DogeCode

Developed by Jovonda Robinson &
Samuel Watson

Students of CS 403 Spring 2022

Jovonda Robinson

Samuel Watson

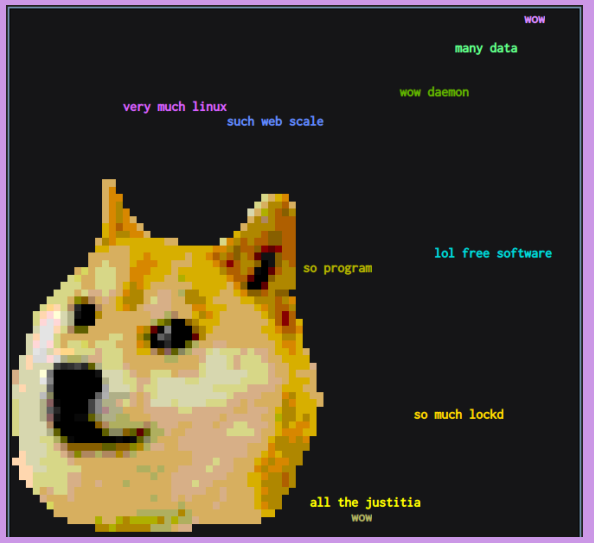
Our Team

Overview


such president
wow
post-soviet russia shibe
world leader
danger to hug
such ex-kgb
cyka

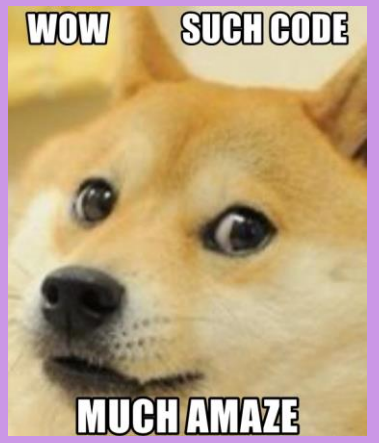

<doge>
code doge thx
click
hello world
much intelligent
many html
such programming
very computer
wow
</doge>
wow how skill


pls no shoot
such danger
i hav
family
wow :(
very scare
is it gune?



DogeCode

DogeCode is a kid-friendly programming language that incorporates playful language from the global phenomenon and meme—Doge



Overview | Justification | Grammar | Examples | Working Implementation

Doge Backstory

The popular dog seen in all Doge memes is actually named Kabosu and lives in Japan. Kabosu is a Shiba Inu

Overview | Justification | Grammar | Examples | Working Implementation

Justification

Why DogeCode?

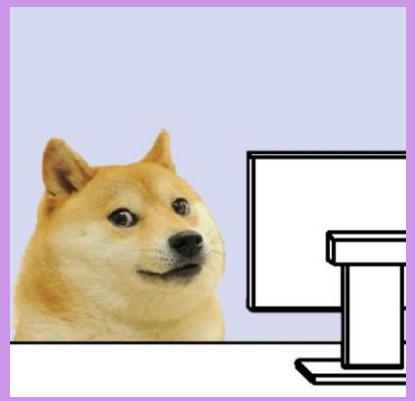
We, Jovonda & Samuel, both have extensive experience teaching local youth in Alabama how to code

Jovonda mentors high school and college-bound students and prepares them for a career in STEM. She also serves as a leader in a STEM camp in Tuscaloosa

Samuel volunteers with elementary and middle schools in Huntsville to teach students to code. He also serves as an English-as-a-second-language teacher in Tuscaloosa

Why DogeCode?

Our experience has shown us that the best way to engage children in STEM activities is to ensure they are understanding what they are doing and having fun at the same time



So we developed DogeCode, a programming language which incorporates popular meme words like these: excite, very, wow, and many. According to The University of Melbourne, these keywords appear most frequently in Doge-related memes

Why DogeCode?

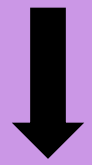
Just like **DogeCoin** brought an entire generation into the world of investing and encouraged millions to invest in cryptocurrencies, we foresee DogeCode being a useful learning tool for those hoping to learn how to program

Grammar

Grammar Note

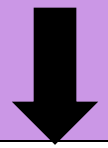
The idea for the grammar came from the craftinginterpreters, where only a few syntaxes were changed. Majority of the grammar is from here: http://www.craftinginterpreters.com/appendix-i.html

Grammar Syntax



variable = very
print = such
if = excite
else or end = boom
while = many
for = somany
function = wow
class = much

