Uniwersytet Wrocławski

Rust i JavaScript w wyścigu o wydajność

Wiktor Zychla
Zakład Inżynierii Oprogramowania

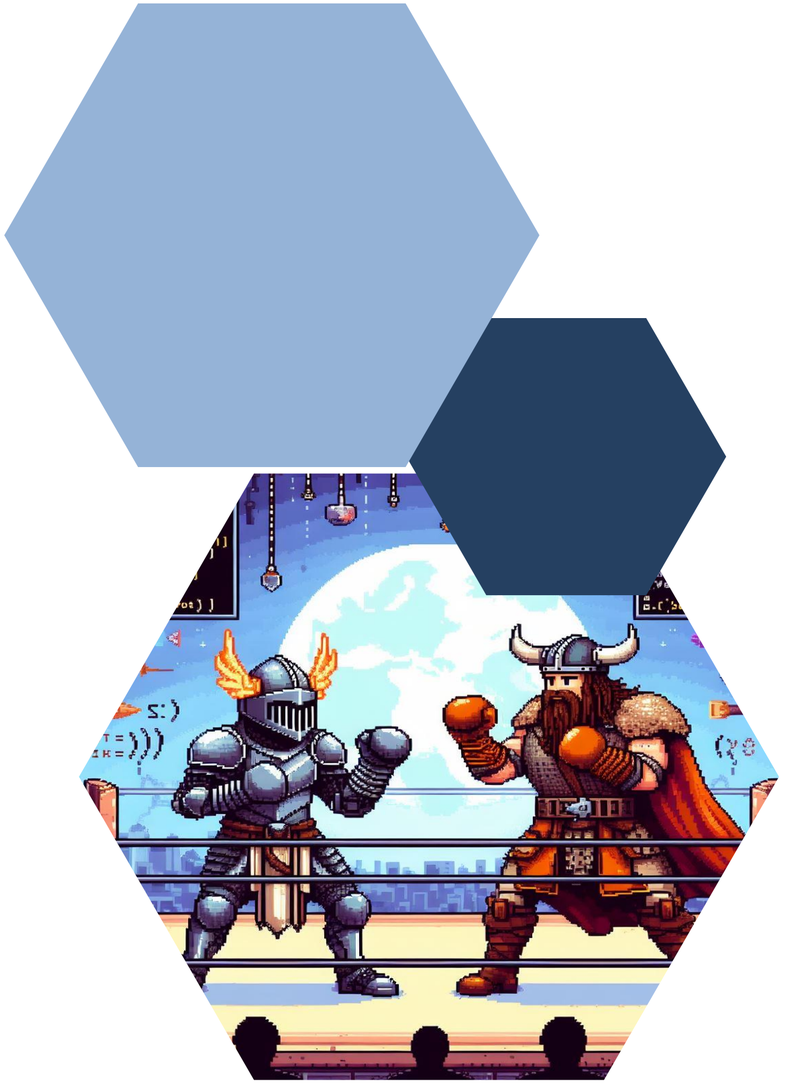
wiktor.zychla@uwr.edu.pl
https://github.com/wzychla
https://stackoverflow.com/users/941240/wiktor-zychla

Dla kogo i dlaczego

- Instytut Informatyki jest cenionym ośrodkiem naukowym, z uznanym dorobkiem w obszarach teorii języków programowania, algorytmiki czy data science
- Prowadzimy zajęcia we wszystkich kluczowych obszarach, w tym również (oprócz powyższych):
  - Sieci komputerowe
  - Grafika komputerowa
  - Inżynieria oprogramowania
  - Bazy danych

