Formation JavaScript - ECMAScript 6 à 15 - Les nouveautés

Code : DEV-JS-300 Durée : 2 jours (14 heures)

Public visé

Développeurs ayant à réaliser des applications Web responsives.

{{< formation-prerequis >}}

Objectifs pédagogiques

À l'issue de cette formation, vous serez capable de :

- Reconnaître et utiliser les apports de la norme ES2015 (qui correspond à ECMAScript 6)
- Expliquer ES2016 (ECMAScript 7), ES2017 et ES2018 (ECMAScript 8 et 9), ES2019 et ES2020 (ECMAScript 10 et 11), ES2021 et ES2022 (ECMAScript 12 et 13), ES2023 (ECMAScript 14) et ES2024 (ECMAScript 15)
- Exploiter ces particularités dans les nouveaux projets

Programme

Jour 1

Introduction

- Rappels sur les aspects avancés de JavaScript
- Synthèse des apports de ES2015 et ES2016
- Compatibilité actuelle des browsers
- Tour d'horizon des outils de développement et d'intégration
- Compilateurs disponibles

Apports de ES2015 (ECMAScript 6)

- Mot-clé "let"
- Assignation des variables
- Constantes
- Modification des API
- Nouvelle syntaxe des "arrow functions"
- Assignations déstructurées
- Formatage des chaînes de caractères
- Object API, les nouvelles méthodes

Programmation objet en ES2015

- Classe et héritage
- Méthodes statiques
- Création de proxy
- Nouveaux types (Set, Map)
- Nouveaux objets héritables

Modularisation en ES2015

- Modularisation avec AMD et CommonJS
- Modularisation avec ES2015
- Différences entre les trois approches
- Façons pour l'utiliser
- Gestion des dépendances
- "Dynamique loading"

Jour 2

Itérateurs et générateurs

- Création d'un itérateur
- Toutes les nouvelles boucles "For"
- Création d'un générateur
- Exploitation d'un générateur

Asynchronisme avec JavaScript

- Présentation des "promises"
- Création et utilisation des "promises"

Déployer une application JavaScript à partir de ES2015

- JavaScript et TypeScript
- Transpileurs
- Package managers
- Traceur
- Nécessité de packager son code
- Gestion des packages avec npm
- Outils de Lint et de test

Apports de ES2016 (ECMAScript 7)

- La fonction Array.prototype.includes()
- L'opérateur Exponentiation

Apports de ES2017 (ECMAScript 8)

- Async functions
- Shared memory et les atomics

Apports de ES2018 (ECMAScript 9)

- Les itérations asynchrones
- Les propriétés REST / Spread

- Nouvelles expressions régulières
- La fonctionnalité Promise.prototype.finally()

Apports de ES2019 (ECMAScript 10)

- Nouvelles fonctions sur le type Array

Apports de ES2020 (ECMAScript 11)

- Le type BigInt
- Modifications dans les opérateurs

Apports de ES2021 (ECMAScript 12)

- Séparateur numérique
- String replaceAll()
- Opérateur logique pour l'assignation
- Promise.any()
- Accessibilité "private" pour les méthodes
- WeakReference pour le "Garbage Collector"

Apports de ES2022 (ECMAScript 13)

- Accessibilité "private" pour les attributs
- Bloc d'initialisation statique
- Ajout d'indice pour les expressions régulières
- Nouvelles fonctionnalités des promesses
- Nouvelles fonctionnalités des tableaux
- Object.hasOwn()
- La cause des erreurs

Apports de ES2023 (ECMAScript 14)

- Nouvelles fonctionnalités de recherche des tableaux
- L'opérateur Pipeline
- Les Records et les Tuples
- WeakMap pour le "Garbage Collector"
- Nouvelles fonctionnalités des tableaux
- Générateur et itérateur asynchrone

Apports de ES2024 (ECMAScript 15)

- Top Level Await
- Opérateur Pipeline
- Les Records et les Tuples
- L'API Temporal
- L'API Realms
- Les décorateurs

Modalités d'évaluation des acquis

En cours de formation, par des études de cas ou des travaux pratiques. En fin de formation, par un questionnaire d'auto-évaluation.

{{< formation-require >}}

{{< formation-seealso >}}