Advanced and Performant Web Apps

Taking advantage of multiple threads with Web Workers, executing binary modules on the web with Web Assembly and Rust as a systems programming language

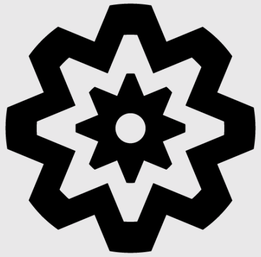
I'm **Alastair Paragas**

I build server-side, web, mobile and desktop apps with **Python**, **Scala**, **Java**, **Javascript**, **PHP**, **Haskell**, **Rust**, **Swift** and **C**. Fellow ACM FIU Alumni!

aparagas.com

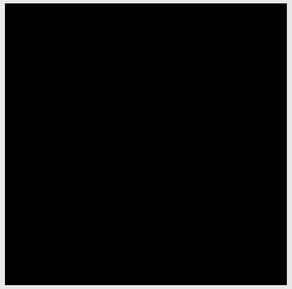
github.com/alastairparagas

devpost.com/alastairparagas



Web Workers

- Browser Javascript API
- Enables “Multi-threading”
- No DOM access
- Take advantage of processing parallelism

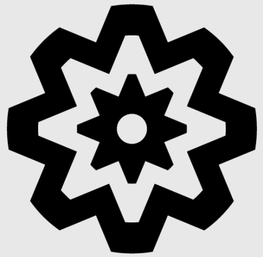


Rust

- Systems Programming Language
- Modern alternative to C and C++
- Developed by Mozilla, runs Mozilla’s browser engine - Servo

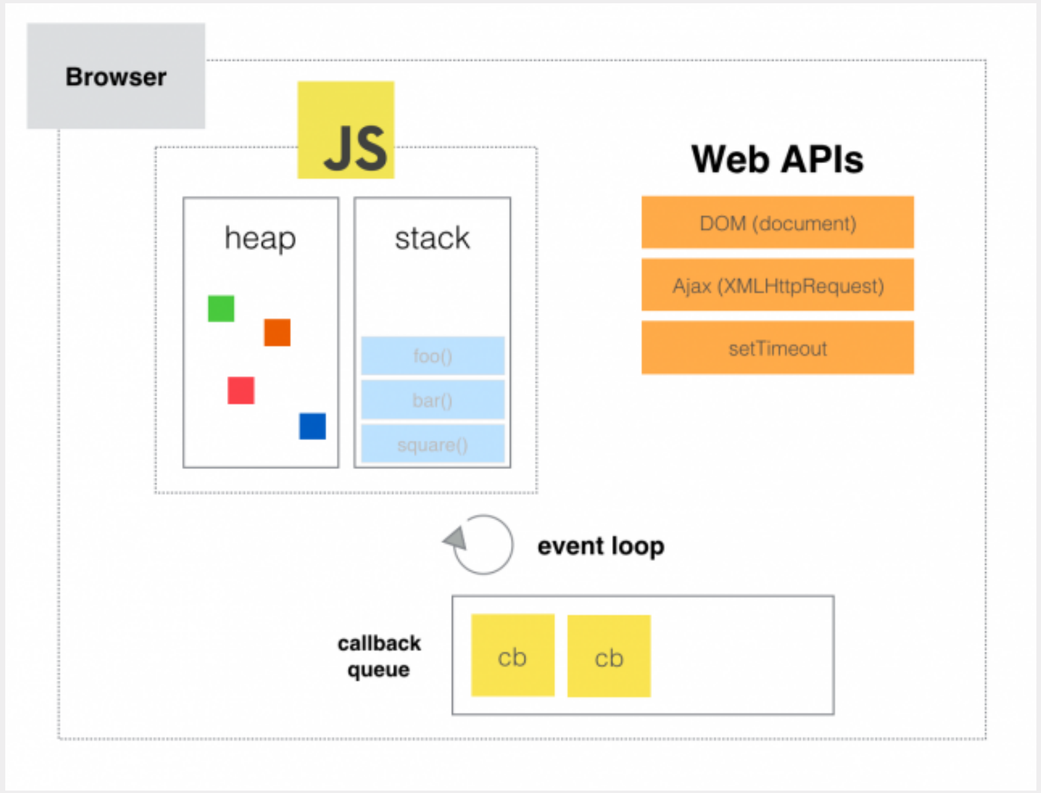
Web Assembly

- A binary module that gets executed on the browser with closer-to-the-metal performance than standard Javascript
- Assembly-like programs that are meant to be quickly downloaded and compiled on the fly



