RacketScript — язык будущего?

Сергей Головин

CSSSR

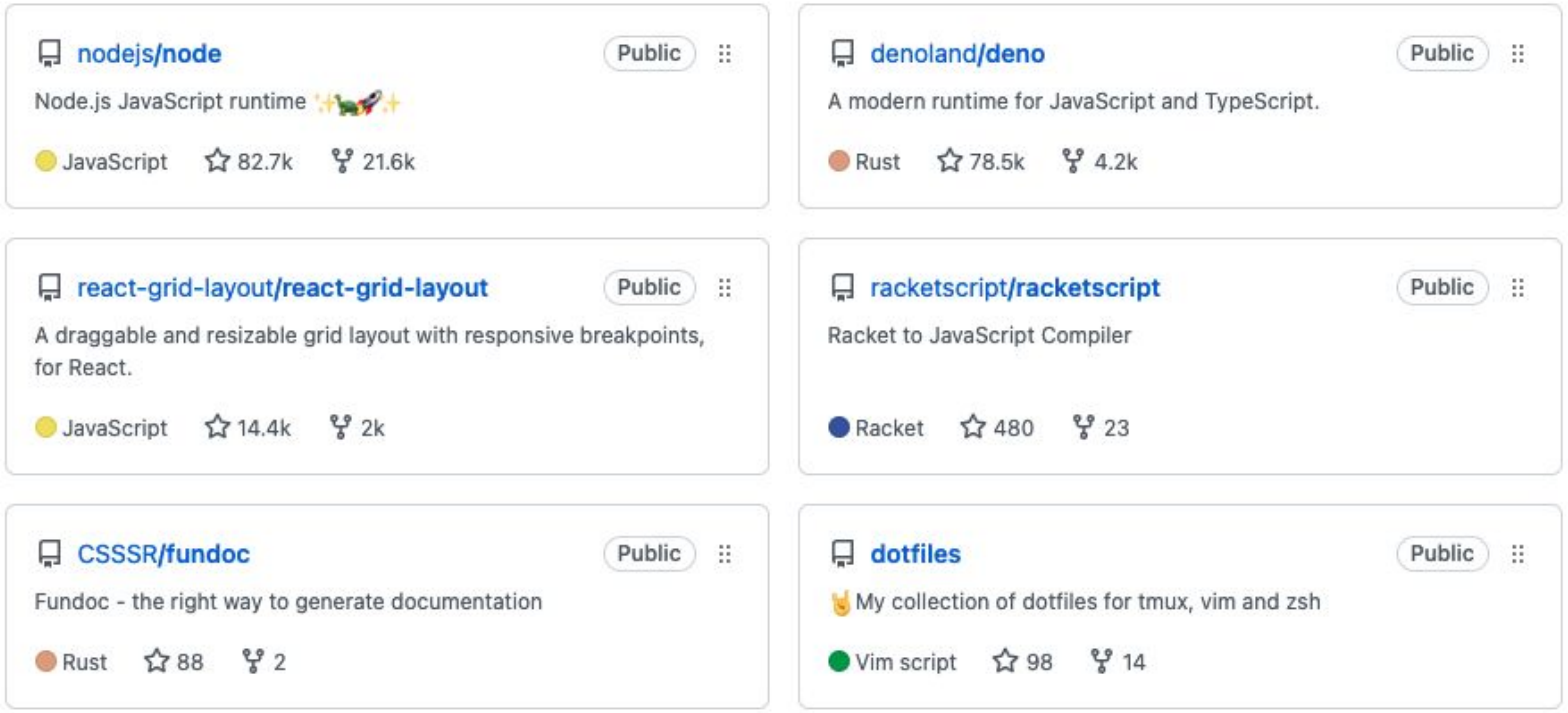
daynin

@_sgolovin

sgolovin

CSSSR

Контрибьютил в разное



nodejs/node Public
Node.js JavaScript runtime ✨🐢🚀✨
● JavaScript ☆ 82.7k 21.6k

denoland/deno Public
A modern runtime for JavaScript and TypeScript.
● Rust ☆ 78.5k 4.2k

react-grid-layout/react-grid-layout Public
A draggable and resizable grid layout with responsive breakpoints, for React.
● JavaScript ☆ 14.4k 2k

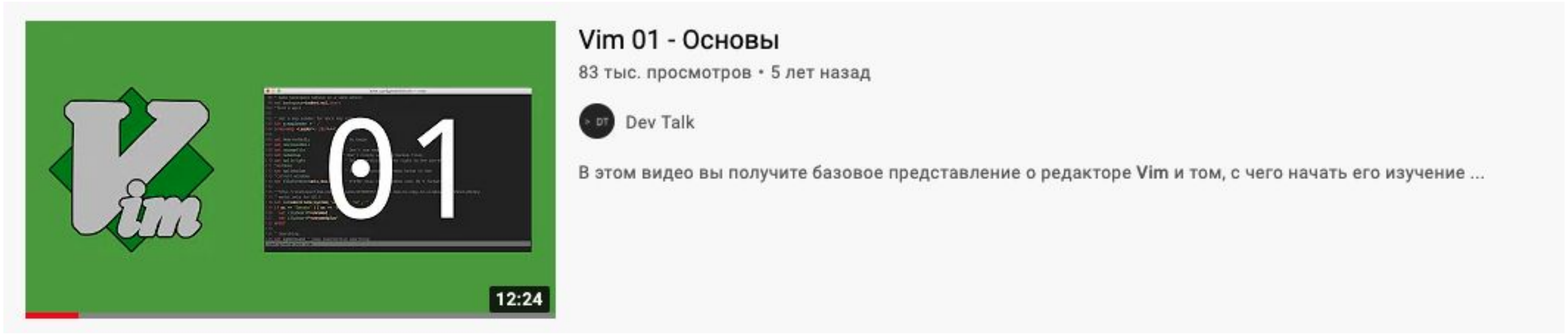
racketscript/racketscript Public
Racket to JavaScript Compiler
● Racket ☆ 480 23

