JS WA

郭力恒 广发证券 GF SECURITIES IT

面向未来的原生化web开发

cutting edge frontend skills

2018.4.20

wasm前世今生

1995年Javascript诞生 Brendan Eich

前端网页时代

2008年V8诞生 即时编译JIT

前端App时代

. . .

两个问题

- 性能　　　　移动设备，视频，游戏
- 单一语言

  Typescript, Clojurescript, Coffeescript, Elem...

原生化技术

- ActiveX，NPAPI
- NaCl

2013 asmjs

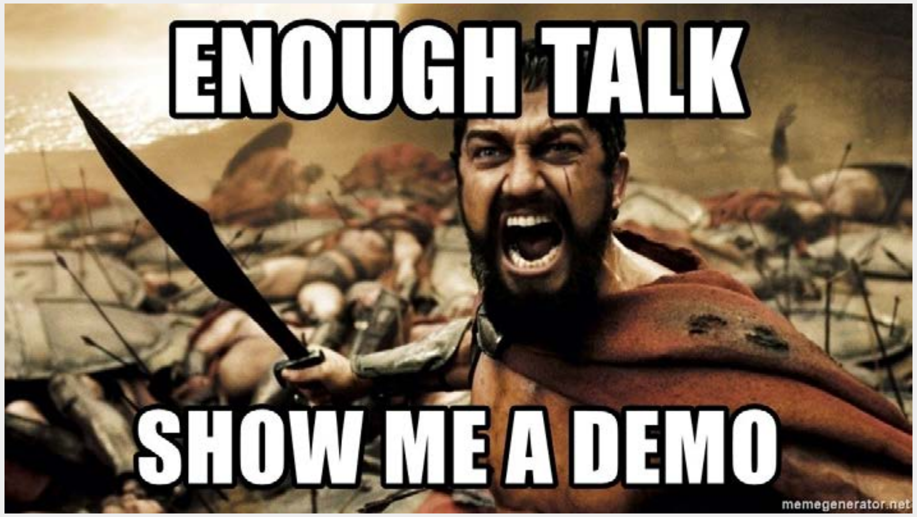
```
function add(x, y) {
  x = x|0;
  y = y|0;
  return (x + y)|0;
}
```

2015

WEBASSEMBLY

```
0x00   0x61   0x73   0x6d   0x01   0x00   0x00
0x00   0x01   0x07   0x01   0x60   0x02   0x7f
0x7f   0x01   0x7f   0x03   0x02   0x01   0x00
0x07   0x07   0x01   0x03   0x61   0x64   0x64
0x00   0x00   0x0a   0x09   0x01   0x07   0x00
0x20   0x00   0x20   0x01   0x6a   0x0b
```

```
(module
  (func
    (export "add")
    (param $lhs i32)
    (param $rhs i32)
    (result i32)

    (get_local $lhs)
    (get_local $rhs)
    (i32.add)))
```



https://gliheng.github.io/rust-wasm/sdl2-mandelbrot

```
WebAssembly.instantiate(bufferSource, importObject)
```

```
WebAssembly.instantiate(buffer, {})
    .then(result => result.instance)
    .then(mod => {
        mod.exports.add(100, 1000);
    });
```

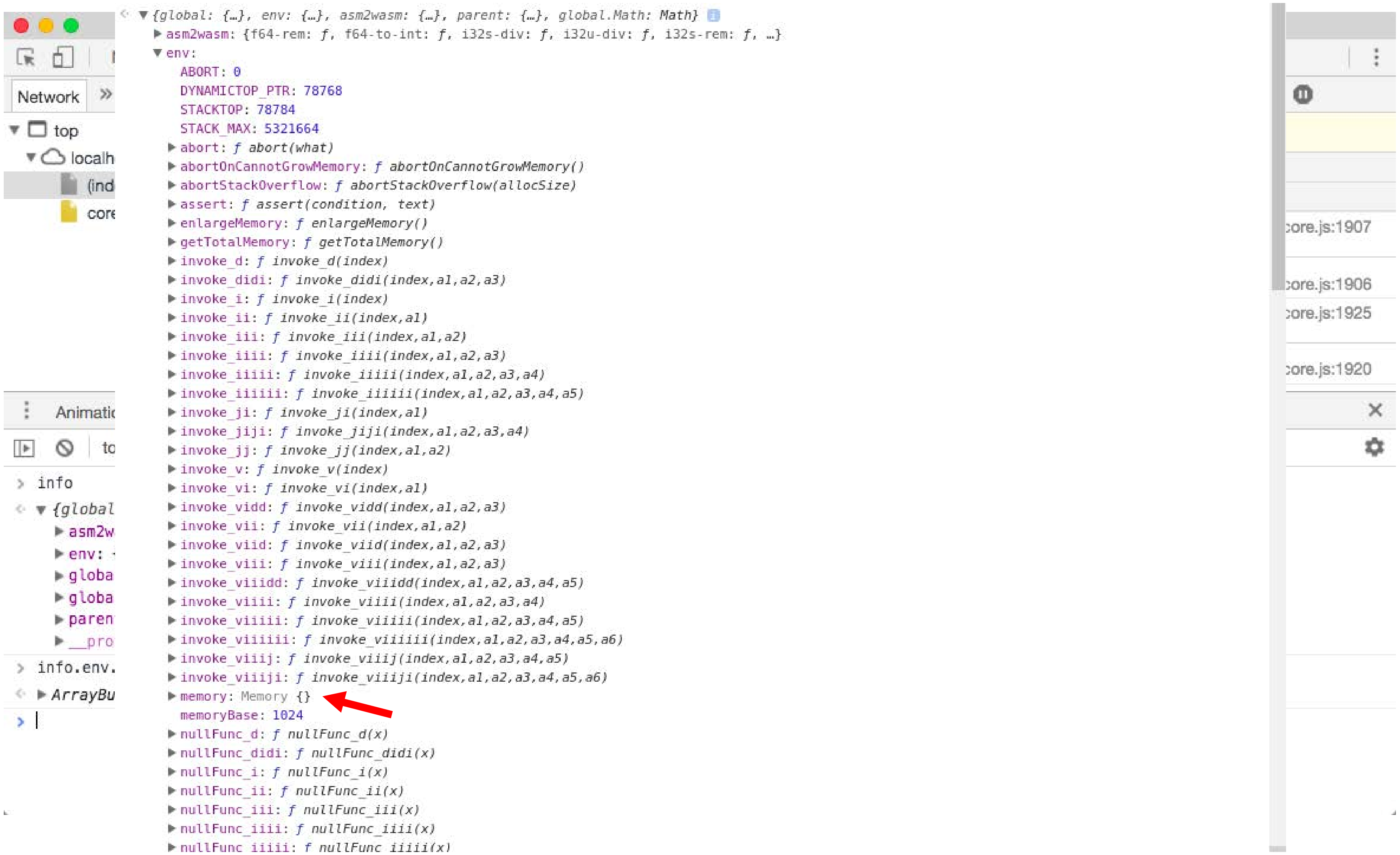
```
WebAssembly.instantiateStreaming(fetch('app.wasm'), importObject)
WebAssembly.instantiate(bufferSource, importObject)
```

```
WebAssembly.instantiate(bufferSource, {
    constants: {
        n: 9
    },
    console: {
        log: function(arg) {
            console.log(">>>", arg);
        }
    },
    js: {
        mem: new WebAssembly.Memory({
            initial:10, maximum:100
        })
    }
});
```