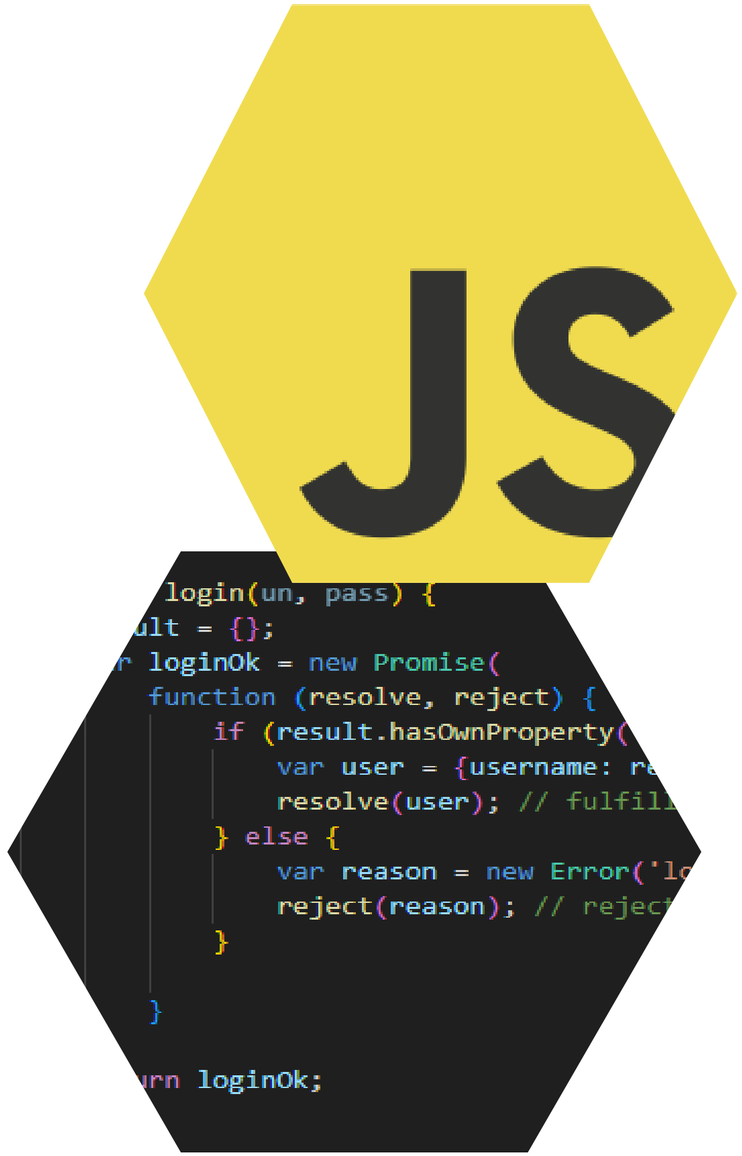
Plan

- Zawodnicy
  - Javascript
  - Rust
- Pojedynek
  - Ring / problem
  - Wyniki
- I co dalej?



