**Documentation Technique**

Dans ce document, vous trouverez les différentes méthodes et classes réalisé pour le fonctionnant des exercices.

1. **Partie Web**
2. **Exercice `Type-Script`**

**Les variables globales** (ces dernières ne seront pas représenté car elles sont les mêmes pour tous les exercices) :

/\*\* @type {boolean} \*/

let hasExerciceStarted = false; // L'exercice a commencé ?

/\*\* @type {boolean} \*/

let hasExerciceEnded = false; // L'exercice est terminé ?

/\*\* @type {Date} \*/

let tempsDebut = new Date(); // Date de début de l'exercice

/\*\* @type {boolean} \*/

let isFetching = false; // Est-ce qu'on est en train de récupérer une phrase depuis l'API python ?

/\*\* @type {string} \*/

let langue = 'fr'; // Langue de la phrase à récupérer

**Variables relatives à l’exercice :**

/\*\* @type {boolean} \*/

let caseSensitive = true; // Majuscules sensibles ?

/\*\* @type {boolean} \*/

let accentSensitive = true; // Accents sensibles ?

/\*\* @type {number} \*/

let nbreMotsTapes = 0; // Nombre de mots tapés

/\*\* @type {number} \*/

let nbreMotsTotal = 0; // Nombre de mots total

/\*\* @type {Array<string>} \*/

$: letters = sentences.split(''); // La phrase à taper, séparée en lettres

/\*\* @type {string} \*/

const ESPACE = ' ⸱ '; // Caractère de remplacement pour les espaces

/\*\* @type {string}\*/

let sentences = ''; // La phrase à taper

/\*\* @type {string} \*/

let source = ''; // La source de la phrase

// Résultats de l'exercice calculé par l’API Python

/\*\* @type {{

 \* tempsMisString: string,

 \* nbErreurs: number,

 \* nbCaracteres: number,

 \* vitesse: number,

 \* precision: number,

 \* score: number}} \*/

let resultats = {}

**Méthodes relatives au fonctionnement de l’exercice :**

/\*\*

 \* Appelé lorsqu'une touche est appuyée

\* Ici est vérifié si la lettre tapé par l’utilisateur correspond bien à la lettre attendue.

 \* @param {KeyboardEvent} event

 \*/

function keyDown(event) {

}

/\*\*

 \* Appelé lorsque l'utilisateur a terminé de taper les phrases.

\* Envois les résultats à Python pour récupérer le score de l’

 \* @returns {void}

 \*/

function handleSentencesCompleted() {

}

/\*\*

 \* Démarre l'exercice

 \* Récupère une phrase aléatoire depuis l'API python

 \* @returns {Promise<void>}

 \*/

async function startExercice() {

}

1. **Exercice `Sténographie`**

**Variables relatives à l’exercice :**

/\*\*

 \* Les audios des phrases ainsi que les phrases au format [(texte, audio file path), ...]

 \* @type {Array<[string, string]>}

 \*/

let audios = [];

/\*\* @type {string} \*/

let tapedSentence = ''; // La phrase tapée par l'utilisateur

/\*\* @type {number | null} \*/

let similitude = null; // Le pourcentage de similitude entre la phrase tapée et la phrase de l'audio

/\*\* @type {boolean} \*/

let checkMaj = false; // Autoriser les majuscules dans les chaines de caractères ?

/\*\* @type {boolean} \*/

let checkOrthographe = true; // Autoriser les accents dans les chaines de caractères ?

/\*\* @type {boolean} \*/

let checkPonctuations = false; // Autoriser les caractères spéciaux dans les chaines de caractères ?

