REACT - ANGULAR - VUE : Que choisir, comment choisir ?

I- Introduction & contexte

Dans le monde du développement front-end, le choix d'un framework est crucial. Que ce soit pour des projets personnels ou à des fins professionnelles, le choix de ce dernier influence non seulement la rapidité de développement, mais aussi la maintenabilité, l'evolutivité et la performance de l'application à long terme.

Cet article a pour but d'accompagner et d'éclairer les jeunes développeurs front-end ou full stack dans leur prise de décision en matière de framework.

Nous passerons en revue trois des frameworks les plus populaires : React, Angular et Vue.js, en détaillant leurs caractéristiques et en comparant leurs forces et faiblesses.

II- Un framework c'est quoi ?

Définition

En anglais un framework signifie « un **cadre de travail** ».

Mais qu'entend-t-on par cadre de travail ?

Dans une approche orientée logiciel, un framework représente :

*« Un **ensemble d'outils et de composants autonomes qui ont pour objectif de faciliter et uniformiser le travail des développeurs.** Cela se rapporte à une architecture sur laquelle se baser pour construire des applications ou des sites web. »*

Cependant un framework ne se limite pas à **fournir des outils,** il impose également une **organisation de code**, qui, dans un contexte de travail collaboratif, est essentiel pour maintenir une cohérence dans le développement d'une application.

L'utilisation d'un framework garantit une structuration du code se basant sur des **conventions prédéfinies**, rendant ainsi les projets plus faciles à maintenir et à faire évoluer dans le temps.

Les frameworks modernes, incluent également des solutions pour la **gestion des états**, le **routage**, et des fonctionnalités avancées comme la **gestion des formulaires** ou l'**authentification**. (cf : mot clés)

III- Librairie ? Framework ?

La *Wildcodeschool* définie une librairie comme :

« *Un **ensemble de fonctions, de classes ou de modules pré-écrits, compilés et réutilisables** qui peuvent être intégrés dans un programme pour effectuer des tâches spécifiques. On peut les visualiser comme des fonctions prêtes à l'emploi qui **évitent aux développeurs de devoir coder ou recoder** certaines fonctionnalités triviales.* »

En sorte, une bibliothèque fournit des outils spécialisés pour accomplir une tâche précise, un framework, quant à lui, prend en charge la structure globale du projet.

Par exemple, React est souvent perçu comme une bibliothèque car il se concentre uniquement sur l'interface utilisateur, laissant le choix aux développeurs d'ajouter des solutions comme React **Router** pour le routage ou **Redux** pour la gestion des états.

En revanche, Angular est un framework complet qui inclut nativement des solutions pour ces aspects, ce qui en fait un choix privilégié pour des applications complexes et à grande échelle.

IV- Scalabilité et modularité

La **scalabilité** caractérise la capacité d'adaptation d'un système informatique (ou de l'un de ses composants) d'un point de vue dimensionnel, tant vers des tailles inférieures que vers des tailles supérieures. La scalabilité peut ainsi concerner un flux, un volume, un espace-temps, etc…

