Engineer

Interface utilisateur

partie d'un programme qu'une personne voit et utilise

Ce que c’est

UI est l’abréviation de user interface, ou interface utilisateur : les écrans, menus, champs, boutons, messages et sons par lesquels une personne indique à un programme quoi faire et découvre ce qu’il a fait. C’est aussi l’ensemble des règles qui les régissent : quel contrôle est accessible depuis quel endroit, ce qui se passe après un clic et ce que dit le programme lorsqu’un problème survient.

Le glossaire du NIST adopte la définition la plus large : une interface utilisateur est « le moyen physique ou logique par lequel les utilisateurs interagissent avec un système, un appareil ou un processus ». Le côté physique comprend le clavier, l’écran tactile, le bouton rotatif d’un appareil. Le côté logique couvre tout ce qui est dessiné ou imprimé en réponse : une fenêtre avec des menus, une invite attendant une commande saisie, une voix demandant un numéro. Une interface graphique, une ligne de commande et un menu vocal sont tous des interfaces utilisateur, et un même programme en propose souvent plusieurs.

Sur le web, les WCAG nomment l’élément de base. Un *composant d’interface utilisateur* est « une partie du contenu perçue par les utilisateurs comme un contrôle unique pour une fonction distincte ». Les notes comptent un composant selon ce que l’utilisateur perçoit comme un contrôle, et non selon sa programmation. Un seul élément programmatique peut donc contenir plusieurs composants ; l’exemple du standard est un contrôle permettant de parcourir le contenu par ligne, par page ou jusqu’à un point quelconque : chaque mode de déplacement doit avoir son nom et compte comme son propre composant.
Les éléments de formulaire, les liens et les contrôles créés par des scripts sont concernés.

Deux termes voisins l'accompagnent. L'*UX*, user experience ou expérience utilisateur, s'intéresse plus largement à ce que l'on vit en utilisant le produit du début à la fin : la tâche est-elle accomplie, combien de temps faut-il et comment les erreurs sont-elles traitées ? L'UI est la partie de cette expérience que l'on peut désigner sur un écran. Un *système de design* est un ensemble partagé de composants, de jetons et de règles dont une équipe tire son interface, pour que deux écrans réalisés par deux personnes à deux mois différents gardent l'apparence et le comportement d'un même produit.

Utilisez le concept pour prendre une décision

Utilisez « Qu'est-ce qu'une interface utilisateur ? » pour éclairer une décision réelle. Nous pouvons vous aider à relier ce concept au système que vous construisez ou achetez. Des interfaces rapides, utilisables sur téléphone et cohérentes dans tous les navigateurs de vos clients.

Découvrir nos prestations Contactez-nous →

Vérifiez les hypothèses derrière le terme

Appuyez-vous sur l'explication et les réalisations liées pour formuler les questions de votre projet. Nous pouvons examiner les hypothèses avec vous avant le choix d'un outil. Développement cartographique GIS full-stack — cartographie full-stack, de la base au navigateur

Développement cartographique GIS full-stack Contactez-nous →

Pourquoi c'est important

Un programme n'apporte de valeur aux personnes que par son interface. Une fonctionnalité que seules trouvent les personnes sachant déjà où elle se situe n'a, pour la plupart des utilisateurs, jamais été livrée. Un tel écart apparaît rarement dans une suite de tests, puisque chaque fonction marche ; il se révèle dans les demandes d'assistance et les formulaires abandonnés.

La cohérence s'apprend une fois et se réutilise. Les personnes reportent ce qu'elles ont appris d'un écran au suivant. Le critère de succès 3.2.4 des WCAG 2.2, Identification cohérente, demande que les composants ayant la même fonction au sein d'un ensemble de pages web soient identifiés de façon cohérente. Son document explicatif en donne la raison simplement : lorsqu'une même fonction porte deux noms à deux endroits, les personnes peuvent supposer qu'il s'agit de deux choses différentes ou passer du temps à découvrir que ce n'est pas le cas.

Les erreurs font partie de l'interface. Chaque formulaire reçoit des données incorrectes, et sa façon d'y répondre relève autant de la conception que sa disposition. Le critère 3.3.1, Identification des erreurs, demande qu'une erreur détectée désigne le champ et décrive le problème par du texte. Le critère 3.3.3, Suggestion après une erreur, va plus loin : lorsque le programme sait comment corriger des données incorrectes détectées automatiquement, il propose des

corrections, sauf si cela compromet la sécurité ou la finalité du contenu. Le document explicatif du critère donne l'exemple d'un champ qui attend un nom de mois : à une personne saisissant « 12 », il propose soit la liste des valeurs acceptées, soit la question « Voulez-vous dire décembre ? ».

Chaque contrôle doit aussi pouvoir être décrit aux logiciels. Le critère 4.1.2, Nom, rôle et valeur, demande que les programmes puissent déterminer le nom et le rôle de chaque composant d'interface utilisateur, y compris ceux créés par des scripts, et que les modifications de son état leur soient signalées. Un contrôle qui ne fait que ressembler visuellement à un contrôle est invisible pour les technologies d'assistance dont beaucoup de personnes dépendent pour l'atteindre.