Web Workers

- Browser Javascript API
- Enables “Multi-threading”
- No DOM access
- Take advantage of processing parallelism



```
18 const armstrongNumberWorker = createWorker(() => {
19   let currentNumber = 1;
20
21   while (true) {
22     // Convert a number to a list of digits
23     const digitArray = currentNumber
24       .toString()
25       .split('')
26       .map(digitChar => Number(digitChar));
27
28     // Cube each digit and add them up
29     const cubedAddResult = digitArray
30       .map(digit => Math.pow(digit, 3))
31       .reduce((acc, current) => acc + current, 0);
32
33     if (cubedAddResult === currentNumber) {
34       // Send a message back to the main thread
35       self.postMessage(currentNumber);
36     }
37
38     currentNumber += 1;
39   }
40 });
```

Armstrong Numbers

Find all numbers whose digits cubed and summed equals the digit itself

$$1 = 1^3$$

$$153 = 1^3 + 5^3 + 3^3$$

$$370 = 3^3 + 7^3 + 0^3$$

```
19 v    let nthFibonacciNumber = (nthTerm) => {
20 v      if (nthTerm == 1) {
21          return 1;
22        }
23 v      else if (nthTerm == 2) {
24          return 1;
25        }
26
27        return nthFibonacciNumber(nthTerm - 1) +
28          nthFibonacciNumber(nthTerm - 2);
29      };
30
```

