Engineer

macOS

Système d'exploitation de bureau d'Apple

Ce que c'est

macOS est le système d'exploitation livré par Apple sur chaque Mac : le bureau et le Finder, les applications fournies et, dessous, un système Unix avec terminal, shell et outils habituels en ligne de commande. Apple le construit pour son propre matériel et vend les deux ensemble : choisir macOS revient donc en pratique à choisir un Mac.

Son socle est Darwin, dont Apple publie les sources. Son noyau XNU est un hybride réunissant le noyau Mach de l'université Carnegie Mellon, du code de FreeBSD et IOKit, framework C++ de pilotes ; son nom signifie « X is Not Unix », soit « X n'est pas Unix ». Une grande part de l'espace utilisateur vient aussi de FreeBSD, et les pages de manuel de `sed` et `date` sur Mac décrivent les versions FreeBSD de ces outils. Depuis macOS 10.15 Catalina, le shell par défaut est zsh.

Le matériel sous-jacent a changé récemment. Fin 2020, Apple a commencé à remplacer les processeurs Intel des Mac par ses propres puces, Apple silicon ; chaque gamme, du MacBook Air au Mac Pro, possède désormais des modèles Apple silicon. Ces puces sont arm64, ce qui compte dès qu'un logiciel construit sur Mac doit fonctionner sur un serveur x86-64.

Sur les ordinateurs portables et de bureau, macOS se compare à Windows et Linux.

Utilisez le concept pour prendre une décision

Utilisez « Qu'est-ce que macOS ? » pour éclairer une décision réelle. Nous pouvons vous aider à relier ce concept au système que vous construisez ou achetez. Portable, imprimante, connexion et messagerie : des problèmes réglés rapidement, en termes simples, sans mettre personne mal à l'aise.

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. Création d'un service d'assistance technique — 10,000 clients servis par un service de support créé de toutes pièces

Création d'un service d'assistance technique Contactez-nous →

Pourquoi c’est important

Pour une entreprise de développement logiciel, macOS joue deux rôles faciles à confondre : c’est l’environnement de travail de nombreux développeurs et une plateforme sur laquelle les clients exécutent des logiciels.

Comme poste de travail, un Mac diffère sur trois points des serveurs Linux ciblés par ses déploiements. Ses outils en ligne de commande viennent de FreeBSD plutôt que de GNU : une même option peut donc changer de sens dans un script. Son processeur est arm64, alors que le serveur peut être x86-64. Enfin, il n’a pas de noyau Linux, tandis que les conteneurs Linux partagent celui de leur hôte : les outils qui exécutent ces conteneurs sur Mac démarrent donc d’abord une machine virtuelle Linux. L’outil `container` d’Apple, qui exige Apple silicon et macOS 26, donne à chaque conteneur sa propre machine virtuelle légère. Rien de cela n’écarte le Mac : il faut tester sur le système de déploiement plutôt que se fier à une exécution réussie sur un portable.

Comme plateforme, macOS pose la question de la manière de développer pour lui. Une enveloppe multiplateforme, une seule base de code avec moteur de navigateur embarqué livrée sur Windows, Linux et Mac, permet de maintenir à peu de frais des versions cohérentes sur plusieurs systèmes. Mais sur Mac, elle transporte un environnement de navigateur inutile à une application native et dessine ses propres contrôles plutôt que ceux du système. Une application native en Swift et SwiftUI utilise les contrôles et matériaux visuels du système, et suit sans code supplémentaire les réglages de présentation, couleur d’accentuation, contraste, taille du texte et réduction des animations. Le prix est un client distinct et des personnes connaissant la plateforme. Le natif se justifie lorsque les clients utilisent des Mac.

La livraison sur Mac passe aussi par un contrôle. Un logiciel distribué hors du Mac App Store est signé avec un certificat Developer ID et envoyé au service de notarisation d’Apple : une analyse automatique du contenu malveillant et des problèmes de signature, distincte d’App Review. Depuis macOS 10.15, tout logiciel construit après le 1er juin 2019 et distribué avec Developer ID doit être notarisé. Apple renvoie un ticket à agrafer au logiciel et publie le même en ligne,

où Gatekeeper le cherche à la première ouverture. Le certificat est inclus dans l'Apple Developer Program, à 99 $ par an. Prévoyez l'adhésion et la signature dès la première version.

À quoi cela ressemble en pratique

L'expérience de bureau d'un produit de messagerie sur macOS était une enveloppe multiplateforme qui n'avait ni l'apparence ni le comportement d'une application Mac. Son cœur Rust bien testé expose une interface C, et Swift sait appeler du C : un client natif en Swift et SwiftUI a donc été construit directement dessus. Le cœur est lié comme archive statique depuis une révision amont fixée, ce qui rend la construction reproductible ; une fine couche d'adaptation gère la propriété de chaque pointeur traversant la frontière des langages. Le client démarre comme application Mac, utilise les matériaux et contrôles de la plateforme et n'embarque aucun navigateur. La page expose franchement le manque : aucune distribution notarisée ni aucun canal de mise à jour. C'est donc une application fonctionnelle, plutôt qu'un produit utilisé par d'autres personnes. La notarisation décrite plus haut sépare les deux.

L'apparence de la plateforme peut être imposée plutôt qu'imitée. Un système de conception de 48 jetons nommés, répartis en six groupes, fonde couleurs et matériaux sur les couleurs sémantiques et le matériau vitré de la plateforme plutôt que sur des valeurs fixes. L'interface suit ainsi la présentation du système, la couleur d'accentuation et le contraste sans code de surveillance, et ses styles de texte suivent la taille choisie par l'utilisateur. Un linter refuse un nombre brut là où un jeton est requis, une taille de police codée en dur ou une animation ignorant la réduction des animations. La page précise aussi la limite : les jetons rendent une interface cohérente, ce qui ne suffit pas à la rendre bonne.

