

Formation TypeScript : Initiation

| | |
|--|--|
| ■ Durée : | 3 jours (21 heures) |
| ■ Tarifs inter-entreprise : | 1 875,00 CHF HT (standard) 1 500,00 CHF HT (remisé) |
| ■ Public : | Développeurs front-end ou full-stack souhaitant sécuriser et structurer leurs développements JavaScript avec TypeScript - Chefs de projet techniques amenés à cadrer ou relire du code TypeScript - Développeurs orientés objet voulant transposer leurs réflexes POO vers l'écosystème TypeScript |
| ■ Pré-requis : | Bases de programmation orientée objet (réflexes type classes/héritage) - Connaissances JavaScript ou expérience de développement applicatif - Aisance avec un IDE et la ligne de commande |
| ■ Objectifs : | Prendre en main l'environnement TypeScript et les outils de développement - Manipuler fonctions et tableaux de manière efficace - Maîtriser le typage et produire un code plus robuste et maintenable - Utiliser classes et héritage pour structurer une approche orientée objet - Organiser et modulariser le code pour un usage côté serveur (assemblage / build) |
| ■ Modalités pédagogiques, techniques et d'encadrement : | <ul style="list-style-type: none">• Formation synchrone en présentiel et distanciel.• Méthodologie basée sur l'Active Learning : 75 % de pratique minimum.• Un PC par participant en présentiel, possibilité de mettre à disposition en bureau à distance un PC et l'environnement adéquat.• Un formateur expert. |
| ■ Modalités d'évaluation : | <ul style="list-style-type: none">• Définition des besoins et attentes des apprenants en amont de la formation.• Auto-positionnement à l'entrée et la sortie de la formation.• Suivi continu par les formateurs durant les ateliers pratiques.• Évaluation à chaud de l'adéquation au besoin professionnel des apprenants le dernier jour de formation. |

| | |
|---|---|
| ■ Sanction : | Attestation de fin de formation mentionnant le résultat des acquis |
| ■ Référence : | SIT102884-F |
| ■ Note de satisfaction des participants: | Pas de données disponibles |
| ■ Contacts : | commercial@dawan.fr - 09 72 37 73 73 |
| ■ Modalités d'accès : | Possibilité de faire un devis en ligne (www.dawan.fr , moncompteformation.gouv.fr , maformation.fr , etc.) ou en appelant au standard. |
| ■ Délais d'accès : | Variable selon le type de financement. |
| ■ Accessibilité : | Si vous êtes en situation de handicap, nous sommes en mesure de vous accueillir, n'hésitez pas à nous contacter à referenthandicap@dawan.fr , nous étudierons ensemble vos besoins |

Mettre en place l'environnement TypeScript et maîtriser les outils

Comprendre la place de TypeScript comme surcouche de JavaScript et ses bénéfices en développement Front-End

Installer et configurer l'environnement (Node.js, TypeScript, IDE) et valider une chaîne de build simple

Comprendre le rôle de tsconfig et des options clés (strict, target, module, lib, sourceMap)

Compiler, exécuter et déboguer un projet TypeScript en local

Lire et interpréter les erreurs du compilateur pour corriger rapidement et fiabiliser le code

Atelier fil rouge : initialiser un mini-projet TypeScript, configurer tsconfig, compiler et exécuter un premier module typé

Créer et manipuler les fonctions et les tableaux efficacement

Déclarer des types de base et composer des types utiles (union, intersection, littéraux, tuples)

Typier des fonctions : paramètres, retours, surcharges, fonctions fléchées et callbacks

Structurer des données avec des tableaux typés et appliquer des transformations sûres (map, filter, reduce)

Définir des interfaces et des types pour modéliser des objets métier simples

Éviter les pièges courants (any, unknown, null/undefined) et sécuriser les accès

Atelier fil rouge : implémenter des fonctions de traitement de données

typées, manipuler des collections et produire des résultats vérifiés par le compilateur

Structurer le code avec la POO, modulariser et appliquer les bonnes pratiques

Définir des classes, propriétés, méthodes, constructeurs et modificateurs d'accès

Mettre en œuvre l'héritage, le polymorphisme et la composition selon le besoin

Utiliser des génériques pour rendre le code réutilisable et sûr (collections, services, wrappers)

Modulariser et organiser le code pour un usage côté serveur (imports/exports, découpage, conventions)

Appliquer des pratiques de productivité : typage progressif, gestion d'erreurs, lisibilité, refactoring guidé

Préparer un code maintenable : structure de projet, noms, contrats de types, documentation minimale

Atelier fil rouge : construire un mini-domaine orienté objet (modèles + services), modulariser le code et préparer un build/exécution côté serveur