Rust y WebAssembly para JavaScripters

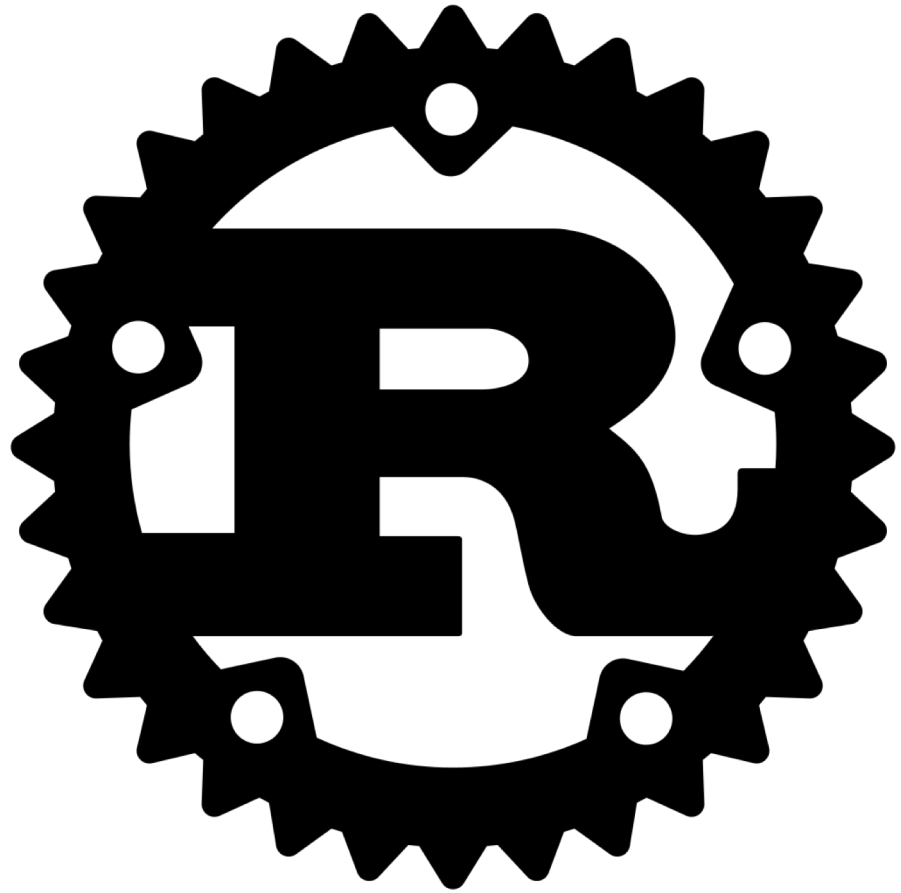
侍 Pablo Magaz
https://pablomagaz.com

github.com/pmagaz

twitter.com/pablo_magaz

Un lenguaje ultra rápido
Ultra eficiente
Ultra seguro

El futuro de la Web
Gran Performance
Compatibilidad APIS y JavaScript



No Garbage Collector

Control / Rendimiento

Seguridad

C / C++

Java / C#

Python / JavaScript

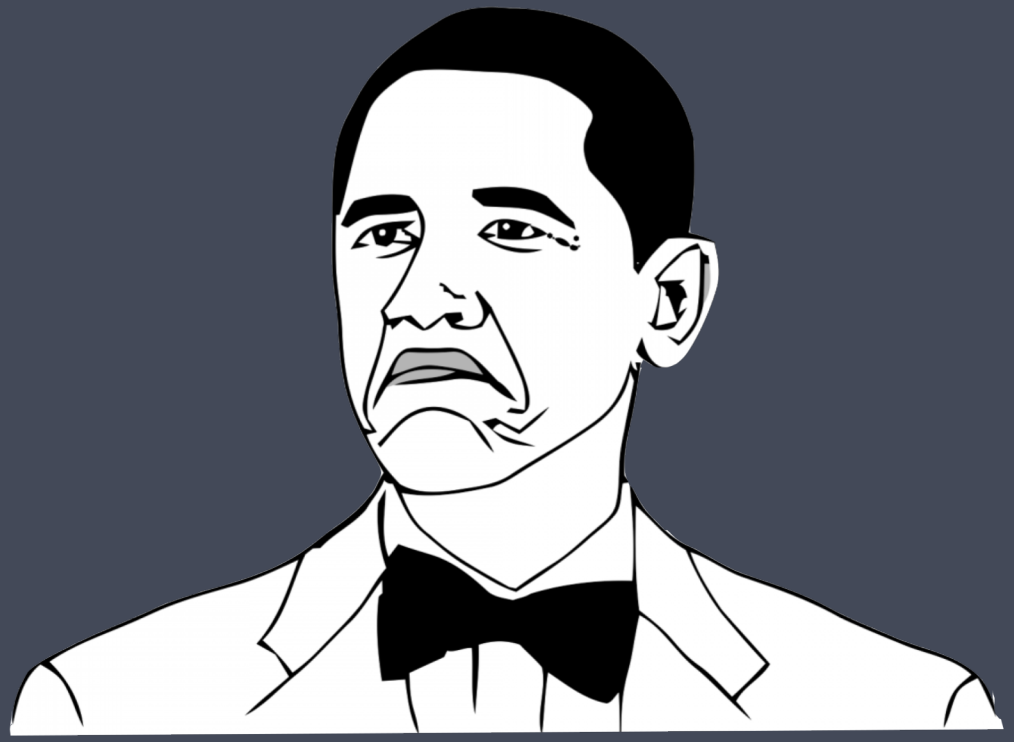
Control / Rendimiento

Seguridad

C / C++

Java / C#

Python / JavaScript



NOT BAD

Lo básico

侍 Variables

```
fn main(){
    let var;
    println!("Var value is {}", var); // Error!
    let var2 : i32 = 5;
    var2 = 2; // Error!
    let mut mutable_var : f32 = 5.5;
    mutable_var = 10.10;
    println!("New value: {}", mutable_var);
   // OUTPUT: New value: 10
}
```

侍 Tipos

```
fn main(){
    let mut vec: Vec<i32> = Vec::new();
    let my_string = String::from("Hello Commit!");
    let infered_number = 123;
    let  negative_number2: usize  = -123;
    let float_number3: f32  = 500.50;
}
```

侍 Funciones

```
fn square(x: i32) -> i32 {
    x * 2
}

fn main(){
    let number: i32  = 5;
    let square = square(number);
    println!("Square of {} is {}", number, square);
   // OUTPUT: Square of 5 is 10
}
```

Programación Orientada a Objetos

侍 Structs

```
struct Rectangle {