EBNF Grammar Rule

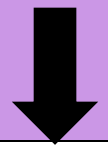


```
program → declaration* EOF ;
declaration → classDecl | funDecl | varDecl | statement ;
classDecl → "much" IDENTIFIER ( "<" IDENTIFIER )? "{" function* "}" ;
funDecl → "wow" function ;
varDecl → "very" IDENTIFIER ( "=" expression )? ";" ;
statement → exprStmt | forStmt | ifStmt | printStmt | returnStmt| whileStmt | block ;
exprStmt → expression ";" ;
forStmt → "somany" "(" ( varDecl | exprStmt | ";" ) expression? ";" expression? ")" statement ;
ifStmt → "excite" "(" expression ")" statement ( "boom" statement )? ;
printStmt → "such" expression ";" ;
returnStmt → "return" expression? ";" ;
whileStmt → "while" "(" expression ")" statement ;
block → "{" declaration* "}" ;
```

EBNF Grammar Rule Cont.

```
expression → assignment ;
assignment → ( call "." )? IDENTIFIER "=" assignment | logic_or ;
logic_or → logic_and ( "or" logic_and )* ;
logic_and → equality ( "and" equality )* ;
equality → comparison ( ( "!=" | "==" ) comparison )* ;
comparison → term ( ( ">" | ">=" | "<" | "<=" ) term )* ;
term → factor ( ( "-" | "+" ) factor )* ;
factor → unary ( ( "/" | "*" ) unary )* ;
unary → ( "!" | "-" ) unary | call ;
call → primary ( "(" arguments? ")" | "." IDENTIFIER )* ;
primary → "true" | "false" | "nil" | "this"| NUMBER | STRING | IDENTIFIER | "(" expression ")" | "super" "." IDENTIFIER ;
function → IDENTIFIER "(" parameters? ")" block ;
parameters → IDENTIFIER ( "," IDENTIFIER )* ;
arguments → expression ( "," expression )* ;
NUMBER → DIGIT+ ( "." DIGIT+ )? ;
STRING → "\"" <any char except "\"">* "\"" ;
IDENTIFIER → ALPHA ( ALPHA | DIGIT )* ;
ALPHA → "a" ... "z" | "A" ... "Z" | "_" ;
DIGIT → "0" ... "9" ;
```

Small BNF Grammar Rule



<letter>::= "A" | "B" | "C" | "D" | "E" | "F" | "G" | "H" | "I" | "J" | "K" | "L" | "M" | "N" | "O" | "P" | "Q" | "R" | "S" | "T" | "U" | "V" | "W" | "X" | "Y" | "Z" | "a" | "b" | "c" | "d" | "e" | "f" | "g" | "h" | "i" | "j" | "k" | "l" | "m" | "n" | "o" | "p" | "q" | "r" | "s" | "t" | "u" | "v" | "w" | "x" | "y" | "z"
<digit>::= "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
<string> ::= <>|<string>|<string><letter>
<number>::=<digit>
<operator>::= “+” | “-”| “/” | “*”|”<”|”<=”|”>”|”=>”|”==”|”!=”
<literal>::= <number> |<string>|”true”|”false”|”nil”
<unary> ::= “-”<expression>|”!”<expression>
<binary>::= <expression><operator><expression>
<expression>::=<literal>|<unary>|<binary>|<grouping>

Examples

Examples

Grammar Syntax

variable = very
print = such
if = excite
else or end = boom
while = many
for = somany
function = wow
class = much

```
very message = "Hello, World!"
    such message;

excite(condition)
{ such "yes"; }
boom
{ such "no"; }
```

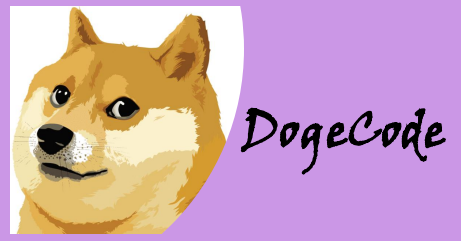
```
somany(very a = 1; a < 10; a = a +1)
{ such a; }

very a = 1;
many(a < 10)
{
        such a;
        a = a +1;
}
```

```
wow printSum(a, b)
{ such a + b; }
very printSumAgain = printSum(4, 5);
printSum( 1, 2);
printSum("Hello", World");
```

Working Implementation

Creating DogeCode with C#



To make the implementation easier, our team revised the C# code for our Lox interpreter to develop DogeCode. We only implemented up to functions because of a time constraint

You can find the GitHub repository here: https://github.com/Jovonda/Csharp-implements-dogecode

SCAN ME

GitHub Repository

Test Run Prompt

```
D:\New_Schedule_2022\CS_403_001_Programming Languages\Csharp implements dogecode\DogeApp>dotnet run
> somany(very a = 1; a < 10; a = a + 1){such a;}
1
2
3
4
5
6
7
8
9
> wow printSum(a,b){return a + b;}
> very printSumAgain = printSum(4,5);
> such printSumAgain;
9
> such printSum("Hello", " World");
Hello World
> very num1 = 2;
> very num2 = 3;
> excite(num1 == num2){such "yes";}exit{such "no";}
no
```