数字

函数

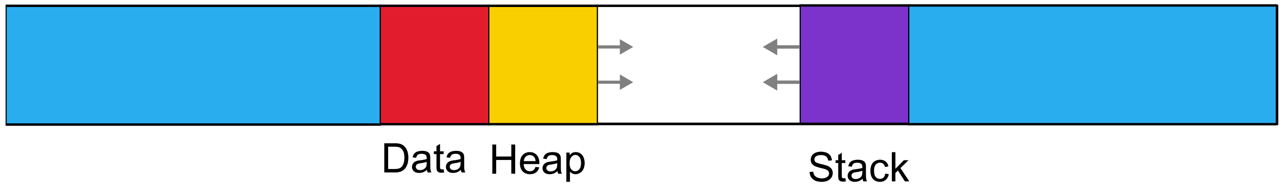
内存对象



```
> info
< ▼{global
   ▶asm2w
   ▶env: 
   ▶globa
   ▶globa
   ▶paren
   ▶__pro
> info.env.
< ▶ArrayBu
>
```

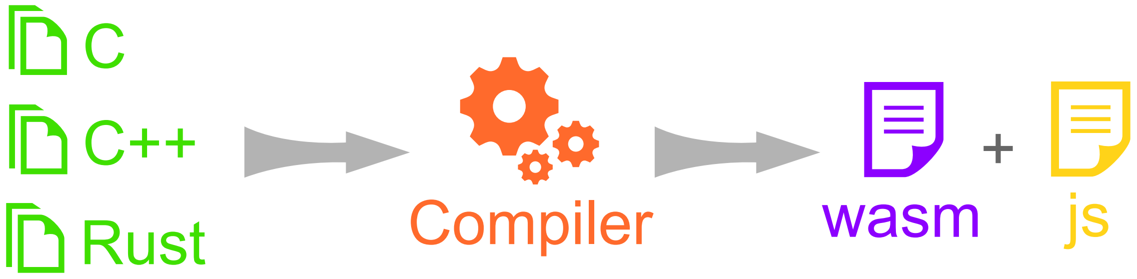
```
< ▼{global: {…}, env: {…}, asm2wasm: {…}, parent: {…}, global.Math: Math} ℹ
   ▶asm2wasm: {f64-rem: ƒ, f64-to-int: ƒ, i32s-div: ƒ, i32u-div: ƒ, i32s-rem: ƒ, …}
   ▼env:
      ABORT: 0
      DYNAMICTOP_PTR: 78768
      STACKTOP: 78784
      STACK_MAX: 5321664
     ▶abort: ƒ abort(what)
     ▶abortOnCannotGrowMemory: ƒ abortOnCannotGrowMemory()
     ▶abortStackOverflow: ƒ abortStackOverflow(allocSize)
     ▶assert: ƒ assert(condition, text)
     ▶enlargeMemory: ƒ enlargeMemory()
     ▶getTotalMemory: ƒ getTotalMemory()
     ▶invoke_d: ƒ invoke_d(index)
     ▶invoke_didi: ƒ invoke_didi(index,a1,a2,a3)
     ▶invoke_i: ƒ invoke_i(index)
     ▶invoke_ii: ƒ invoke_ii(index,a1)
     ▶invoke_iii: ƒ invoke_iii(index,a1,a2)
     ▶invoke_iiii: ƒ invoke_iiii(index,a1,a2,a3)
     ▶invoke_iiiii: ƒ invoke_iiiii(index,a1,a2,a3,a4)
     ▶invoke_iiiiii: ƒ invoke_iiiiii(index,a1,a2,a3,a4,a5)
     ▶invoke_ji: ƒ invoke_ji(index,a1)
     ▶invoke_jiji: ƒ invoke_jiji(index,a1,a2,a3,a4)
     ▶invoke_jj: ƒ invoke_jj(index,a1,a2)
     ▶invoke_v: ƒ invoke_v(index)
     ▶invoke_vi: ƒ invoke_vi(index,a1)
     ▶invoke_vidd: ƒ invoke_vidd(index,a1,a2,a3)
     ▶invoke_vii: ƒ invoke_vii(index,a1,a2)
     ▶invoke_viid: ƒ invoke_viid(index,a1,a2,a3)
     ▶invoke_viii: ƒ invoke_viii(index,a1,a2,a3)
     ▶invoke_viiidd: ƒ invoke_viiidd(index,a1,a2,a3,a4,a5)
     ▶invoke_viiii: ƒ invoke_viiii(index,a1,a2,a3,a4)
     ▶invoke_viiiii: ƒ invoke_viiiii(index,a1,a2,a3,a4,a5)
     ▶invoke_viiiiii: ƒ invoke_viiiiii(index,a1,a2,a3,a4,a5,a6)
     ▶invoke_viiij: ƒ invoke_viiij(index,a1,a2,a3,a4,a5)
     ▶invoke_viiiji: ƒ invoke_viiiji(index,a1,a2,a3,a4,a5,a6)
     ▶memory: Memory {}
      memoryBase: 1024
     ▶nullFunc_d: ƒ nullFunc_d(x)
     ▶nullFunc_didi: ƒ nullFunc_didi(x)
     ▶nullFunc_i: ƒ nullFunc_i(x)
     ▶nullFunc_ii: ƒ nullFunc_ii(x)
     ▶nullFunc_iii: ƒ nullFunc_iii(x)
     ▶nullFunc_iiii: ƒ nullFunc_iiii(x)
     ▶nullFunc_iiiii: ƒ nullFunc_iiiii(x)
```

```
let memory = new WebAssembly.Memory({
    initial: 1024, maximum: 10240
});
```



```
let HEAP8    = new Int8Array(memory.buffer);
let HEAP16   = new Int16Array(memory.buffer);
let HEAP32   = new Int32Array(memory.buffer);

let HEAPU8   = new Uint8Array(memory.buffer);
let HEAPU16  = new Uint16Array(memory.buffer);
let HEAPU32  = new Uint32Array(memory.buffer);

let HEAPF32  = new Float32Array(memory.buffer);
let HEAPF64  = new Float64Array(memory.buffer);
```

