

skyguide

ENSG  
Géomatique

# Compilation & langages

Présenté par Yann Caron  
skyguide

ENSG Géomatique

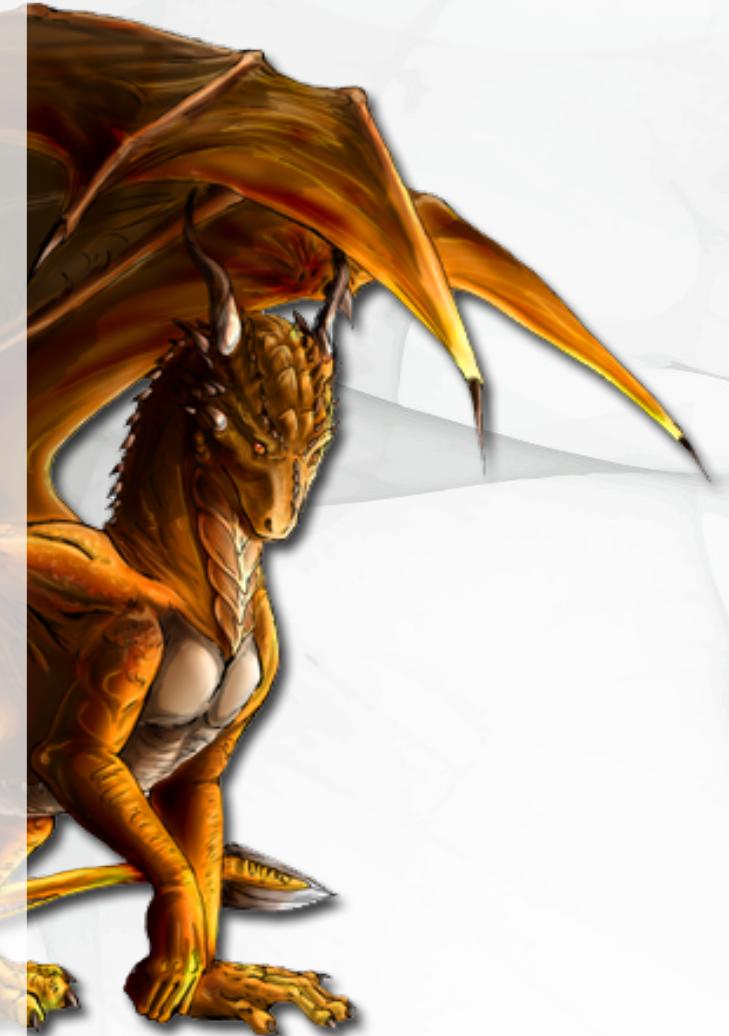
# Plan du cours

Introduction

Théorie des langages

La compilation

Principes et paradigmes



# Présentation

Yann Caron

skyguide – SCADA – Java – C# - C++

Ingénieur EI-CNAM

Projet Algoid [www.algoid.net](http://www.algoid.net) & Wikipédia

Projet JinyParser

<https://github.com/YannCaron/JinyParser>

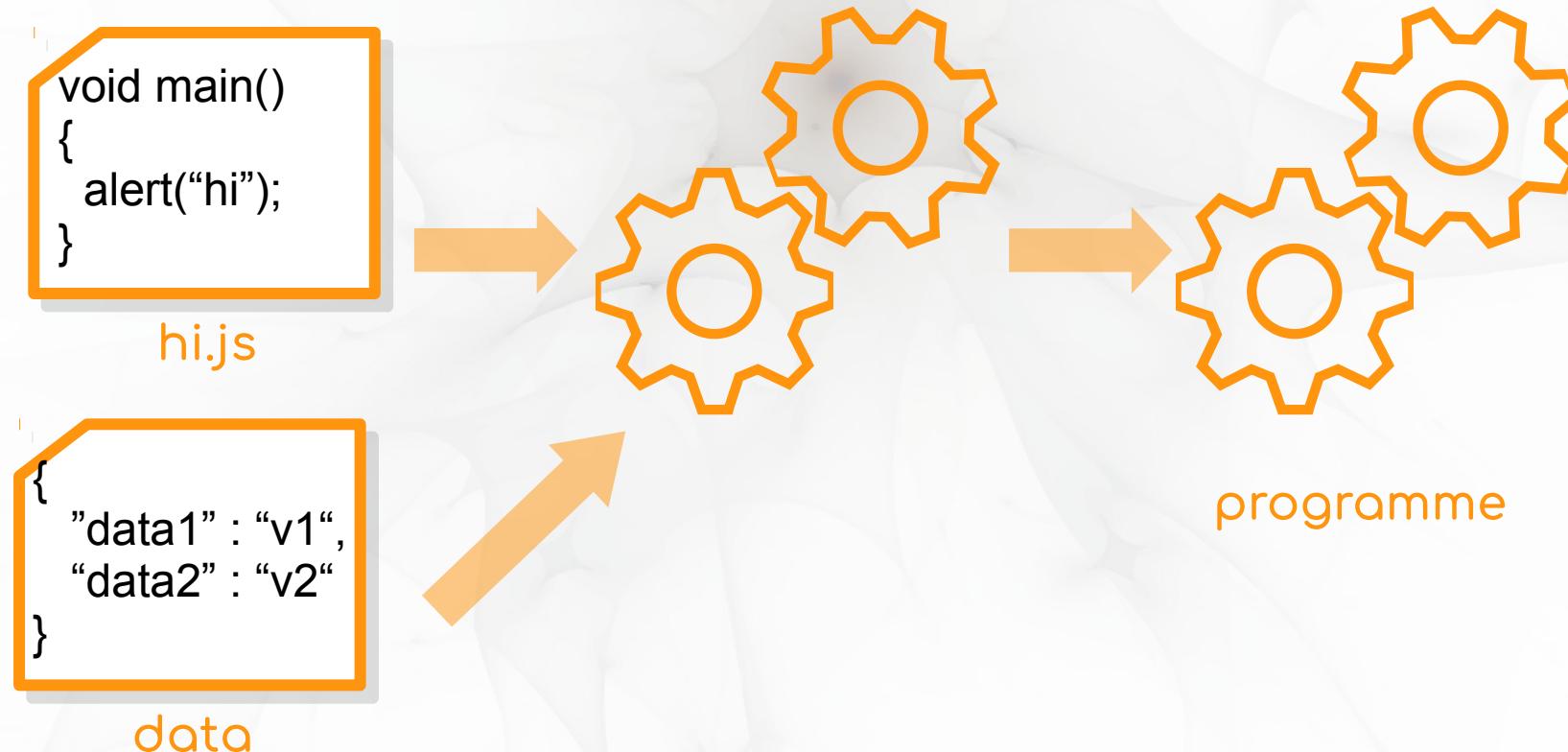
# Compilateur - définition

Programme qui transforme un code source (langage source) dans un langage cible  
Ex : binaire ou C++ vers du C



# Interpréteur - définition

Programme qui transforme un code source (langage source) en un résultat



# Philosophie

Quel est le meilleur langage ?

# Philosophie

- ✓ pas réellement de meilleurs langages que d'autres
- ✓ des langages pour des besoins différents
- ✓ script
  - ✓ pro : rapide, simple efficace
  - ✓ cons : librairies, bugs difficiles à identifier
- ✓ compilés
  - ✓ pro : langages sûr, correction à la compilation
  - ✓ cons : verbeux, complexes, rigides

# Par domaines

Domaine	Fonctionnalités	Langage
Scientifique	Virgule flottante Tableaux Parallélisme	Fortran
Business	Persistante Génération de rapports Analyse des données	SQL
Programmation système	Contrôle des ressources Temps réel	C / C++

# Compilation ? Pourquoi ?

- ✓ Challenge intellectuel
  - ✓ Paradigmes
  - ✓ Algorithmes
  - ✓ Compréhension en profondeur des langages
- ✓ Économiques ex : Oracle vs Microsoft
- ✓ Besoin de configuration avancé ! (exemple projet Stars)
- ✓ Domain Specific Languages

