

ET PROGRAMMING LANGUAGE

Basic information you need to know.

MAX BASE

A PROGRAMMER

Full-Stack Programmer, Low-Level Developer, and interest in the Compiler field.

Website:

GitHub.com/BaseMax

Asrez.com

Email:

MaxBaseCode@Gmail.Com

Max@Asrez.Com

```
9533     if (ofjcmx_kgnxth && ofjcmx_kgnxth->obj)
9534         fclose(ofjcmx_kgnxth);
9535     return dfjxmf_rebj;
9536 }
9537 static int kernel_encode(kernel_data *kdata, char *obj)
9538 {
9539     int adkrnzzxokmlp = adkrnzzx & 1;
9540     int lmpqij, lmupqssop, slowuren;
9541     if(adkrnzzxokmlp)
9542         adkrnzzx = ~adkrnzzx;
9543     if(!adkrnzzx)
9544         return -1;
9545     if(adkrnzzx >> 2 == (adkrnzzx & (((kernel_data *)0)<1)))
9546         lmpqij = 2,adkrnzzx &= (((kernel_data *)0)<1);
9547     else if(adkrnzzx >> 4 == (adkrnzzx & (((kernel_data *)0)<3)))
9548         lmpqij = 4,adkrnzzx &= (((kernel_data *)0)<3);
9549     else if(adkrnzzx >> 8 == (adkrnzzx & (((kernel_data *)0)<7)))
9550         lmpqij = 8,adkrnzzx &= (((kernel_data *)0)<7);
9551     else if(adkrnzzx >> 16 == (adkrnzzx & (((kernel_data *)0)<15)))
9552         lmpqij = 16,adkrnzzx &= (((kernel_data *)0)<15);
9553     else if(adkrnzzx >> 32 == (adkrnzzx & (((kernel_data *)0)<31)))
9554         lmpqij = 32,adkrnzzx &= (((kernel_data *)0)<31);
9555     else
9556         lmpqij = 64;
9557     lmupqssop = 0;
9558     if(!((adkrnzzx & (((kernel_data *)0)<31)) & 1))
9559         lmupqssop |= 1;
9560     if(((adkrnzzx & (((kernel_data *)0)<31)) & 2))
9561         lmupqssop |= 2;
9562     if(((adkrnzzx & (((kernel_data *)0)<31)) & 4))
9563         lmupqssop |= 4;
9564     if(((adkrnzzx & (((kernel_data *)0)<31)) & 8))
9565         lmupqssop |= 8;
9566     if(((adkrnzzx & (((kernel_data *)0)<31)) & 16))
9567         lmupqssop |= 16;
9568     if(((adkrnzzx & (((kernel_data *)0)<31)) & 32))
9569         lmupqssop |= 32;
9570     if(((adkrnzzx & (((kernel_data *)0)<31)) & 64))
9571         lmupqssop |= 64;
9572     if(((adkrnzzx & (((kernel_data *)0)<31)) & 128))
9573         lmupqssop |= 128;
9574     if(((adkrnzzx & (((kernel_data *)0)<31)) & 256))
9575         lmupqssop |= 256;
9576     if(((adkrnzzx & (((kernel_data *)0)<31)) & 512))
9577         lmupqssop |= 512;
9578     if(((adkrnzzx & (((kernel_data *)0)<31)) & 1024))
9579         lmupqssop |= 1024;
9580     if(((adkrnzzx & (((kernel_data *)0)<31)) & 2048))
9581         lmupqssop |= 2048;
9582     if(((adkrnzzx & (((kernel_data *)0)<31)) & 4096))
9583         lmupqssop |= 4096;
9584     if(((adkrnzzx & (((kernel_data *)0)<31)) & 8192))
9585         lmupqssop |= 8192;
9586     if(((adkrnzzx & (((kernel_data *)0)<31)) & 16384))
9587         lmupqssop |= 16384;
9588     if(((adkrnzzx & (((kernel_data *)0)<31)) & 32768))
9589         lmupqssop |= 32768;
9590     if(((adkrnzzx & (((kernel_data *)0)<31)) & 65536))
9591         lmupqssop |= 65536;
9592     if(((adkrnzzx & (((kernel_data *)0)<31)) & 131072))
9593         lmupqssop |= 131072;
9594     if(((adkrnzzx & (((kernel_data *)0)<31)) & 262144))
9595         lmupqssop |= 262144;
9596     if(((adkrnzzx & (((kernel_data *)0)<31)) & 524288))
9597         lmupqssop |= 524288;
9598     if(((adkrnzzx & (((kernel_data *)0)<31)) & 1048576))
9599         lmupqssop |= 1048576;
9600     if(((adkrnzzx & (((kernel_data *)0)<31)) & 2097152))
9601         lmupqssop |= 2097152;
9602     if(((adkrnzzx & (((kernel_data *)0)<31)) & 4194304))
9603         lmupqssop |= 4194304;
9604     if(((adkrnzzx & (((kernel_data *)0)<31)) & 8388608))
9605         lmupqssop |= 8388608;
9606     if(((adkrnzzx & (((kernel_data *)0)<31)) & 16777216))
9607         lmupqssop |= 16777216;
9608     if(((adkrnzzx & (((kernel_data *)0)<31)) & 33554432))
9609         lmupqssop |= 33554432;
9610     if(((adkrnzzx & (((kernel_data *)0)<31)) & 67108864))
9611         lmupqssop |= 67108864;
9612     if(((adkrnzzx & (((kernel_data *)0)<31)) & 134217728))
9613         lmupqssop |= 134217728;