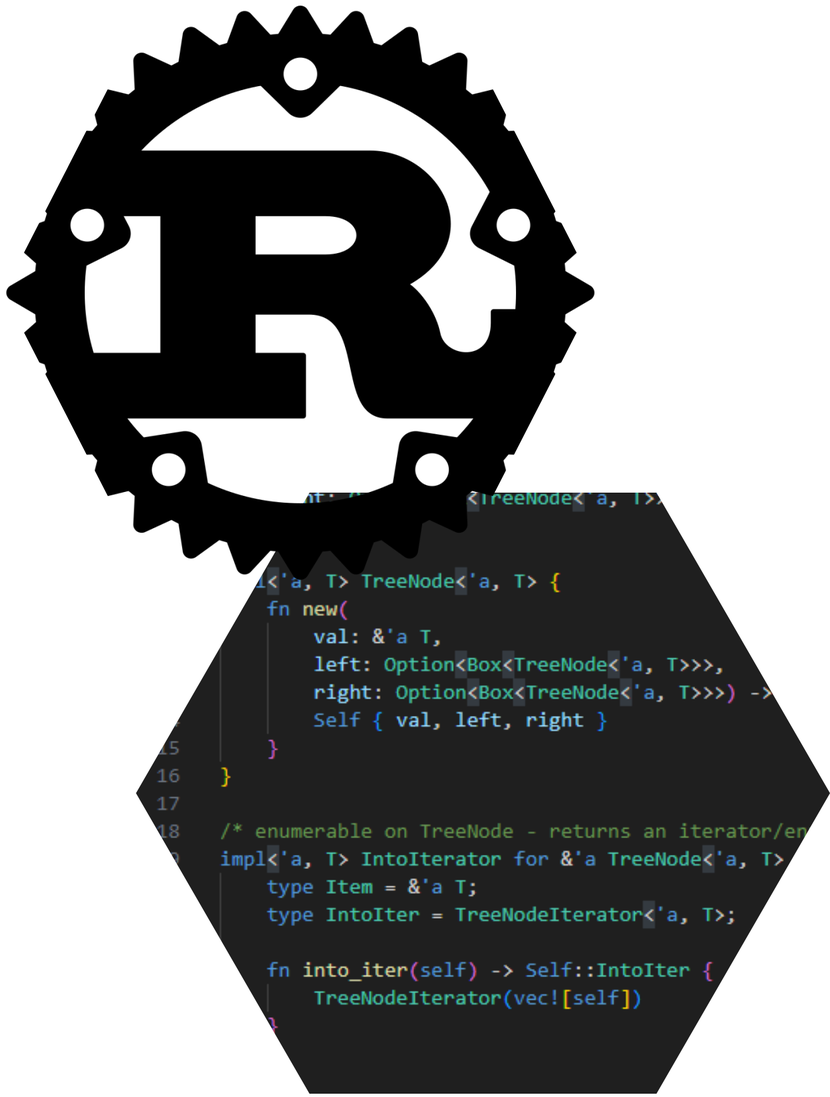
JavaScript – główne cechy

- Dynamiczne typowanie
  - Brak oznaczeń typów w kodzie
  - Brak kompilacji
  - Ale: istnieje TypeScript
- Hybryda
  - Elementy funkcyjne (m.in. Domknięcia)
  - Elementy obiektowe (obiektowość prototypowa)
- Wieloplatformowość
  - Przeglądarka, serwery, urządzenia mobilne
- Dawniej – interpretowany, obecnie – Just-in-Time