# Économiques

- ✓ Atlassian
- ✓ Sonar-source
- ✓ Oracle vs Microsoft

# Nouveau mais !

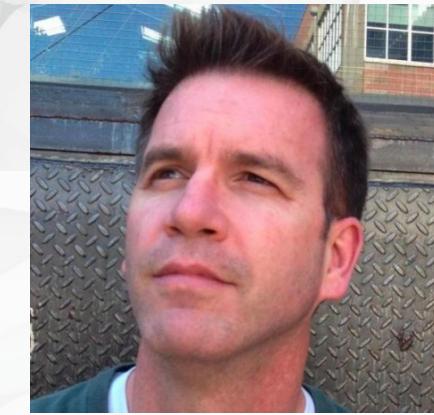
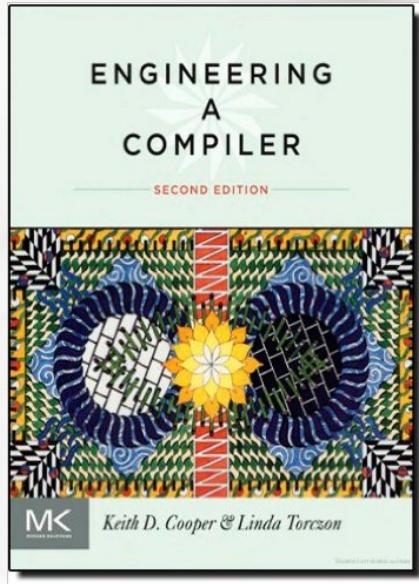
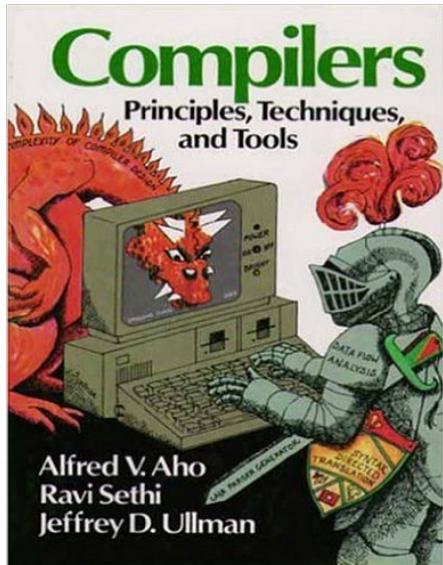
- ✓ Les nouveaux langages créés tendent à ressembler aux anciens
- ✓ Exemple : C++ - Java - C#
- ✓ JavaScript - Algoid
- ✓ Parce que l'apprentissage d'un nouveau langage en rapporte à ce qu'il apporte de nouveau, coûte cher

# Références

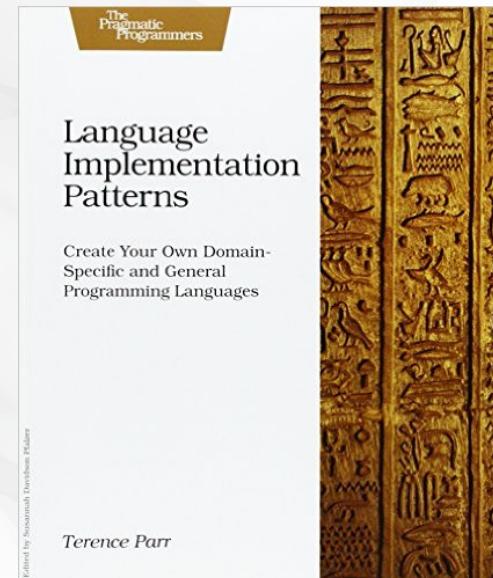


# Bibliographie

- ✓ Dragon book
- ✓ Engineering a compiler
- ✓ Terrens Parr



Terrens Parr -1964



# MOOCs



Jeff Ullman - 1942



Alex Aiken

- ✓ Stanford Online - Automata - Jeffrey D. Ullman

<http://online.stanford.edu/course/automata-theory-self-paced>

- ✓ Stanford Online - Compilers - Alex Aiken (language COOL)

<https://lagunita.stanford.edu/courses/Engineering/Compilers/Fall2014/about>

# MOOCs



Peter Van Roy



Sedgewick - 1946

- ✓ Louvain – Paradigms of Computer Programming (Langage Oz)

<https://www.edx.org/course/paradigms-computer-programming-louvainx-louv1-1x-2>

- ✓ Princeton – Algorithms Part II – Robert Sedgewick

<https://www.coursera.org/learn/algorithms-part2>

# Acteurs



Alan Turing  
1912 - 1954



Donald Knuth  
1938

- ✓ Alan Turing
  - ✓ Machine formelle de Turing
- ✓ Donald Knuth
  - ✓ Tex → LaTex (Domain specific language)
  - ✓ Parser LR
  - ✓ Grammaire attribués (attributs de l'AST)