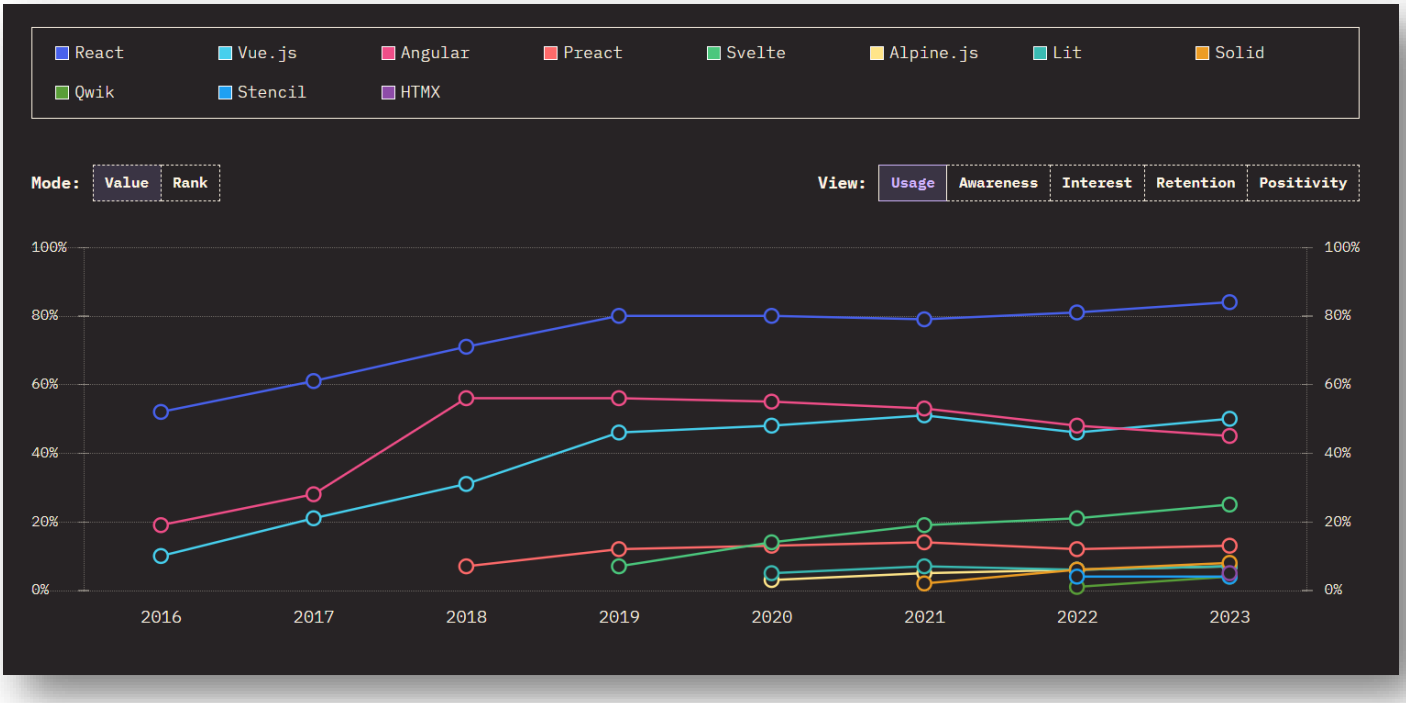
V- Popularité et tendances actuelles

Selon le rapport *State of JS 2023* :

React est le choix préféré des développeurs front-end, en grande partie grâce à sa flexibilité et à sa large communauté. **React** est particulièrement apprécié dans des contextes où la **performance de l'interface utilisateur** est primordiale, comme dans les applications **SPA** (Single Page Applications) ou les applications mobiles grâce à **React Native**.

Angular, en revanche, continue d'être populaire dans les grandes entreprises, notamment pour des applications complexes et exigeantes.

Enfin, **Vue.js** voit sa cote de popularité augmenter de plus en plus dans les PME et les startups, en raison de sa simplicité d'utilisation et de son adoption croissante au sein de la communauté.



Source: *State of JS 2023*

VI- Critères de sélection d'un framework

- Paramètre (**besoins spécifiques dans un projet**) :

- Pour une **application complexe à grande échelle** avec une équipe nombreuse, **Angular** est souvent le meilleur choix car il offre une structure robuste et un ensemble complet d'outils.

- Pour **développer rapidement une application légère** ou une interface utilisateur interactive, **React** est la solution idéale. Sa flexibilité et sa large communauté en font un choix populaire pour les applications web modernes.

- Pour une **approche progressive** ou que vous travaillez sur un **projet à taille réduite ou moyenne**, **Vue.js** est une excellente option. Il est facile à intégrer et à adapter selon les besoins du projet.

- Paramètre (**apprentissage**) :

- **React** est souvent plus facile à prendre en main pour les développeurs débutants, surtout ceux familiers avec JavaScript.

- **Angular** peut avoir une courbe d'apprentissage plus raide, en raison de la complexité de son écosystème et de ses nombreuses fonctionnalités intégrées.

- **Vue.js**, quant à lui, est réputé pour sa simplicité, ce qui en fait un bon point de départ pour les développeurs novices tout en étant suffisamment puissant pour des projets plus avancés.

- Paramètre (**évolutivité**) :

En 2024, les trois frameworks continuent d'évoluer, mais chacun suit des directions spécifiques :

- **React** se concentre de plus en plus sur l'optimisation de la performance via des fonctionnalités comme les Server Components.

