IT po mojemu

Jak stworzyć swój własny język programowania?

Czyli im dalej w las, tym więcej Drzewek...

Dariusz Jędrzejczak
darius.j.pl

początki

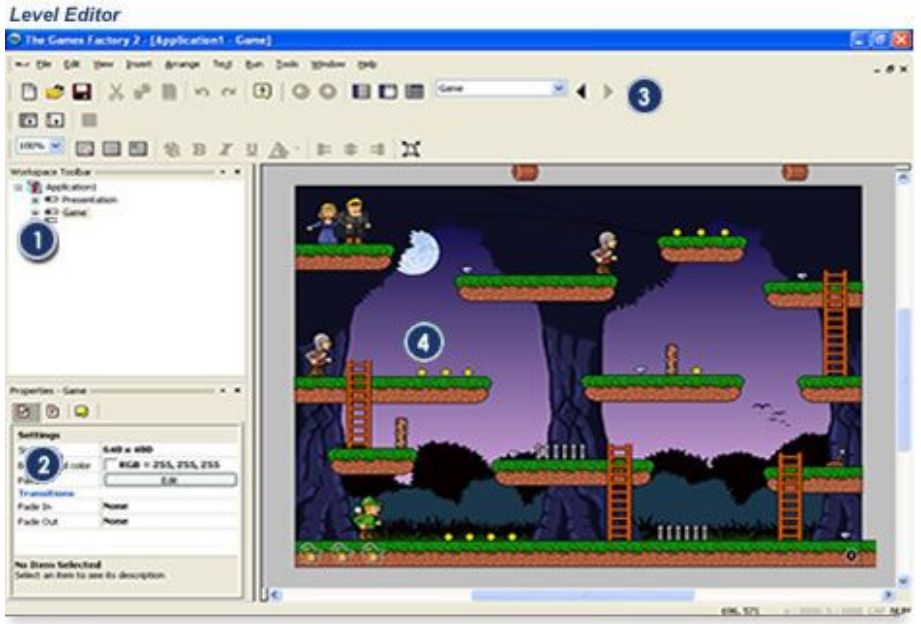
- pierwszy komputer:
  - druga połowa lat 90.
  - Apple IIc
  - kilkunastoletni, przytargany z Niemiec
  - magiczny

początki

- wczesne oznaki kreatywności:
  - bazgranie komiksów
- ważne dla rozwoju:
  - podlewanie takich przejawów

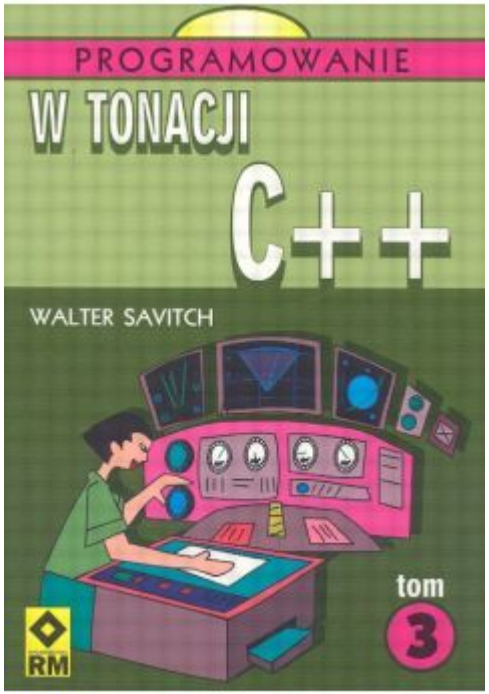
początki

- pierwsze stworzone oprogramowanie:
  - gry w The Games Factory
  - m.in. na podstawie komiksów
  - początek milenium