Fibonacci Numbers

The nth Fibonacci number is the sum

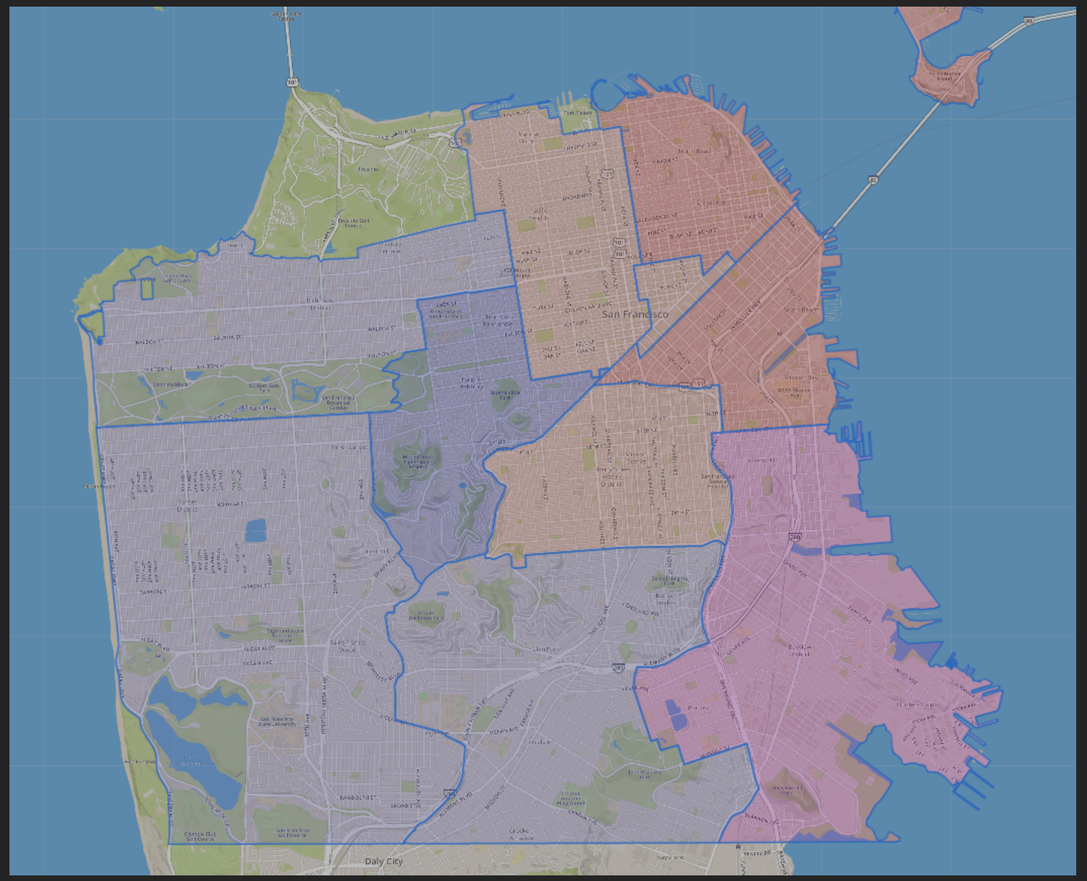
of the n-1 and the n-2 Fibonacci number

1st Fibonacci Number: 0

2nd Fibonacci Number: 1

3rd Fibonacci Number: 0 + 1 = 1

4th Fibonacci Number: 1 + 1 = 2



Real World Application

Ranking San Francisco Police Districts

by crime frequency since May 2018 and

clustering them into 5 cluster labels

Deep Blue - Least frequent areas

Deep Red - Most frequent areas

Web Assembly

- A binary module that gets executed on the browser with closer-to-the-metal performance than standard Javascript
- Assembly-like programs that are meant to be quickly downloaded and compiled on the fly

C++	Binary	Text

`int factorial(int n) {`
`  if (n == 0)`
`  return 1;`
`  else`
`  return n * factorial(n-1);`
`}`
	`20 00`
	`42 00`
	`51`
	`04 7e`
	`42 01`
	`05`
	`20 00`
	`20 00`
	`42 01`
	`7d`
	`10 00`
	`7e`
	`0b`
		`get_local 0`
		`i64.const 0`
		`i64.eq`
		`if i64`
		`  i64.const 1`
		`else`
		`  get_local 0`
		`  get_local 0`
		`  i64.const 1`
		`  i64.sub`
		`  call 0`
		`  i64.mul`
		`end`

```
(module
  (func additionOperation
    (param firstOperand i32)
    (param secondOperand i32)
    (result i32)
    get_local firstOperand
    get_local secondOperand
    i32.add)
  (export "add" (func additionOperation))
)

```

```
(module
  (func $additionOperation
    (param $firstOperand f32)
    (param $secondOperand f32)
    (result f32)
    get_local $firstOperand
    get_local $secondOperand
    f32.add)
  (export "add" (func $additionOperation))
)

```

32-bit Addition

Add two 32-bit integers together

vs

Adding two 32-bit floats together

Integer Addition:

1st and 2nd parameters can be any values between −2,147,483,648 to 2,147,483,647

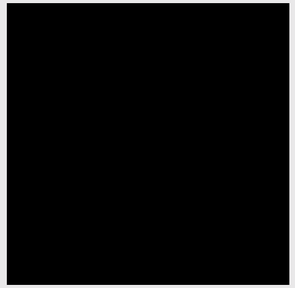
```
(module
 (export "fib" (func $fib))
 (func $fib (param $n i32) (result i32)
  (if
   (i32.lt_s
    (get_local $n)
    (i32.const 2)
   )
   (return
    (i32.const 1)
   )
  )
  (return
   (i32.add
    (call $fib
     (i32.sub
      (get_local $n)
      (i32.const 2)
     )
    )
    (call $fib
     (i32.sub
      (get_local $n)
      (i32.const 1)
     )
    )
   )
  )
 )
)
```

