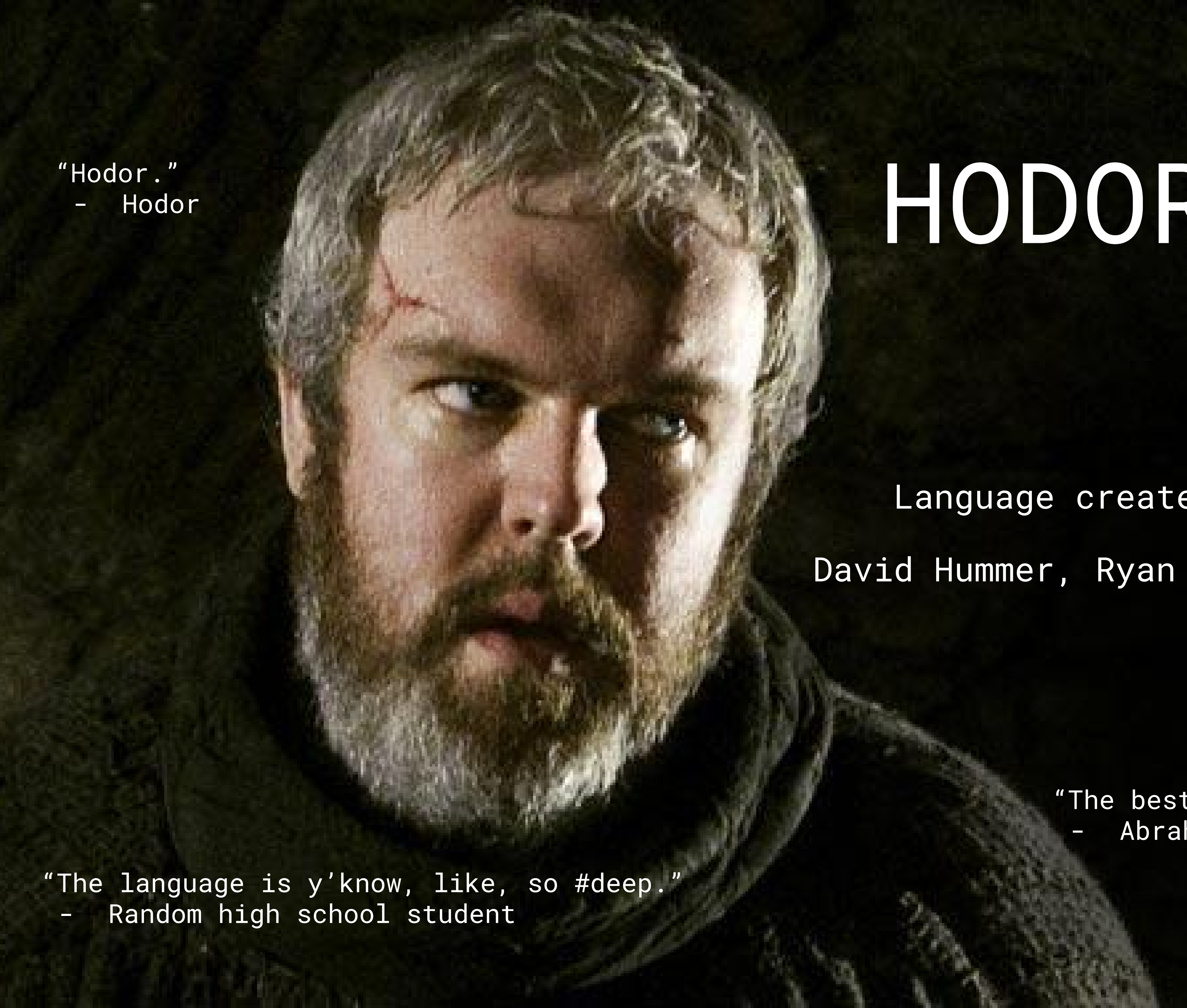


“Hodor.”
- Hodor

HODOR-lang

