Auteurs : Vu Trung HOANG, Benjamin WITTERS

Comparaison entre Elm et JavaScript

TcTurtle est un langage permettant à une application web qui transforme des commandes textuelles en dessins interactifs représentant les déplacements d'un crayon virtuel. Ce projet met en avant les différences entre Elm et JavaScript dans le développement d'applications web, notamment pour les applications monopages.

1. **Programmation et fiabilité**
   Elm adopte un paradigme fonctionnel avec un typage statique strict, ce qui permet de détecter les erreurs dès la compilation. À l'inverse, JavaScript, avec son typage dynamique et ses effets de bord imprévisibles, peut entraîner des bugs difficiles à anticiper. La pureté fonctionnelle d'Elm améliore ainsi la stabilité et la lisibilité du code, un atout clé pour TcTurtle.
2. **Gestion de l'affichage et de l'état**
   Elm repose sur "The Elm Architecture" (Model, Update, View), garantissant une gestion claire et prévisible de l'état. En JavaScript, l'organisation de l'interface repose souvent sur des bibliothèques comme React ou Vue.js, nécessaires pour structurer les mises à jour asynchrones. Elm est particulièrement adapté aux applications monopages (SPA), assurant un flux de données cohérent et une maintenance simplifiée.
3. **Interprétation des commandes TcTurtle**

   Le module Parser d'Elm permet une conversion efficace des commandes textuelles en structure de données exploitable. En JavaScript, cette tâche nécessiterait une bibliothèque externe comme PEG.js ou une implémentation plus complexe. De plus, Elm étant déterministe, une même entrée produit toujours le même résultat, ce qui évite les comportements aléatoires possibles en JavaScript.

4. **Création des dessins en SVG**

   Le module Svg d'Elm facilite la génération d'éléments graphiques intégrés à son architecture. En JavaScript, la manipulation des SVG repose soit sur l'API native, soit sur des bibliothèques comme D3.js, nécessitant un travail plus direct sur le DOM.

5. **Robustesse et maintenance**

   Grâce à son typage strict, Elm empêche l'exécution d'un programme contenant des incohérences, garantissant un code plus fiable dès la conception. JavaScript, en revanche, détecte certaines erreurs uniquement à l'exécution, nécessitant des tests plus approfondis. Cette rigueur d'Elm facilite la maintenance et garantit un code plus structuré sur le long terme.

En conclusion, Elm se distingue par sa stabilité, son typage strict et sa gestion optimisée de l'état, offrant un développement plus sûr et prévisible. Ces caractéristiques en font un choix idéal pour les applications monopages. Dans notre contexte, ces avantages ont permis de concevoir un interpréteur fiable et performant, garantissant une exécution fluide et un rendu graphique précis.