**Méthodes relatives au fonctionnement de l’exercice :**

/\*\*

 \* Appelée lorsqu'une touche est appuyée

 \* Si la touche est ENTRÉE et que l'exercice n'a pas encore commencé, démarre l'exercice

 \* @param {KeyboardEvent} event

 \*/

function keyDown(event) {

}

/\*\*

 \* Démarre l'exercice

 \* Récupère depuis l'API python les phrases et leurs audios

 \*/

async function startExercice() {

}

/\*\*

 \* Joue l'audio suivant à retrancrire

 \* Si il n'y a plus d'audios, récupère de nouvelles phrases depuis l'API python

 \*/

function playNextAudio() {

}

/\*\*

 \* Appelée lorsqu'on clique sur le bouton de soumission

 \* Envoye la phrase tapée par l'utilisateur à l'API python pour vérifier si elle est correcte en fonction des options choisies

 \*/

async function handleSubmissionButtonClicked() {

}

1. **Exercice `Rythme`**

**Variables relatives à l’exercice :**

/\*\* @type {Array<import('$lib/classes/Niveau').Niveau>} \*/

let niveaux = []; // Les différents niveaux récupérés depuis python

/\*\* @type {import('$lib/classes/Niveau').Niveau} \*/

let selectedLevelObj; // Le niveau sélectionné

/\*\* @type {number} \*/

let audioPosition = 0; // La position actuelle de l'audio

/\*\* @type {number} \*/

let score = 0; // Le score actuel

/\*\* @type {number} \*/

let scoreMax = 0; // Le score maximum possible

/\*\* @type {Array<string>} \*/

let keys = []; // Les lettres dans l'ordre du clavier

/\*\* @type {Array<any>} \*/

let keysScored = []; // Les lettres déjà scorés

/\*\* @type {Array<string>} \*/

let keyboardLayout = ['a', 'z', 'e', 'r', 't', 'y', 'u', 'i', 'o', 'p', 'q', 's', 'd', 'f', 'g', 'h', 'j', 'k', 'l', 'm', 'w', 'x', 'c', 'v', 'b', 'n']; // Le layout du clavier pour trier les touches dans l'ordre

/\*\* @type {Array<string>} \*/

let touchesAppuyees = []; // Les touches actuellement appuyées

**Méthodes relatives au fonctionnement de l’exercice :**

/\*\*

 \* Appelée lorsque la fenêtre est chargée

 \* Récupère les niveaux depuis python

\*/

onMount(async () => {

}

/\*\*

 \* Appelée lorsqu'une touche est appuyée

 \* Si la touche est ENTRÉE et que l'exercice n'a pas encore commencé, démarre l'exercice

 \* Sinon, vérifie si la touche appuyée est dans les touches à appuyer pour le temps actuel

 \* @param {KeyboardEvent} event

 \*/

function keyDown(event) {

}

/\*\*

 \* Démarre l'exercice

 \* Récupère le niveau sélectionné et démarre l'audio

 \*/

function startExercice() {

}

1. **Exercice `Réaction`**

**Variables relatives à l’exercice :**

/\*\* @type {number} \*/

let nombreDeReactions = 6; // Le nombre de réactions à afficher

/\*\* @type {number} \*/

let nombreDeReactions = 6; // Le nombre de réactions à afficher

/\*\* @type {boolean} \*/

let hasExerciceEnded = false; // L'exercice est terminé ?

/\*\* @type {boolean} \*/

let allowUppercase = true; // Autoriser les majuscules dans les chaines de caractères ?

/\*\* @type {boolean} \*/

let allowAccents = true; // Autoriser les accents dans les chaines de caractères ?

/\*\* @type {boolean} \*/

let allowSpecialCharacters = false; // Autoriser les caractères spéciaux dans les chaines de caractères ?

/\*\* @type {string}\*/

let reaction = ''; // La chaine de caractères à écrire

/\*\* @type {number} \*/

let tempsDebut; // Temps auquel la réaction a été affichée (pour calculer le temps de réaction de l'utilisateur)

/\*\* @type {string} \*/

let typedReaction = ''; // La chaine de caractères tapée par l'utilisateur

/\*\* @type {Array<Array<any>>} \*/

let tempsDeReactions = []; // Les temps de réaction de l'utilisateur en millisecondes

/\*\* @type {number} \*/

let indexReaction = 0; // Index de la réaction actuelle

/\*\* @type {number} \*/

let score = 0; // Le score du joueur

/\*\* @type {number} \*/

let temps\_moyen\_difficulte = 0; // Le temps moyen de réaction par rapport à la difficulté