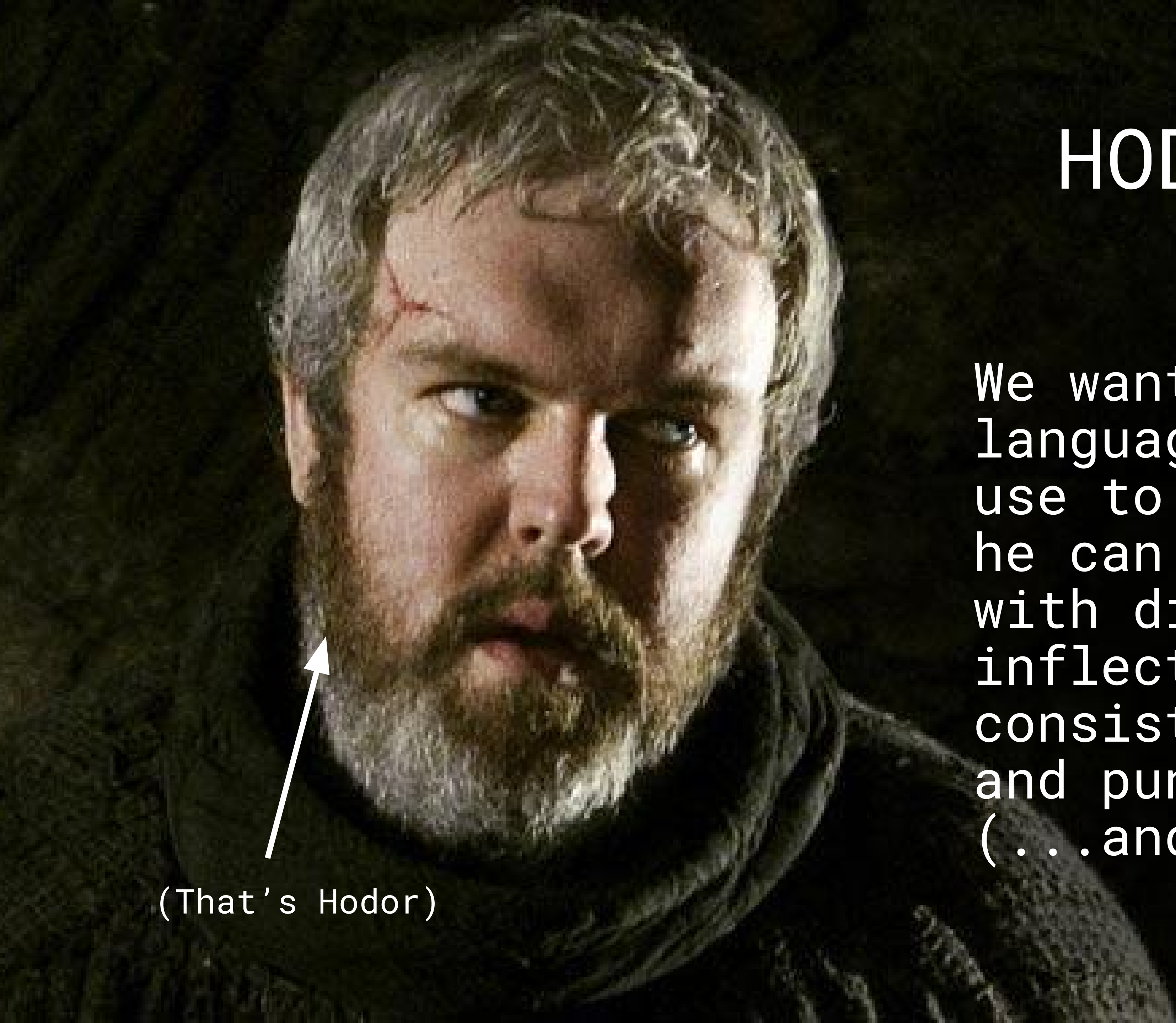
Language created and developed by:

