+33 (0)1 48 10 09 46
formation@ambient-it.net
140 Avenue Jean Lolive 93500 Pantin - France

Mis à jour le 05/02/2026

S'inscrire

Formation Tauri : Applications Desktop Modernes

3 jours (21 heures)

Présentation

Tauri est un toolkit de développement permettant de créer des applications desktop multiplateformes à partir d'un frontend Web. Basé sur un cœur Rust et une WebView, il permet de construire des applications légères, rapides et avec une approche sécurisée pour l'exposition des capacités natives.

Notre formation Tauri vous permettra de concevoir une application desktop moderne en maîtrisant l'architecture WebView + backend Rust, la communication IPC, la gestion des permissions, ainsi que l'intégration avec vos frameworks front (React, Vue, Svelte, etc.).

Vous apprendrez à structurer un projet Tauri, exposer des commandes Rust fiables, sécuriser les accès aux fonctionnalités natives et packager l'application pour Windows, macOS et Linux, tout en adoptant des pratiques orientées production.

À l'issue de la formation, vous serez en mesure en mesure de développer une application Tauri complète, d'industrialiser son cycle de livraison (build, bundling, CI/CD), de durcir sa configuration et d'optimiser l'expérience utilisateur grâce à une approche pratique et opérationnelle.

Comme toutes nos formations, celle-ci vous présentera **la dernière version stable** de la technologie et ses nouveautés.

Objectifs

- Comprendre l'architecture Tauri et le modèle WebView + backend Rust.
- Mettre en place un projet Tauri et intégrer un frontend Web existant.
- Développer des commandes IPC et structurer la couche d'intégration.
- Sécuriser l'application via permissions, bonnes pratiques et durcissement.
- Packager et distribuer l'application sur plusieurs OS.

- Industrialiser avec tests, observabilité et CI/CD.

Public visé

- Développeurs Front-end
- Développeurs Fullstack
- Développeurs souhaitant créer des applications desktop multiplateformes

Pré-requis

- Bonnes bases en JavaScript/TypeScript et tooling Web
- Notions de ligne de commande et de gestion de dépendances (npm/pnpm/yarn)
- La connaissance de Rust est un plus (mais non indispensable)

Formation Tauri

[Jour 1 - Matin]

Fondations Tauri et mise en route

- Comprendre Tauri : toolkit desktop à frontend web et cœur Rust
- Installer l’environnement : Rust, Node.js, Tauri CLI
- Structure d’un projet : src-tauri, config, cycle de vie
- Premiers écrans : fenêtre, assets, navigation
- Bonnes pratiques de démarrage (templates, conventions, repo)
- Atelier pratique : Initialiser une app Tauri.

[Jour 1 - Après-midi]

Architecture, sécurité et modèle d’exécution

- Architecture WebView + backend Rust et échanges par messages
- Modèle de sécurité : surface d’API, principe du moindre privilège
- Gestion des permissions et réduction de l’exposition des commandes
- Configuration : fichiers, environnements, profils dev/prod
- Gestion des secrets et des variables d’environnement
- Atelier pratique : Verrouiller une commande, tester un appel autorisé/refusé.

IPC et API JavaScript Tauri

- Concepts IPC : commandes, events et sérialisation
- Utiliser @tauri-apps/api : window, dialog, fs, shell (selon besoins)

- Gestion d’erreurs : typage, retours, affichage UI
- Patterns d’architecture : services front, couche “bridge”
- Debug : logs, devtools, traces côté Rust et côté UI

[Jour 2 - Matin]

Intégration Frontend (React/Vue/Svelte/Vanilla)

- Brancher un framework : build web + intégration Tauri
- Routage, state management, stockage local (selon stack)
- UI desktop : raccourcis, menus, tray (principes)
- Gestion des assets, i18n, thèmes, packaging UI
- Atelier pratique : Intégrer un frontend existant et construire un écran “Settings”.

[Jour 2 - Après-midi]

Rust côté Tauri : commandes, state et async

- Créer des commandes Rust robustes (entrées/sorties typées)
- Gestion du state applicatif (cache, configuration runtime)
- Async : tâches longues, threads, non-blocant pour l’UI
- Gestion d’erreurs idiomatique (Result, erreurs métier)
- Sécurité des entrées : validation, limites

Plugins et fonctionnalités natives

- Comprendre l’écosystème de plugins
- Ajouter une capacité native (fichiers, notifications, OS integration)
- Séparer “core” et “features” activables
- Stratégies de mise à jour et compatibilité
- Atelier pratique : Ajouter un plugin + exposer une fonctionnalité native via une commande.

[Jour 3 - Matin]

Build, bundling et distribution

- Build multi-plateformes : contraintes et bonnes pratiques
- Bundling : artefacts, signatures, icônes, métadonnées
- Gestion des versions, changelog, stratégie de release
- Optimisation : features, stripping, gestion des assets
- Atelier pratique : Produire un bundle/installateur et valider sur 2 OS.

[Jour 3 - Après-midi]

Qualité, tests et CI/CD

- Tests : unitaires Rust, tests UI, tests d’intégration
- Pipeline CI : caches, matrix OS, artefacts
- Gestion des secrets CI et signature (selon contexte)
- Analyse statique : lint, format, audit dépendances
- Atelier pratique : Pipeline CI build + artefacts de release.

Performance, durcissement et mise en production

- Observabilité : logs, crash reports, diagnostics
- Hardening : réduction des permissions, revue d’API exposées
- Gestion des updates et cycle de maintenance
- Check-list prod : sécurité, perf, UX, conformité
- Atelier pratique : Audit final (surface API + perf) et plan de mise en production.

Sociétés concernées

Cette formation s’adresse à la fois aux particuliers ainsi qu’aux entreprises, petites ou grandes, souhaitant former ses équipes à une nouvelle technologie informatique avancée ou bien à acquérir des connaissances métiers spécifiques ou des méthodes modernes.

Positionnement à l'entrée en formation

Le positionnement à l'entrée en formation respecte les critères qualité Qualiopi. Dès son inscription définitive, l'apprenant reçoit un questionnaire d’auto-évaluation nous permettant d’apprécier son niveau estimé sur différents types de technologies, ses attentes et objectifs personnels quant à la formation à venir, dans les limites imposées par le format sélectionné. Ce questionnaire nous permet également d’anticiper certaines difficultés de connexion ou de sécurité interne en entreprise (intraentreprise ou classe virtuelle) qui pourraient être problématiques pour le suivi et le bon déroulement de la session de formation.

Méthodes pédagogiques

Stage Pratique : 60% Pratique, 40% Théorie. Support de la formation distribué au format numérique à tous les participants.

Organisation

Le cours alterne les apports théoriques du formateur soutenus par des exemples et des séances de réflexions, et de travail en groupe.

Validation

À la fin de la session, un questionnaire à choix multiples permet de vérifier l’acquisition correcte des compétences.

Sanction

Une attestation sera remise à chaque stagiaire qui aura suivi la totalité de la formation.