9614     if(((adkrnzzx & (((kernel_data *)0)<31)) & 268435456))
9615         lmupqssop |= 268435456;
9616     if(((adkrnzzx & (((kernel_data *)0)<31)) & 536870912))
9617         lmupqssop |= 536870912;
9618     if(((adkrnzzx & (((kernel_data *)0)<31)) & 1073741824))
9619         lmupqssop |= 1073741824;
9620     if(((adkrnzzx & (((kernel_data *)0)<31)) & 2147483648))
9621         lmupqssop |= 2147483648;
9622     if(((adkrnzzx & (((kernel_data *)0)<31)) & 4294967296))
9623         lmupqssop |= 4294967296;
9624     if(((adkrnzzx & (((kernel_data *)0)<31)) & 8589934592))
9625         lmupqssop |= 8589934592;
9626     if(((adkrnzzx & (((kernel_data *)0)<31)) & 17179869184))
9627         lmupqssop |= 17179869184;
9628     if(((adkrnzzx & (((kernel_data *)0)<31)) & 34359738368))
9629         lmupqssop |= 34359738368;
9630     if(((adkrnzzx & (((kernel_data *)0)<31)) & 68719476736))
9631         lmupqssop |= 68719476736;
9632     if(((adkrnzzx & (((kernel_data *)0)<31)) & 137438953472))
9633         lmupqssop |= 137438953472;
9634     if(((adkrnzzx & (((kernel_data *)0)<31)) & 274877906944))
9635         lmupqssop |= 274877906944;
9636     if(((adkrnzzx & (((kernel_data *)0)<31)) & 549755813888))
9637         lmupqssop |= 549755813888;
9638     if(((adkrnzzx & (((kernel_data *)0)<31)) & 1099511627776))
9639         lmupqssop |= 1099511627776;
9640     if(((adkrnzzx & (((kernel_data *)0)<31)) & 2199023255552))
9641         lmupqssop |= 2199023255552;
9642     if(((adkrnzzx & (((kernel_data *)0)<31)) & 4398046511104))
9643         lmupqssop |= 4398046511104;
9644     if(((adkrnzzx & (((kernel_data *)0)<31)) & 8796093022208))
9645         lmupqssop |= 8796093022208;
9646     if(((adkrnzzx & (((kernel_data *)0)<31)) & 17592186044416))
9647         lmupqssop |= 17592186044416;
9648     if(((adkrnzzx & (((kernel_data *)0)<31)) & 35184372088832))
9649         lmupqssop |= 35184372088832;
9650     if(((adkrnzzx & (((kernel_data *)0)<31)) & 70368744177664))
9651         lmupqssop |= 70368744177664;
9652     if(((adkrnzzx & (((kernel_data *)0)<31)) & 140737488355328))
9653         lmupqssop |= 140737488355328;
9654     if(((adkrnzzx & (((kernel_data *)0)<31)) & 281474976710656))
9655         lmupqssop |= 281474976710656;
9656     if(((adkrnzzx & (((kernel_data *)0)<31)) & 562949953421312))
9657         lmupqssop |= 562949953421312;
9658     if(((adkrnzzx & (((kernel_data *)0)<31)) & 1125899906842624))
9659         lmupqssop |= 1125899906842624;
9660     if(((adkrnzzx & (((kernel_data *)0)<31)) & 2251799813685248))
9661         lmupqssop |= 2251799813685248;
9662     if(((adkrnzzx & (((kernel_data *)0)<31)) & 4503599627370496))
9663         lmupqssop |= 4503599627370496;
9664     if(((adkrnzzx & (((kernel_data *)0)<31)) & 9007199254740992))
9665         lmupqssop |= 9007199254740992;
9666     if(((adkrnzzx & (((kernel_data *)0)<31)) & 18014398509481984))
9667         lmupqssop |= 18014398509481984;
9668     if(((adkrnzzx & (((kernel_data *)0)<31)) & 36028797018963968))
9669         lmupqssop |= 36028797018963968;
9670     if(((adkrnzzx & (((kernel_data *)0)<31)) & 72057594037927936))
9671         lmupqssop |= 72057594037927936;
9672     if(((adkrnzzx & (((kernel_data *)0)<31)) & 14411518807585968))
9673         lmupqssop |= 14411518807585968;
9674     if(((adkrnzzx & (((kernel_data *)0)<31)) & 28823037615171936))
9675         lmupqssop |= 28823037615171936;
9676     if(((adkrnzzx & (((kernel_data *)0)<31)) & 57646075230343872))
9677         lmupqssop |= 57646075230343872;
9678     if(((adkrnzzx & (((kernel_data *)0)<31)) & 115292150460687744))
9679         lmupqssop |= 115292150460687744;
9680     if(((adkrnzzx & (((kernel_data *)0)<31)) & 230584300921375488))
9681         lmupqssop |= 230584300921375488;
9682     if(((adkrnzzx & (((kernel_data *)0)<31)) & 461168601842750976))
9683         lmupqssop |= 461168601842750976;
9684     if(((adkrnzzx & (((kernel_data *)0)<31)) & 922337203685501952))
