Le quiz est structuré autour d'un **tableau (Array)** nommé `lesQuestions`, défini au début du fichier `quizzjs.js`, et contenant des **objets littéraux**.

Chaque objet du tableau correspond à une question unique et possède les quatre propriétés suivantes :

- **`id`** : un identifiant unique (nombre entier, ex: 1).
- **`question`** : l'intitulé de la question sous forme de chaîne de caractères.
- **`contexte`** : une citation ou information historique supplémentaire (chaîne de caractères) affichée au-dessus de la question.
- **`possibilites`** : un tableau de chaînes de caractères listant les trois choix de réponses disponibles.

Cette conception sépare clairement les données de la logique d'affichage, assurant une bonne modularité du script : l'ajout d'une nouvelle question se fait simplement en insérant un nouvel objet dans le tableau.

Un point dont je suis particulièrement satisfait est la gestion des interactions utilisateur. Plutôt que de forcer l'utilisateur à cliquer précisément sur la petite case du bouton radio, j'ai implémenté dans `quizzjs.js` une gestion de l'événement `click` sur l'intégralité de l'élément de liste (`<li>`) de la réponse. Le script récupère l'élément `input[type="radio"]` enfant via `item.querySelector('input')` et active sa propriété `.checked`. Cette approche améliore significativement l'ergonomie, notamment sur mobile, en rendant l'interface plus agréable à utiliser.