

    console.log("South Park gespeichert:", quote\_3);

  }

}

// Prüfen, ob Quotes für heute vorhanden

if (!quotesAreValidForToday()) {

  loadandStoreQuotes();

} else {

  // Quotes wurden heute gültig gespeichert = lade sie aus LocalStorage

  const quote1 = localStorage.getItem("quote\_GoT");

  const quote2 = localStorage.getItem("quote\_advice");

  const quote3 = localStorage.getItem("quote\_southpark");

  console.log("Quote GoT:", quote1);

  console.log("Quote Advice:", quote2);

  console.log("Quote South Park:", quote3);

}

/\*Hole aktuelles Datum aus Local Storage

const savedDate = localStorage.getItem("quotes\_saved\_date");

// Prüfen, ob Quotes schon für heute gespeichert wurden

if (savedDate !== formattedDate) {

  loadandStoreQuotes();

   } else {

  // Quotes wurden heute bereits gespeichert – lade sie aus localStorage

  const quote1 = localStorage.getItem("quote\_GoT");

  const quote2 = localStorage.getItem("quote\_advice");

  const quote3 = localStorage.getItem("quote\_southpark");

  console.log("Quote GoT:", quote1);

  console.log("Quote Advice:", quote2);

  console.log("Quote South Park:", quote3);

} \*/

///////////////////////////////Cookie Breaks ////////////////////////////////

document.addEventListener("DOMContentLoaded", () => {

  console.log("Script läuft – Fortune Cookie Animation Etappe 1");

  const cookieIds = ["leftcookie", "middlecookie", "rightcookie"];

  const container = document.getElementById("cookie-container");

  cookieIds.forEach((id) => {

    const cookie = document.getElementById(id);

    cookie.addEventListener("click", async () => {

      console.log(`Geklickt: ${id}`);

      // Alle anderen Cookies ausblenden

      cookieIds.forEach((otherId) => {

        if (otherId !== id) {

          const otherCookie = document.getElementById(otherId);

          if (otherCookie) otherCookie.style.visibility = "hidden";

        }

      });

      // Klone das Cookie-Bild als neues Element zur Animation

      const animatedCookie = cookie.cloneNode(true);

      animatedCookie.style.position = "absolute";

      animatedCookie.style.top = "50%";

      animatedCookie.style.left = "50%";

      animatedCookie.style.transform = "translate(-50%, -50%)";

      animatedCookie.style.zIndex = "10";

      animatedCookie.style.width = "30%"; // gleiche Größe wie Original

      container.appendChild(animatedCookie);

      // Animation je nach Cookie-ID zuweisen

      if (id === "leftcookie") {

        animatedCookie.style.animation = "leftToCenter 0.7s ease forwards";

      } else if (id === "rightcookie") {

        animatedCookie.style.animation = "rightToCenter 0.7s ease forwards";

      } else if (id === "middlecookie") {

        animatedCookie.style.animation = "middlePop 0.6s ease forwards";

      }

      // Originale Cookies entfernen/verstecken

      cookie.style.visibility = "hidden";

      // Warte auf Animationsende

      await new Promise((resolve) => setTimeout(resolve, 700));

      // Entferne das animierte Cookie-Bild

      animatedCookie.remove();

      // Zeige die zerbrochenen Hälften

      /\*const leftHalf = document.createElement("img");

      leftHalf.src = "img/FortuneCookieLeft.svg";

      leftHalf.alt = "Linke Hälfte";

      leftHalf.className = "cookie-half animate-left";

      const rightHalf = document.createElement("img");

      rightHalf.src = "img/FortuneCookieRight.svg";

      rightHalf.alt = "Rechte Hälfte";

      rightHalf.className = "cookie-half animate-right"; \*/

      const leftHalf = document.createElement("img");

      leftHalf.src = "img/FortuneCookieLeft.svg";

      leftHalf.alt = "Linke Hälfte";

      leftHalf.className = "cookie-half";

      const rightHalf = document.createElement("img");

      rightHalf.src = "img/FortuneCookieRight.svg";

      rightHalf.alt = "Rechte Hälfte";

      rightHalf.className = "cookie-half";

      container.appendChild(leftHalf);

      container.appendChild(rightHalf);

      console.log("Cookie zerbrochen!");

      // Schritt 1: Hole das zugehörige Quote aus dem LocalStorage

            let quoteText = "";

      if (id === "leftcookie") {

        const quote = JSON.parse(localStorage.getItem("quote\_GoT"));

        quoteText = quote?.sentence || quote?.character?.name || "Zitat fehlt.";

      } else if (id === "middlecookie") {

        const quote = JSON.parse(localStorage.getItem("quote\_advice"));

        quoteText = quote?.slip?.advice || "Zitat fehlt.";