/\*\* @type {number} \*/

let temps\_moyen\_total = 0; // Le temps moyen de réaction total

**Méthodes relatives au fonctionnement de l’exercice :**

/\*\*

 \* Appelée depuis l'API python lorsqu'une réaction est à afficher

 \* @param {string} reaction\_a\_afficher

 \*/

function afficherReaction(reaction\_a\_afficher) {

}

/\*\*

 \* Démarre l'exercice

 \* Envoie les paramètres choisis à l'API et attend que l'API initialise l'exercice

 \*/

async function startExercice() {

}

/\*\*

 \* Appelée lorsque l'utilisateur a tapé une chaine de caractères dans l'input de la réaction

 \* Compare la chaine de caractères tapée par l'utilisateur avec la chaine de caractères attendue

 \*/

$: if (typedReaction) {

}

/\*\*

 \* Appelée lorsque l'exercice est terminé

 \* Envoie les temps de réaction à l'API python pour récupérer les statistiques

 \*/

async function exerciceEnded() {

}

1. **Exercice `Jeu du Bac`**

**Variables relatives à l’exercice :**

/\*\* @type {Array<string>} \*/

let themes = []; // Les thèmes disponibles. Récupérés depuis l'API python lors du montage du composant

/\*\* @type {Array<string>} \*/

let selectedThemes = ['Animaux', 'Pays', 'Fruits', 'Légumes', 'Métiers']; // Les thèmes sélectionnés par l'utilisateur (par défaut les 5 ci-gauche)

/\*\* @type {number} \*/

let nombreDeRound = 3; // Le nombre de ligne de l'exercice

let chronometre = 0; // Le chronometre de la round actuelle (en secondes)

let max\_temps = 60; // Le temps maximum pour chaque round (en secondes)

let chronometreTotal = 0; // Le temps total pour finir l'exercice (en secondes)

/\*\* @type {Array<JeuBacRow>} \*/

let rows = []; // Les lignes du tableau

**Classes relatives au fonctionnement de l’exercice :**

/\*\*

 \* La classe JeuBacRow représente une ligne de la grille du jeu du Bac.

 \* Elle contient une lettre et une liste de mots entrés par les joueurs.

 \*/

export default class JeuBacRow {

}

 \* La classe Mot représente un mot entré par un joueur dans une grille du jeu du Bac.

 \* L'attribut "valide" est null si le mot n'a pas encore été validé, "true" si le mot est valide et "false" s'il ne l'est pas.

 \*/

export class Mot {

}

**Méthodes relatives au fonctionnement de l’exercice :**

/\*\*

 \* Appelée lorsqu'on appuie sur ENTRÉE

 \* Vérifie que l'utilisateur a sélectionné 5 thèmes

 \* Si oui, démarre l'exercice

 \*/

async function startExercice() {

}

/\*\*

 \* Appelée lorsqu'un thème est sélectionné ou désélectionné

 \* Si le thème est déjà sélectionné, le retire des thèmes sélectionnés

 \* Sinon, l'ajoute aux thèmes sélectionnés

 \* Si l'utilisateur a déjà sélectionné 5 thèmes, l'empêche de sélectionner plus de thèmes

 \* @param {any} e

 \* @param {string} theme

 \*/

function handleThemeSelected(e, theme) {

}

/\*\*

 \* Démarre un round

 \* Récupère une lettre aléatoire et crée une ligne pour cette lettre

 \*/

async function startRound() {

}

/\*\*

 \* Démarre la roulette

 \* Retourne une lettre aléatoire après un certain nombre de tours de roulette

 \* @returns {Promise<string>}

 \*/

function startRoulette() {

}

/\*\*

 \* Appelée lorsqu'on clique sur le bouton de validation

 \* Envoie les réponses à l'API python pour vérifier si les mots sont corrects depuis la base de données

 \*/

async function validateRow() {

}

/\*\*

 \* Appelée lorsqu'on a fini l'exercice

 \* Trouve le nombre de mots valides

 \*/

function endExercice() {

}

1. **Partie Python**
2. **Clavier**
3. **Exercice `nom`**

**Variables relatives à l’exercice :**

**Méthodes relatives au fonctionnement de l’exercice :**

1. **Souris**
2. **Exercice `nom`**

**Variables relatives à l’exercice :**

**Méthodes relatives au fonctionnement de l’exercice :**