Nom du Module

BILAN TP DE PROGRAMMATION

Nom : Defossez
Prénom : Théo
Nom du TD/TP : Messagerie en javascript

I. Instructions

Format

- Le bilan peut être **rédigé** sous forme de document écrit ou présenté sous forme de diaporama si cela est plus adapté.
- Le document final doit être soumis au format **PDF**. Assurez-vous que les fichiers sont bien nommés (évitez les espaces et utilisez des noms explicites, ex. `:bilanTP_NomPrenom.pdf`).

Clarté et Précision

- Assurez-vous que toutes les sections sont complétées avec soin. Prenez le temps de rédiger des explications claires et argumentées.
- Présentez les informations dans un ordre logique, avec des titres et sous-titres bien marqués pour faciliter la lecture.

Respect des délais

- Remettez votre bilan dans les délais impartis pour permettre une évaluation rapide et efficace. Les travaux remis en retard peuvent ne pas être pris en compte.

Étape N°1 : II. Fonctionnalités Implémentées

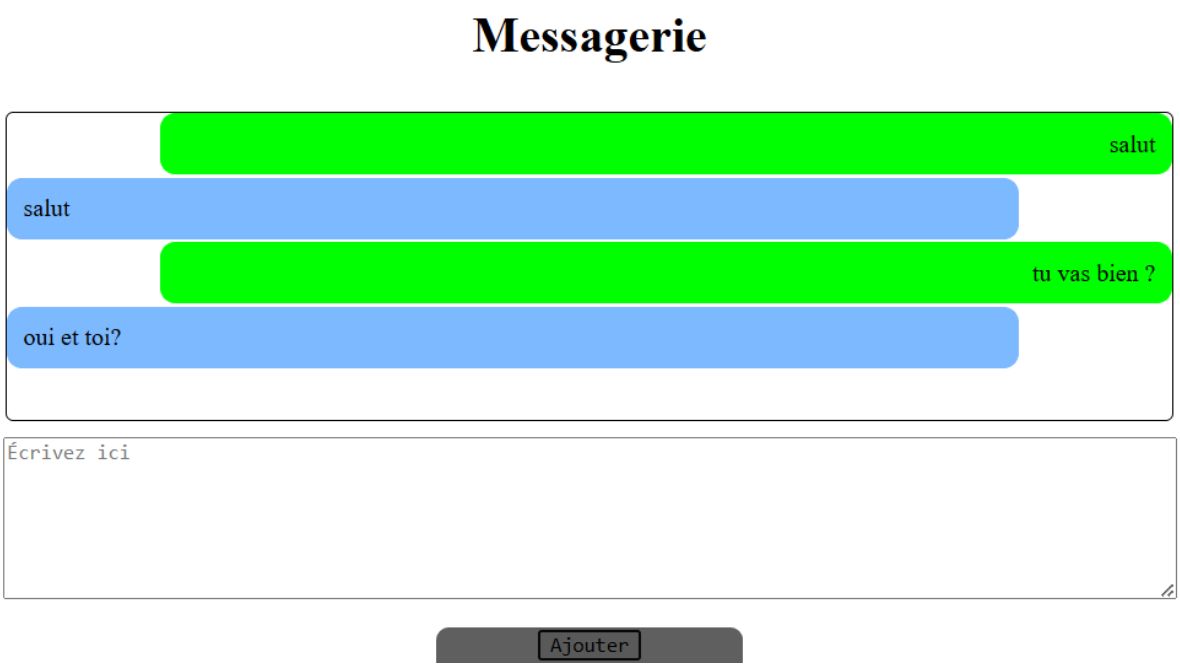
Liste des fonctionnalités développées et présentez des copies d'écran de votre application finale :

Fonctionnalité 1 : Saisie et ajout de messages: L'utilisateur peut saisir un message dans la zone de texte et l'ajouter à la messagerie en cliquant sur le bouton Ajouter.

BILAN TP DE PROGRAMMATION

Fonctionnalité 2 : Alternance des styles de messages: Les messages s'affichent alternativement à droite (en vert) et à gauche (en bleu) pour distinguer les émetteurs ou conversations.

BILAN TP DE PROGRAMMATION



Description détaillée :

- Chaque capture d'écran doit être accompagnée d'une explication claire et détaillée.
- Expliquez le rôle de chaque fonctionnalité (ex. : à quoi elle sert, pour qui elle est destinée).
- Précisez son fonctionnement (ex. : interaction utilisateur, logique derrière la fonctionnalité) et comment vous l'avez programmé.
- Mentionnez les problèmes rencontrés lors de son implémentation et les solutions apportées.
- Fonctionnalités visibles : pour chaque capture, précisez les fonctionnalités mises en avant (ex. : barre de navigation, formulaire, tableau de données).
- Aspects techniques : expliquez les choix techniques réalisés (ex. : technologies utilisées, structure du code, organisation des composants) et leur impact sur l'efficacité ou la performance de l'application.

Critères d'évaluation :

- Fonctionnalités listées avec précision.
- Explications approfondies et compréhensibles.
- Explications précises et bien argumentées pour chaque capture.
- Captures claires et pertinentes.
- Explications du code

Étape N°2 : III. Difficultés Rencontrées et Solutions Apportées

Difficultés :

Difficulté 1 : Gestion de l'alignement des messages : Initialement, il était difficile de gérer l'alignement alternatif des messages à droite et à gauche dans l'affichage. Le problème venait d'une mauvaise gestion des classes CSS associées.