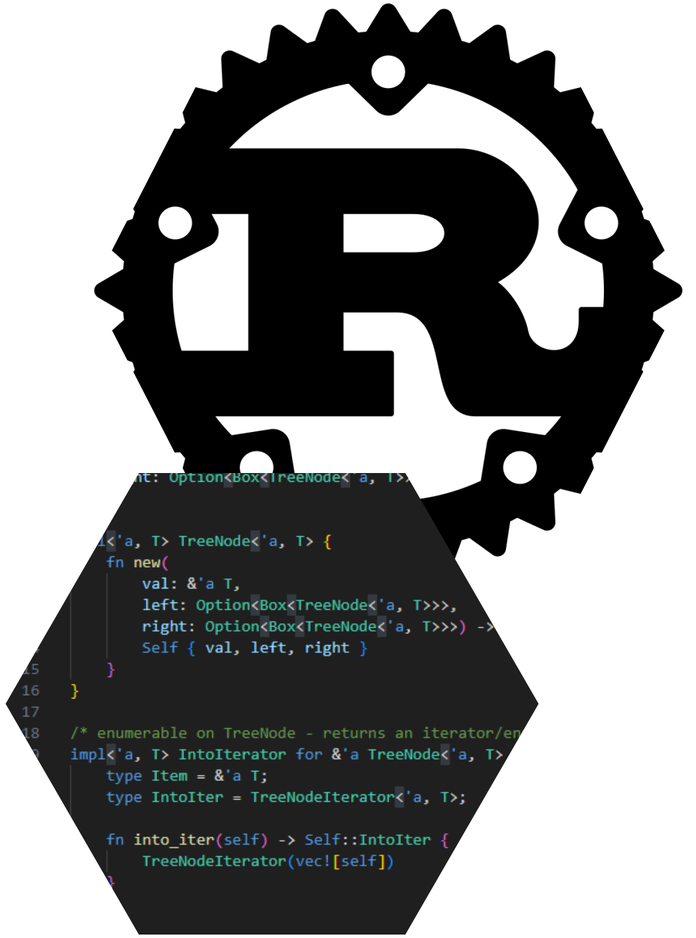
Rust – główne cechy

- Bezpieczeństwo
  - Silne statyczne typowanie, inferencja typów
  - Bezpieczeństwo pamięci *by-design*
  - Dostęp do pamięci w kodzie wielowątkowym
  - Brak *null* (zamiast tego – Option<T>)
- Efektywność
  - wydajność porównywalna z C/C++
  - *Zero-cost abstractions*
- Czytelna komunikacja
  - Kompilator podpowiada korekty literówek
- Ale: **trudniejszy do nauki niż wiele języków**



Rust w świecie

- 2021/2022/2023 StackOverflow most loved language
- 2021 NSA aktualizuje listę rekomendowanych języków (m.in. Rust/C#/Java/Python ale nie C/C++)
- 2022 Linux 6.1 dodaje Rust jako drugi język (obok C) do pisania komponentów jądra
- 2023 Microsoft ogłasza przepisanie kodu jądra Windows z C++ do Rust
- Rusta używa Facebook/Dropbox/Discord/Cloudflare
- Zastosowania w IoT/blockchain/itd.