À quoi cela ressemble en pratique

Le cas le plus clair est une plateforme maritime dont l'UI et l'UX du produit ont été conçues de bout en bout. Elle présente des professionnels, des entreprises, des navires, des emplois et des avis ; elle devait être assez dense pour accueillir de vraies données maritimes sans devenir un mur de données. Quelques décisions ont fait l'essentiel. Les grilles d'annuaire basculent entre cartes et tableau, et l'application mémorise la vue choisie. Les formulaires affichent leurs exigences au-dessus du champ dans une liste dynamique : chaque règle devient verte lorsqu'elle est satisfaite et reste ambre sinon, ce qui guide les personnes pendant la saisie plutôt que de les corriger après l'envoi. Le fil d'Ariane et la disposition à deux colonnes se répètent sur chaque page d'entité, donnant aux différents écrans la cohérence d'un même produit. Les erreurs ouvrent une voie : une redirection plutôt qu'une impasse. La page pose directement la question de la confiance : sur une plateforme utilisée pour une carrière professionnelle, une apparence négligée donnerait moins confiance dans les données, à juste titre.

L'enjeu commercial apparaît le plus clairement dans 200 améliorations d'UI et d'UX pour une application cartographique SIG. L'interface était devenue puissante et compliquée, avec de bonnes fonctionnalités derrière des interactions maladroites. Au lieu de deviner, le travail est parti des retours et des données d'utilisation pour produire un backlog de 200 modifications concrètes, les classer par impact et faire aboutir les plus importantes avec l'équipe jusqu'à leur livraison comme véritables fonctionnalités. L'interface est devenue sensiblement plus facile à utiliser et ce travail a contribué à multiplier le chiffre d'affaires par dix.

Maintenir la cohérence d'une interface dans le temps est un autre problème, traité par un système de design iOS 26 « Liquid Glass ». Le langage visuel associe des panneaux translucides au fond flouté, des ombres superposées, une légère réfraction et des animations à ressort, à partir des utilitaires Tailwind. Chaque écran est composé à partir d'un inventaire fixe de composants, dont Card, Label, Button, GlassIconButton et DirectoryGrid. Les règles de lint rejettent un badge ou une puce codés sur mesure et orientent vers le composant de référence ; la documentation associe les erreurs concrètes à leurs solutions. La cohérence cesse de dépendre de la personne qui relit la modification ce jour-là : les outils la maintiennent.

Où les problèmes surviennent

- **La couleur porte seule le sens.** Un champ entouré de vert lorsqu'il est valide et de rouge sinon ne dit rien à une personne incapable de distinguer ces couleurs. Le document Understanding pour Use of Color cite précisément ce cas et demande un second indice visuel. Un mot, une icône ou un changement de forme à côté de la couleur suffit.
- **Un élément qui ressemble à un contrôle n'en est pas un.** Une boîte mise en forme et dotée d'un gestionnaire de clic ressemble à un bouton pour qui utilise une souris, mais n'existe pas pour qui utilise un clavier ou un lecteur d'écran. Les contrôles natifs de la plateforme possèdent déjà un rôle, un focus et un comportement au clavier qu'une imitation doit reconstruire à la main, pièce par pièce.
- **Chaque écran réinvente ses composants.** Chaque nouvelle vue reçoit ses propres boutons, badges et cartes, tous légèrement différents, jusqu'à ce qu'une seule modification exige de retoucher cinq versions du même élément. Un inventaire unique de composants, imposé par un linter plutôt que par la mémoire, évite cette dérive.
- **Les erreurs arrivent tard et loin de leur cause.** Un formulaire qui accepte tout puis renvoie en haut un bloc rouge listant les problèmes oblige les personnes à chercher les champs. Nommer le champ, décrire le problème juste à côté avec des mots et suggérer une correction transforme la même vérification en aide.
- **L'interface est conçue pour le bureau de son créateur.** Un grand écran, une souris, une bonne vue et une connexion rapide décrivent la personne qui l'a faite, pas celles qui l'utiliseront. Un téléphone tenu d'une main, une page agrandie ou un clavier sans souris révèlent chacun des problèmes qu'un bureau ne montre jamais.

Pour aller plus loin

- Glossaire NIST CSRC : interface utilisateur — la définition en une ligne de NIST SP 800-152, qui couvre les moyens physiques et logiques d'interagir avec un système.
- WCAG 2.2 : composant d'interface utilisateur — la définition par le contrôle perçu, avec des notes sur la relation entre les composants et les éléments qui les mettent en œuvre.
- Comprendre le critère 3.2.4 : Consistent Identification — pourquoi les composants de même fonction doivent porter le même nom sur un site et quels composants répétés sont les plus concernés.
- Comprendre le critère 3.3.3 : Error Suggestion — ce qu'un formulaire doit dire lorsqu'il sait comment corriger des données incorrectes, avec des exemples détaillés.
- Comprendre le critère 1.4.1 : Use of Color — quand la couleur peut porter le sens et quand un second indice visuel est nécessaire.

Glossaire

Choisissez selon vos contraintes réelles

Présentez le processus de travail, les interfaces existantes et les utilisateurs qui ont besoin du changement. Nous pouvons définir une livraison testable et une transmission permettant à votre équipe d'assurer la maintenance des éléments livrés.

Consulter les tarifs Contactez-nous →

Rendez la terminologie concrète

Demandez comment « Qu'est-ce qu'une interface utilisateur ? » s'applique à votre organisation. Un exemple concret tiré de votre système actuel suffit pour engager la discussion.

Découvrir nos prestations

welcome@engineer.company +45 50 10 61 84 Copenhague, Danemark

Emportez cette page avec vous

Fiche de contact

Créé avec soin et amour par D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Droit d'auteur © 2025 – aujourd'hui · Engineer ApS · Tous droits réservés.

Imprimé depuis https://engineer.company/fr/glossary/ui/