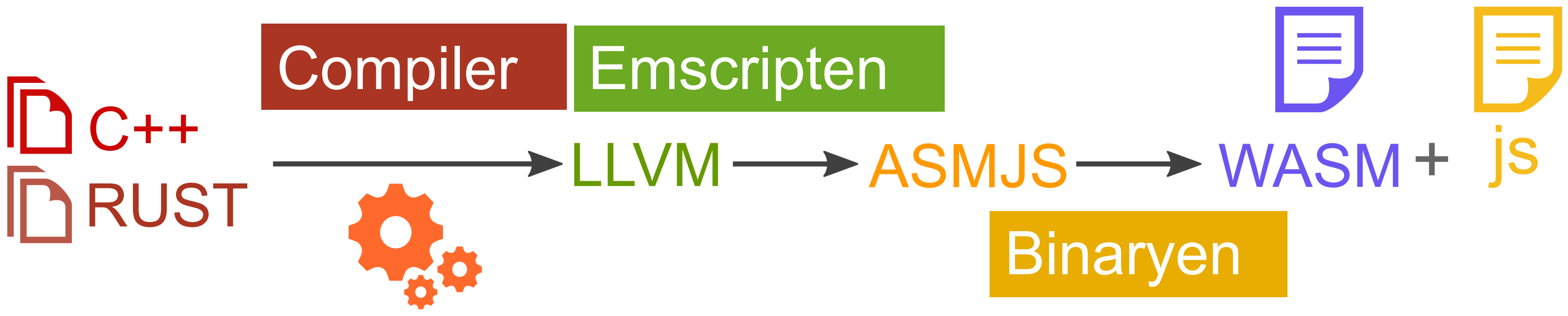
js wasm



wasm开发

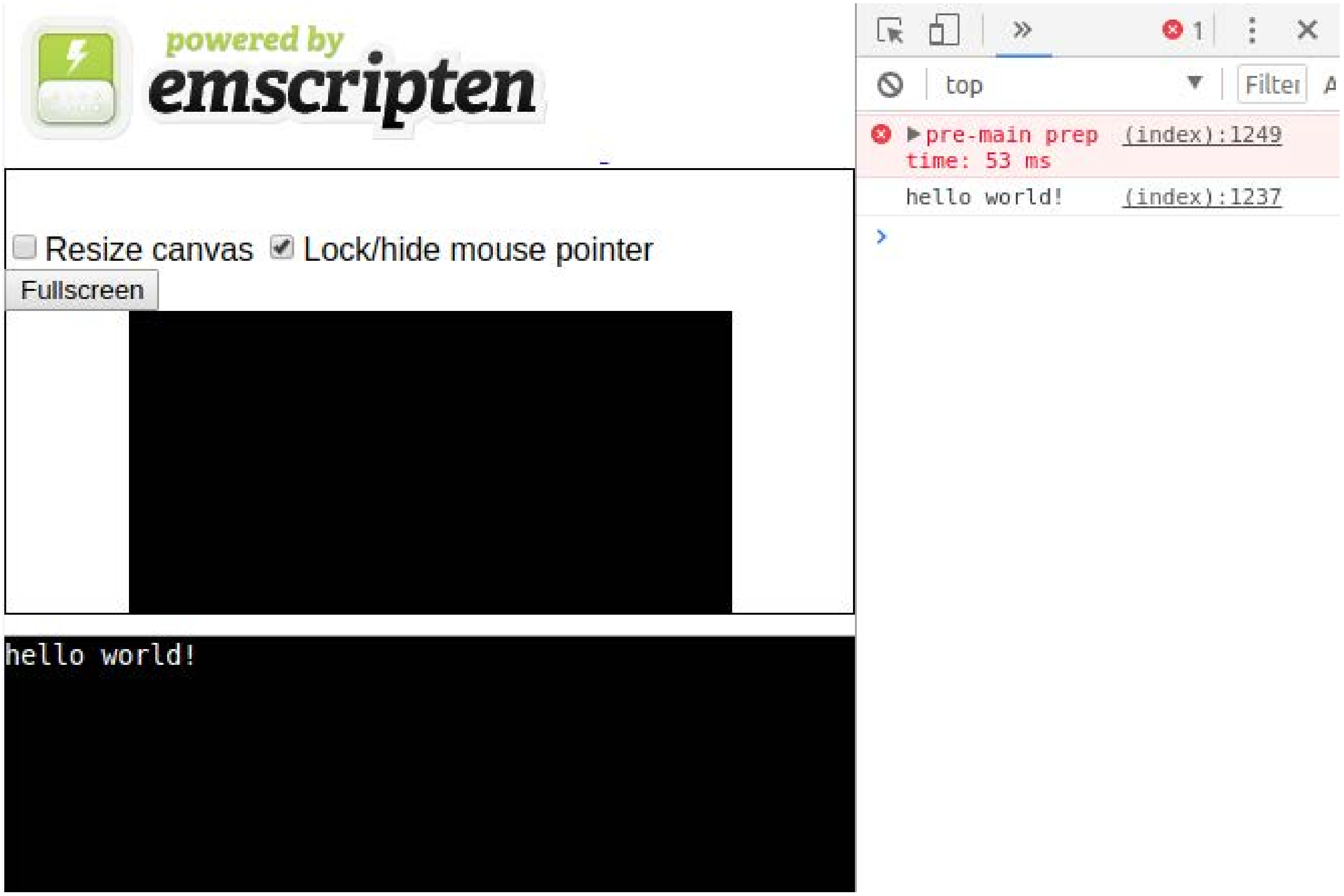
emscripten

An LLVM-to-JavaScript Compiler



```c
#include <stdio.h>

int main()
{
    printf("hello world!");
    return 0;
}
```

```
emcc hello_world.c
        -o hello_world.html
        -s WASM=1
```



```
hello world!
```

```
pre-main prep time: 53 ms    (index):1249
hello world!                 (index):1237
```

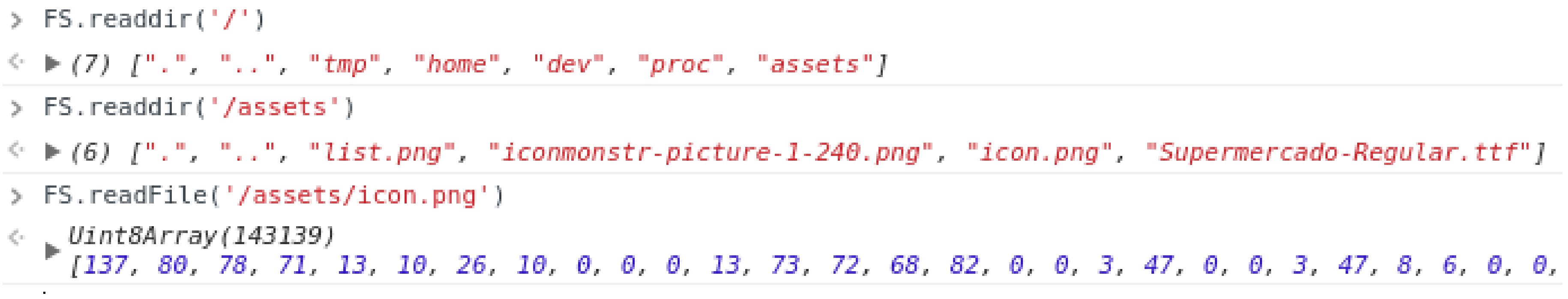
浏览器运行环境

- 异步
- 受限的IO

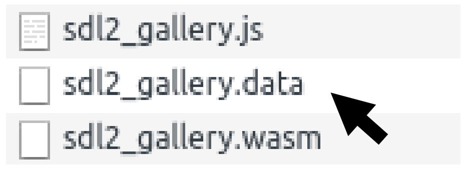
文件IO

```c
FILE *file = fopen("./file.txt", "rb");
while (!feof(file)) {
    char c = fgetc(file);
    if (c != EOF) {
        putchar(c);
    }
}
fclose(file);
```

FS API



```
> FS.readdir('/')
<· ▶ (7) [".", "..", "tmp", "home", "dev", "proc", "assets"]
> FS.readdir('/assets')
<· ▶ (6) [".", "..", "list.png", "iconmonstr-picture-1-240.png", "icon.png", "Supermercado-Regular.ttf"]
> FS.readFile('/assets/icon.png')
<·   Uint8Array(143139)
   ▶ [137, 80, 78, 71, 13, 10, 26, 10, 0, 0, 0, 13, 73, 72, 68, 82, 0, 0, 3, 47, 0, 0, 3, 47, 8, 6, 0, 0,
emcc --preload-file ./assets/
```