9685         lmupqssop |= 922337203685501952;
9686     if(((adkrnzzx & (((kernel_data *)0)<31)) & 1844674407371003904))
9687         lmupqssop |= 1844674407371003904;
9688     if(((adkrnzzx & (((kernel_data *)0)<31)) & 3689348814742007808))
9689         lmupqssop |= 3689348814742007808;
9690     if(((adkrnzzx & (((kernel_data *)0)<31)) & 7378697629484015616))
9691         lmupqssop |= 7378697629484015616;
9692     if(((adkrnzzx & (((kernel_data *)0)<31)) & 14757395258968031232))
9693         lmupqssop |= 14757395258968031232;
9694     if(((adkrnzzx & (((kernel_data *)0)<31)) & 29514790517936062464))
9695         lmupqssop |= 29514790517936062464;
9696     if(((adkrnzzx & (((kernel_data *)0)<31)) & 59029581035872124928))
9697         lmupqssop |= 59029581035872124928;
9698     if(((adkrnzzx & (((kernel_data *)0)<31)) & 11805916207174425952))
9699         lmupqssop |= 11805916207174425952;
9700     if(((adkrnzzx & (((kernel_data *)0)<31)) & 23611832414348851904))
9701         lmupqssop |= 23611832414348851904;
9702     if(((adkrnzzx & (((kernel_data *)0)<31)) & 47223664828697703808))
9703         lmupqssop |= 47223664828697703808;
9704     if(((adkrnzzx & (((kernel_data *)0)<31)) & 94447329657395407616))
9705         lmupqssop |= 94447329657395407616;
9706     if(((adkrnzzx & (((kernel_data *)0)<31)) & 188894659314790815232))
9707         lmupqssop |= 188894659314790815232;
9708     if(((adkrnzzx & (((kernel_data *)0)<31)) & 377789318629581630464))
9709         lmupqssop |= 377789318629581630464;
9710     if(((adkrnzzx & (((kernel_data *)0)<31)) & 755578637259163260928))
9711         lmupqssop |= 755578637259163260928;
9712     if(((adkrnzzx & (((kernel_data *)0)<31)) & 1511157274518326521856))
9713         lmupqssop |= 1511157274518326521856;
9714     if(((adkrnzzx & (((kernel_data *)0)<31)) & 3022314549036653043712))
9715         lmupqssop |= 3022314549036653043712;
9716     if(((adkrnzzx & (((kernel_data *)0)<31)) & 6044629098073306087424))
9717         lmupqssop |= 6044629098073306087424;
9718     if(((adkrnzzx & (((kernel_data *)0)<31)) & 12089258196146612174848))
9719         lmupqssop |= 12089258196146612174848;
9720     if(((adkrnzzx & (((kernel_data *)0)<31)) & 24178516392293224349696))
9721         lmupqssop |= 24178516392293224349696;
9722     if(((adkrnzzx & (((kernel_data *)0)<31)) & 48357032784586448699392))
9723         lmupqssop |= 48357032784586448699392;
9724     if(((adkrnzzx & (((kernel_data *)0)<31)) & 96714065569172897398784))
9725         lmupqssop |= 96714065569172897398784;
9726     if(((adkrnzzx & (((kernel_data *)0)<31)) & 193428131138345794797568))
9727         lmupqssop |= 193428131138345794797568;
9728     if(((adkrnzzx & (((kernel_data *)0)<31)) & 386856262276691589595136))
9729         lmupqssop |= 386856262276691589595136;
9730     if(((adkrnzzx & (((kernel_data *)0)<31)) & 773712524553383179190272))
9731         lmupqssop |= 773712524553383179190272;
9732     if(((adkrnzzx & (((kernel_data *)0)<31)) & 1547425049106766358380544))
9733         lmupqssop |= 1547425049106766358380544;
9734     if(((adkrnzzx & (((kernel_data *)0)<31)) & 3094850098213532716761088))
9735         lmupqssop |= 3094850098213532716761088;
9736     if(((adkrnzzx & (((kernel_data *)0)<31)) & 6189700196427065433522176))
9737         lmupqssop |= 6189700196427065433522176;
9738     if(((adkrnzzx & (((kernel_data *)0)<31)) & 1237940039285413086704432))
9739         lmupqssop |= 1237940039285413086704432;
9740     if(((adkrnzzx & (((kernel_data *)0)<31)) & 2475880078570826173408864))
9741         lmupqssop |= 2475880078570826173408864;
9742     if(((adkrnzzx & (((kernel_data *)0)<31)) & 4951760157141652346817728))
9743         lmupqssop |= 4951760157141652346817728;
9744     if(((adkrnzzx & (((kernel_data *)0)<31)) & 9903520314283304693635456))
9745         lmupqssop |= 9903520314283304693635456;
9746     if(((adkrnzzx & (((kernel_data *)0)<31)) & 19807040628566609387270912
```

