Date de publication : 20/02/2024

Formation : WebAssembly, booster les performances de ses applications web

Mettre du binaire dans le moteur de ses navigateurs web

Formation pratique - 3j - 21h00 - Réf. WAY

Prix : 2150 CHF H.T.

5 / 5

WebAssembly (WASM), standard W3C officiel depuis 2019, permet l'écriture d'applications ultra-rapides et ultra-légères sur le web. Ces applications peuvent être déjà écrites en toutes sortes de langages sources existants : C/C++, Rust, Go, Java, etc. Elles n'ont plus qu'à être portées pour être accessibles dans un navigateur ou un container sécurisé. Cette formation pratique donne les clés pour développer du code WASM et compiler des programmes existants en WebAssembly.

Objectifs pédagogiques

À l'issue de la formation, le participant sera en mesure de :

- Comprendre l'architecture et l'environnement du standard W3C WebAssembly
- Maîtriser le jeu d'instructions binaires du langage WASM et sa représentation textuelle, le format WAT
- Utiliser l'API JavaScript pour interagir avec des modules WASM
- Savoir mettre en œuvre une compilation C/C++ avec la suite Emscripten
- Développer avec le langage AssemblyScript
- Porter un programme ou une librairie C/C++ en WASM

Public concerné

Développeurs, développeurs web, intégrateurs, architectes techniques, responsables de solutions techniques.

Prérequis

Connaître les bases de HTML, de langages tels que JavaScript, C, et de langages de commandes tels que shell, Bash ou CMD (DOS).

Vérifiez que vous avez les prérequis nécessaires pour profiter pleinement de cette formation en faisant ce test.

PARTICIPANTS

Développeurs, développeurs web, intégrateurs, architectes techniques, responsables de solutions techniques.

PRÉREQUIS

Connaître les bases de HTML, de langages tels que JavaScript, C, et de langages de commandes tels que shell, Bash ou CMD (DOS).

COMPÉTENCES DU FORMATEUR

Les experts qui animent la formation sont des spécialistes des matières abordées. Ils ont été validés par nos équipes pédagogiques tant sur le plan des connaissances métiers que sur celui de la pédagogie, et ce pour chaque cours qu'ils enseignent. Ils ont au minimum cinq à dix années d'expérience dans leur domaine et occupent ou ont occupé des postes à responsabilité en entreprise.

MODALITÉS D'ÉVALUATION

Le formateur évalue la progression pédagogique du participant tout au long de la formation au moyen de QCM, mises en situation, travaux pratiques...

Le participant complète également un test de positionnement en amont et en aval pour valider les compétences acquises.

Modalités d'évaluation

Le formateur évalue la progression pédagogique du participant tout au long de la formation au moyen de QCM, mises en situation, travaux pratiques…

Le participant complète également un test de positionnement en amont et en aval pour valider les compétences acquises.

Programme de la formation

1 Introduction à WASM

- À quelles problématiques répond WebAssembly ?
- Historique de WASM.
- Architecture.
- Portabilité, sécurité, performance.
- Organisation de la spécification.
- Documentation.
- WASI, Bytecode Alliance.
- Structure d'un module.

Travaux pratiques

Écriture de modules simples en WASM à l'aide de WAT. Compilation et exécution avec wat2wasm et node.

2 Le langage textuel WAT

- Description d'un environnement de développement en WAT.
- Extensions Visual Studio Code.
- Les différentes déclarations d'un module.
- Les commentaires, les S-expressions.
- Les fonctions et la pile d'instruction.
- Importer/exporter une fonction ou autre artefact.
- Les objets globaux.
- La mémoire linéaire.
- Les tables de pointeurs.
- Les différentes instructions : boucle, conditions, opérations, trap.
- Fonction de démarrage "start".
- L'interface JavaScript pour utiliser un module WASM.

Travaux pratiques

Écriture et compilation d'un module en WAT offrant quelques fonctions mathématiques de base (factorielle, Fibonacci, etc.). Exécution du fichier dans Node et dans un navigateur.

3 Runtimes WASM

- Prérequis d'un runtime.
- Liste des runtimes.
- Description de WASI.
- Installation de runtime.
- Exécution de programmes WASM avec des runtimes.

Travaux pratiques

Compilation en WASM d'un programme simple écrit en Rust et exécution sur plusieurs runtimes (wasm3, wasmtime, etc.)

MOYENS PÉDAGOGIQUES ET TECHNIQUES

• Les moyens pédagogiques et les méthodes d'enseignement utilisés sont principalement : aides audiovisuelles, documentation et support de cours, exercices pratiques d'application et corrigés des exercices pour les formations pratiques, études de cas ou présentation de cas réels pour les séminaires de formation.
• À l'issue de chaque formation ou séminaire, ORSYS fournit aux participants un questionnaire d'évaluation du cours qui est ensuite analysé par nos équipes pédagogiques.
• Une feuille d'émargement par demi-journée de présence est fournie en fin de formation ainsi qu'une attestation de fin de formation si le participant a bien assisté à la totalité de la session.

MODALITÉS ET DÉLAIS D'ACCÈS

L'inscription doit être finalisée 24 heures avant le début de la formation.

ACCESSIBILITÉ AUX PERSONNES HANDICAPÉES

Pour toute question ou besoin relatif à l'accessibilité, vous pouvez joindre notre équipe PSH par e-mail à l'adresse psh-accueil@orsys.fr.

4 AssemblyScript

- Installation du module Node AssemblyScript.
- Initialisation d'un projet avec asinit.
- Garbage collector et mémoire.
- Programmation avec des objets.
- Intégration d'une librairie WASM fabriquée en AssemblyScript.

Travaux pratiques

Écriture d'un module WASM en AssemblyScript calculant les couleurs des points d'une fractale de Mandelbrot et intégration de cette librairie dans un front end visualisant la fractale.

5 L'outil Emscripten

- Les langages pouvant être portés en WASM.
- Présentation générale de Emscripten.
- Historique.
- Installation officielle.
- Installation sous Debian/Ubuntu avec apt.
- Le compilateur emcc.
- Le fichier JavaScript d'enveloppe.
- Les options de compilation.
- Les stratégies d'appels depuis JavaScript (ccall, cwrap, etc.).

Travaux pratiques

Écrire un programme simple en C, le compiler en WASM et l'utiliser avec Node et dans un navigateur.

6 Portage de librairie

- Compiler et configurer avec Emscripten et Autoconf.
- Compiler et configurer avec Emmake et Emconfigure.
- Interaction avec les makefiles.
- Option de compilations MODULARIZE, EXPORTED_FUNCTIONS, EXPORTED_RUNTIME_METHODS.
- Le système de fichier virtuel.
- Les variables d'environnement.

Dates et lieux

CLASSE À DISTANCE
2026 : 24 juin, 16 nov.