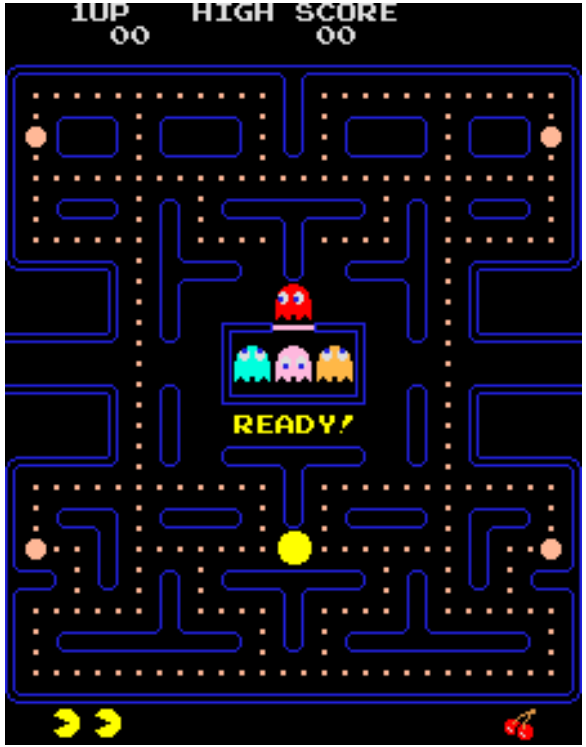
początki

- pierwszy język programowania:
  - C++
  - parę lat później



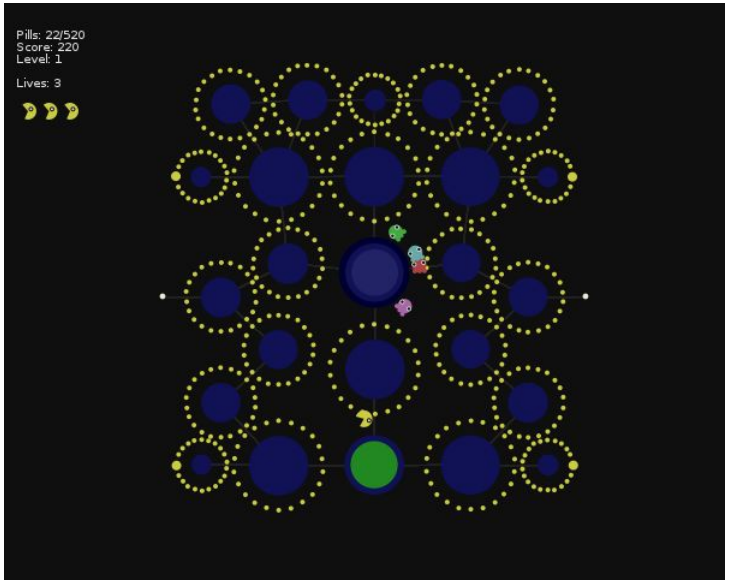
nauka programowania

- motywacja do nauki programowania:
  - chęć tworzenia gier
- nauka dzięki internetowej społeczności gamedev
  - konkursy robienia gier na czas



nauka programowania

- wykształcenie: Politechnika Łódzka
  - symulacje i gry komputerowe

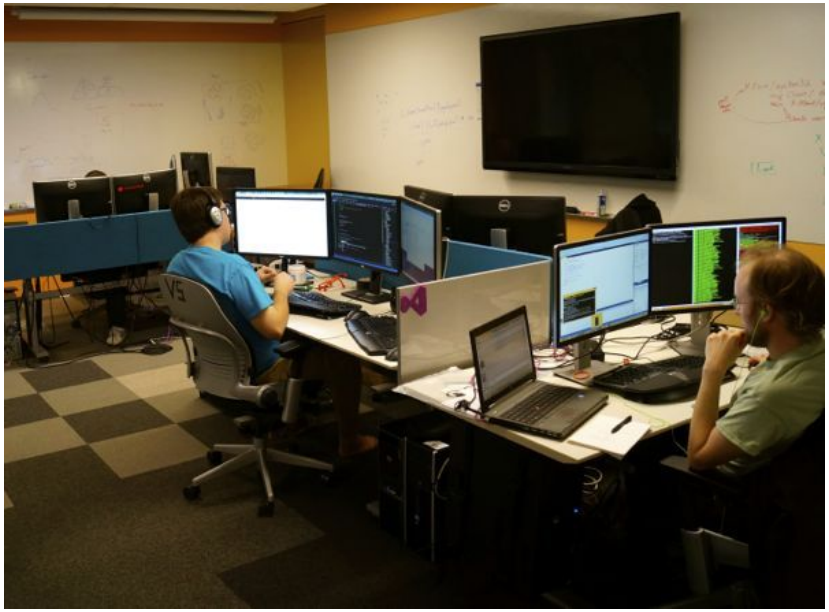


nauka programowania

- staż na Uniwersytecie Edynburskim
  - tworzenie języka programowania

praca

- marzenie: tworzenie gier indie
- rzeczywistość: praca w korporacji przy R&D



praca

- kolejne prace w kolejnych korporacjach: Łódź, Warszawa, Berlin
- kolejna praca w startupie

praca

- dzisiaj: własne projekty
  - co sprowadza mnie tutaj

rozwój projektów

- stopniowo, organicznie
- spontanicznie
- wiele lat rozwoju

rozwój projektów

- interakcja przypadków z wewnętrzną potrzebą tworzenia
- kierowanie się ciekawością, upór i obsesyjne podejście
- dużo różnych projektów przez lata
  - niektóre trwają do dziś
- jeden z nich: Jevko

czym jest Jevko?