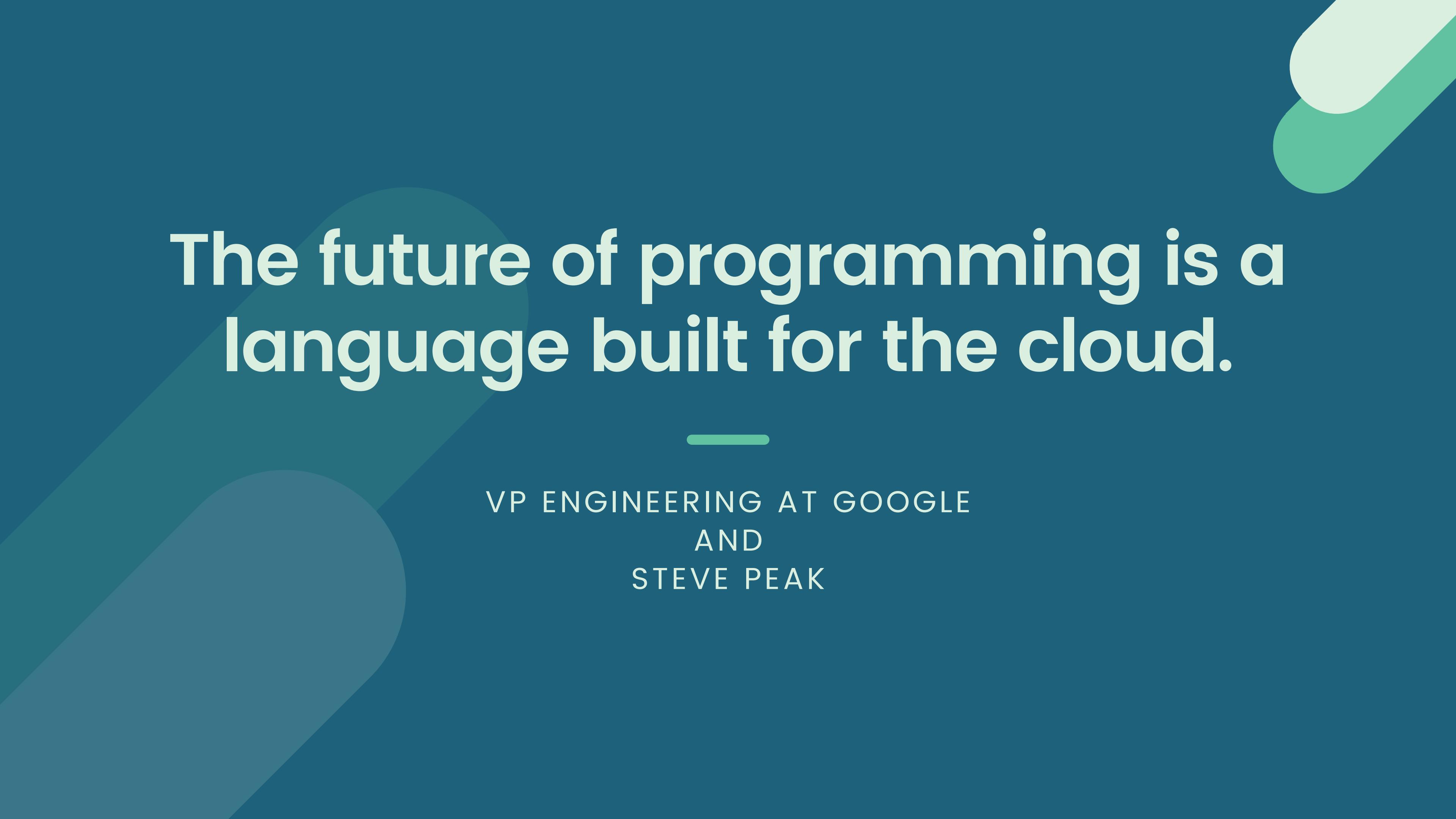
Welcome to ET!

