



Informatique 1

2. Structure d'un programme

Objectifs

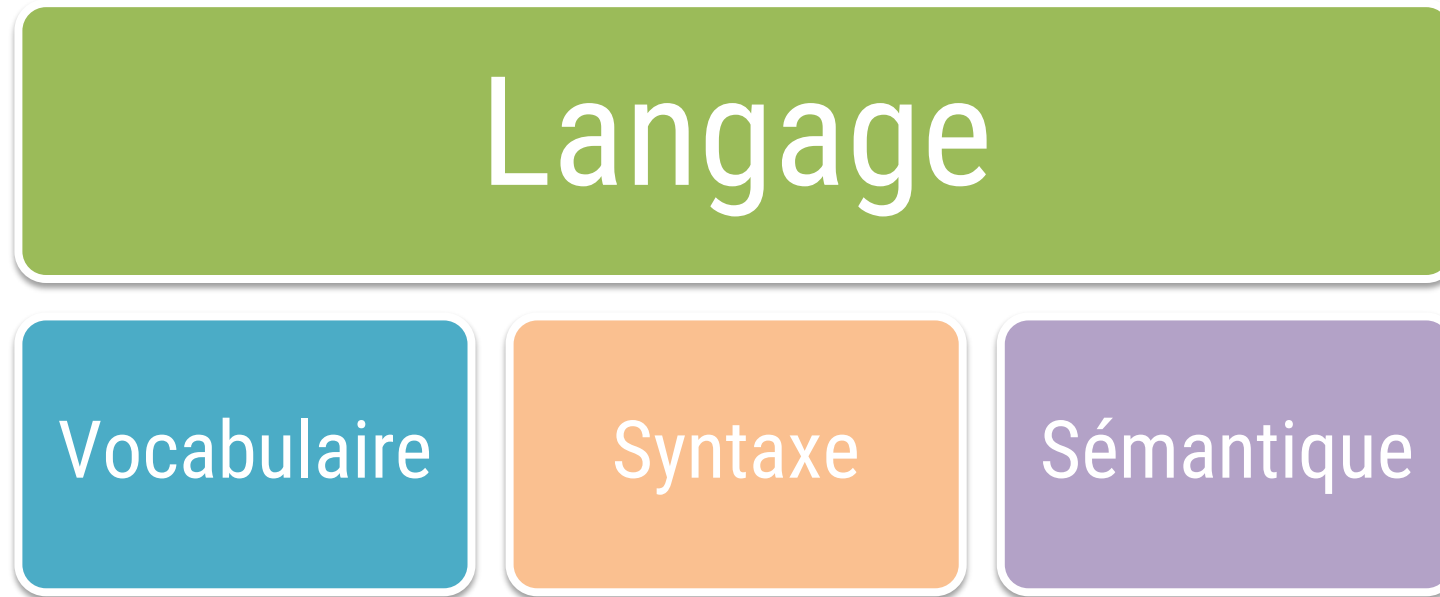
Comprendre les règles d'écriture du Java

- ▶ Vocabulaire
- ▶ Syntaxe et sémantique
- ▶ Analyse d'un programme
- ▶ Standards d'écriture

Exemple introductoire

```
public class MyProgram
{
    //Program entry point
    public static void main(String[] args)
    {
        int var1=3;
        int var2=4;
        System.out.println("Sum is");
        System.out.println(var1 + var2);
    }
}
```

Langage de programmation

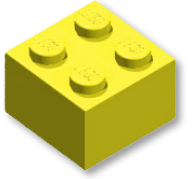


- Les phrases de ces langages sont appelées **instructions**.