La mémoire compressée et le swap permettent à une application macOS de dépasser largement la mémoire physique. Un compte rendu d'incident documente des rapports de diagnostic indiquant 111 Go de pages compressées sur une machine de 36 Go le 2026-09-01, et un échantillon distinct d'empreinte mémoire montrant une croissance d'environ 41 Mo par seconde lors d'une boucle de reconnexion. Le circuit de traitement des événements conservait les événements de journalisation du cœur dans un flux principal sans limite et les copiait vers des abonnés qui ne les utilisaient pas. Les tampons ont été bornés, les abonnements filtrés par type d'événement et la journalisation dotée d'un budget de débit. Une comparaison d'exécutions complètes des tests a enregistré une baisse de 610 996 à 1 411 lignes du cœur : ce sont des mesures de tests, ni une mesure de mémoire après correction ni un test de performance en

production. Le compromis est explicite : des événements de diagnostic répétitifs peuvent être abandonnés et l'état concerné actualisé au lieu de conserver des événements sans limite. Des délais d'arrêt distincts ont exigé une correction distincte.

Où les problèmes surviennent

Le Moniteur d'activité indique une mémoire libre presque nulle sur un Mac fonctionnant parfaitement. macOS met les fichiers récents en cache dans la mémoire inutilisée et, lorsqu'elle se remplit, compresse les applications inactives pour laisser de la place aux actives. Il est donc normal que Memory Used (mémoire utilisée) approche Physical Memory (mémoire physique) et que Compressed (mémoire compressée) soit élevé. Apple indique qu'une mémoire libre ne rend pas nécessairement le Mac plus rapide. Lisez plutôt le graphique Memory Pressure (pression sur la mémoire) : vert signifie une utilisation efficace de la RAM, jaune un éventuel besoin futur de mémoire supplémentaire, rouge un besoin présent. Le swap est le signal le plus net :

```
sysctl vm.swapusage
vm_stat | grep -E '^Swap(ins|outs)'
```

Les compteurs de `vm_stat` comptabilisent les événements depuis le démarrage : comparez deux lectures à une minute d'intervalle. Des écritures vers le swap qui continuent d'augmenter signifient que le travail a dépassé la capacité de la RAM.

Un script fonctionnant sur Mac s'arrête avec une erreur étrange sur le serveur Linux, ou l'inverse. Le `sed` dérivé de FreeBSD sur Mac traite le mot après `-i` comme suffixe de sauvegarde : l'habitude GNU `sed -i 's/a/b/' file` prend donc l'expression pour le suffixe et le nom du fichier pour le script. L'habitude Mac `sed -i '' …` fait lire à GNU `sed` l'expression comme un nom de fichier. Un suffixe accolé à l'option fonctionne des deux côtés :

```
sed -i.bak 's/^port=80$/port=8080/' app.conf && rm app.conf.bak
```

La correction durable consiste à exécuter les scripts en CI sur le système qu'ils rencontreront en production.

Une version partagée hors de l'équipe est bloquée par Gatekeeper à son ouverture. L'application a été signée pour le développement, ou signée sans être notarisée. Signez-la avec un certificat Developer ID, l'environnement d'exécution renforcé activé et un horodatage sécurisé, envoyez-la au service de notarisation et agrafez le ticket pour que Gatekeeper le trouve même hors ligne. Une archive ZIP peut être notarisée mais ne peut pas porter de ticket : agrafez-le donc à l'application qu'elle contient :

```
ditto -c -k --keepParent App.app App.zip
xcrun notarytool submit App.zip --keychain-profile notary --wait
xcrun stapler staple App.app
```

Une image de conteneur fonctionne sur le Mac du développeur puis s'arrête sur le serveur avec `exec format error`. Docker construit par défaut pour la plateforme du démon BuildKit ; sur un Mac Apple silicon, celui-ci fonctionne dans une machine virtuelle arm64. Un conteneur partage le noyau de l'hôte : sans émulation, un hôte x86-64 n'exécute donc que du code x86-64. Vérifiez le serveur avec `uname -m`, puis construisez pour lui avec `docker build --platform linux/amd64`, ou en CI sur une machine du même type de processeur.

Pour aller plus loin

- Guide d'utilisation du Mac — le guide macOS d'Apple, maintenu pour chaque version actuelle.
- Vérifier si votre Mac a besoin de plus de RAM — le graphique de pression sur la mémoire et le sens de ses trois couleurs.
- Notariser un logiciel macOS avant distribution — Developer ID, l'environnement d'exécution renforcé et l'utilisation du ticket par Gatekeeper.
- Constructions multiplateformes — pourquoi les hôtes arm64 et amd64 ont besoin d'images différentes, et comment une construction produit les deux.
- container — l'outil d'Apple exécutant des conteneurs Linux sur Mac, chacun dans une machine virtuelle légère.
- XNU — les sources du noyau macOS et un bref exposé de l'articulation entre Mach, FreeBSD et IOKit.

Glossaire

Choisissez selon vos contraintes réelles

Décrivez l'appareil ou le service, les symptômes observés et les données à préserver. Nous pouvons déterminer l'étape de diagnostic utile la plus sûre.

Consulter les tarifs Contactez-nous →

Rendez la terminologie concrète

Demandez comment « Qu'est-ce que macOS ? » 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/macos/