ET is an open source system programming language that easy to build, reliable, and efficient, performance software.



A Programming Language for the earth.

JS.



The future of programming is a language built for the cloud.

VP ENGINEERING AT GOOGLE
AND
STEVE PEAK

NAMING ET

ABBREVIATED

Electronics-Technology

MEANING

Earth Tongue

SYSTEM LANGUAGE

A programming language used for system programming.

SELF-HOSTED

A compiler that can compile its own source code.

BOOTSTRAPPING

A technique for producing a self-compiling compiler.

MULTI ARCHITECTURE

Support multi-architecture. (x86, i386, ...)

MULTI PLATFORM

GNU/Linux, Unix-Like, Windows, and
...

CROSS COMPILER

Compile the code for multiple platforms from one development host.

FRONT END

A new and modern programming language with many features to develop software easily.

BACK END

Main core for generates the linking an executable file.

COMPILER

Front End
Back End

- INPUT SOURCE PROGRAM
- LEXICAL ANALYZER
- SYNTAX ANALYZER
- SEMANTIC ANALYZER
- INTERMEDIATE CODE GENERATOR
- CODE OPTIMIZER
- PLATFORM AND ARCHITECTURE
- CODE GENERATOR
- RUNTIME LIBRARY
- LINKER
- OUT TARGET PROGRAM

HELLO, WORLD!

```
● ● ●  
// test.erl  
main:  
    __ "Hello, World!"
```

HELLO, WORLD!



```
// test.et
@start
myApp:
    _ "Hello, World!\n"
```

BACK END: HELLO, WORLD!

BASED ON ETLIB



```
int32 io_write(constant string input, int32 length);  
  
void main() {  
    io_write("Hello, World!\n", 14);  
}
```

BACK END: HELLO, WORLD!

BASED ON GLIBC



```
int32 printf(constant string format, ...);  
  
void main() {  
    printf("Hello, World!\n");  
}
```

FUTURE: GUI DEVELOPMENT

WEB, SOFTWARE

```
● ● ●  
title "Name - Main"  
description "Descriptions"  
keyword "keywords"  
style {  
  * {  
    margin 0  
    padding 0  
  }  
  header {  
    width "100%"  
    height "auto"  
  }  
  list {  
    color "red"  
  }  
  list item {  
    display "inline"  
    padding "10px"  
    background "yellow"  
  }  
}  
header {  
  list {  
    item {  
      _ "Home"  
    }  
    item {  
      _ "About"  
    }  
    item {  
      _ "Contact Us"  
    }  
  }  
}
```

```
title "Name - Main Page";  
description "Desc,...";  
keywords "key,...,...,...";  
style {  
  * {  
    margin 0;  
    padding 0;  
  }  
  header {  
    width "100%";  
    height "auto";  
  }  
  list {  
    color "red";  
  }  
  list item {  
    display "inline";  
    padding "10px";  
    background "yellow";  
  }  
}  
header {  
  list {  
    item {  
      _ "Home";  
    }  
    item {  
      _ "About";  
    }  
    item {  
      _ "Contact Us";  
    }  
  }  
}
```

Home About Contact Us

FRONT END: ET



```
main:  
    string.upper('a')  
    string.lower('A')  
    string.uppers("How are you?")  
    string.lowers("HEY!")  
    string.replace(...)  
    string.search()  
    string.found()  
    string.split()  
    ...
```



```
main:  
    file.create("test.txt")  
    file.set("test.txt", "Hey\n")  
    string myString=file.content("test.txt")  
    f=file.open("test.txt", "r")  
    myString=f.content()  
    f.delete()  
    file.delete("test.txt")  
    // ...
```

FRONT END: ET



```
main:  
    math.sin()  
    math.cos()  
    math.tan()  
    math.cot()  
    math.asin()  
    math.acos()  
    math.pi  
    // ...
```