- oficjalnie, za jevko.org:
  - Jevko to minimalna składnia ogólnego przeznaczenia – może służyć jako podstawa do konstrukcji prostych i przenośnych formatów, języków i notacji w różnych dziedzinach, takich jak wymiana danych, konfiguracja lub znaczenie tekstu.

Jevko is a minimal general-purpose syntax.

It can be used as a basic building block for simple and portable formats, languages, and notations in a variety of domains, such as data interchange, configuration, or text markup.

jak wygląda Jevko?

XML

```xml
<person
  first-name="John"
  last-name="Smith"
  is-alive="true"
  age="27"
>
  <address
    street-address="21 2nd Street"
    city="New York"
    state="NY"
    postal-code="10021-3100"
  />
  <phone-numbers>
    <phone-number
      type="home"
      number="212 555-1234"
    />
    <phone-number
      type="office"
      number="646 555-4567"
    />
  </phone-numbers>
  <children />
  <spouse xsi:nil="true" />
</person>
```

JSON

```json
{
  "first name": "John",
  "last name": "Smith",
  "is alive": true,
  "age": 27,
  "address": {
    "street address": "21 2nd Street",
    "city": "New York",
    "state": "NY",
    "postal code": "10021-3100"
  },
  "phone numbers": [
    {
      "type": "home",
      "number": "212 555-1234"
    },
    {
      "type": "office",
      "number": "646 555-4567"
    }
  ],
  "children": [],
  "spouse": null
}
```

Jevko

```
first name [John]
last name [Smith]
is alive [true]
age [27]
address [
  street address [21 2nd Street]
  city [New York]
  state [NY]
  postal code [10021-3100]
]
phone numbers [
  [
    type [home]
    number [212 555-1234]
  ]
  [
    type [office]
    number [646 555-4567]
  ]
]
children [seq]
spouse [nil]
```

HTML

```
<html>
  <head>
    <title>This is a title</title>
  </head>
  <body>
    <div>
      <p>Hello world!</p>
      <abbr
        id="anId"
        class="jargon"
        style="color: purple;"
        title="Hypertext Markup Language">
      HTML</abbr>
      <a href="https://www.wikipedia.org/">
        A link to Wikipedia!
      </a>
      <p>
        Oh well,
        <span lang="fr">c'est la vie</span>,
        as they say in France.
      </p>
    </div>
  </body>
</html>
```

Jevko

```
html [
  head [
    title [This is a title]
  ]
  body [
    div [
      p [Hello world!]
      abbr [
        id=[anId]
        class=[jargon]
        style=[color: purple;]
        title=[Hypertext Markup Language]
      HTML]
      a [href=[https://www.wikipedia.org/]
        A link to Wikipedia!
      ]
      p [
        Oh well,
        span [lang=[fr]c'est la vie],
        as they say in France.
      ]
    ]
  ]
]
```

Jevko: początki

- korzenie sięgają do początku drogi programistycznej
- fascynacja językami programowania i minimalizmem

Jevko: wcześniej

- praca magisterska: język programowania do gier
- składnia oparta o Lisp, z własnymi ulepszeniami

2.10.1 Destructuring

Pattern matching works with bindings, the language allows destructuring assignments and definitions[28]. An example of a such definition would be:

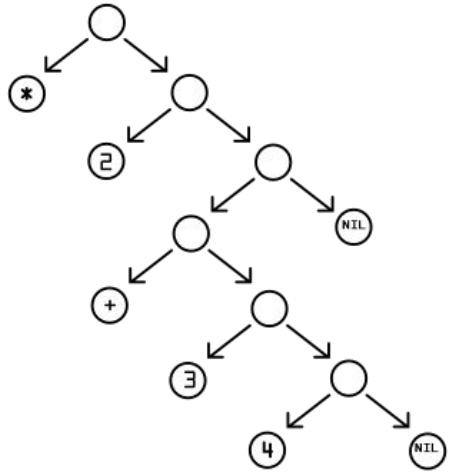
```
bind [
    $[a b $[c d]]
    $[1 2 $[3 4]]
]

bind [
    $[ _   x   y  {   rest   }]
    $['|a '|b '|c  '|d '|e  '|f]
]

-- logs `1 2 3 4`:
log [a b c d]

-- logs `b c ["d", "e", "f"]`:
log [x y rest]
```