Fibonacci Sequence

The nth Fibonacci number is the sum of

the n-1th Fibonacci number and

n-2th Fibonacci number

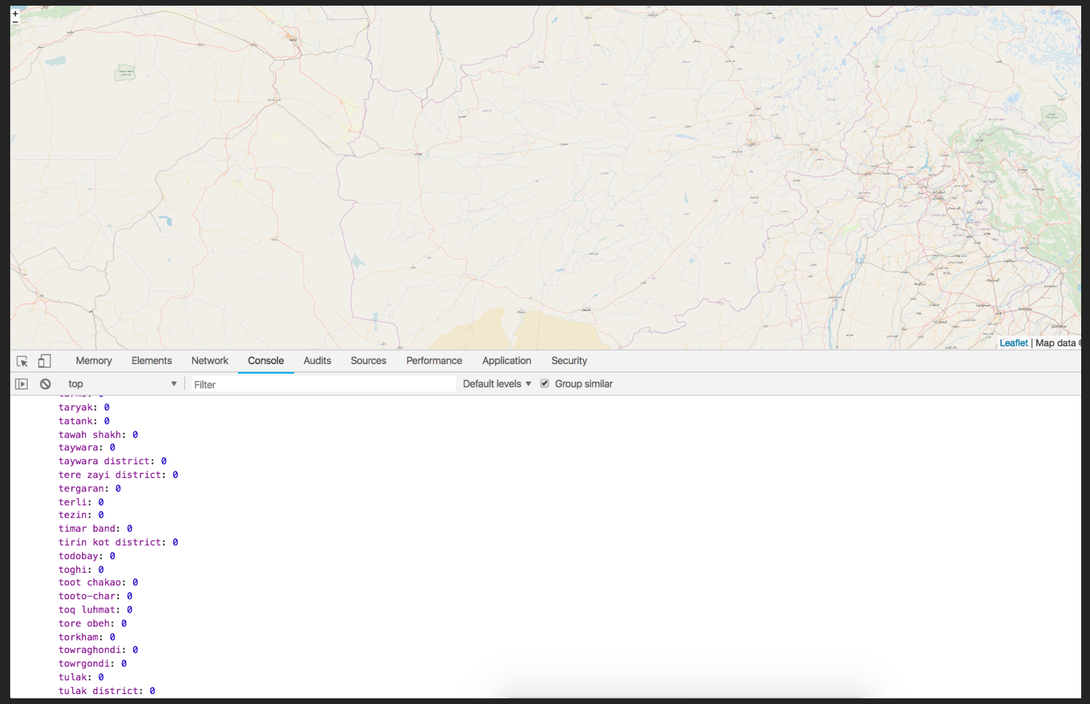


Rust

- Systems Programming Language
- Modern alternative to C and C++
- Developed by Mozilla, runs Mozilla's browser engine - Servo

```
let afghan_cities_incidents: HashMap<String, Vec<HashMap<&str, &str>>> = match row_titles.len() {
  0 => HashMap::new(),
  _ => file_contents_iterator
    .map(|line|
      row_titles
        .iter()
        .zip(line.split(','))
        .map(|(x, y)| (*x, y))
        .collect::<HashMap<&str, &str>>()
    )
    .filter(|incident_record| match incident_record.get("country") {
      Some(country) => *country == "4",
      None => false
    })
    .fold(HashMap::new(), |mut city_incidents, incident_record| match incident_record.get("city") {
      Some(city) => {
        let normalized_city = normalized_city_regex.replace_all(city, "").into_owned().to_lowercase();
        city_incidents
          .entry(normalized_city)
          .or_insert(vec![])
          .push(incident_record.clone());
        return city_incidents;
      },
      None => {
        city_incidents.insert(String::from("unclassified"), vec![]);
        return city_incidents;
      }
    })
};
```

```rust
extern crate wasm_bindgen;

use wasm_bindgen::prelude::*;

#[wasm_bindgen]
pub fn fibonacci(nth_term: u32) -> u32 {

  if nth_term == 1 {
    return 1;
  }
  if nth_term == 2 {
    return 2;
  }

  return fibonacci(nth_term - 1) + fibonacci(nth_term - 2);
}

```



Real World Application

Ranking cities in Afghanistan into clusters based on their levels of insurgency since 2008

1 - Safest areas

5 - Most dangerous areas