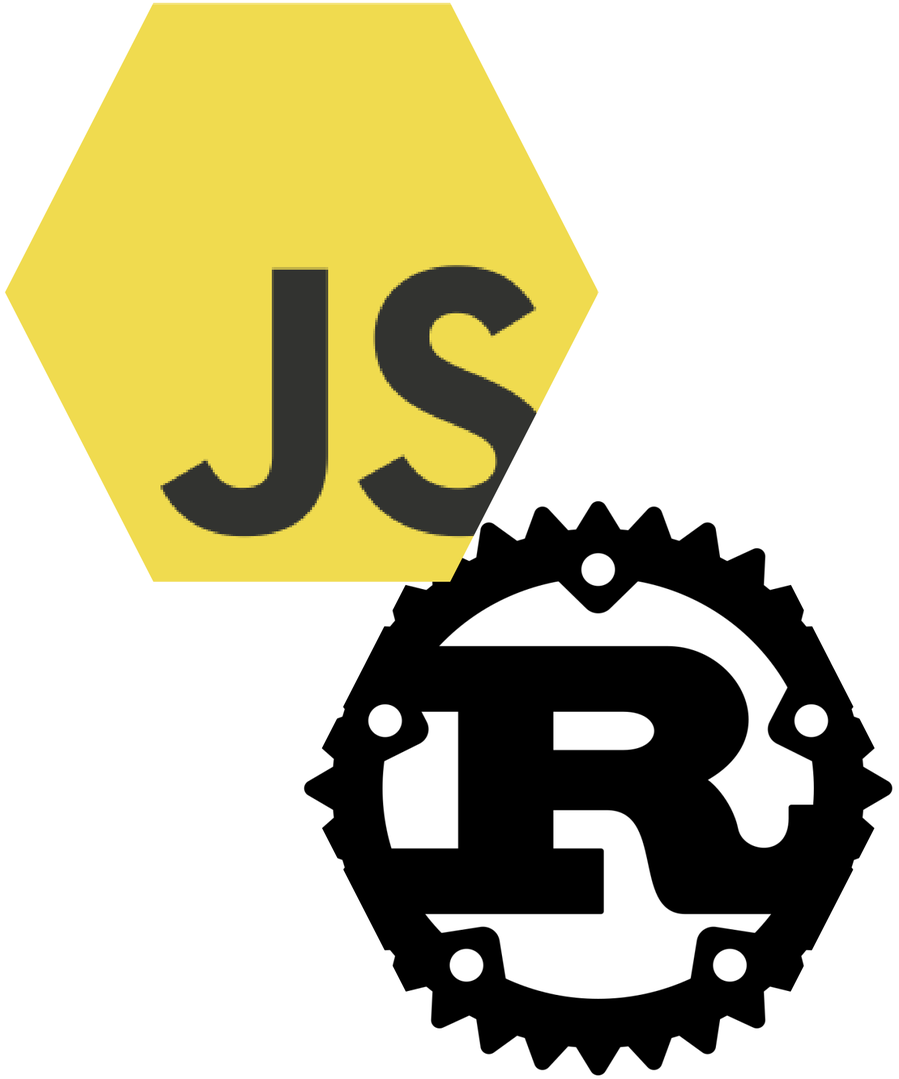
Hello Rust

- `cargo new hello-rust`
- `cargo build`
- `cargo run`

Ring

Jak badać wydajność w porównywalnych warunkach?

- Wybór hosta
  - Konsola
  - Przeglądarka internetowa
- Wybór problemu
  - Jakieś zagadnienie obliczeniowe



Ring

Dwa silniki obliczeniowe w przeglądarce

- V8/SpiderMonkey/JavaScriptCore (JavaScript)
- WASM – silnik kodu binarnego maszyny stosowej (2019+)

WASM jest łatwym celem kompilacji różnych języków, m.in. C/C++/Rust (Emscripten/LLVM)

Poza naszym konkursem: Emscripten umożliwia kompilację C/C++ zarówno do WASM jak i do Javascript. W ten sposób można skompilować nawet np. jądro Linuksa/Windows czy emulować cokolwiek.

Przykład chaosu

Prosta funkcja iteracyjna

$$x_{n+1} = 4 * x_n * (1 - x_n)$$

prowadzi do zachowania chaotycznego dla bliskich wartości początkowych, z przedziału [0, 1]
Początkowo nic tego nie zapowiada

xa=0.2000000, xb=0.2000001, błąd=0.0000001
xa=0.6400000, xb=0.6400002, błąd=0.0000002
xa=0.9216000, xb=0.9215997, błąd=0.0000003
xa=0.2890138, xb=0.2890147, błąd=0.0000009
xa=0.8219392, xb=0.8219408, błąd=0.0000015
...

ale po 20-tej iteracji

...
xa=0.8708927, xb=0.0382300, błąd=0.8326627
xa=0.4497544, xb=0.1470737, błąd=0.3026806
xa=0.9899015, xb=0.5017722, błąd=0.4881293
xa=0.0399860, xb=0.9999874, błąd=0.9600014
xa=0.1535486, xb=0.0000503, błąd=0.1534983
...

Liczby zespolone - przypomnienie

Zbiór **liczb zespolonych** to takie rozszerzenie zbioru **liczb rzeczywistych** w którym liczby są postaci

$$a + bi$$

gdzie

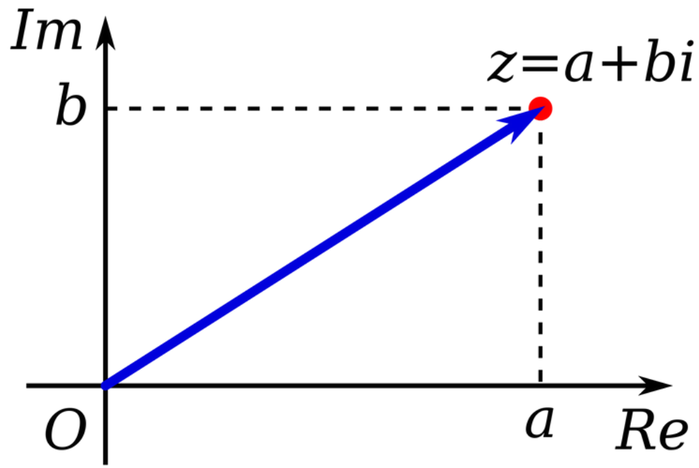
$$i^2 = -1$$

Liczby zespolone można dodawać, odejmować, mnożyć i dzielić czy liczyć ich "długość". Zbiór liczb zespolonych jest **ciałem**

