Interfaces utilisateur pilotées par les composants

Présenté le 12 octobre 2021 (mis à jour le 13 octobre)

Atomic Design

- La pratique de développement dites “Atomique” est de conception et de construction d'interfaces utilisateur avec des composants modulaires. Les interfaces utilisateur sont construites « de bas en haut » en commençant par les composants de base, puis progressivement combinées pour assembler les écrans.

Pourquoi des composants ?

Les interfaces utilisateur modernes sont plus compliquées que jamais. Les gens s'attendent à des expériences convaincantes et personnalisées sur tous les appareils. Cela signifie que les développeurs et les concepteurs frontaux doivent intégrer plus de logique dans l'interface utilisateur.

Mais les interfaces utilisateur deviennent lourdes à mesure que les applications se développent. Les grandes interfaces utilisateur sont fragiles, difficiles à déboguer et longues à expédier. Les décomposer de manière modulaire facilite la création d'interfaces utilisateur robustes mais flexibles.

Les composants permettent l'interchangeabilité en isolant l'état de la logique métier de l'application. De cette façon, vous pouvez décomposer des écrans complexes en composants simples. Chaque composant a des attributs bien définies et une série fixe d'états peuvent être simulés. Cela permet aux composants d'être démontés et recomposés pour créer différentes interfaces utilisateur.

Quels sont les composants ?

Les composants sont des blocs de construction standardisés et interchangeables d'interfaces utilisateur. Ils encapsulent l'apparence et la fonction des éléments de l'interface utilisateur.

Pensez aux briques LEGO. Les LEGO peuvent être utilisés pour tout construire, des châteaux aux vaisseaux spatiaux ; les composants peuvent être démontés et utilisés pour créer de nouvelles fonctionnalités.

Comment être piloté par les composants ?

- Construire un composant à la fois
- Combiner des composants
- Assembler des pages
- Intégrez des pages dans votre projet
- Tester de bout en bout

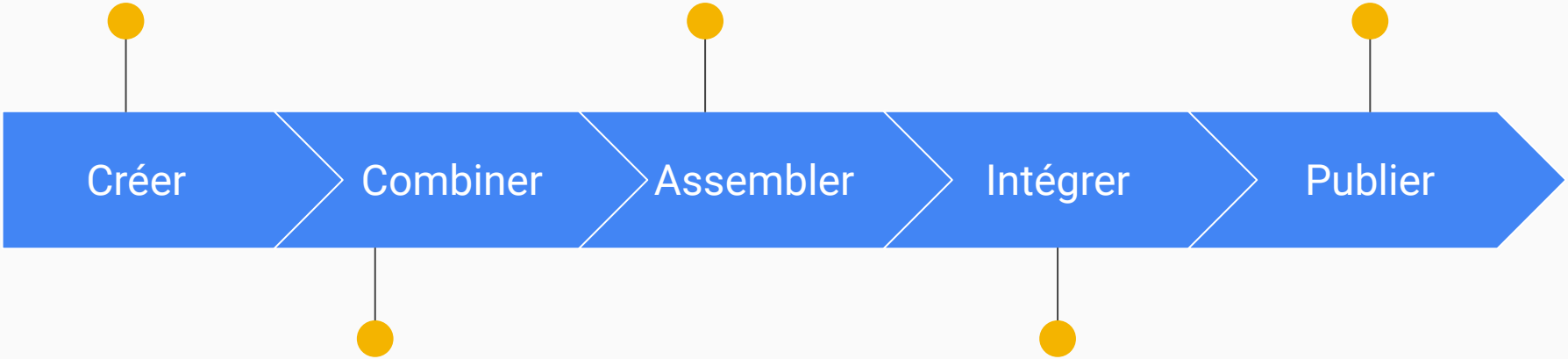
Processus de développement

Avatar / Bouton /

Input / Info-bulle

Page d'accueil / Page des paramètres / Page de profil

Compiler et mettre en production



Forms / Entête

Liste / Table

Application Web

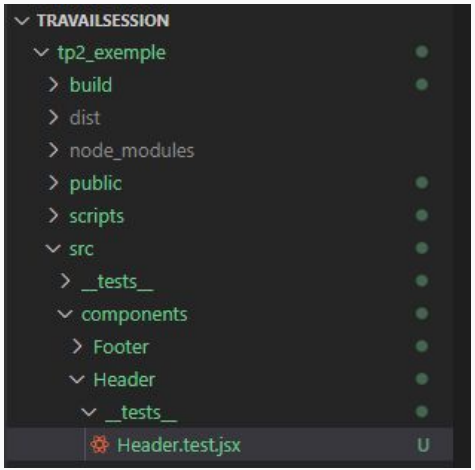
Site de commercialisation

Site de documentation

Tester ?

Les tests doivent être le plus près possible

Dans l’exemple que vous avez vu pendant le cours, il est important de conserver la mentalité d’avoir les tests le plus près des composantes possibles. Que ce soit un bouton, un header ou une page, la meilleure pratique est d’avoir les tests le plus près possible du composante visé par le test.