sdl2_gallery.js
sdl2_gallery.data
sdl2_gallery.wasm

网络IO

- BSD sockets API (WebSocket backend)

  listen failed: Not supported

- emscripten_async_wget

- emscripten_fetch

图形界面

- DOM
- Canvas
    - OpenGL ES，SDL，游戏引擎
    - Canvas App, Canvas Game

事件循环

```
while( true ) {
  while (SDL_PollEvent(&e)) {


  }
}
```

emscripten_set_main_loop
emscripten_pause_main_loop
emscripten_cancel_main_loop

```
int main(int argc, char *argv[])
{
  emscripten_set_main_loop(iter, 0, 1);
}

void iter()
{
  SDL_Event e;
  while (SDL_PollEvent(&e)) {

  }
}
```

requestAnimationFrame

使用C库

```
> emcc --show-ports
Available ports:
    zlib (USE_ZLIB=1; zlib license)
    libpng (USE_LIBPNG=1; zlib license)
    SDL2 (USE_SDL=2; zlib license)
    SDL2_image (USE_SDL_IMAGE=2; zlib license)
    ogg (USE_OGG=1; zlib license)
    vorbis (USE_VORBIS=1; zlib license)
    bullet (USE_BULLET=1; zlib license)
    freetype (USE_FREETYPE=1; freetype license)
    SDL2_ttf (USE_SDL_TTF=2; zlib license)
    SDL2_net (zlib license)
    Binaryen (Apache 2.0 license)
    cocos2d
```

```
emcc -s  USE_SDL=2 -s USE_SDL_IMAGE=2 main.cpp
```

