

Six10

Version 1.0

# Frågor att ta ställning till vid skapande:

Eftersom jag aldrig har gjort någonting liknande varken i storlek eller funktionellt så har jag tänkt att jag får börja med att bena ut lite om vad jag vill/tänker göra. För att sen ha lite kött på benen och börja efterforska. De frågorna som jag spontant tänker att jag behöver fråga mig själv är:

* Vad är språkets syfte, finns det någon utpekad målgrupp?
* Vilken typ av språk vill jag göra?
* Vilket språk ska jag skriva det i?
* Ska språket vara kompilerat, interpreterat eller transpilerat eller någonting helt annat?
* Hur ska syntaxen se ut?
* Hur fungerar semantiken?
* Ska språket vara dynamiskt eller statiskt typat?
* Ska man jobba med pekare eller sköter programmet det?

Jag har inte skrivit något programmeringsspråk, tagit någon kurs om det eller ens läst en bok om det. Min resa börjar här, nu, idag – med dessa frågorna. Så då kör vi!

## Vad är språkets syfte, finns det någon utpekad målgrupp?

Språkets huvudsakliga syfte är att jag ska lära mig mer om hur ett programmeringsspråk fungerar, vilka byggstenar det är uppbyggt av och att jag ska bli en bättre mjukvaruingenjör.

Jag har absolut inte några förväntningar att någon annan kommer att använda sig av detta språket, men tanken är att jag ska skriva ett så enkelt och intuitivt språk som möjligt. Tanken är att det ska vara ett enkelt språk att använda i utbildning/skola som instegsspråk för att lära sig om programmeringens grundkoncept.

## Vilken typ av språk ska det bli?

Mitt sikte är att det ska bli ett skriptspråk med möjligheter att köras både interpreterat och som skall kunna kompileras för ökad prestanda. Det skall dessutom vara dynamiskt typat.

## Vilket språk skall det skrivas i?

Eftersom jag är mest bekväm i C++ och Python så blir det logiska valet C++, där jag kan få bättre prestanda och ett snabbare interpreterat språk.

## Hur ska syntaxen och semantiken se ut?

Syntaxen skall vara så enkel att läsa som möjligt och skall skrivas med engelska nyckelord. Klammerparantes skall användas för att definiera ett scope och indentering används för att markera vilket statement ett kodblock tillhör.

Radbryt markerar att ett statement är färdigt. Om radbryt kommer i rundparantes fortsätter statment tills slutparantes kommer. Backslash indikerar att statment fortsätter på kommande rad.

Kolon markerar ett kommande kodblock.

Exempel:

foreach number in range(0, 10):

    output("Printing number: " + number)

Detta kodstycke ger utskriften:

Printing number: 0

Printing number: 1

Printing number: 2

Printing number: 3

Printing number: 4

Printing number: 5

Printing number: 6

Printing number: 7

Printing number: 8

Printing number: 9

I kodexemplet ovan är foreach och in exempel på nyckelord, number är en variabel, 0, 10 och stängen är litterals och range och output är funktioner.

### Regler för att skriva

1. Identifiers kan endast skrivas med **a-z**, **0-9** och understreck (**\_**). Namn som **minKlass**, **variabel1**, **print\_to\_screen** är giltliga.
2. En identifier får inte börja med en siffra. Ex. **1variable** är **inte** giltig, men **variable1** är giltigt.
3. Nyckelord kan inte användas som identifierare
4. Specialtecken kan inte användas som identifierare, ex. **!”$%&** etc.
5. En variabel kan ha valfri längd
6. Alla identifierare och nyckelord är skiftlägeskänsliga, **var1** är alltså inte samma som **VAR1**
7. En brädgård (#) markerar starten för en kommentar. Kommentaren sträcker sig till kommande radbyte i källkoden.
8. Flerradskommentar kan skrivas genom att öppna med /# och stänga med #/. Exempel:

/#

Detta är en flerradskommentar

Den skrivs över flera rader

och behöver bara markeras i

början och slut.

#/

## Hur ska språket arbeta med pekare?

På samma sätt som Python, variabler med allting som inte är en fundamental datatyp hanteras som pekare, medan fundamentala datatyper – ex. int, float, string etc – placeras rakt in i variabeln.

# Skiss på lexer

1. Om [a-z], [A-Z] eller [\_], :
   1. KEYWORD
   2. DATATYPE
   3. annars IDENTIFIER
2. Om [”] eller [’] : sträng, om det finns ett stängande citattecken
3. Om startar med en siffra [0–9]:
   1. INT om det inte finns någon punkt
   2. FLOAT om det finns en punkt
   3. INVALID om fler än en punkt
4. Om [space]:
   1. TAB, Om 4 stycken följande [space]
   2. Annars SPACE
5. Om [\n]: EOL
6. Om [\r\n]: WIN\_EOL
7. Om [(], [)], [{], [}], [[], []], [<] eller [>]: PARENTHESIS
8. Om [/#] eller [#/]: MULTILINECOMMENT
9. Om [#]: COMMENT
10. Om [,]: COMMA
11. Om [.]: MEMBEROP
12. Om [==], [!=], [<], [>], [<=] eller [>=]: RELATIONALOP
13. Om [&&], [||] eller [!]: LOGICALOP
14. Om [=], [+=], [-=], [\*=], [/=] eller [%=]: ASSIGNMENTOP
15. Om [+], [-], [/], [\*], [%], [\*\*], [++] eller [--]: ARITHMETICOP