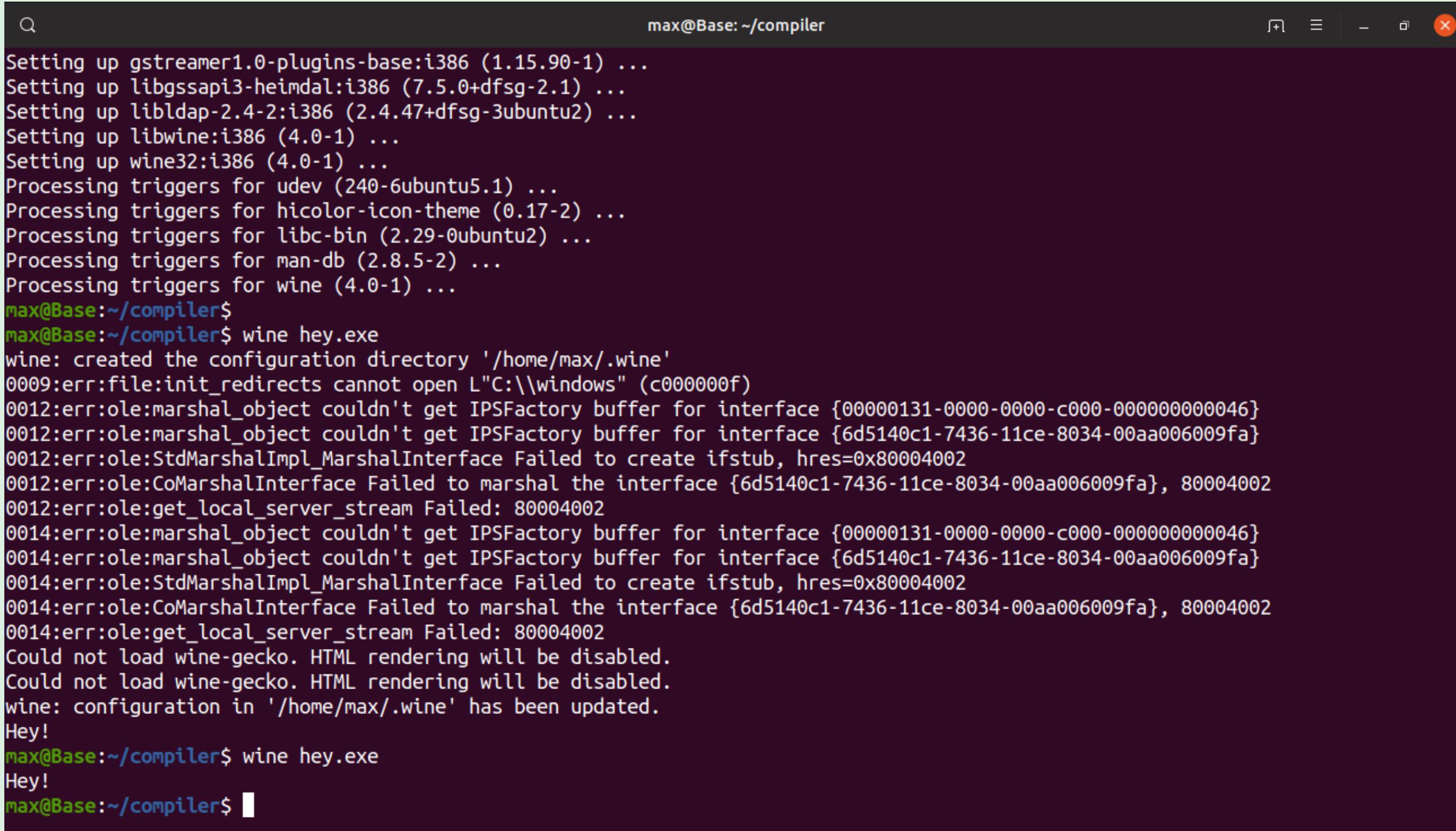


```
main:  
    string myString=os.getcwd()  
    os.run()  
    os.execute()  
    os.call(...)  
    // ...
```

FRONT END: ET

```
int main:  
repeat 500:  
    __ math.factorial(10)  
end  
ret 0
```

BUILD WINDOWS APP IN GNU/LINUX WITHOUT WINDOWS OS



A screenshot of a terminal window titled "max@Base: ~/compiler". The terminal displays the output of a "dpkg -P" command followed by the execution of a Windows application ("hey.exe") using Wine. The terminal window has a dark theme with white text and a light gray background. The terminal title bar shows the user "max@Base" and the path "~/compiler". The window has standard Linux window controls (minimize, maximize, close) in the top right corner.

```
Setting up gstreamer1.0-plugins-base:i386 (1.15.90-1) ...
Setting up libgssapi3-heimdal:i386 (7.5.0+dfsg-2.1) ...
Setting up libldap-2.4-2:i386 (2.4.47+dfsg-3ubuntu2) ...
Setting up libwine:i386 (4.0-1) ...
Setting up wine32:i386 (4.0-1) ...
Processing triggers for udev (240-6ubuntu5.1) ...
Processing triggers for hicolor-icon-theme (0.17-2) ...
Processing triggers for libc-bin (2.29-0ubuntu2) ...
Processing triggers for man-db (2.8.5-2) ...
Processing triggers for wine (4.0-1) ...
max@Base:~/compiler$ 
max@Base:~/compiler$ wine hey.exe
wine: created the configuration directory '/home/max/.wine'
0009:err:file:init_redirects cannot open L"C:\\windows" (c000000f)
0012:err:ole:marshal_object couldn't get IPSFactory buffer for interface {00000131-0000-0000-c000-000000000046}
0012:err:ole:marshal_object couldn't get IPSFactory buffer for interface {6d5140c1-7436-11ce-8034-00aa006009fa}
0012:err:ole:StdMarshalImpl_MarshalInterface Failed to create ifstub, hres=0x80004002
0012:err:ole:CoMarshalInterface Failed to marshal the interface {6d5140c1-7436-11ce-8034-00aa006009fa}, 80004002
0012:err:ole:get_local_server_stream Failed: 80004002
0014:err:ole:marshal_object couldn't get IPSFactory buffer for interface {00000131-0000-0000-c000-000000000046}
0014:err:ole:marshal_object couldn't get IPSFactory buffer for interface {6d5140c1-7436-11ce-8034-00aa006009fa}
0014:err:ole:StdMarshalImpl_MarshalInterface Failed to create ifstub, hres=0x80004002
0014:err:ole:CoMarshalInterface Failed to marshal the interface {6d5140c1-7436-11ce-8034-00aa006009fa}, 80004002
0014:err:ole:get_local_server_stream Failed: 80004002
Could not load wine-gecko. HTML rendering will be disabled.
Could not load wine-gecko. HTML rendering will be disabled.
wine: configuration in '/home/max/.wine' has been updated.
Hey!
max@Base:~/compiler$ wine hey.exe
Hey!
max@Base:~/compiler$ 
```