    width: i32
    height: i32
}

fn main(){
    let rectangle = Rectangle{width: 100, height: 200};
    println!("Width is {}", rectangle.width );
   // OUTPUT: Width is 100
}
```

侍 Structs methods

```
struct Rectangle {
    width: i32
    height: i32
}
Impl Rectangle {
    fn get_area(&self) -> i32 {
        self.width * self.height
    }
 }
...
println!("Area is {}", rectangle.get_area() );
```

Programación Funcional

侍 Programación Funcional

```
fn main(){
    let numbers =  vec![1,2,3,4,5,6];
    let even_numbers: usize  = numbers
        .into_iter()
        .take(4) // 1,2,3,4
        .filter(|n| n %2 == 0) // 2,4
        .count();

    println!("Total even numbers: {}", even_numbers);
     // OUTPUT: Total even numbers: 2
}
```

侍 Iteradores

```
fn main(){
    let my_vector =  vec![1,2,3];
    let mut my_iterator = my_vector.iter();
    println!("First Value is {}", my_iterator.next() );
     // OUTPUT: First Value is Some(1)

    for value in my_iterator {
        println!("Value: {}", value);
         // OUTPUT: Value: 2, Value: 3
    }
}
```

Ownership

侍 Ownership

```
fn main(){
    let str1 = String::from("El Blog Isomórfico");
    let str2 = str1; // El valor de str1 es MOVIDO a str2 y str1 es DESTRUIDO
    println!("Str1 : {}", str1); // ERROR: str1 ya NO existe
}
```