- **Angular** se renforce avec des améliorations de la gestion des standalone components. (v.18.0.1)

- **Vue.js**, de son côté, améliore son écosystème avec des outils comme Nuxt 3, qui facilite le rendu côté serveur et le développement d'applications full-stack

VII- Tableau comparatif

Caractéristique	React	Angular	Vues.js
Type	Bibilothèque UI	Framework complet	Framework progressif
Langage	JS (support TS)	Typescript	JS (support TS)
Créateur	Meta	Google	Evan You (créateur de Angular)
Date de lancement	2013	2010	2014
Architecture	Basé sur des composants	Basé sur des composants et services	Basé sur des composants
Courbe d'apprentissage	Modérée	Difficile, impose une structure stricte	Facile à modérée, syntaxe intuitive
Ecosystème	Large, fragmenté (beaucoup de bibliothèques)	Intégré avec de nombreux outils	Integré avec un bon écosystème
Routing	Nécéessite des bibliothèques tierces (Redux…)	Integré avec RxJS et NgRx	Integré avec Vuex ou Composition API
Performance	Léger grâce au Virtual DOM	Lourd	Léger
Flexibilité	Flexible mais nécessite des outils externes	Approche opiniâtre, moins flexible	Modérément flexible, approche progressive
Popularité	Très populaire pour app web et mobiles	Populaire dans les grandes entreprises	En forte croissance particulièrement dans les PME
Cas d'utilisation idéal	Application interactivé nécessitant flexibilité et customization	Application à grande échelle	Projet de taille moyenne
Support mobile	React Native	Angular NativeScript	Quasar (Vue)
Philosophie	Vue centrée ('library-first')	Full stack avec gestion complete	Progressivité et simplicité

VIII- Glossaire technique

Single Page Application (SPA) : Application web qui se charge une seule fois et dont le contenu est mis à jour dynamiquement, sans recharger la page entière, ce qui améliore l'expérience utilisateur.

Virtual DOM : Concept utilisé par **React** et **Vue.js** pour optimiser la mise à jour du DOM en minimisant les manipulations réelles de la page, rendant l'interface utilisateur plus performante.

Components : Blocs réutilisables et indépendants qui encapsulent le HTML, CSS et JavaScript, utilisés dans la construction d'interfaces utilisateurs modulaires.

Data Binding : Mécanisme permettant de lier automatiquement les données de l'interface utilisateur aux modèles de données sous-jacents. **Angular** utilise notamment le **two-way data binding**.

Reactivity : Caractéristique permettant aux interfaces de réagir automatiquement aux changements de données sous-jacentes, un concept clé dans **Vue.js**.

State Management : Gestion centralisée de l'état de l'application, souvent assurée par des bibliothèques comme **Redux** pour **React** ou **Vuex** pour **Vue.js**, afin de maintenir la cohérence de l'interface.

Routing : Gestion de la navigation entre différentes vues ou pages dans une application web, pris en charge nativement dans **Angular** et via des bibliothèques comme **React Router** ou **Vue Router**.

TypeScript : Superset de JavaScript utilisé par **Angular** et compatible avec **React** et **Vue.js**, apportant un typage statique et une meilleure robustesse dans le développement de grandes applications.

SSR (Server-Side Rendering) : Technique permettant de générer les pages côté serveur pour améliorer les performances et le **SEO**, utilisée avec des outils comme **Next.js** (pour **React**) ou **Nuxt.js** (pour **Vue.js**).

Ecosystem : Ensemble des outils, bibliothèques et extensions disponibles autour d'un framework. Par exemple, l'écosystème de **React** est vaste et inclut des outils comme **Redux, React Router**, et Jest pour les tests.

Performance Optimization : Techniques pour améliorer les temps de chargement et la réactivité de l'application, notamment via des mécanismes comme la gestion du **lazy loading** ou l'optimisation des composants.

IX- Conclusion

Il est intéressant d'affirmer en guise de conclusion que le choix entre **React**, **Angular** et **Vue.js** ne repose pas unique sur une préférence visuelle du logo mais bien sur de **nombreux facteurs de divers** natures : la taille de l'équipe, la complexité du projet, les préférences personnelles du développeur etc…

Chaque framework propose son lot d'atouts et de faiblesses mais ce qu'il faut retenir c'est qu'ils sont tous **suffisamment matures pour répondre à la majorité des besoins en développement** front-end.

Il est donc essentiel d'évaluer les caractéristiques de chaque solution pour trouver celle qui correspond le mieux aux exigences de votre projet.

Faites-vous la main et essayez chacun des framework, je suis sûr que vous n'êtes pas de ceux qui portent des habits sans les essayer n'est-ce pas ? Attention quand même à en acheter plusieurs, personne n'a dans sa garde-robe un seul et unique t-shirt ? 😉

Bon coding !

Article rédigé par : Cyril Dupont

X- Source

https://www.youtube.com/watch?v=ekytObUD4KQ

https://2023.stateofjs.com/

https://www.amiltone.com/javascript-top-frameworks/

https://uk.linkedin.com/company/amigoscode

https://uk.linkedin.com/in/nelsonamigoscode?trk=org-employees