Difficulté 2 : Défilement automatique de la boîte d'affichage: Lorsque de nombreux messages étaient ajoutés, les messages en bas n'étaient pas visibles sans faire défiler manuellement.

Solutions :

Difficulté 1 : Gestion de l'alignement des messages : Utilisation d'un compteur incrémenté à chaque message pour alterner les classes `droite` et `gauche`. La documentation sur la méthode JavaScript `classList.add()` a été consultée pour résoudre ce problème.

Difficulté 2 : Défilement automatique de la boîte d'affichage: Mise en place de la propriété CSS `overflow: auto` pour la boîte d'affichage. Cette solution a été trouvée en explorant des forums en ligne (comme Stack Overflow).

Critères d'évaluation :

- Difficultés bien identifiées et pertinentes.
- Solutions claires, adaptées et bien expliquées.

Étape N°3 : IV. Commentaires et Suggestions

Remarques générales :

Points Positifs :

Le projet a permis de consolider la gestion du DOM en JavaScript, notamment pour la création dynamique d'éléments HTML et la manipulation des classes CSS.

L'interface est simple et fonctionnelle, offrant une expérience utilisateur fluide.

L'alternance des styles de message (droite/gauche) fonctionne bien et améliore la lisibilité.

Points à Améliorer :

Le projet pourrait être enrichi avec des fonctionnalités supplémentaires, comme un bouton pour effacer les messages ou un horodatage des messages.

Suggestions d'amélioration :

Ajouter des fonctionnalitées avancée, comme l'ajout de fichiers multimédias (images ou vidéos) dans la messagerie.

Proposer une base de données ou une API, simulant un environnement plus proche du monde réel.

Focus sur les Apprentissages

Sur le plan technique

J'ai appris à mieux gérer l'interaction dynamique entre le DOM et JavaScript, en particulier pour créer et manipuler des éléments d'interface tels que des messages dynamiques avec des styles alternés. La gestion des événements, via l'ajout d'écouteurs (`eventListeners`), m'a permis d'approfondir mes connaissances sur la manipulation interactive des éléments HTML.

L'implémentation d'un système d'alternance des styles (droite/gauche) m'a également permis de mieux comprendre l'utilisation des classes CSS dynamiques en fonction des conditions dans le code. De plus, la validation des entrées utilisateur (empêcher l'ajout de messages vides) a renforcé mes compétences en écriture de code robuste et fonctionnel.

Sur le plan méthodologique

La gestion des erreurs, comme la saisie de messages vides ou les problèmes d'affichage avec des longueurs de texte inattendues, m'a appris l'importance de tester chaque fonctionnalité dans différents scénarios. L'utilisation des outils de debug, comme les `console.log`, m'a permis d'identifier et de résoudre rapidement les problèmes.

Ce projet m'a également sensibilisé à l'importance de structurer le code pour éviter des comportements inattendus et faciliter la lisibilité. Ces apprentissages méthodologiques seront précieux pour des projets plus complexes, nécessitant une organisation et une optimisation rigoureuses.

Critères d'évaluation :

- Observations pertinentes et réflexion approfondie.
- Suggestions concrètes et réalistes.

Étape N°4 : V. Auto-évaluation

Évaluation de votre travail :

Planification : 7/10

La planification a été assez simple étant donné que le projet était relativement basique. Cependant, je n'ai pas pris le temps de découper chaque fonctionnalité en étapes claires. Cela a parfois causé des retards, notamment lors de l'ajout de la validation des entrées et de l'ajustement de l'affichage dynamique. Une meilleure organisation en amont aurait rendu le processus de développement plus fluide.

Codage : 8/10

BILAN TP DE PROGRAMMATION

Le codage a été fluide pour les fonctionnalités principales, comme l'ajout des messages et l'alternance des styles (droite/gauche). Les concepts de manipulation du DOM étaient familiers, ce qui a accéléré le travail. Cependant, des petits ajustements ont été nécessaires pour gérer les cas particuliers, comme les messages vides ou très longs. Ces problèmes ont été résolus efficacement après quelques débogages.

Tests : 6/10

Les tests ont été réalisés manuellement en vérifiant les fonctionnalités après chaque ajout. Bien que cela ait permis de détecter rapidement certains bugs.

Critères d'évaluation :

- Auto-évaluation objective et bien argumentée.
- Objectifs pertinents et réalistes.

Étape N°5 : VI. Code

Lien GitHub :

- Ajoutez un lien vers votre dépôt GitHub et vérifiez qu'il est accessible.
- Organisez votre dépôt en plusieurs dossiers (ex. : `src` pour le code source, `images` pour les images et ressources, `docs` pour la documentation ….). Assurez-vous que chaque dossier a une utilité claire et est bien structuré.

README :

- Incluez un fichier README complet et structuré contenant :
  - Une description du projet.
  - Les instructions d'installation et d'utilisation.
  - Les prérequis et outils utilisés.

Critères d'évaluation :

- Lien GitHub fonctionnel.
- README bien élaboré.
- Code propre, bien organisé et documenté