2.10.2 `match` primitive

The above examples show pattern matching used in function definitions and for destructuring values by binding their components to identifiers. There is also the `match` primitive, which can serve the role of a `switch` statement from C-like languages. It is however much more powerful, as any complex values support
pattern matching system can be matched, including
multiple values and in any
The

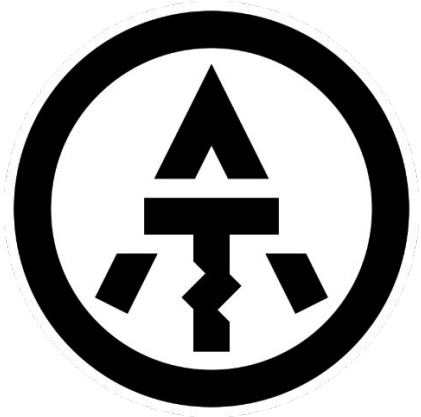
Jevko: wcześniej

- doświadczenie zawodowe i poza z różnymi językami, formatami, składniami
- chęć znalezienia esencji



Jevko: wcześniej

- odkrycie obiecującego diamencika
  - nazwanie go TAO i próba pokazania szerszej publiczności
  - rok 2020



TAO: a simple alternative to XML, JSON, S-expressions

TAO (Tree Annotation Operator) is a unique and extremely simple Tree Annotation Syntax which can be used in all kinds of contexts. It is easily readable and writable by humans and machines and has only three basic constructs:

- **Tree**, the generic data structure.
- **Annotation** or `note`, the generic data primitive.
- **Operator** or `op`, the generic extension mechanism.

These can be used to encode arbitrary

Jevko: dzisiaj

- zmiana nazwy na Jevko
  - rok 2021
- oznaki zainteresowania
  - dyskusje na Hacker News, Lobsters, reddicie
  - rok 2022

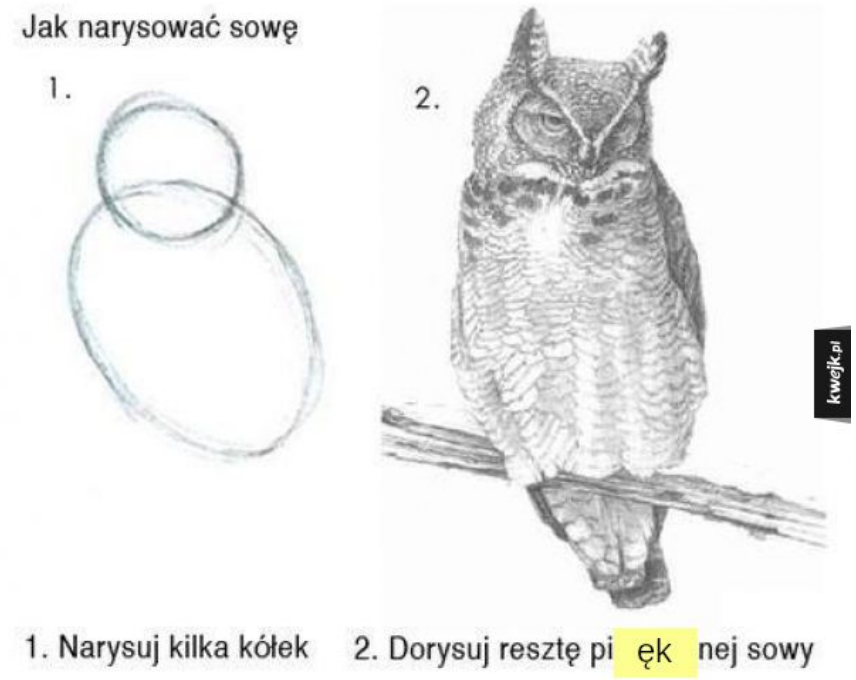
Jevko: dzisiaj

- na tym nie koniec
  - formaty
  - warianty
  - zastosowania

```
This is a comment

title [djed example]

owner [
  name [tester]
  dob [`2020-08-05T20:30:01+09:00[Asia/Tokyo][u-ca=japanese]`]
]

database [
  enabled [true]
  quoted [`true`]
  ports [
    [8000]
    [8001]
    [8002]
  ]
  data [ [[delta] [phi]] [3.14] ]
  temp targets [ cpu [79.5] case [72.0] ]
]
```

własny język programowania – jak?

