# Article information:

import CMake; C++20 Modules
<https://www.kitware.com/import-cmake-c20-modules/>

# Article summary:

1. Le support des modules de la norme de programmation en langage de programmation (C++) 20 est en cours d'implémentation dans l'outil de construction logicielle, CMake.

2. Les modules sont une alternative au préprocesseur et à l'inclusion (#include) dans les fichiers source en langage de programmation (C++).

3. Pour que les fichiers source soient compilés correctement avec les modules, il est nécessaire que leur ordre soit déterminé avant la compilation, ce qui nécessite un analyseur syntaxique pour extraire les informations "fournies" et "requises" des fichiers source pendant la construction.

# Article rating:

Appears moderately imbalanced: The article provides some useful information, but is missing several important points or pieces of evidence that would be required to present the discussed topics in a balanced and reliable way. You are encouraged to seek a more balanced perspective on the presented issues by exploring the provided research topics and looking at different information sources.

# Article analysis:

L'article traite de l'implémentation des modules C++20 dans CMake et explique les difficultés que cela pose pour les systèmes de construction. L'auteur commence par une brève introduction aux modules C++20, qui remplacent le préprocesseur et #include. Il explique ensuite comment les modules sont difficiles à gérer pour les systèmes de construction car ils nécessitent un ordre de compilation spécifique et doivent être analysés avant d'être compilés.

L'article présente également l'historique de la prise en charge des modules Fortran dans CMake depuis 2005, ainsi que la façon dont le projet Trilinos a financé l'ajout du support des modules Fortran au ninja build tool en 2015. Enfin, l'article décrit comment le Kitware CMake team a présenté un document à SG15 pour définir un format standard pour que les fournisseurs de compilateurs décrivent les informations de dépendance des modules C++20.

Dans l'ensemble, l'article est bien écrit et fournit une explication claire des défis liés à la prise en charge des modules C++20 dans CMake. Toutefois, il y a quelques biais potentiels dans l'article. Par exemple, il ne mentionne pas les alternatives possibles à CMake pour la gestion des modules C++20 ou ne discute pas suffisamment des inconvénients potentiels liés à l'utilisation de cette technologie. De plus, certaines affirmations ne sont pas étayées par des preuves solides ou ne présentent qu'un seul point de vue sur la question.

En fin de compte, cet article est utile pour ceux qui cherchent à comprendre comment implémenter les modules C++20 dans leur système de construction, mais il est important de garder à l'esprit qu'il ne présente qu'un point de vue limité sur la question.

# Topics for further research:

* Alternatives to CMake for managing C++20 modules
* Disadvantages of using C++20 modules
* Comparison of different build tools for C++20 modules
* Best practices for implementing C++20 modules in CMake
* Performance considerations for using C++20 modules
* Impact of C++20 modules on code organization and architecture

# Report location:

<https://www.fullpicture.app/item/5b38c12a3a8d728efc4f889c89079bbe>