Database

Dieser Code bezieht sich auf eine Simulation von Feuerwerken. Hierbei werden Daten über Feuerwerksraketen in einer Datenbank gespeichert und ausgelesen.

Es wird eine Schnittstelle "Rocket" definiert, die die Eigenschaften einer Feuerwerksrakete beschreibt. Die Schnittstelle besitzt einen Index-Signaturnamen, der einen Namen als String angibt.

Es wird auch eine URL definiert, die auf eine Datenbank-API verweist, über die Daten gespeichert und gelesen werden können.

Die Funktion "handleLoad()" liest die Daten von dem Server aus und gibt ein Array von Raketen zurück.

Die Funktion "saveRocket()" speichert eine Feuerwerksrakete in der Datenbank. Dabei werden die Daten als Parameter übergeben und an die Datenbank-API gesendet. Wenn die Daten erfolgreich gespeichert wurden, wird eine Benachrichtigung angezeigt, ansonsten wird eine Fehlermeldung angezeigt.

Diese Klasse stellt eine Schnittstelle für Raketen und ein paar Funktionen bereit, um Daten von und auf einen Server zu speichern.

Die Schnittstelle "Rocket" definiert ein Dictionary-ähnliches Objekt, dessen Schlüssel ein String und dessen Wert ein Objekt mit den Eigenschaften "particleAmount", "color", "size" und "lifetime" ist.

Die Funktion "handleLoad" liest Daten vom Server aus, indem sie einen HTTP GET-Request an eine URL sendet und dann den Response des Servers in ein JavaScript-Objekt umwandelt. Die Funktion gibt ein Array von Raketen zurück, das die Daten enthält, die aus dem Server gelesen wurden.

Die Funktion "saveRocket" speichert eine Rakete auf dem Server, indem sie einen HTTP POST-Request an eine URL sendet. Dabei werden die Daten als Teil des Requests in einem formatierten String gesendet. Nach dem Absenden des Requests wird das Ergebnis überprüft und eine entsprechende Benachrichtigung angezeigt.

Script

Dieser Code ist in TypeScript geschrieben und implementiert eine Simulation eines Feuerwerks auf einer Leinwand. Die Hauptlogik der Simulation ist in der Funktion createExplosion enthalten, die ausgelöst wird, wenn der Benutzer auf die Leinwand klickt. Die Funktion ruft die Mausposition ab, erstellt eine Instanz der Klasse Explosion und speichert sie im Array explosions.

Die Klasse Explosion benötigt fünf Parameter: die Position der Explosion, die Anzahl der zu erzeugenden Partikel, die Farbe der Explosion, die Größe der Explosion und die Lebensdauer der Explosion.

Der Benutzer kann verschiedene Raketentypen mit unterschiedlichen Parametern speichern und auswählen, und die Parameter der ausgewählten Rakete werden in den Eingabefeldern angezeigt. Wenn der Benutzer auf die Schaltfläche "Rakete bauen" klickt, wird eine neue Rakete erstellt und gespeichert. Die gespeicherten Raketen werden in dem Auswahlmenü "Gespeicherte Raketen" angezeigt.

Es scheint, dass der Code auf die Existenz der Klassen Rocket, Explosion und Vector sowie auf die Funktionen handleLoad und saveRocket angewiesen ist, die in diesem Codeschnipsel nicht gezeigt werden.

function createExplosion

Diese Funktion erstellt eine Explosion an einer bestimmten Stelle auf einem Canvas. Die Position der Explosion wird basierend auf den Koordinaten des Mausklicks berechnet. Die Informationen über die Anzahl der Kreise, Farbe, Größe und Lebensdauer der Explosion werden aus den Werten von HTML-Eingabefeldern abgerufen. Die neue Explosion wird in einem explosions-Array gespeichert.

function addRocket(\_rocket: Rocket): void {

Diese Funktion fügt ein Raketen-Objekt (\_rocket) dem Array "rockets" hinzu. Zusätzlich wird die Funktion "showRocketsToSelect" aufgerufen, die wahrscheinlich die aktualisierten Raketen in einer Auswahl anzeigt.

buildRocketButton.addEventListener("click", () => {

Diese Funktion handelt das Klicken auf einen "buildRocketButton".