- tutaj pokażę tylko główne kroki
- a szczegóły:
  - https://github.com/djedr/szczebiot
  - pytania na końcu



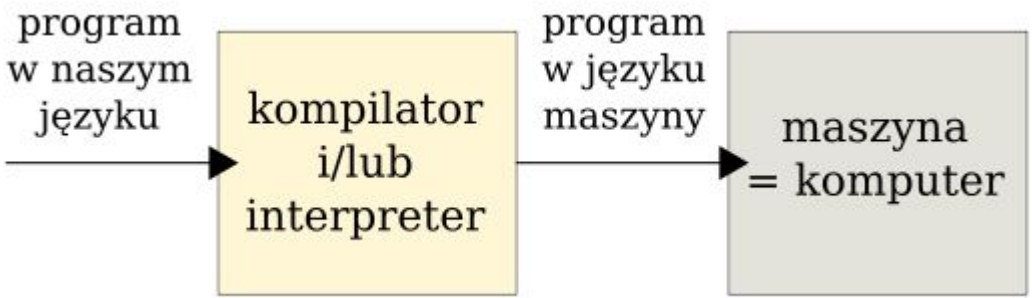
własny język programowania – jak?

- wymyślamy składnię
  - tutaj możemy wykorzystać Jevko
- wymyślamy semantykę
- miejsce na kreatywność

```
zdefiniuj [[a] [5]]
zdefiniuj [[b] [10]]
szczebiocz [[a][b]]

komentarz [nieważne]
zdefiniuj [[f] funkcja [[[a][b]]
  dodaj [[a] pomnóż [[b][b]]]
]]
zdefiniuj [[silnia] funkcja [[n]
  warunkowo [
    równe? [[n] [1]] [1]
    pomnóż [[n] silnia [odejmij [[n] [1]]]]
  ]
]]
zdefiniuj [[liczba Fibonacciego] funkcja [[n]
  warunkowo [
    równe? [[n] [0]] [0]
    równe? [[n] [1]] [1]
    dodaj [
      liczba Fibonacciego [odejmij [[n][1]]]
      liczba Fibonacciego [odejmij [[n][2]]]
    ]
  ]
]]
szczebiocz [f [[a][b]]]
szczebiocz [silnia [20]]
szczebiocz [liczba Fibonacciego [10]]
```

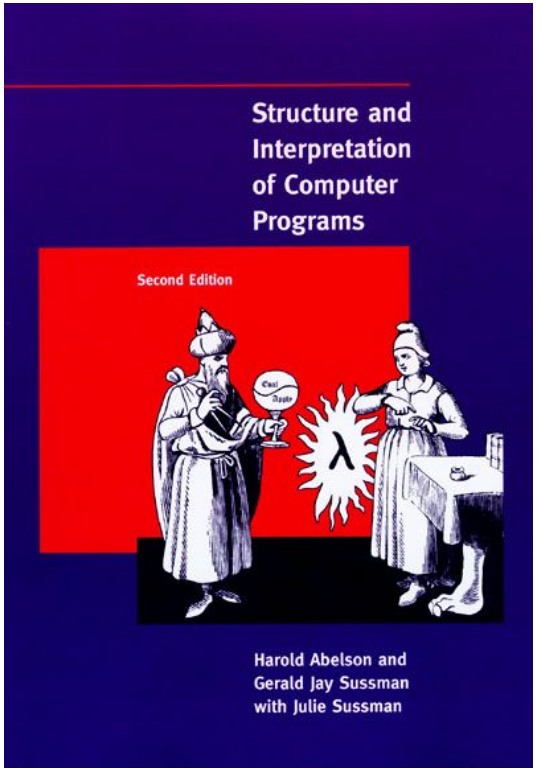
własny język programowania – jak?

- piszemy program, który przetłumaczy nasz język na wybrany istniejący, zgodnie z wymyślonymi przez nas zasadami składniowymi i semantycznymi
- ten program to interpreter i/lub kompilator
- istniejący język to np. JavaScript, Python, C, język maszynowy, lub (na najwyższym poziomie trudności) prawa fizyki