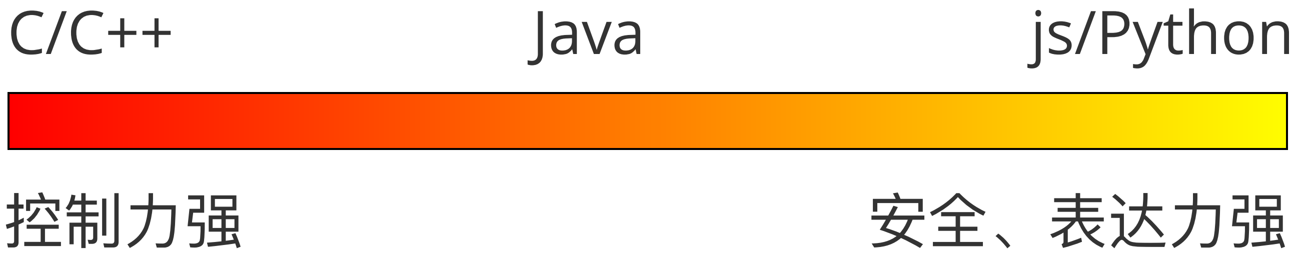
C/C++

Rust

Go

Rust是Mozilla发起的一个开源项目。

目标是构建高效，安全的系统级编程语言。


C/C++
Java
js/Python
控制力强
安全、表达力强

零成本抽象

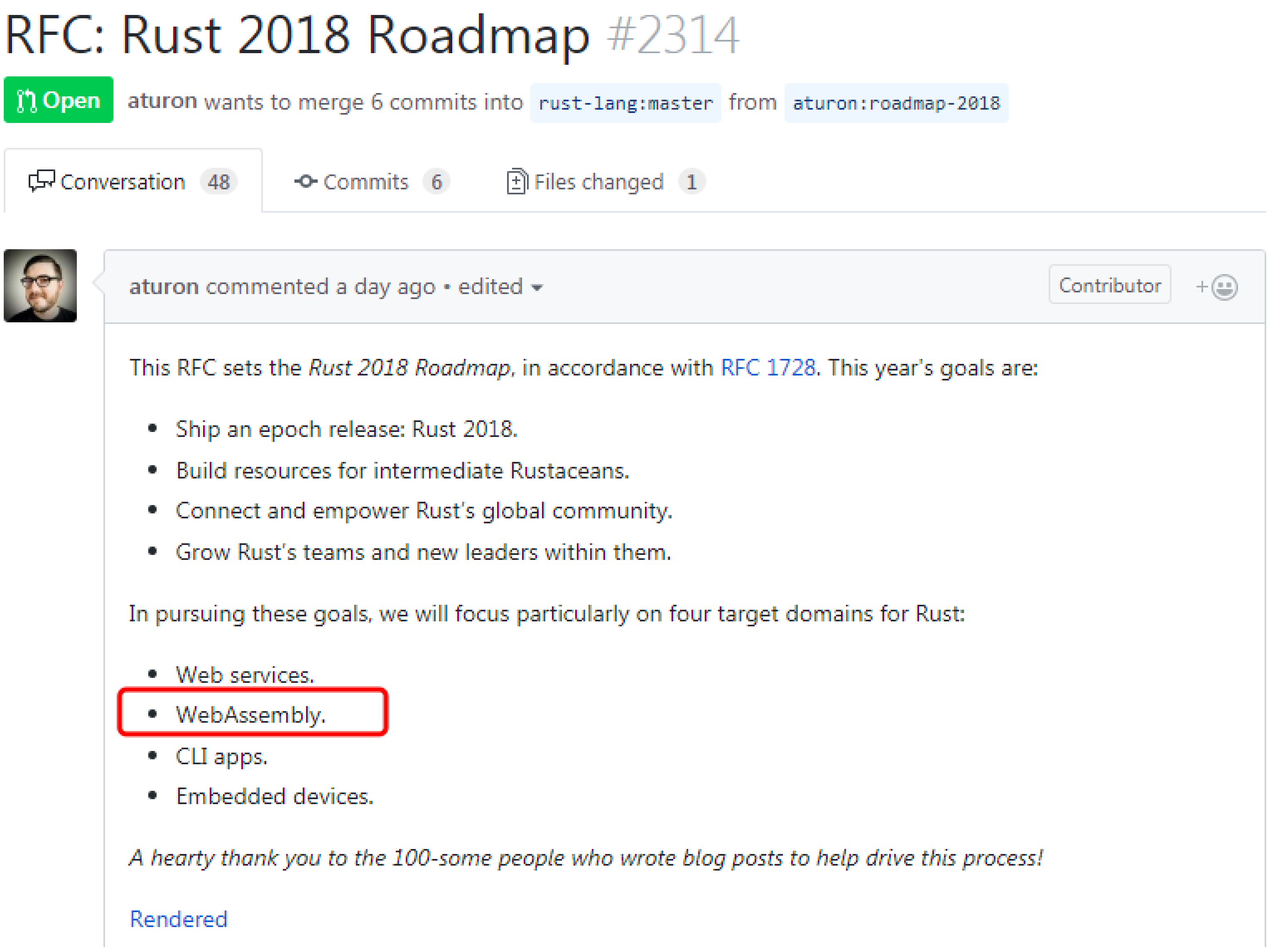
Iterator

```
(0..).map(|n| n * n)          ⇐ 取平方
    .filter(is_odd)           ⇐ 过滤
    .take(100)                ⇐ 取100个
    .fold(0, |acc, n| acc + n) ⇐ 加起来
```

Rust特性

- 性能与C/C++相当，无GC
- 类型推断
- 更现代的语法
- 独特的内存管理机制
- 无空指针
- 快速的发布周期 （< 2个月）

...