      } else if (id === "rightcookie") {

        const quote = JSON.parse(localStorage.getItem("quote\_southpark"));

        quoteText = quote?.quote || "Zitat fehlt.";

      }

      // Schritt 2: Erstelle den Zettel (fortune-paper)

      const paper = document.createElement("div");

      paper.className = "fortune-paper";

      paper.textContent = ""; // zuerst leer

      container.appendChild(paper);

      // Schritt 3: Setze Breite dynamisch basierend auf Textlänge

      // (nur grobe Schätzung, du kannst statisch lassen, wenn gewünscht)

      const paperWidth = Math.min(

        quoteText.length \* 8 + 100,

        window.innerWidth \* 0.6

      );

      paper.style.width = `${paperWidth}px`;

      // Schritt 4: Positioniere Cookie-Hälften passend weiter auseinander

      /\*const cookieHalfOffset = leftHalf.offsetWidth / 2 || 100;

      const cookieHalfrightOffset = 115;

      leftHalf.style.left = `calc(50% - ${

        paperWidth / 2 + cookieHalfOffset

      }px)`;

      rightHalf.style.right = `calc(50% - ${

        paperWidth / 2 + cookieHalfrightOffset

      }px)`; \*/

      // Schritt 4: Warte auf Rendering, um tatsächliche Breite zu bekommen

      requestAnimationFrame(() => {

        const paperActualWidth = paper.offsetWidth;

        // Linke Hälfte links vom Zettel positionieren

        leftHalf.style.left = `calc(50% - ${paperActualWidth / 2}px)`;

        leftHalf.style.transform = `translate(-100%, -50%) rotate(-20deg)`;

        // Rechte Hälfte rechts vom Zettel positionieren

        rightHalf.style.left = `calc(50% + ${paperActualWidth / 2}px)`;

        rightHalf.style.transform = `translate(0%, -50%) rotate(20deg)`;

        /\* // Schritt 4: Setze CSS-Variable --offset dynamisch für Animation

      const offset = Math.min(paperWidth / 2 + 290, 390); // max 120px

      const offsetr = Math.min(paperWidth / 2 + 50);

      leftHalf.style.setProperty("--offset", `${offset}px`);

      rightHalf.style.setProperty("--offset", `${offsetr}px`); \*/

        // Start Animation

        leftHalf.classList.add("animate-left");

        rightHalf.classList.add("animate-right");

        // Schritt 5: Quote erscheint 1.5s nach Zettel-Erscheinen

        setTimeout(() => {

          const quoteTextElem = document.createElement("div");

          quoteTextElem.className = "fortune-text";

          quoteTextElem.textContent = quoteText;

          paper.appendChild(quoteTextElem);

        }, 50);

      });

    });

  });

});

**Java cookie break**

document.addEventListener("DOMContentLoaded", () => {

  console.log("Script läuft – Fortune Cookie Animation Etappe 1");

  const cookieIds = ["leftcookie", "middlecookie", "rightcookie"];

  const container = document.getElementById("cookie-container");

  cookieIds.forEach((id) => {

    const cookie = document.getElementById(id);

    cookie.addEventListener("click", async () => {

      console.log(`Geklickt: ${id}`);

      // Alle anderen Cookies ausblenden

      cookieIds.forEach((otherId) => {

        if (otherId !== id) {

          const otherCookie = document.getElementById(otherId);

          if (otherCookie) otherCookie.style.visibility = "hidden";

        }

      });

      // Klone das Cookie-Bild als neues Element zur Animation

      const animatedCookie = cookie.cloneNode(true);

      animatedCookie.style.position = "absolute";

      animatedCookie.style.top = "50%";

      animatedCookie.style.left = "50%";

      animatedCookie.style.transform = "translate(-50%, -50%)";

      animatedCookie.style.zIndex = "10";

      animatedCookie.style.width = "30%"; // gleiche Größe wie Original

      container.appendChild(animatedCookie);

      // Animation je nach Cookie-ID zuweisen

      if (id === "leftcookie") {

        animatedCookie.style.animation = "leftToCenter 0.7s ease forwards";

      } else if (id === "rightcookie") {

        animatedCookie.style.animation = "rightToCenter 0.7s ease forwards";

      } else if (id === "middlecookie") {

        animatedCookie.style.animation = "middlePop 0.6s ease forwards";

      }

      // Originale Cookies entfernen/verstecken

      cookie.style.visibility = "hidden";

      // Warte auf Animationsende

      await new Promise((resolve) => setTimeout(resolve, 700));

      // Entferne das animierte Cookie-Bild

      animatedCookie.remove();