Avantages

- **Qualité :** vérifiez que les interfaces utilisateur fonctionnent dans différents scénarios en créant des composants isolément et en définissant leurs états pertinents.
- **Durabilité :** identifiez les bugs dans les moindres détails en testant au niveau des composants. C'est moins de travail et plus précis que de tester des écrans.
- **Vitesse :** assemblez les interfaces utilisateur plus rapidement en réutilisant les composants existants d'une bibliothèque de composants ou d'un système de conception.
- **Efficacité :** Parallélisez le développement et la conception en décomposant l'interface utilisateur en composants discrets, puis en partageant la charge entre les différents membres de l'équipe.

Quand les interfaces utilisateur ne sont pas pilotées par les composants ?

- **Basé sur les pages :** processus de développement et de conception qui traitent un site Web comme un ensemble de pages. Peu d'efforts sont faits pour réutiliser les éléments communs sur les pages.
- **Outils conçus pour les pages :** outils qui se concentrent sur l'affichage de documents tels que Wordpress et Drupal. Des frameworks backend tels que Rails, Django et PHP qui traitent la réutilisation de l'interface utilisateur comme une réflexion après coup et découragent la réutilisation généralisée des composants.

Tendances complémentaires

Systèmes de conception : approche holistique de la conception d'interfaces utilisateur qui documente tous les modèles d'interface utilisateur dans un système centralisé comprenant des actifs (Sketch, Figma, etc.), des principes de conception, une gouvernance et une bibliothèque de composants.

JAMStack : une méthodologie de création de sites Web qui pré-rend des fichiers statiques et les sert directement à partir d'un CDN (par opposition à un serveur). Les interfaces utilisateur des sites JAMStack reposent sur des frameworks JavaScript à composants.

Agile : Une méthode de développement logiciel qui favorise des boucles de rétroaction courtes et une itération rapide. Les composants aident les équipes à expédier plus rapidement en réutilisant des blocs de construction prêts à l'emploi. Cela permet aux équipes agiles de se concentrer davantage sur l'adaptation aux besoins des utilisateurs.

Notions importantes

Développement axé sur les composants : Une méthodologie de développement qui ancre le processus de construction autour des composants. Il s'agit d'un processus qui construit des interfaces utilisateur « de bas en haut » en commençant au niveau des composants et en se terminant au niveau des pages ou des écrans.

Conception atomique : La conception atomique est constituée d'atomes, de molécules, d'organismes, de modèles et de pages fonctionnant simultanément pour créer des systèmes de conception d'interface efficaces. C'est un modèle mental pour nous aider à penser à nos interfaces utilisateur à la fois comme un tout cohérent et une collection de parties en même temps.

Pourquoi un format standard ?

Les composants ont augmenté pour dominer le paysage de l'interface utilisateur. Il existe de nouveaux outils orientés composants pour le développement, les tests, la conception et le prototypage. Ces outils s'engagent dans la création et la consommation de composants et d'exemples de composants (aka histoires). Mais chaque outil a son propre format propriétaire car il n'existe pas encore de moyen simple et indépendant de la plate-forme d'exprimer des exemples de composants.

Format de l'histoire du composant

Le Component Story Format est un standard ouvert pour les exemples de composants basés sur les modules JavaScript ES6. Cela permet l'interopérabilité entre les outils de développement, de test et de conception.

Qui utilise CSF ?

Outils: Storybook , WebComponents.dev , RedwoodJS , UXPin

Compatible avec : Jest , Enzyme , Testing Library , Cypress , Playwright , Mocha , etc.

Le storybook est une bibliothèque de composantes ES6 qui peuvent être basé sur l’un des frameworks suivant.

Les composantes de base d’un storybook

Exemple : https://storybook.js.org/docs/react/get-started/examples

Framework de composants gratuits

https://technostacks.com/blog/react-component-libraries

https://react.libhunt.com/ant-design-alternatives

Un storybook peut être une présentation des composantes disponible d’un framework avec une thématique de couleurs / fonts … etc. Un storybook peut également contenir un regroupement de composantes qui peuvent, ensemble, formé une fonctionnalité. Exemple : Réserver une table à un restaurant.

Du code au design

En résumé, à partir des outils web de conception graphique tel que Sketch, Figma, Zeplin, etc, vous pouvez connecter vos composantes react à partir d'un storybook et l'importer dans l'outil de design graphique de votre choix tel que Figma.

Exemple : Projet React avec des composantes => StoryBook => Composantes Figma

(https://help.figma.com/hc/en-us/articles/360045003494-Storybook-and-Figma)

Il peut être bénéfique de ré-utiliser les composantes du framework de composantes pré-sélectionné de votre choix, tel que material-ui, avec une touche personnalisé afin de démarrer un storybook.

Apprendre à créer des composantes react avant un storybook est un prérequis important.

Pour terminer, pour faire de bons gâteaux, ça prend de bons moules et surtout un bon appétit.