RFC: Rust 2018 Roadmap #2314

aturon wants to merge 6 commits into `rust-lang:master` from `aturon:roadmap-2018`

aturon commented a day ago • edited

This RFC sets the *Rust 2018 Roadmap*, in accordance with RFC 1728. This year's goals are:

- Ship an epoch release: Rust 2018.
- Build resources for intermediate Rustaceans.
- Connect and empower Rust's global community.
- Grow Rust's teams and new leaders within them.

In pursuing these goals, we will focus particularly on four target domains for Rust:

- Web services.
- WebAssembly.
- CLI apps.
- Embedded devices.

A hearty thank you to the 100-some people who wrote blog posts to help drive this process!

Rendered

Rust wasm生态圈

stdweb： js rust交互，DOM操作

yew： MV*框架, 虚拟DOM

rust-sdl2： 硬件加速的2D画图API

rust-wasm-loader: webpack loader

wasm-bindgen: 对象绑定

serde: 序列化

future: 异步

stdweb

```
let message = "Hello, 世界!";
let result = js! {
    alert( @{message} );
    return 2 + 2 * 2;
};
```

传递数字，字符串，函数

stdweb

```
let man = Man {
    name: String::new("Tom"),
    age: 12,
};
js! {
    var man = @{man};
    console.log( man.name + " is " + man.age + " years old."
};
```

传递结构化数据

调用emscripten SDK方法

```
extern "C" {
    pub fn emscripten_async_wget(url: *const c_char,
                                 file: *const c_char,
                                 onload: em_str_callback_func,
                                 onerror: em_str_callback_func);
}


unsafe {
    let url = concat!("icon.png", "\0");
    let file = concat!("local_image.png", "\0");
    emscripten::emscripten_async_wget(url as *const _ as *const c_char,
                                      file as *const _ as *const c_char,
                                      onload, onerror);
}
```

```
> cargo new demo-app
```

```
[package]
name = "demo-app"
version = "0.1.0"
authors = ["python <python@MacBook.local>"]

[dependencies]
stdweb = "0.4.3"
```

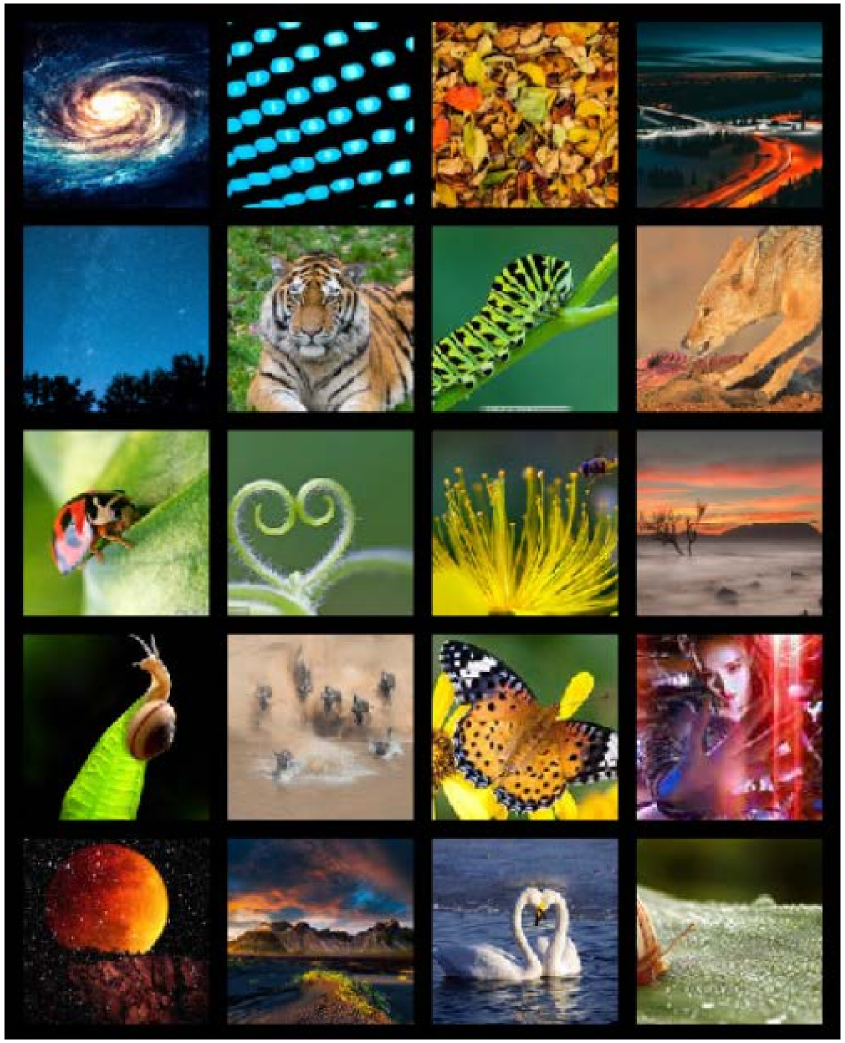
```
> tree .

.
├── Cargo.toml
└── src
    └── main.rs
```