Running Tests 1-5

```
D:\New_Schedule_2022\CS_403_001_Programming Languages\Csharp implements dogecode\DogeApp>dotnet run Tests/Test1.
txt
Seven
false
7
7
false

D:\New_Schedule_2022\CS_403_001_Programming Languages\Csharp implements dogecode\DogeApp>dotnet run Tests/Test2.
txt
50

D:\New_Schedule_2022\CS_403_001_Programming Languages\Csharp implements dogecode\DogeApp>dotnet run Tests/Test3.
txt
1 is greater

D:\New_Schedule_2022\CS_403_001_Programming Languages\Csharp implements dogecode\DogeApp>dotnet run Tests/Test4.
txt
President of the United States
Joe Biden
Joe Biden is the President of the United States.

D:\New_Schedule_2022\CS_403_001_Programming Languages\Csharp implements dogecode\DogeApp>dotnet run Tests/Test5.
txt
A is greater
2
3
4
```

Running All Tests

```
D:\New_Schedule_2022\CS_403_001_Programming Languages\Csharp implements dogecode\DogeApp>dotnet run Tests/All_Tests.txt
Seven
false
7
7
false
50
1 is greater
President of the United States
Joe Biden
Joe Biden is the President of the United States.
A is greater
2
3
4
5
For Loop 1
10
20
30
40
For Loop 2
0.5
1
1.5
2
2.5
3
```

Running All Tests Cont.

```
3.5
4
4.5
Hello World
4
49
144
1
2
21
301
This has not ended yet.
This has not ended yet.
This has not ended yet.
This has not ended yet.
This has not ended yet.
This has not ended yet.
This has not ended yet.
This has not ended yet.
This has not ended yet.
This has not ended yet.
cake
icecream
cookie
brownie
*
*
*
*
```

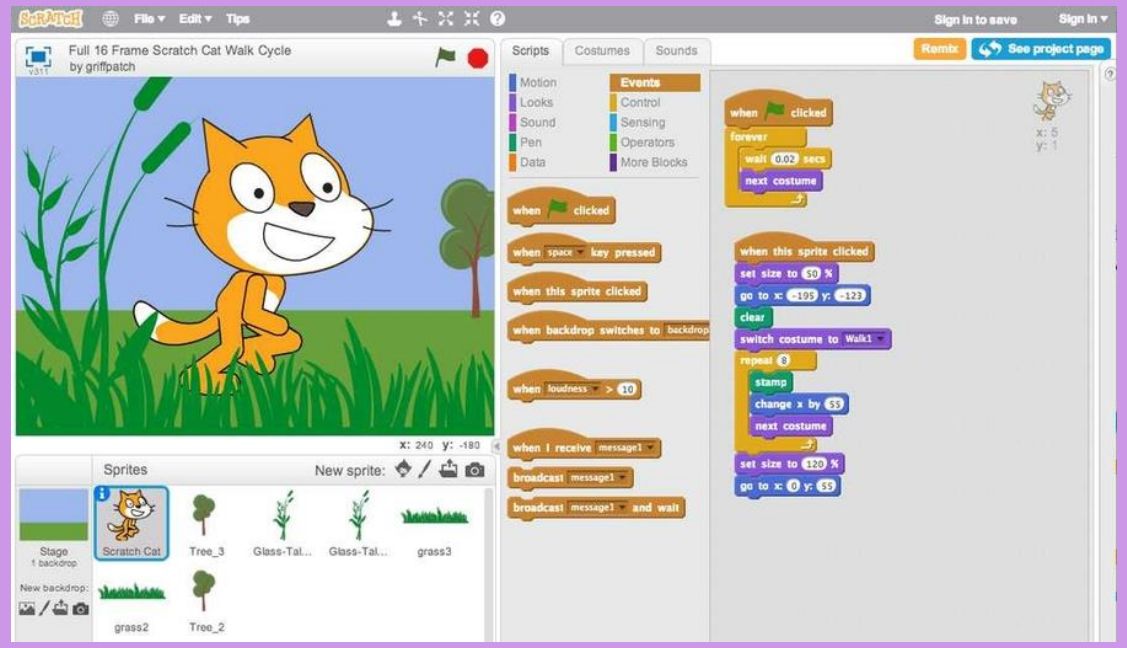
Running All Tests Cont.

```
brownie
*
*
*
*
*
*
*
*
*
*
*
24
3628800
6.204484017332394E+23
8.841761993739701E+30
The sum of three is
6
The sum of nine is
45
The sum of five is
15
Janay
```

The Future of DogeCode?

If we were to continue developing DogeCode, we would strive to make it even more kid-friendly by creating a drag-and-drop programming interface for kids to use

Scratch is a good example of the kid-friendly drag-and-drop interface we would mirror



Scratch Programming

Closing Remarks

How to Speak Doge — The University of Melbourne

https://blogs.unimelb.edu.au/sciencecommunication/2016/10/22/how-to-speak-doge/

Doge Grammar Explained by a Linguist

https://the-toast.net/2014/02/06/linguist-explains-grammar-doge-wow/

DrewBanas GitHub Repo
https://github.com/drewbanas/jloxcs

Sources

Thank you