$$(a + bi) + (c + di) = (a + c) + (b + d)i$$

$$(a + bi) * (c + di) = (ac - bd) + (ad + bc)i$$

$$|(a + bi)| = \sqrt{a^2 + b^2}$$



https://en.wikipedia.org/wiki/Complex_number#/media/File:A_plus_bi.svg

Zbiory Julii i Mandelbrota

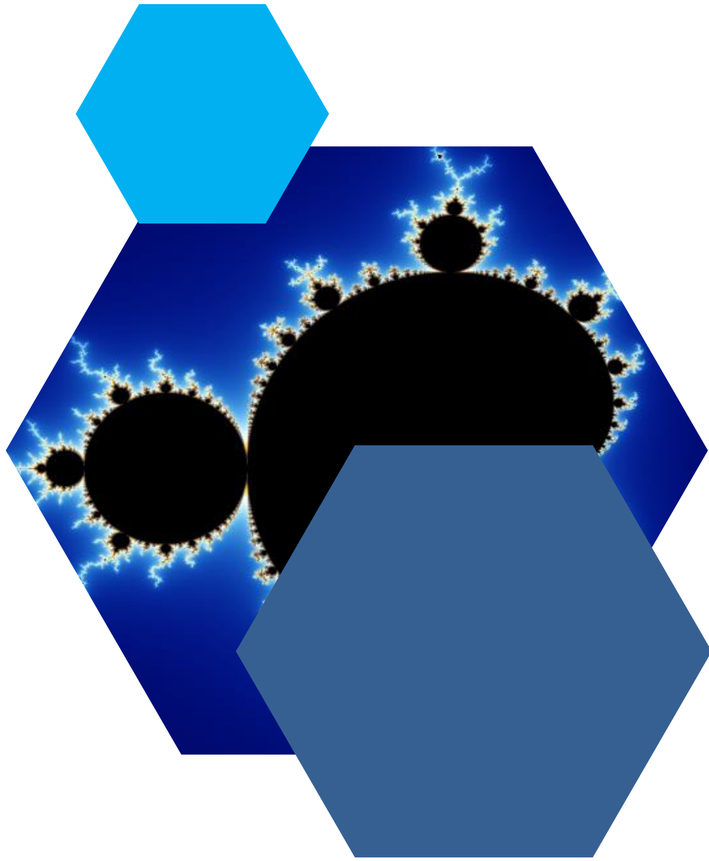
Na początku XX w. **Gaston Julia** i **Pierre Fatou** badali iterowanie funkcji analitycznych

$$f_c(x) = \begin{cases} x_0 = \text{współrzedne punktu na płaszczyźnie zespolonej} \\ x_{n+1} = x_n * x_n + c \end{cases}$$

Obserwowali, że dla pewnych, bardzo bliskich sobie punktów początkowych, iterowanie tej funkcji albo zbiega do zera albo jest rozbieżne.

Czy zbiór punktów płaszczyzny zespolonej dla których iterowanie tej funkcji jest zbieżne można zwizualizować?