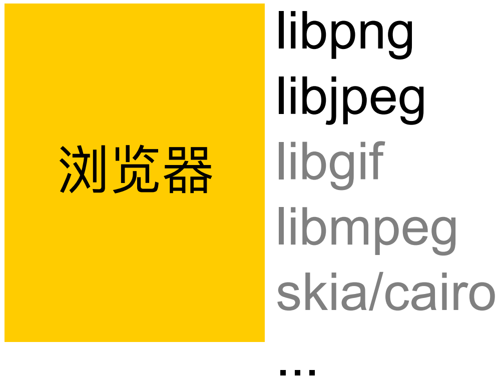
```rust
#[macro_use]
extern crate stdweb;

fn main() {
    stdweb::initialize();
    js! {
        alert("Hello from rust");
    }
    stdweb::event_loop();
}
```

```
> cargo build --target=wasm32-unknown-emscripten
```

demo-app.js demo_app.wasm

项目： 图片浏览器





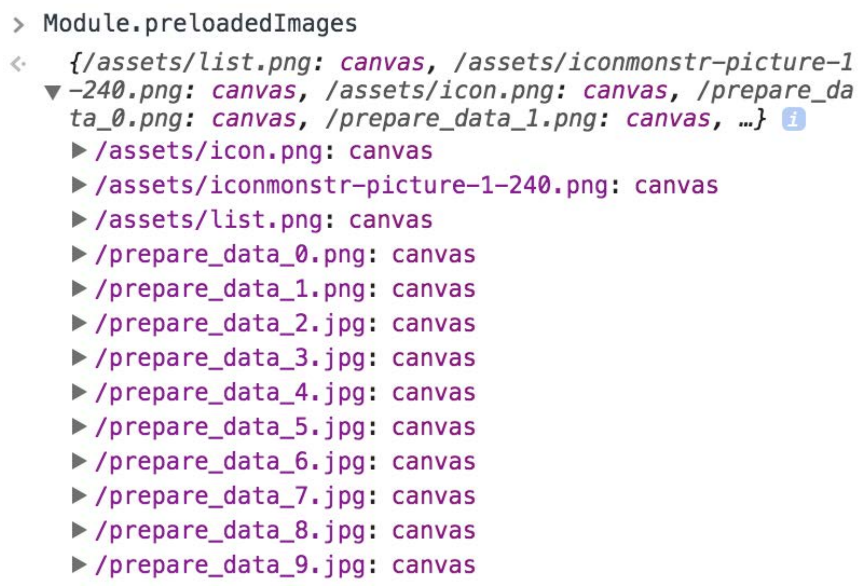
复用浏览器能力?

Preload Plugin

图片： jpg, png, bmp

音频： ogg, mp3, wav

emcc --use-preload-plugins



```
> Module.preloadedImages
<· ▼{/assets/list.png: canvas, /assets/iconmonstr-picture-1-240.png: canvas, /assets/icon.png: canvas, /prepare_data_0.png: canvas, /prepare_data_1.png: canvas, …} i
   ▶ /assets/icon.png: canvas
   ▶ /assets/iconmonstr-picture-1-240.png: canvas
   ▶ /assets/list.png: canvas
   ▶ /prepare_data_0.png: canvas
   ▶ /prepare_data_1.png: canvas
   ▶ /prepare_data_2.jpg: canvas
   ▶ /prepare_data_3.jpg: canvas
   ▶ /prepare_data_4.jpg: canvas
   ▶ /prepare_data_5.jpg: canvas
   ▶ /prepare_data_6.jpg: canvas
   ▶ /prepare_data_7.jpg: canvas
   ▶ /prepare_data_8.jpg: canvas
   ▶ /prepare_data_9.jpg: canvas
```

