Minuteurs

On veut utiliser un minuteur et un émetteur d'événements pour émettre les valeurs d'un tableau à intervalles réguliers et s'en servir pour afficher de différentes façon du texte au fur et à mesure.

Émission de valeurs

Le fichier `eventemitter3.js` exporte la classe `EventEmitter` permettant d'émettre et écouter des événements. **Écrivez** dans le fichier `emissionValeurs.js` un module qui exporte la fonction `emettreValeurs(valeurs, delai, emetteur, type)` qui fait émettre à l'émetteur `emetteur` tous les `delai` ms un événements de type `type` ayant pour valeur la valeur en cours du tableau `valeurs`, et qui renvoie l'identifiant de la programmation.

Testez : récupérez la page HTML `testminuteur.html`, faites-lui charger le module et écrivez dedans un script qui fait afficher dans la console les valeurs d'un tableau exemple toutes les 500 ms.

Afficheur de texte

Récupérez la page HTML `affichertexte.html` contenant l'interface de l'application.

On veut afficher au fur et à mesure le texte multi-lignes de l'élément `<div id="texte">` de plusieurs façons. Les boutons radio `<input type="radio" name="typetxt">` consécutifs `Phrase`, `Phrases` et `Lettres` permettent de choisir la façon dont le texte sera affiché au fur et à mesure. Le bouton `Voir <input type="button" value="voir" id="voir">` permet de lancer l'affichage (depuis le début) et le bouton `Stop <input type="button" value="stop" id="clear">` de le stopper.

On sépare les trois composants, contrôleur, vue et modèle de l'application dans des modules dans les fichiers `controleur.js`, `vue.js` et `modele.js`. **Récupérez** ces fichiers qui contiennent les modules qui s'auto-initialisent au chargement de la page et s'auto-déchargent quand la page est quittée.

Initialisation des données

Écrivez et exportez dans la vue la fonction `texteInitial` qui renvoie le contenu textuel de l'élément `<div id="texte">` et la fonction `effaceTexte` qui vide le contenu de cet l'élément. Faites en sorte que le modèle stocke un texte qui peut être modifié par la fonction `nouveauTexte(txt)` (à exporter). **Faites en sorte** qu'au chargement le contrôleur transmette le texte de la page (contenu de l'élément `<div id="texte">`) au modèle et qu'il soit enlevé de l'affichage. **Testez**.

Émission des données

Le modèle doit contenir en plus un tableau contenant des morceaux du texte et une variable indiquant le type des morceaux (répliques, phrases ou caractères). **Ajoutez** ces données. **Écrivez** la fonction `repliques(texte)` qui renvoie dans un tableau le texte découpé en répliques (une réplique commence par –). **Écrivez** et exportez la fonction `envoyerRepliquesTexte(delai)` qui fait émettre les répliques du texte (événement `'replique'`) tous les `delai` ms. **Écrivez** et exportez la fonction `arreterTexte()` qui fait arrêter l'émission des morceaux de texte.

Affichage

Écrivez et exportez dans la vue la fonction `commencerAffichage` qui ajoute une liste `<ul>` dans élément `<div id="texte">`, la fonction `ajouteReplique(texte)` qui crée un `<li>` contenant le texte `texte` de la réplique en tenant compte des sauts de ligne et l'ajoute dans la liste `<ul>`. **Écrivez** un gestionnaire à attacher sur `<p id="buttons">` qui émet un événement `'commencer'` avec pour valeur le type du bouton radio sélectionné.

Gestion

Écrivez dans contrôleur la fonction `onCommencer` qui doit être appelée quand l'utilisateur veut commencer l'affichage du texte : il faut effacer l'affichage de la vue et demander au modèle d'envoyer les répliques. **Ajoutz** la fonction `onReplique` qui doit être appelée quand le modèle émet une réplique : il faut que la vue affiche cette réplique. **Testez**.

Modifiez la vue pour qu'elle émette l'événement `'stop'` quand l'utilisateur clique le bouton `Stop`. **Ajoutez** dans le contrôleur la fonction `arreterRepliques` qui doit être appelée quand il faut stopper l'affichage et stoppe cetaffichage. **Testez**.

Faites la même chose pour pouvoir visualiser les texte phrase par phrases (elles sont séparées par les sauts de lignes) et caractère par caractère.