Words that make sentences

2.1 ÉLÉMENTS DE VOCABULAIRE

Éléments de syntaxe



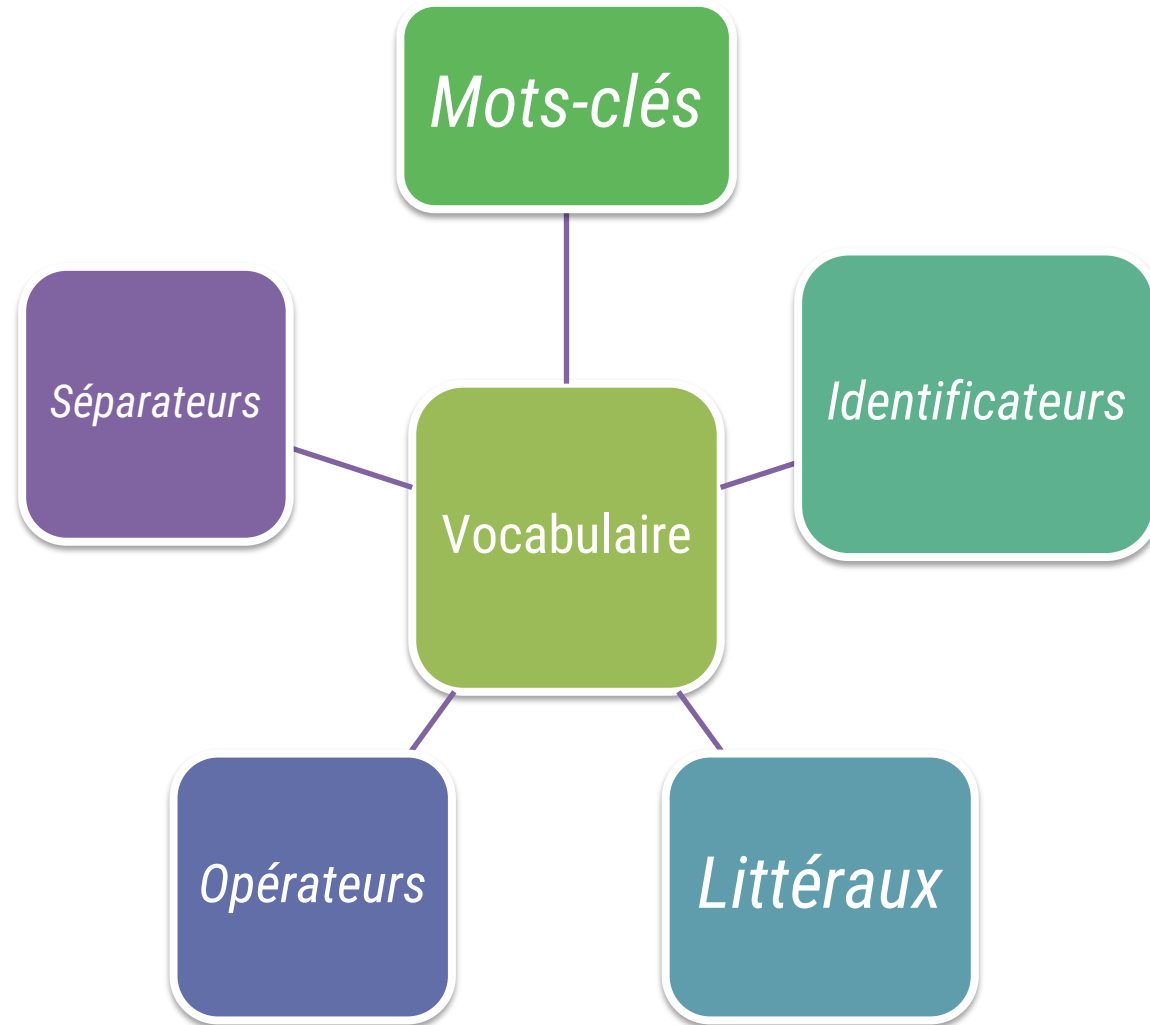
- Le vocabulaire = briques du langage
 - par ex. en français "chat", "chien".
- *Java* → quelques dizaines de mots
- Attention :

case sensitive



Main ≠ main

Éléments de syntaxe (2)



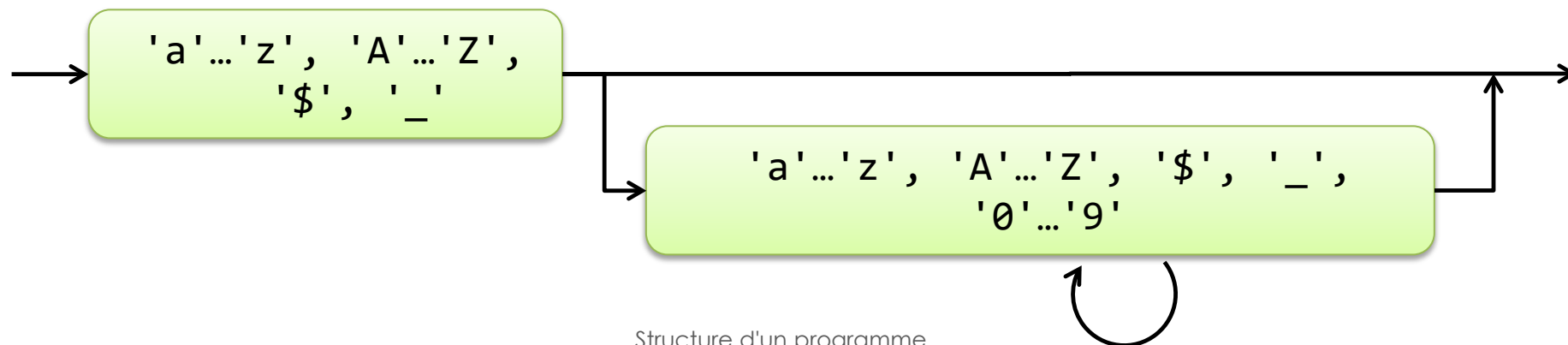
Mots clés en Java

abstract	continue	for	new	switch
assert	default	if	package	synchronized
boolean	do	goto	private	this
break	double	implements	protected	throw
byte	else	import	public	throws
case	enum	instanceof	return	transient
catch	extends	int	short	try
char	final	interface	static	void
class	finally	long	strictfp	volatile
const	float	native	super	while

- Pas besoin "d'apprendre" par cœur cette liste !
- Nous verrons l'usage de ces mots durant le semestre.