IMG_Load

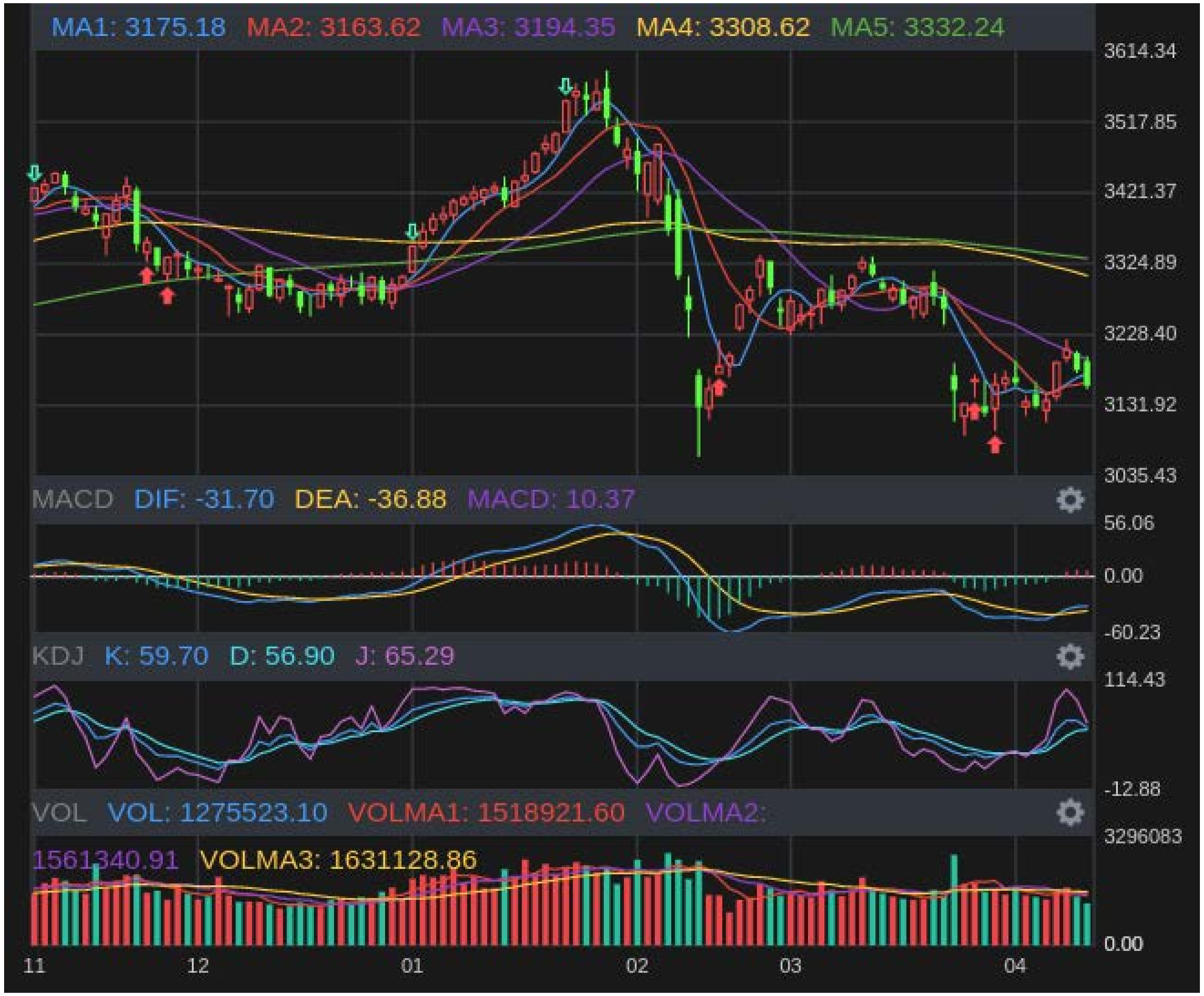
➥emscripten_get_preloaded_image_data

➥ ctx.getImageData

使用C库		使用浏览器能力	
增大打包体积	😨	体积小	😀
兼容性好	😀	浏览器only	😨
性能好	😀	跨语言调用成本高	😨

Q: 如何和现有的js结合?

项目： K线图形重构


MA1: 3175.18
MA2: 3163.62
MA3: 3194.35
MA4: 3308.62
MA5: 3332.24
3614.34
3517.85
3421.37
3324.89
3228.40
3131.92
3035.43
MACD
DIF: -31.70
DEA: -36.88
MACD: 10.37
56.06
0.00
-60.23
KDJ
K: 59.70
D: 56.90
J: 65.29
114.43
-12.88
VOL
VOL: 1275523.10
VOLMA1: 1518921.60
VOLMA2: 1561340.91
VOLMA3: 1631128.86
3296083
0.00
11
12
01
02
03
04

```
> var k = new KLine();
> k.update
ƒ update() { [native code] }

> var a = new Array();
> a.concat
ƒ concat() { [native code] }
```

```
fn main() {
     stdweb::initialize();
     js! {
         Module.Chart = Module.Chart || {};

         var KLine = function(node, t) {
             let ptr = @{kline::new}(node, t);
             Object.defineProperty(this, "ptr", {value: ptr});
         };

         KLine.prototype = {
             update: function(data) {
                 @{kline::update}(this.ptr, data);
             }
         );
         Module.Chart.KLine = KLine;
    }
    stdweb::event_loop();
}
```

wasm vue组件

```
{
    mounted() {
        this._chart = new KLine(this.$refs.root, "day");

        klineData({
          market: this.$props.market,
          code: this.$props.code,
          type: this.$props.type,
        }, ret => {
          this._chart.update(ret.data);
        });
    },
    destroyed() {
        this._chart.destroy();
    }
}
```

- wasm是可以和js完美结合的
- wasm和js相互调用涉及到写wrapper和指针操作

--release --release --release --release --release --release --release --release

--release --release --release --release --release --release --release --release

--release --release --release --release --release --release --release --release

--release --release --release --release --release --release --release --release

--release --release --release --release --release --release --release --release

--release --release --release --release --release --release --release --release

--release --release --release --release --release --release --release --release

--release --release --re cargo build --release ase --release --release

--release --release --release --release --release --release --release --release

--release --release --release --release --release --release --release --release

--release --release --release --release --release --release --release --release

--release --release --release --release --release --release --release --release

--release --release --release --release --release --release --release --release

--release --release --release --release --release --release --release --release

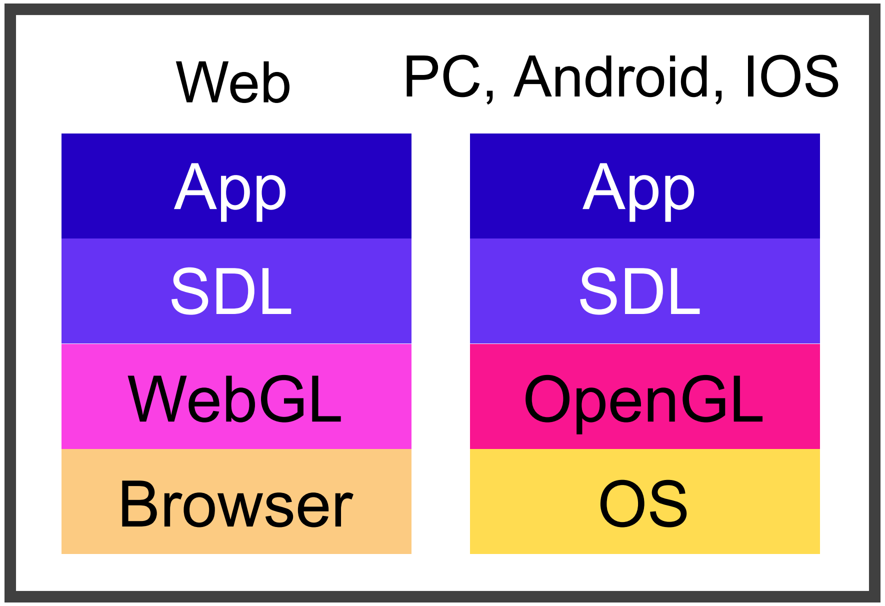
--release --release --release --release --release --release --release --release

总结 & 展望

wasm收益

- 游戏，视频
- 重CPU型程序， 加密，编解码
- webapp: virtual DOM diff
- node C++ addon
- Canvas App

Canvas App



跨平台，性能好

前端职业发展

- 一些C++/Rust程序员投身前端
- 继续提升前端开发的门槛
- 掌握一项原生技术势必成为web开发的必须技能！

Future WebAssembly

- Web API/DOM
- 多线程
- ES6 Module
- 垃圾回收
- SourceMap
- SIMD

...

QA

Thanks!