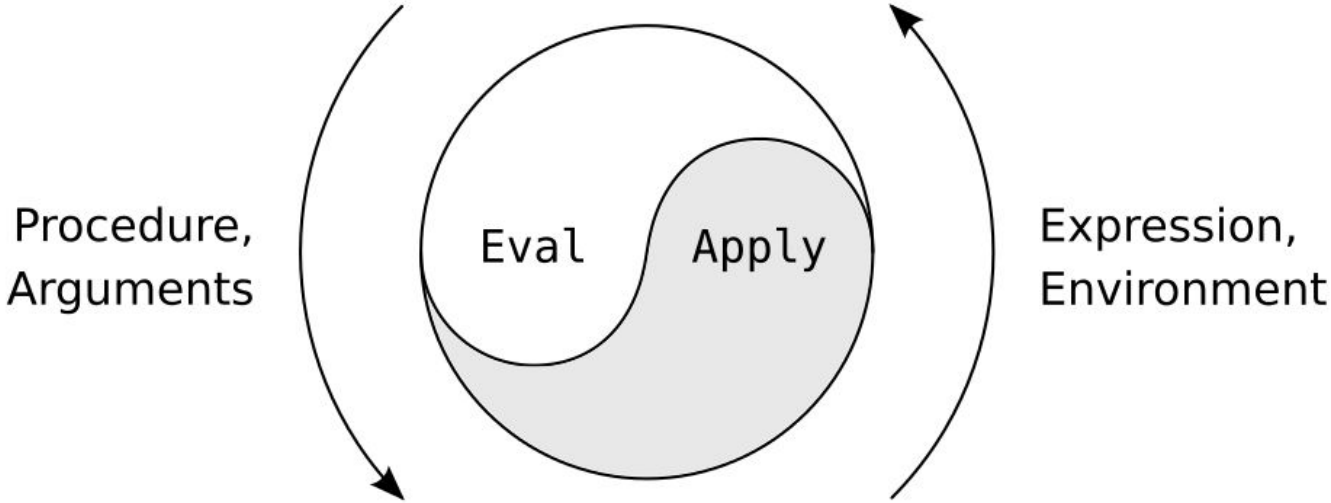


własny język programowania – jak?

- Magiczna Księga Czarodzieja: SICP
- wykłady: https://www.youtube.com/playlist?list=PL8FE88AA54363BC46

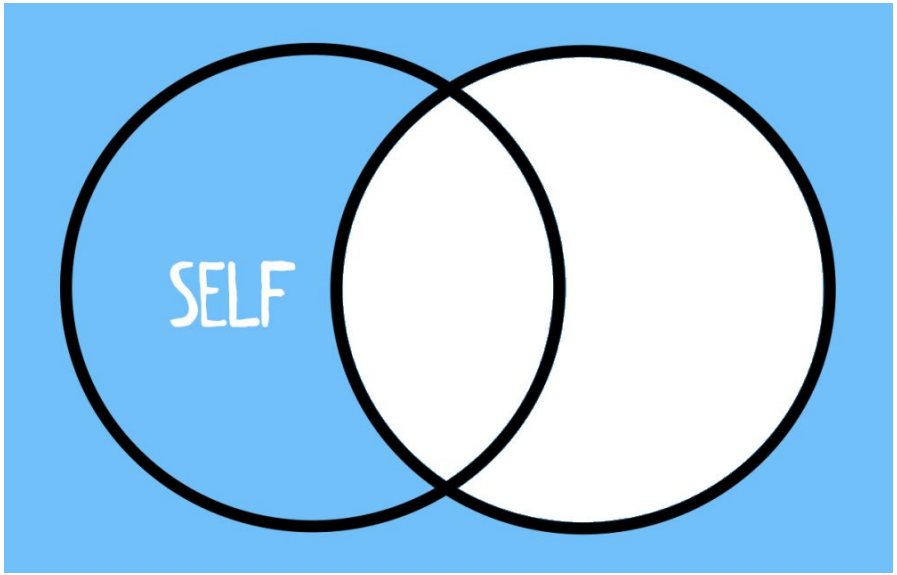


własny język programowania – jak?