David Hummer, Ryan Lorey and Mitch Stephan

“The best programming language I’ve ever seen!”
- Abraham Lincoln

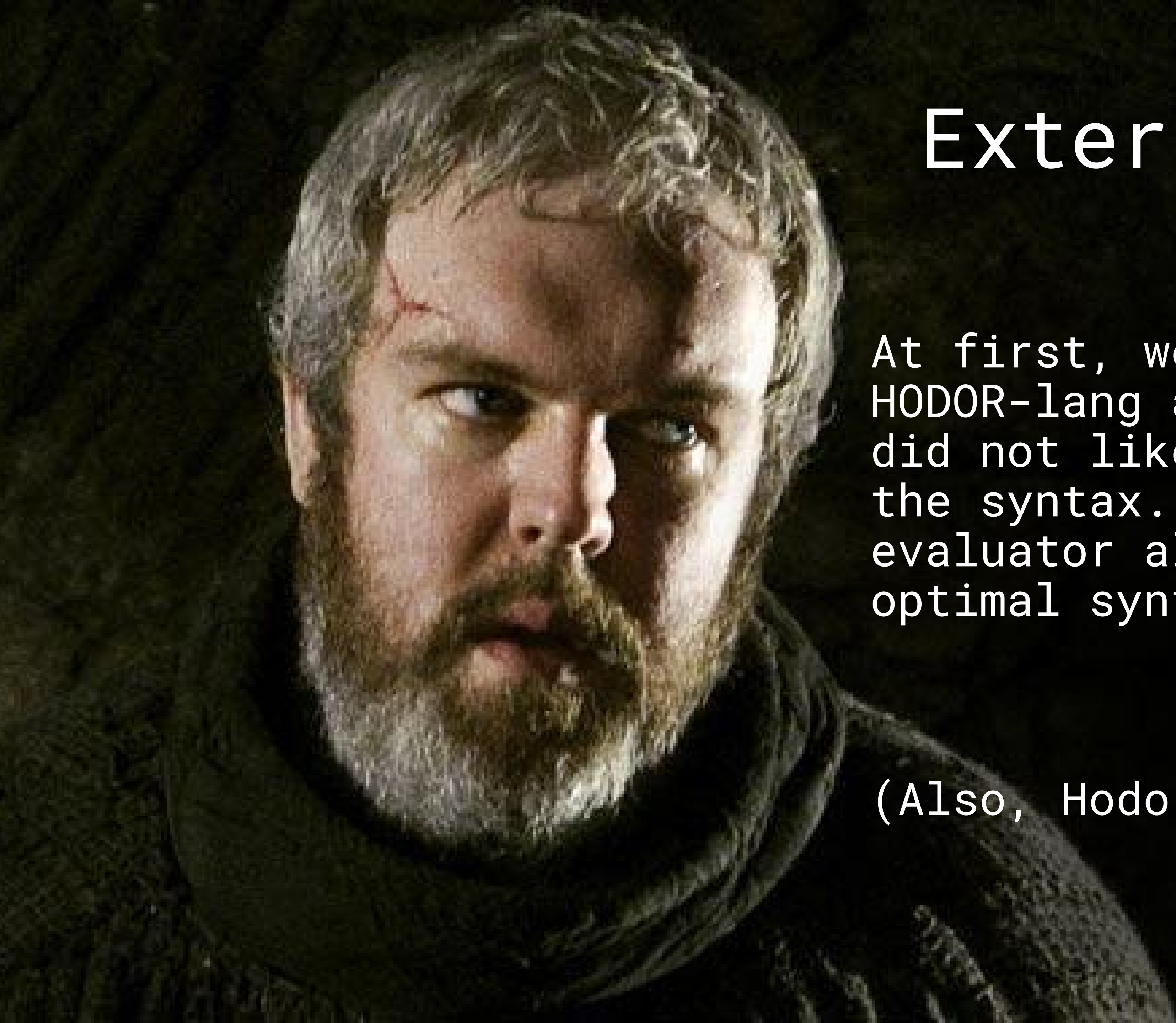
“The language is y’know, like, so #deep.”
- Random high school student

HODOR!!!!!!!



(That’s Hodor)

We wanted to create a language that Hodor could use to program. Seeing as he can only say ‘Hodor’ with different inflections, the language consists of only hodor and punctuation.
(...and steve)



External DSL

At first, we tried implementing HODOR-lang as an internal DSL but did not like the constraints on the syntax. Using a parser and evaluator allowed us to design an optimal syntax for Hodor.

(Also, Hodor doesn’t know Scala)

Static Nested Scoping

Scope in HODOR-lang works like most modern programming languages. We also support variable/function closures and shadowing.

This allows us to write recursive functions that perform as expected.