# Outils



# Outils - parsers

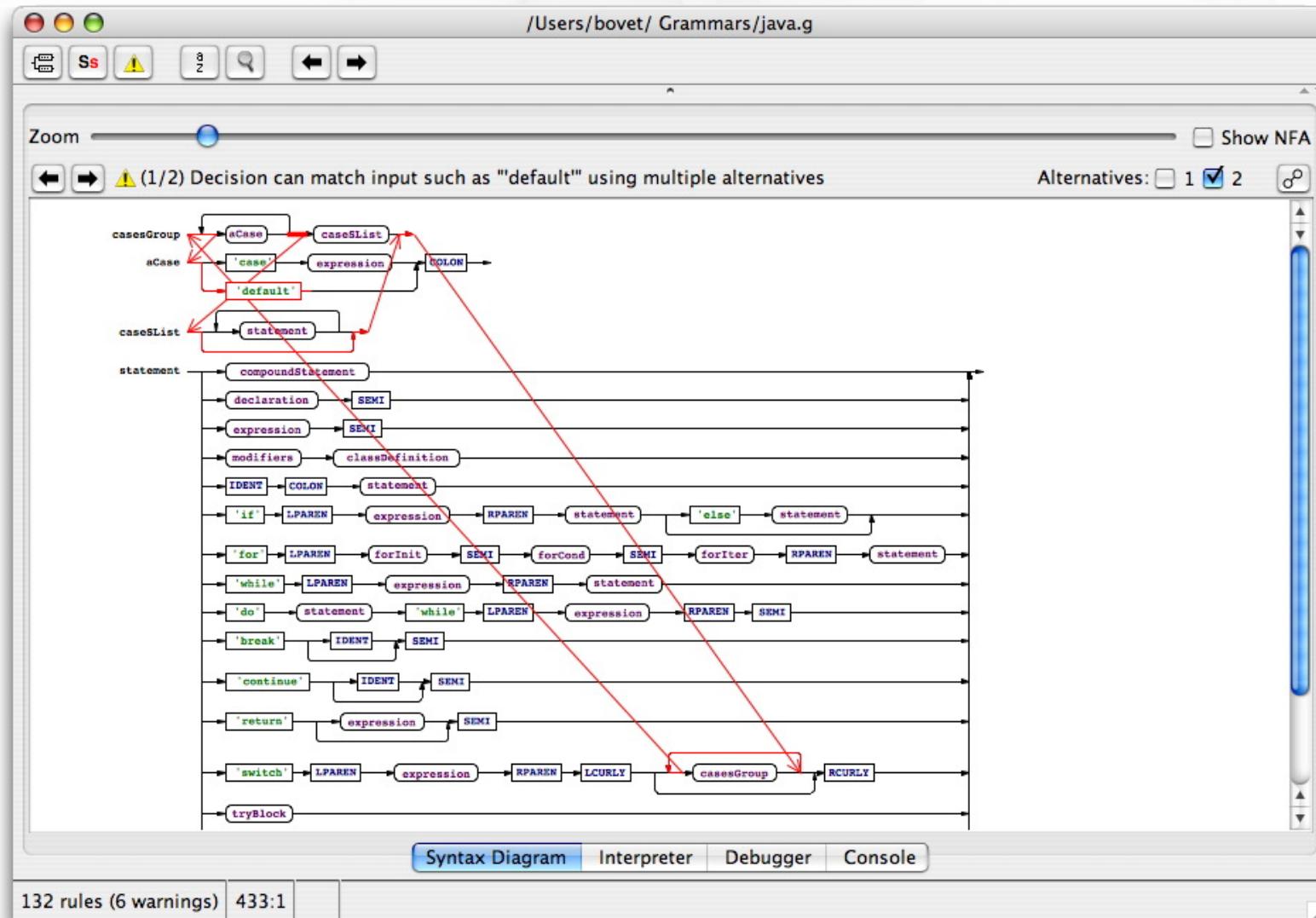
- ✓ Pour une langage : GCC, JavaCC, Roslyn
- ✓ Lex / Yacc - Flex / Bison - Génération de code
- ✓ Antlr - BNF (grammaire externe)
- ✓ Irony.net - Surcharge d'opérateurs  
`ParExpr.Rule = "(" + Expr + ")";`
- ✓ JParsec - Déclaratif fonctionnel
- ✓ JASI - JinyParser