Identificateurs

- Servent à nommer des objets.
- Règles :
 - ▶ Doivent être \neq mots clés
 - ▶ Commencent obligatoirement par une lettre.
 - ▶ Continuent optionnellement par chiffres ou lettres
 - Lettre = toutes les lettres de l'alphabet **anglais** ainsi que '_' et '\$'



Identificateurs (2)

Good



Bad



Littéraux

Les nombres entiers

- 33 9 -1

Les nombres à virgule flottante

- .3 0.3 3.14 -5.7

Les valeurs booléennes

- true false

Les caractères

- '(' 'R' 'r' '{'

Les chaînes de caractères

- "Hello world" "0.2" "r" ""

Opérateurs

- Beaucoup opérateurs en *Java*
- Leur utilisation détaillée fera l'objet du prochain cours

Calcul

+ - * /

Assignment

= += -= *= /=

Incrémentation

++ --

Logiques

|| && !

Comparaison

== < <= > >=

!=

Bit à bit

& | ^

Rotation de bits

<< >> >>>

Langage .

Séparateurs

- Séparateurs \cong ponctuation

() { } [] ; ,

- Utilisation est différente :

Exemple: Une "phrase", càd une instruction, se termine par un ';' et non par un point '.'



Rules to make meaningful sentences

2.2 SYNTAXE ET SÉMANTIQUE

Syntaxe

- Règles d'écriture
- Elle définit :
 - ▶ L'ordre acceptable des mots.
 - ▶ Les mots obligatoires.

Exemple:

"*Chevaux. courent la prairie dans*" est
syntactiquement incorrect.

Sémantique

- La sémantique donne la signification, le **sens**, des instructions
- Sémantique pas respectée = phrase incompréhensible

Exemple syntaxiquement correct:

"Le repas a maintenant tourné contre l'ascension."

Erreurs de programmation (1)

- Dans un langage de programmation, seules les instructions syntaxiquement correctes sont exécutables.
- Le **compilateur** filtre ces erreurs et peut les indiquer facilement.
- En cas d'erreur, lire les messages du compilateur.



Erreurs de programmation (2)

- Règles permettent détecter les erreurs
 - Vocabulaire
 - Syntaxe
- Par analogie, il faut que les phrases soient sans fautes d'orthographe (vocabulaire) ni de grammaire (syntaxe).
- Par contre, la sémantique **NE PEUT PAS** être contrôlée !

Commentaires

- Permettent d'annoter les programmes
- Aucun effet sur l'exécution

```
// comments  
définit le début d'une ligne de  
commentaires  
  
/* comments */  
définit un bloc de commentaires  
(peut être sur plusieurs lignes)  
  
/**  
* comments  
*/  
bloc commentaires (forme  
alternative)
```

Définition de commentaires

A short exercise

2.3 ANALYSE D'UN PROGRAMME

Analyse du programme

```
public class MyProgram
{
    //Program entry point
    public static void main(String[] args)
    {
        int var1=3;
        int var2=4;
        System.out.println("Sum is");
        System.out.println(var1 + var2);
    }
}
```

Informal rules

2.4 STANDARDS D'ÉCRITURE

Lisibilité et compréhension

- Un programme peut être correct
 - ...mais cela ne veut pas dire qu'il est compréhensible.
- Comparez:
 1. `circleArea = 3.1415*radius*radius;`
 2. `a = 3.1415*b*b;`
 3. `кругПлощадь = 3.1415*радиус*радиус;`

Lisibilité : quelques règles


- *Identificateurs*
 - ▶ En anglais (si possible)
 - ▶ Utiliser des mots entiers
 - ▶ L'élément représenté par l'identificateur doit être **compréhensible** par son nom (e.g. **circleRadius**)

Commentaires: quelques règles

- *Commentaires*

- ▶ En anglais (si possible)
- ▶ Ajouter des commentaires **suffisants** et **pertinents**.
- ▶ Il ne sert à rien de lésiner sur les commentaires !

Blocs de programme

- Un bloc \cong unité logique de programme
 - commence par '{', finit par '}'
- On **indente** (avec ) les blocs afin d'améliorer la lisibilité
- Les IDE (comme *Eclipse*) permettent d'automatiser cela.

Indentation des blocs

```
public class MyProgram
{
    //Program entry point
    public static void main(String[] args)
    {
        int var1=3;
        int var2=4;
        System.out.println("Sum is");
        System.out.println(var1 + var2);
    }
}
```

Ce que nous avons vu...

- Éléments programme en *Java*
- Quelques règles de syntaxe
- Quelques règles relatives aux standards d'écriture (indentation)
 - Bases pour plonger dans le langage, ceci dès le prochain cours !