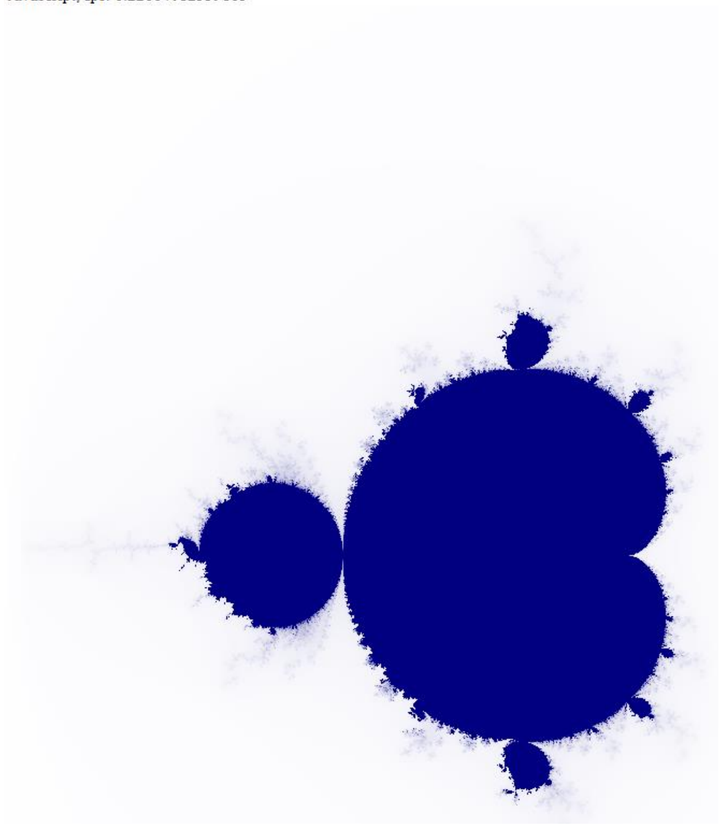
$$K(f_c) = \{x \in C : \forall n \in N, |f_c^n(x)| < 2\}$$



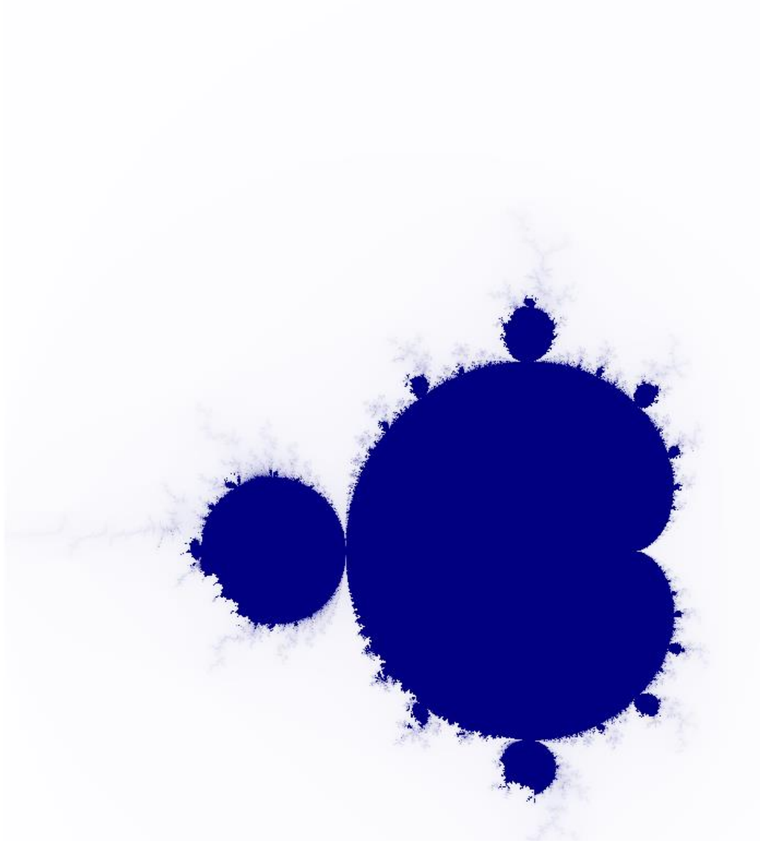
Pojedynek

https://github.com/wzychla/rust-vs-js

Javascript, fps: 8.22884012539185



Rust2WASM, fps: 11.345541401273886



Wnioski/obserwacje

- Rust/WASM **11-14 fps** vs JavaScript **7-9 fps** (i7-1165G7 @ 2.80GHz)
- Czas transferu mapy bitowej 1024x1024 na canvas jest pomijalny
- Dalsze optymalizacje - zastosowanie arytmetyki stałoprzecinkowej lub wręcz przeniesienie obliczeń na GPU (60 fps!)
- Tych (i wielu innych) języków i technologii można nauczyć się w Instytucie

Dziękuję za uwagę

wiktor.zychla@uwr.edu.pl