Statements

Statements terminate with :)

• variable declaration	hodor {var} :)
• variable assignment	{var} Hodor {expr} :)
• expressions that end a block	{expr} :)

Variables are dynamic and are either Double, Boolean or String

Constructs that create a new scope

- Function Declarations
- if (and if else) statements
- while loops

```
HODOR...
     {statements...}
HODOR!
```

Last statement in a code block is the return (like scala)

Expressions

Expressions can be used in

- variable assignment
- functions calls
- mathematic and logical operators

After execution, the return of an expression is one of our 4 types

- logical operators return Booleans
- mathematical operators return Double
- functions return Booleans, Doubles or Strings

Prefix Operators

LOGICAL

AND:	HODOR.HODOR {expr: Boolean} {expr: Boolean}+
OR:	hodor.hodor {expr: Boolean} {expr: Boolean}+
NOT:	steve {expr: Boolean}
EQ:	hodor^hodor {expr} {expr}
LT:	hodorHODOR {expr: Double} {expr: Double}
GT:	HODORhodor {expr: Double} {expr: Double}

MATHEMATIC

ADD:	HoDoR {expr: Double} {expr: Double}+
SUB:	hOdOr {expr: Double} {expr: Double}+
MUL:	HODor {expr: Double} {expr: Double}+
DIV:	hoDOR {expr: Double} {expr: Double}+

Control Flow

IF[-ELSE] STATEMENT

```
Hodor? {expr: Boolean}
    HODOR...
        {statements}
    HODOR!
Hodor/
    HODOR...
        {statements}
    HODOR!
```

WHILE LOOP

```
hoodddooooorrrrr {expr: Boolean}
    HODOR...
        {statements}
    HODOR!
```

Functions and Others

FUNCTION DECLARATION

```
_HODOR {name} {params}_
    HODOR...
        {statements}
    HODOR!
```

COMMENT BLOCKS

```
<hodor>HODORHODORHODORHODOR<hodor>
```

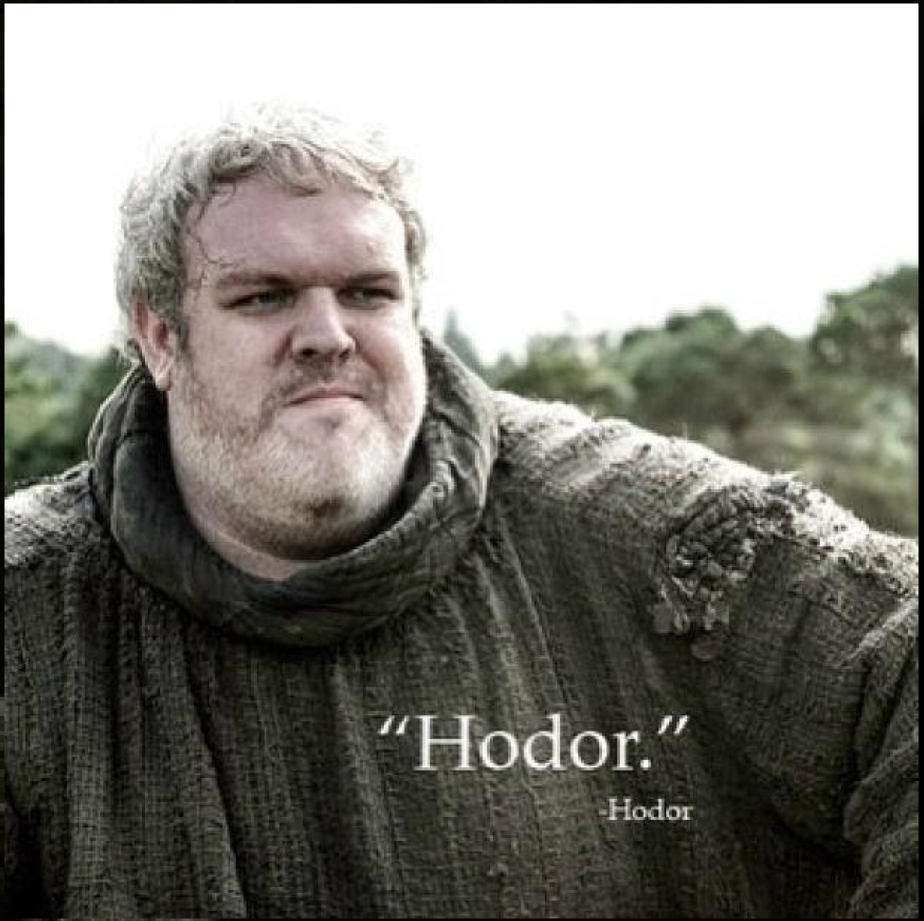
FUNCTION CALL

```
_hodor {name} {params}_
```

PRINT

```
|HODOR| {expr}
```

Demo



https://github.com/MitchStephan/HODOR-lang