# Outils - compilateurs

- ✓ ASM – Générateur de bytecode JVM et Dalvik – ex : Golo
- ✓ CECIL, ILasm – Générateur d'Intermediate Language .net
- ✓ LLVM – VM C / C++
- ✓ Neko VM (bytecode de haut niveau)



# Outils – AntlrWorks



# Outils – Railroad

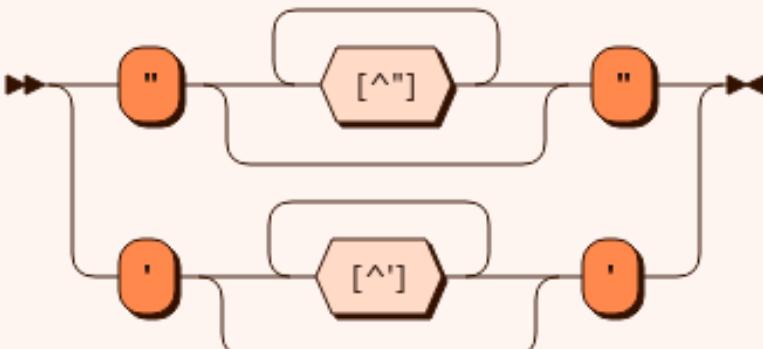
Written by Gunther Rademacher  
grd@gmx.de

**Railroad Diagram Generator**  
v1.47.1494 Apr 15, 2017

**SAXON**  
**XQUER**

**Welcome** **Get Grammar** **Edit Grammar** **View Diagram** **Options**

**StringLiteral:**



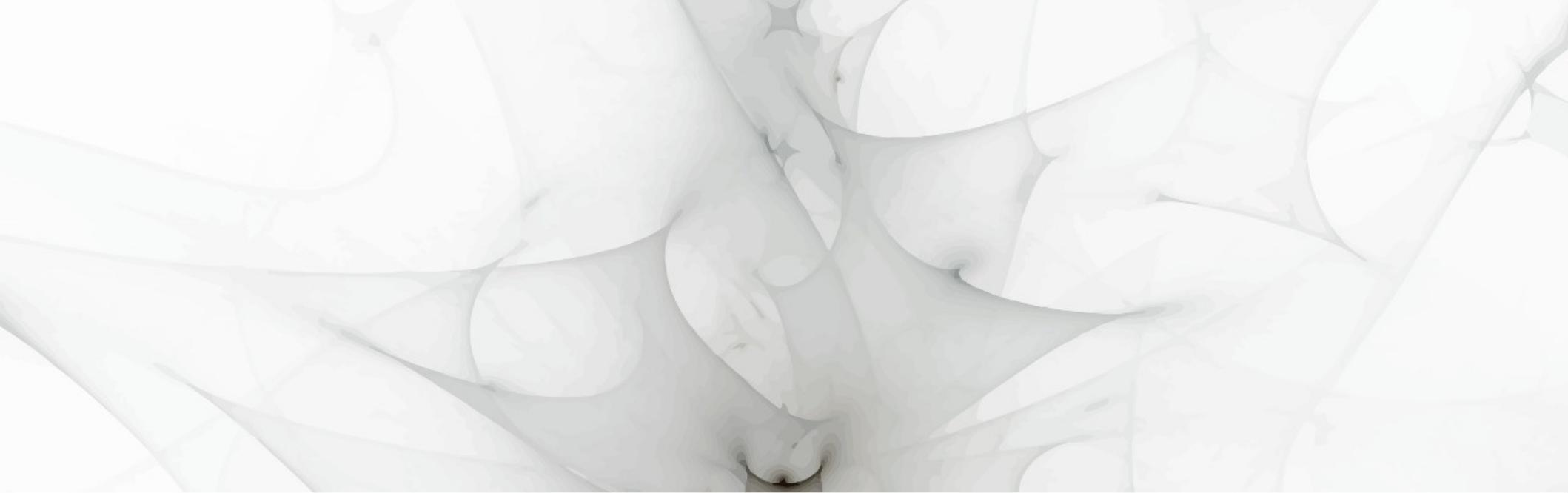
**Download diagram**

**SVG** single XHTML page with embedded SVG graphics

**PNG** zip file containing HTML and linked PNG Images

**Download**

```
StringLiteral
  ::= """ [^"]* """
     | ''' [^']* '''
    /* ws: explicit */
```



Merci de votre attention