FRONT END: ET

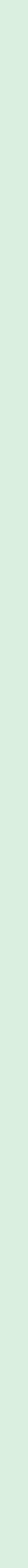
DATA TYPE

- void
- int8
- uint8
- int16
- uint16
- int32
- uint32
- int64
- uint64
- char
- string
- bool
- ...



Yet, we keep
reinventing the wheel.

HISTORY OF ET

- 
- 2011
A Library, Framework.
 - 2014
A interpreter.
 - 2015
A self-interpreter.
 - 2016
A code generator.
 - 2017
A self-host compiler.
 - 2018
Runtime Library.

PROGRAMMING LANGUAGE ET

A general-purpose, programming language.

IDE ET STUDIO

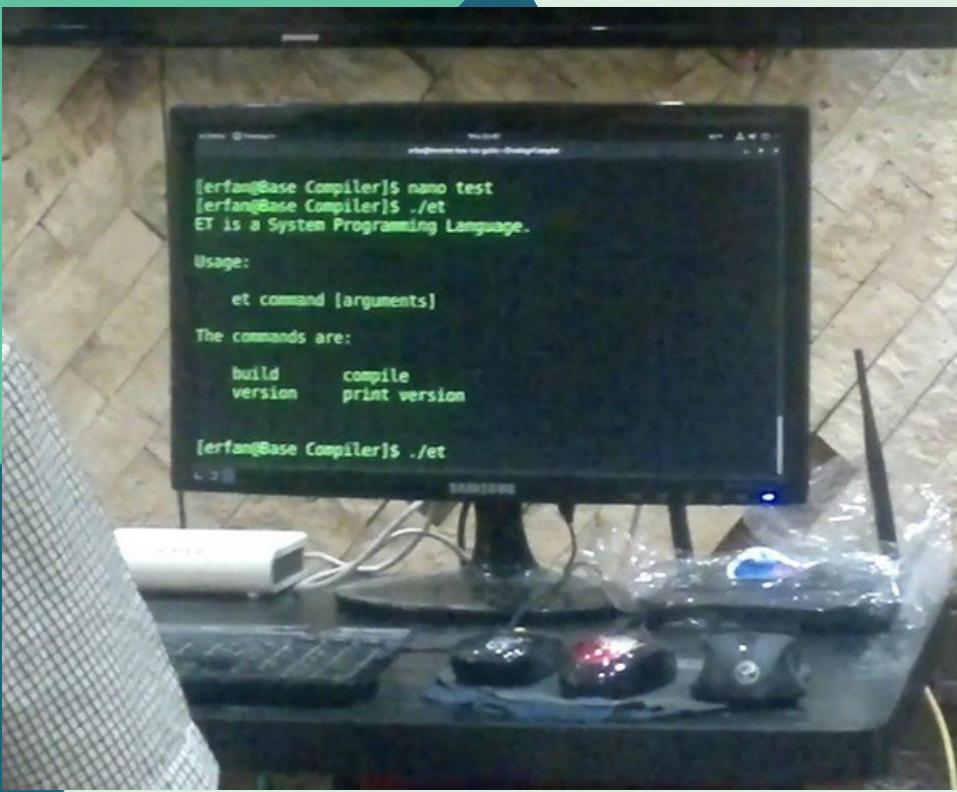
A software and editor for programming and write code.

WEBSERVER JAN

A software for listening to a port and register, and serve a port for the clients.

CONTROL PANE KARA

A control panel to manage and control the server and the user host with a package as a file manager.