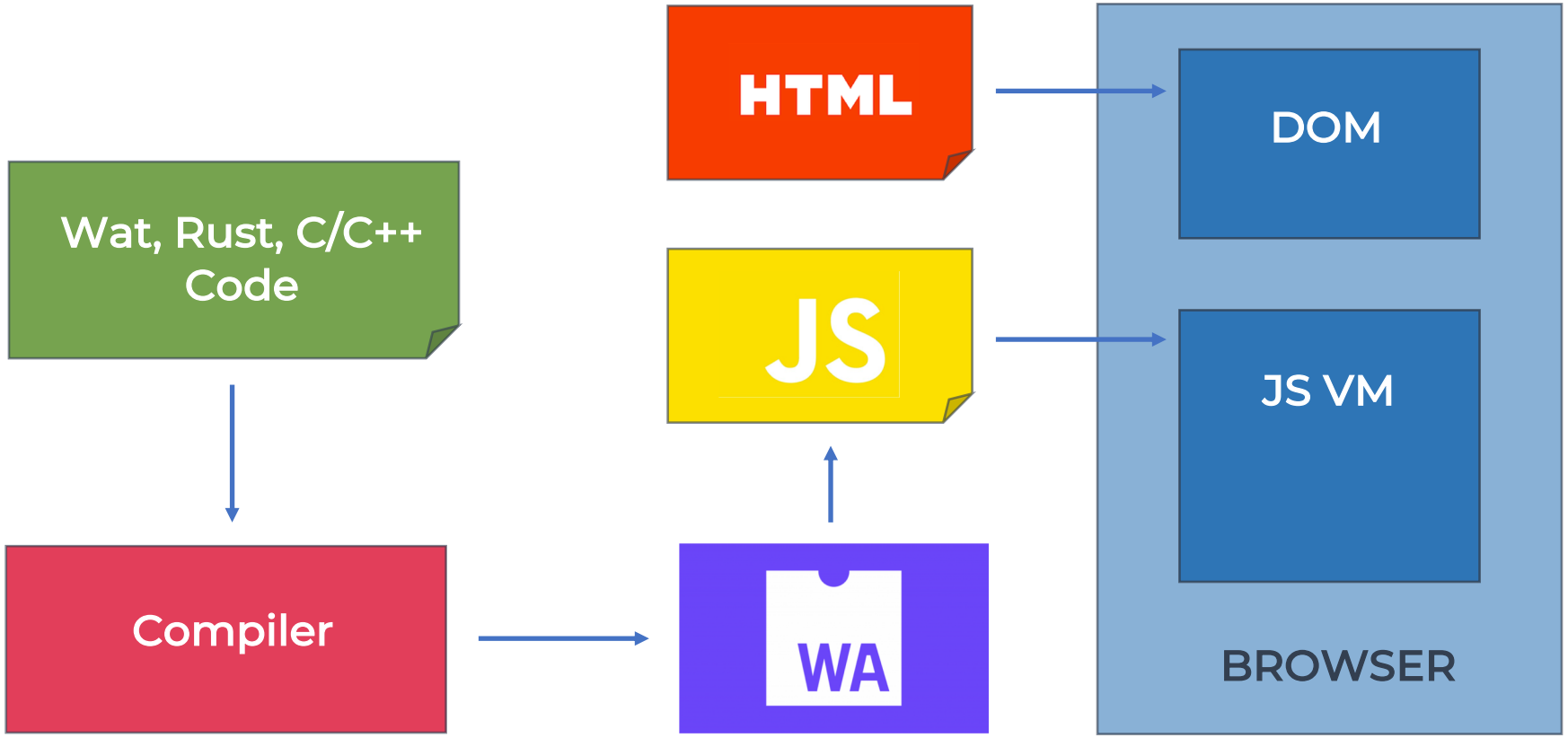
侍 Ownership

```
fn take_ownsership(my_str: String) -> String {
    my_str
}

fn main(){
    let str1 = String::from("El Blog Isomórfico");
    let value = take_ownership(str1); //  str1 es MOVIDO
    println!("Str1 : {}", str1); // ERROR: str1 ya NO existe
}
```

