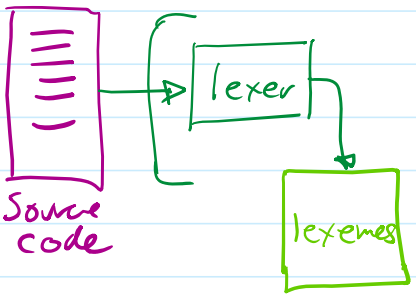
4 Syntax and Regular Expressions

Wednesday, February 4, 2026 3:00 PM



- Borrow from Linguistics & Mathematics

- Orthography : How to write words?

apple manzana pómme. 사과

A complete list of words: Dictionary

in a programming language: Keywords.

- Syntax : How to write sentences?

"has jump apple dog green"

Syntax has rules.

also in programming languages.

- Semantics : What do sentences mean?

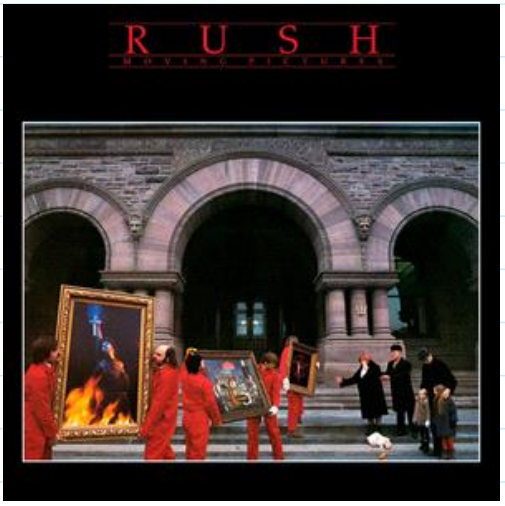
"colorless green ideas sleep furiously"

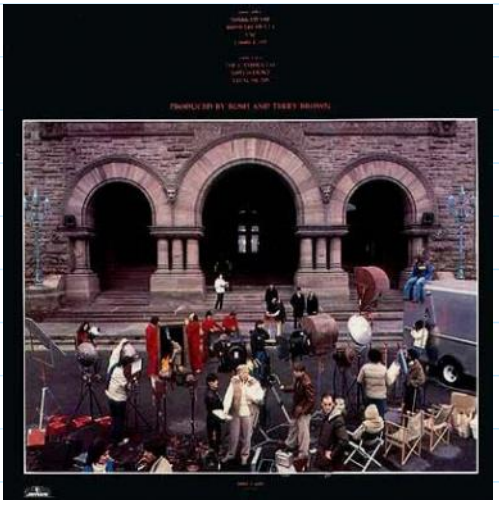
Adj Adj noun verb Adverb

"the chickens are ready to eat"

 or  ?

"Moving Pictures"





In programming languages we should avoid
- non-sensical sentences
- ambiguous sentences.

- Pragmatics .¿How meaning changes over time?
"the container has to handle an unreliable cloud"

- Programming Language Orthography

What are the words of a programming language?

Symbols.- ASCII

- Keywords.
- Operators
- Literals .- constant values
- identifiers.- named entities.
  - variable
  - constants
  - functions
  - classes.
  - modules
  - namespaces
  - templates.

How many identifiers are possible?
infinite !!

- We need rules!
We borrow from mathematics
"Mathematical Languages" S. Kleene E. Post

DEF: An alphabet is a finite set of symbols
e.g $\Sigma = \{a, b\}$ $\Sigma = \{0, 1\}$

DEF: An alphabet is a finite set of symbols

e.g $\Sigma = \{a, b\}$ $\Sigma = \{0, 1\}$

DEF A language is a set of sequences / strings / words made of symbols from an alphabet.

e.g: { a, ab, aaa, bab, baab }

- Problem #1 - the Specification Problem
  i How to precisely describe a language?

  e.g. $L = \{ a, ab, abb, abbb, \ldots \}$??

- Problem #2 - the Recognition problem
  given a word w and a language L

  is w in L? $w \in L$?

- Answer to #1: Regular Expressions
  built from Σ plus | * ( )

DEF: Regular Expressions:

- a word w is a regex
- if r1, r2 are regex then
  - r1 r2
  - r1 | r2
  - r1 *

  } are also regex

- A regular expression specifies a language

- w $\{w\}$
- r1 r2 $\{ w_1 w_2 \text{ s.t. } w_1 \in r1 \wedge w_2 \in r2 \}$
- r1 | r2 $\{ w \text{ s.t. } w \in r_1 \vee w \in r2$
- r * $\{ w \text{ s.t. } w \text{ is zero or more repetitions of a word in } r \}$