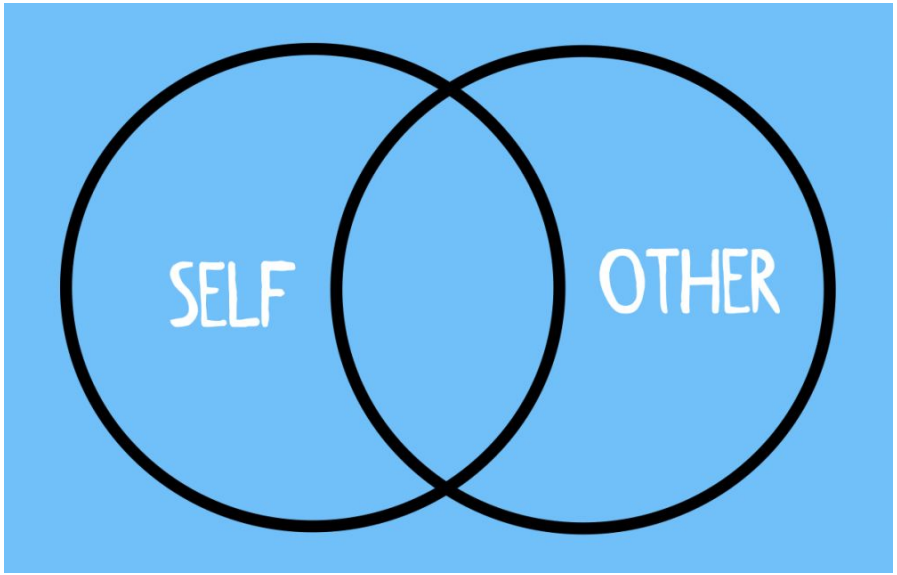
motywacje

- dla siebie
  - rozwiązanie problemu
    - uproszczenie życia sobie
  - wypełnienie luki
    - chcę stworzyć coś, czego mi brakuje



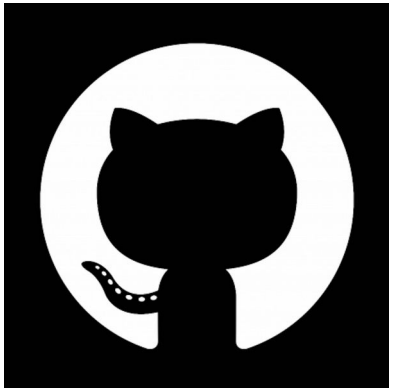
motywacje

- dla innych
  - rozwiązanie problemu
    - uproszczenie życia ludziom
  - podzielenie się rozwiązaniem
    - chcę stworzyć coś użytecznego



dostępność Jevko

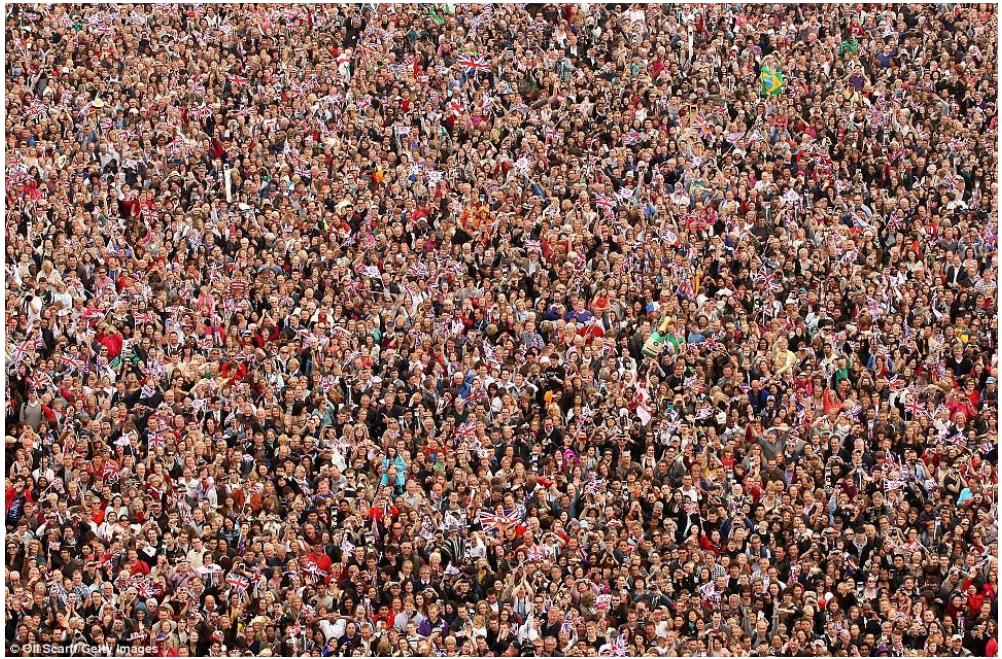
- open-source
- pro bono
- bez ograniczeń