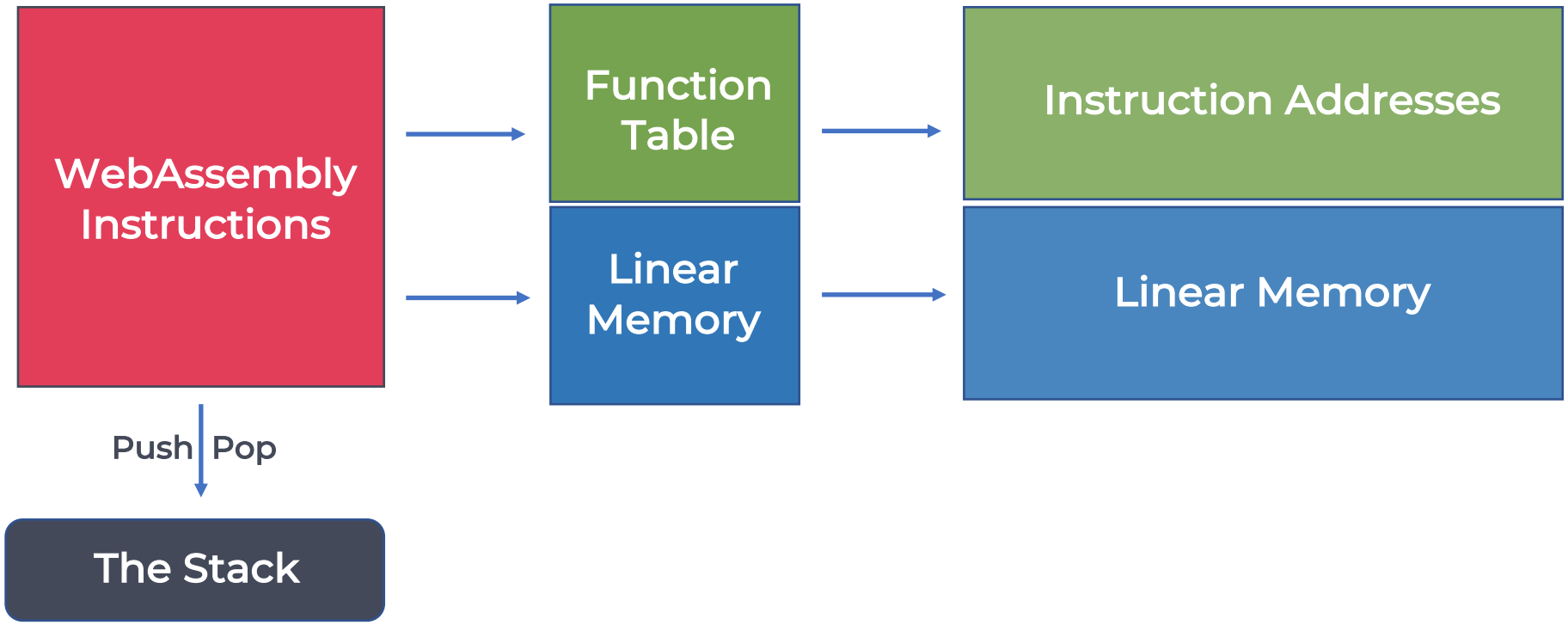
侍 Ownership

```
//  La función recibe una REFERENCIA a un String
fn take_a_reference(my_str: &str) -> &str {
    my_str
}

fn main(){
    let str1 = String::from("El Blog Isomórfico");
    let value = take_a_reference(&str1); //  se pasa la REFERENCIA
    println!("Str1 : {}", str1); //  str1 sigue existiendo!
    //OUTPUT El Blog Isomórfico
}
```

WA


HTML
DOM
Wat, Rust, C/C++ Code
JS
JS VM
Compiler
WA
BROWSER

WebAssembly Stack Machine



侍 WebAssembly Text Format (WAT)

```
(module
  (func multiply(param x i32) (param y i32) (result i32)
    get_local x
    get_local y
    i32.mul)
(export “multiply” (func multiply)))
```

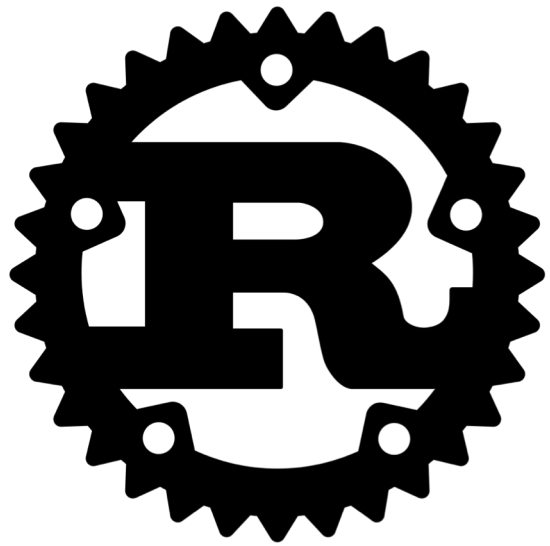
侍 Import WebAssembly Module

```
(async () => {
    const res = await fetch('main.wasm');
    const { instance } = await WebAssembly.instantiateStreaming(res);
    const { multiply } = instance.exports;
    const result = multiply(10,10);
    console.log(result);
    // OUTPUT: 100
})();
```

WA

90% faster

11000% faster...
20% slower...