      // Zeige die zerbrochenen Hälften

      /\*const leftHalf = document.createElement("img");

      leftHalf.src = "img/FortuneCookieLeft.svg";

      leftHalf.alt = "Linke Hälfte";

      leftHalf.className = "cookie-half animate-left";

      const rightHalf = document.createElement("img");

      rightHalf.src = "img/FortuneCookieRight.svg";

      rightHalf.alt = "Rechte Hälfte";

      rightHalf.className = "cookie-half animate-right"; \*/

      const leftHalf = document.createElement("img");

      leftHalf.src = "img/FortuneCookieLeft.svg";

      leftHalf.alt = "Linke Hälfte";

      leftHalf.className = "cookie-half";

      const rightHalf = document.createElement("img");

      rightHalf.src = "img/FortuneCookieRight.svg";

      rightHalf.alt = "Rechte Hälfte";

      rightHalf.className = "cookie-half";

      container.appendChild(leftHalf);

      container.appendChild(rightHalf);

      console.log("Cookie zerbrochen!");

      // Schritt 1: Hole das zugehörige Quote aus dem LocalStorage

            let quoteText = "";

      if (id === "leftcookie") {

        const quote = JSON.parse(localStorage.getItem(`quote\_GoT\_${formattedDate}`));

        quoteText = quote?.sentence || quote?.character?.name || "Zitat fehlt.";

      } else if (id === "middlecookie") {

        const quote = JSON.parse(localStorage.getItem(`quote\_advice\_${formattedDate}`));

        quoteText = quote?.slip?.advice || "Zitat fehlt.";

      } else if (id === "rightcookie") {

        const quote = JSON.parse(localStorage.getItem(`quote\_southpark\_${formattedDate}`));

        quoteText = quote?.quote || "Zitat fehlt.";

      }

      // Schritt 2: Erstelle den Zettel (fortune-paper)

      const paper = document.createElement("div");

      paper.className = "fortune-paper";

      paper.textContent = ""; // zuerst leer

      container.appendChild(paper);

      // Schritt 3: Setze Breite dynamisch basierend auf Textlänge

      // (nur grobe Schätzung, du kannst statisch lassen, wenn gewünscht)

      const paperWidth = Math.min(

        quoteText.length \* 8 + 100,

        window.innerWidth \* 0.6

      );

      paper.style.width = `${paperWidth}px`;

      // Schritt 4: Positioniere Cookie-Hälften passend weiter auseinander

      /\*const cookieHalfOffset = leftHalf.offsetWidth / 2 || 100;

      const cookieHalfrightOffset = 115;

      leftHalf.style.left = `calc(50% - ${

        paperWidth / 2 + cookieHalfOffset

      }px)`;

      rightHalf.style.right = `calc(50% - ${

        paperWidth / 2 + cookieHalfrightOffset

      }px)`; \*/

      // Schritt 4: Warte auf Rendering, um tatsächliche Breite zu bekommen

      requestAnimationFrame(() => {

        const paperActualWidth = paper.offsetWidth;

        // Linke Hälfte links vom Zettel positionieren

        leftHalf.style.left = `calc(50% - ${paperActualWidth / 2}px)`;

        leftHalf.style.transform = `translate(-100%, -50%) rotate(-20deg)`;

        // Rechte Hälfte rechts vom Zettel positionieren

        rightHalf.style.left = `calc(50% + ${paperActualWidth / 2}px)`;

        rightHalf.style.transform = `translate(0%, -50%) rotate(20deg)`;

        /\* // Schritt 4: Setze CSS-Variable --offset dynamisch für Animation

      const offset = Math.min(paperWidth / 2 + 290, 390); // max 120px

      const offsetr = Math.min(paperWidth / 2 + 50);

      leftHalf.style.setProperty("--offset", `${offset}px`);

      rightHalf.style.setProperty("--offset", `${offsetr}px`); \*/

        // Start Animation

        leftHalf.classList.add("animate-left");

        rightHalf.classList.add("animate-right");

        // Schritt 5: Quote erscheint 1.5s nach Zettel-Erscheinen

        setTimeout(() => {

          const quoteTextElem = document.createElement("div");

          quoteTextElem.className = "fortune-text";

          quoteTextElem.textContent = quoteText;

          paper.appendChild(quoteTextElem);

        }, 50);

      });

    });

  });

});

Logik:

Momentan speichert es ab, das Keks geöffnet wurde, wie setzte ich das am ende des Tages zurück?

Manchmal ladet Seite nicht richtig -> wie mache ich funktion, dass wenn flasch geladen alles zurückgesetzt?

Abstand zwischen Zettel und Cookie-Hälften?

Ist es schlimm, wenn kein Kalender und keine Wischanimation mobile Ansicht?

Readme Dokumentation?

Ein Bild, das Text, Screenshot, Schrift enthält.

KI-generierte Inhalte können fehlerhaft sein.

Ein Bild, das Text, Screenshot, Schrift, Reihe enthält.

KI-generierte Inhalte können fehlerhaft sein.

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

KI-generierte Inhalte können fehlerhaft sein.