Wenn der Button geklickt wird, dann werden die Werte aus den Eingabefeldern "nameInput", "numCirclesInput", "colorInput", "sizeInput" und "lifetimeInput" ausgelesen und in Variablen gespeichert. Es wird überprüft, ob die Werte für Größe, Lebensdauer und Anzahl der Kreise einen maximalen Wert überschreiten. Wenn ja, dann wird eine "alert" angezeigt, dass der maximale Wert überschritten wurde.

Wenn ein Name eingegeben wurde, dann wird ein neues Raketen-Objekt mit den eingegebenen Werten erstellt und diese Rakete wird mit der Funktion "addRocket" zu einer Liste von Raketen hinzugefügt.

Am Ende wird die neue Rakete möglicherweise direkt an einen Server gesendet, indem die Funktion "saveRocket" aufgerufen wird.

savedRocketsSelect.addEventListener("change", () => {

Diese Funktion führt eine Aktion aus, wenn eine Rakete aus der Dropdown-Liste "savedRocketsSelect" ausgewählt wird. Es wird zunächst überprüft, ob eine Rakete mit dem ausgewählten Namen in der "rockets" Liste vorhanden ist. Wenn ja, werden die Eingabewerte für "nameInput", "numCirclesInput", "colorInput", "sizeInput" und "lifetimeInput" auf die Werte der ausgewählten Rakete gesetzt, indem die Funktion "setInputValuesToSelectedRocket" mit der ausgewählten Rakete als Argument aufgerufen wird.

function setInputValuesToSelectedRocket(\_rocket: Rocket): void {

Diese Funktion nimmt eine Rakete als Parameter und setzt die Werte der Eingabefelder im Formular auf die Werte der übergebenen Rakete.

Die Funktion extrahiert zunächst den Namen der Rakete aus dem Objekt, indem sie mit **Object.keys(\_rocket)[0]** den ersten (und einzigen) Schlüssel aus dem Objekt extrahiert. Dann werden die Werte der Eingabefelder auf die Werte der Rakete gesetzt: **nameInput.value** wird auf den Namen der Rakete gesetzt, **numCirclesInput.value** auf die Anzahl der Kreise in der Rakete, **colorInput.value** auf die Farbe der Rakete, **sizeInput.value** auf die Größe der Rakete und **lifetimeInput.value** auf die Lebensdauer der Rakete.

function update(): void {

Diese Funktion hat die Aufgabe, den Canvas (das 2D-Zeichenfläche) für jeden Frame neu zu rendern.

Zunächst wird das gesamte Canvas mit einer halbtransparenten grauen Farbe gefüllt, um alte Darstellungen zu überdecken.

Dann wird jede Explosion im explosions-Array durchlaufen. Wenn die Lebenszeit der Explosion auf 0 gesetzt ist, wird die Explosion aus dem explosions-Array entfernt. Andernfalls wird die Funktion "draw" aufgerufen, um die Explosion zu zeichnen.

export class Explosion extends Moveable {

Diese Klasse "Explosion" beschreibt eine Explosion, die aus einer bestimmten Anzahl von Partikeln besteht. Die Explosion ist abgeleitet von der Klasse "Moveable". Die Klasse "Explosion" hat ein Array "particles", in dem alle Partikel der Explosion gespeichert werden.

Im Konstruktor wird für jeden Partikel eine zufällige Geschwindigkeit generiert und eine Kopie des Partikel-Startpunkts erstellt. Dann werden alle Partikel für diese Explosion erstellt.

Die Methode "draw" ruft die Methode "drawParticles" auf, um die Partikel zu zeichnen. In der Methode "drawParticles" werden alle Partikel bewegt und gezeichnet, solange sie noch eine verbleibende Lebenszeit haben. Wenn alle Partikel keine Lebenszeit mehr haben, wird das Array "particles" geleert und die Lebenszeit der Explosion auf 0 gesetzt.