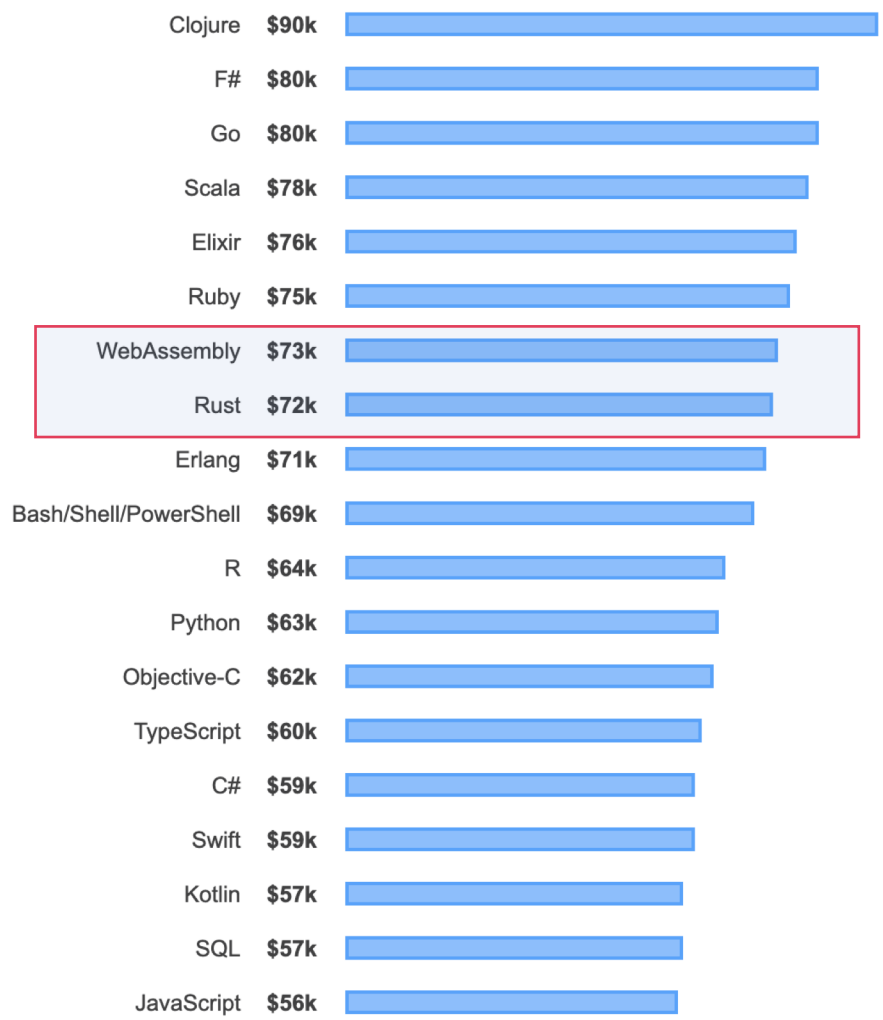



WA

stackoverflow

Top Paying Technologies

Global | United States



侍 Rust – JavaScript Crates

wasm-bindgen	Permite interacciones JavaScript-Rust y Rust-JavaScript
web-sys	Acceso a APIS como DOM, WebGL, console, etc
js-sys	Manejar los objetos globales de Js como Object, Date, etc
wasm-pack	Herramientas para compilación de Wasm y publicación en Npm


PARCEL
Blazing fast, zero configuration web application bundler
webpack
WA
JS

```rust
use wasm_bindgen::prelude::*;
use web_sys::console;

#[wasm_bindgen]
pub fn hello_world() {
    console::log_1(&"Hello Rust!".into());
}
```

```js
import { hello_world } from 'file.rs';

hello_world();
 // OUTPUT: Hello Rust!
```

```rust
use wasm_bindgen::prelude::*;

#[wasm_bindgen]
pub fn multiply(a: i32, b: i32) -> i32 {
    a * b
}
```

```js
import { multiply } from 'file.rs';

let result = multiply(10,10);
console.log(result);
 // OUTPUT: 100
```

```rust
#[wasm_bindgen]
pub struct RustStruct {
    id: i32
}

#[wasm_bindgen]
impl RustStruct {

    #[wasm_bindgen(constructor)]
    pub fn new(val: i32) -> Self {
        Self { id: val }
    }
    pub fn get_id(&self) -> i32 {
        self.id
    }
```

```js
import { RustStruct } from 'file.rs';

const rustStruct = RustStruct(12345);
const id = rustStruct .get_id();
console.log(id);
// OUTPUT: 12345
```

```rust
use wasm_bindgen::prelude::*;
use web_sys::window;

#[wasm_bindgen]
pub fn hello_dom(value: &str) -> Result<(), JsValue> {
    let window = window().expect("Window not found!");
    let document = window.document().expect("Document not found!");
    let body = document.body().expect("Body not found!");
    let div = document.create_element("div")?;
    div.set_inner_html(value);
    body.append_child(&div)?;
    Ok(())
}
```

Gracias.
¿Preguntas?

Slides: pablomagaz.com/#talks

 github.com/pmagaz  twitter.com/pablo_magaz