  { Kleene Star

Eg. $\Sigma = \{a, b\}$

Eg. $\Sigma = \{a, b\}$

- $(a|b)\,b = \{ab, bb\}$
  - $a = \{a\}$
  - $b = \{b\}$
  - $(a|b) = \{a, b\}$
- $(ab)^* = \{\lambda, ab, abab, ababab, \ldots\}$ (λ = empty string)

Note: an infinite set is clearly specified by finite means

- $(aa|b)^* = \{\lambda, b, aa, bb, aab, baa, bbb, aaaa, aabb, \ldots\}$
  - $aa = \{aa\}$
  - $b = \{b\}$
  - $(aa|b) = \{aa, b\}$

- Shorthands:
  - $r? \equiv (r|\lambda)$ "to r or not to r"
  - $r+ \equiv r(r^*)$ "one or more repetitions of r"

e.g $a(b?)(a+) = \{aa, aba, aaa, abaa, aaaa, \ldots\}$

- $a = \{a\}$
- $b? = \{b, \lambda\}$
- $a+ = \{a, aa, aaa, aaaa, \ldots\}$

- Regular Expression and Programming Languages

Σ = ASCII keyboard symbols

L_{C++} = The set of all c++ programs RegEx not enough.

Note: RegEx do not have the power to specify such set.

Why?

e.g $\{a^n b^n \mid n \in \mathbb{Z}^+\} = \{ab, aabb, aaabbb, aaaabbbb, \ldots\}$

consider matching brackets

{ ( ) [ ] } < > { ([ )] }

Consider matching brackets

{ ( ) [ ] } < > { ([ )] }

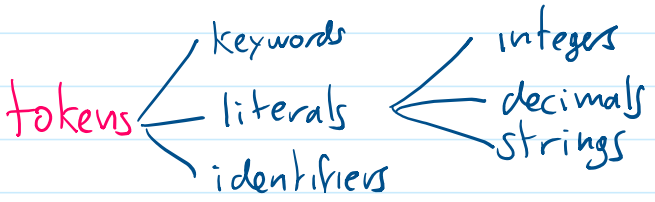
- Famous Stack overflow question:
  what RegEx for validating HTML?

 .....
<h3> ..... </h3>

closed.

- RegEx can be used to validate
  - Phone Numbers
  - Email adresses
  - URL
  - Serial Numbers
  - Credit Card Numbers.

- RegEx is good enough to specify tokens. !!
  token: a category of atomic strings
  lexeme: an instance of a token.



e.g. while foo 3.7

- Describing Tokens using RegEx
  Σ = ASCII keyboard characters.
  - Regex for keywords
    C++ (if | while | for | switch | else | .... | void)
  - Regex for operators
    (+ | * | - | / | = | ....)
  - Shorthands
    [0-9] ≡ (0|1|2|3|...|9)
    [A-Z] ≡ (A|B|C|D|...|Z)
    [a-z] ≡ (a|b|c|d)...|z)

[A-Z] ≡ (A|B|C|D|...|Z)
[a-z] ≡ (a|b|c|d|...|z)
[A-Za-z] ≡ ([A-Z]|[a-z])

- Regex for integer Literals

(+|-)?[0-9]+ = { 15 27 54 333 1257 01 000 +7 +85 -5 -27 -000 +0 }

- Regex for decimal literals

(+|-)?[0-9]+.[0-9]+ { 17.0 0.03 6.3 +0.0 -11.3 -0.0 -27.0 }

- RegEx for identifiers

BASIC [A-Z]+ A ABC APPLE

Fortran [A-Z]([A-Z]|[0-9])* A P2 K7F APPLE3

Pascal ([A-Za-z])([A-Za-z]|[0-9]|_)*
var1 point_2D init_x ThisIsPascalCase.

Python ([A-Za-z]|_)([A-Za-z]|[0-9]|_)* thisIsCamelCase
__main__ __init__
__plus__ var_1 ____ this_is_snake_case
this-is-kebab-case

- RegEx for Scientific Numbers

e.g 3.7e-5 -8.9e7 123e3 +12.7e-8 -5e+7

(+|-)?[0-9]+(.[0-9]+)?e(+|-)?[0-9]+
[1-9][0-9]*
2.5e107


WHENEVER I LEARN A NEW SKILL I CONCOCT ELABORATE FANTASY SCENARIOS WHERE IT LETS ME SAVE THE DAY.
OH NO! THE KILLER MUST HAVE FOLLOWED HER ON VACATION!
BUT TO FIND THEM WE'D HAVE TO SEARCH THROUGH 200 MB OF EMAILS LOOKING FOR SOMETHING FORMATTED LIKE AN ADDRESS!
IT'S HOPELESS!
EVERYBODY STAND BACK.
I KNOW REGULAR EXPRESSIONS.
PERL!
TAP TAP