MIT License

dostępność Jevko

- celem dotarcie do jak największej grupy odbiorców

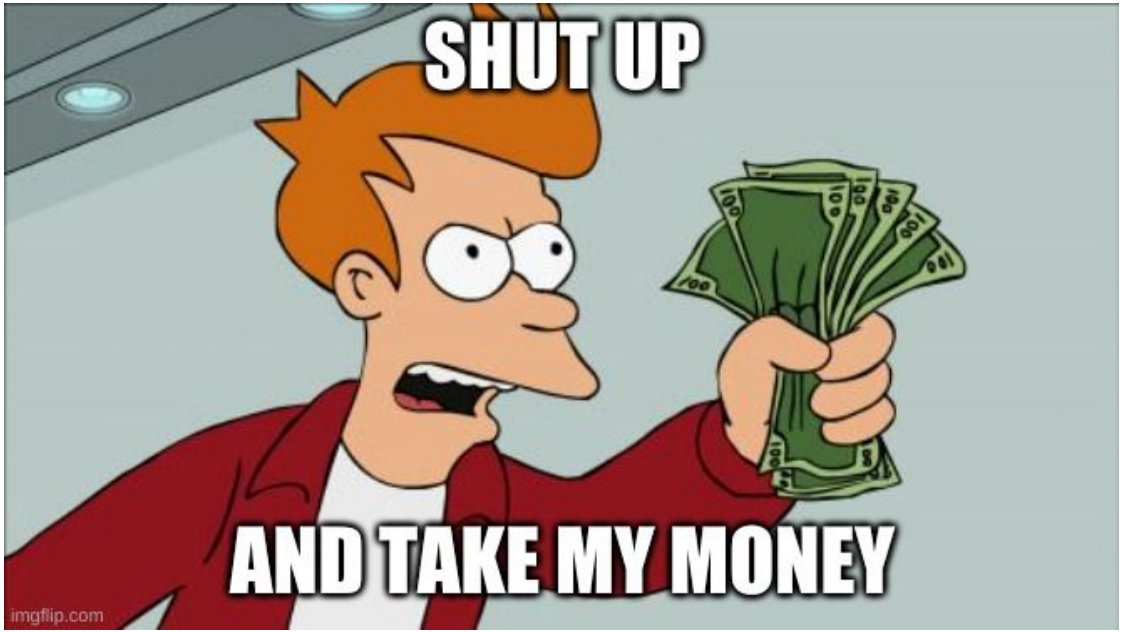


finansowanie

- własne oszczędności
- niemożliwe na dłuższą metę

finansowanie

- potrzebni sponsorzy czy coś



pozytywne aspekty

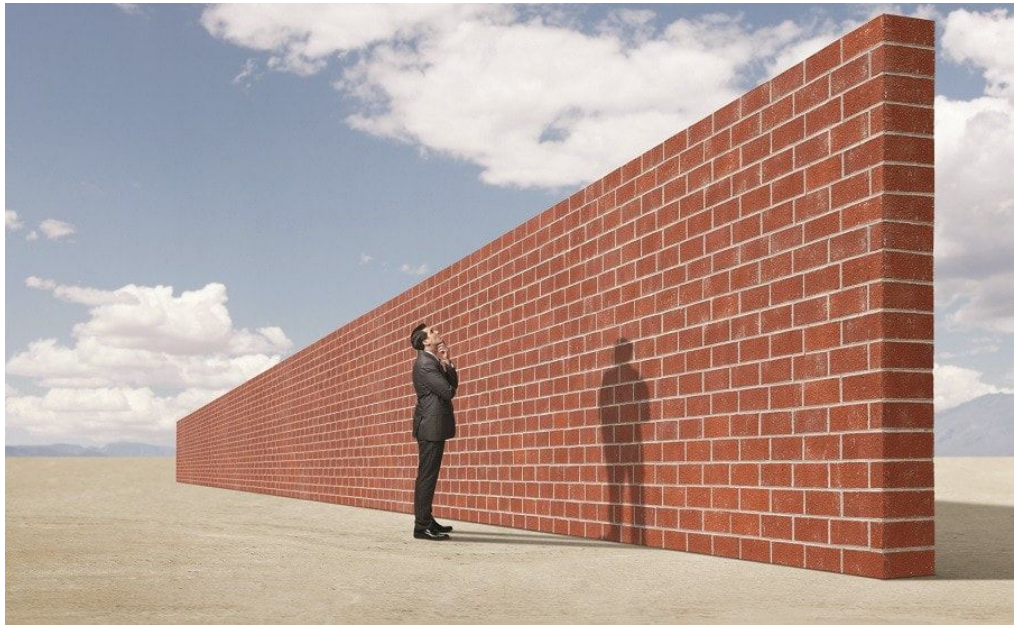
- wolność twórcza
- niezależność
- satysfakcja
- samorealizacja

pozytywne aspekty

- ulga z podrapania się w swędzącym miejscu

trudności i wyzwania

- komunikacja
- dotarcie do odbiorców
- finansowanie
- własne ograniczenia



słowo na koniec

- Rust: Ledwie starcza życia, żeby stać się dobrym w jednej rzeczy.
- Marty: Albo nawet i na to za krótkie.
- Rust: Uważaj zatem w czym się dobry stajesz.

dziękuję