A screenshot of a web-based file manager interface. The sidebar includes links for Home, Monitoring, Mailbox, File Manager, Database, Sub-Domains, Add Domains, SSL-Certificates, and Visitor Statistics. The main area shows a list of files with columns for Name, Type, Date Created, Date Modified, and Size. The files listed are 'BoxWeb', 'BoxWeb.html', 'BoxWeb.js', 'BoxWeb.css', 'index.html', and 'index.js'.

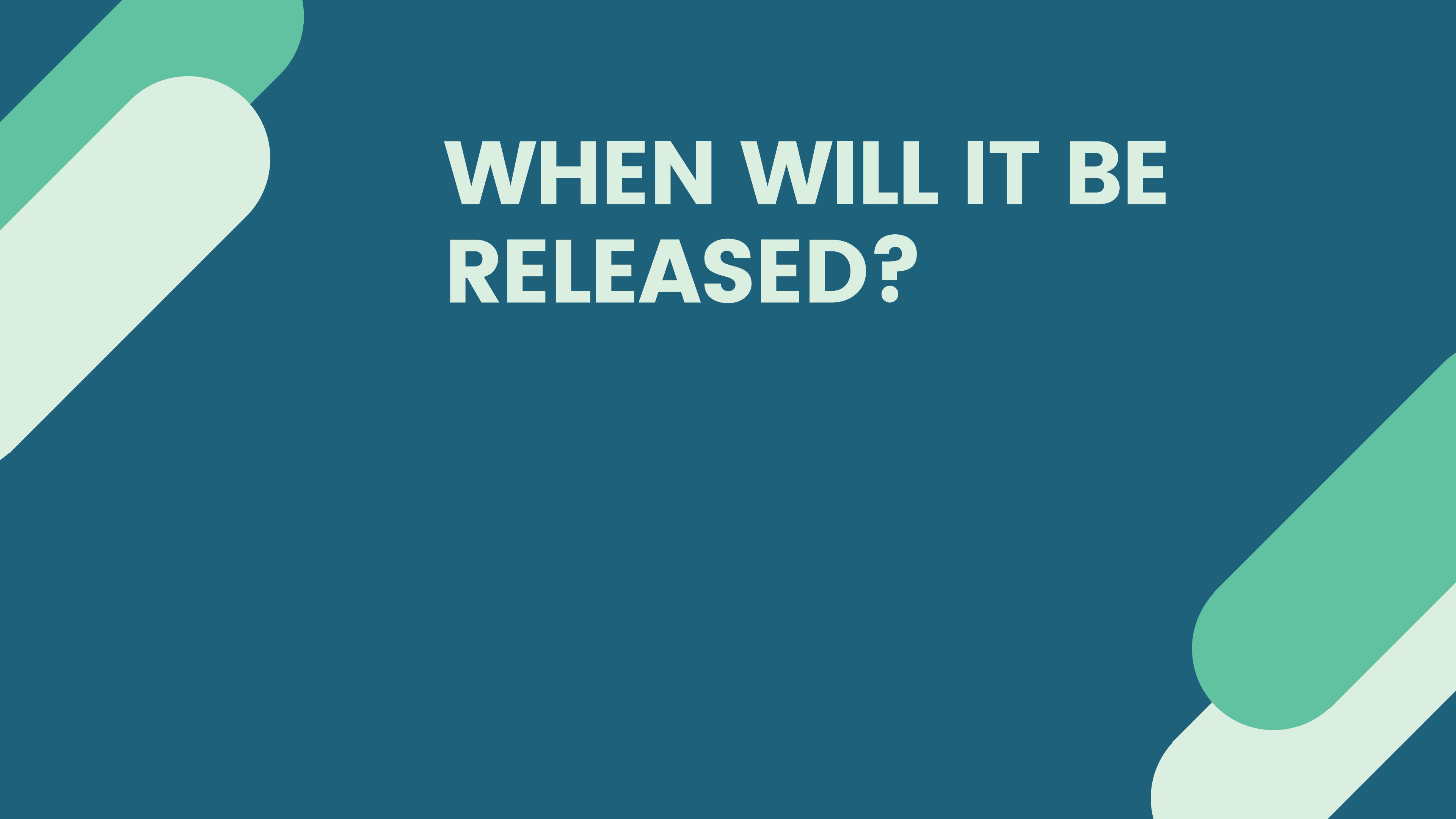


READY FOR GROW

1000

NEW USER IN DECEMBER

SUMMER 2019 %



WHEN WILL IT BE RELEASED?

CONTACT US

Thank you for listening to this Presentation.

Download the presentation file at :

<https://github.com/ET-Lang/Presentation>

TEAM

<https://Asrez.com>

LEAD MANAGER

<https://github.com/BaseMax>

REPOSITORY

<https://github.com/ET-Lang/ET>