CSSSR/fundoc Public
Fundoc - the right way to generate documentation
● Rust ☆ 88 2

dotfiles Public
🤘 My collection of dotfiles for tmux, vim and zsh
● Vim script ☆ 98 14

CSSSR

Любитель Vim



Vim 01 - Основы

83 тыс. просмотров • 5 лет назад

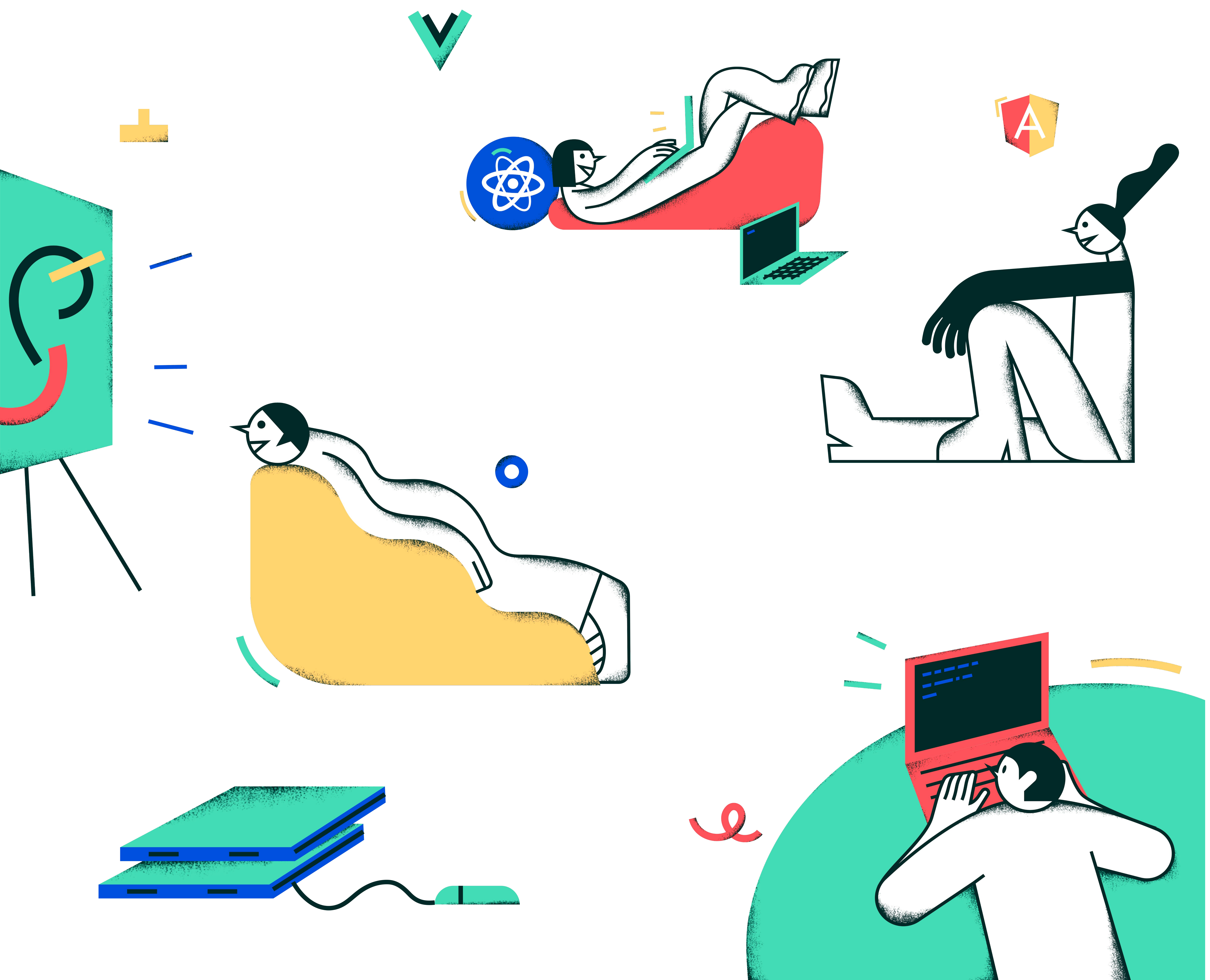
Dev Talk

В этом видео вы получите базовое представление о редакторе **Vim** и том, с чего начать его изучение ...

CSSSR

И немного подкастер

CSSSR Group



Headquartered **in Singapore**, CSSSR is a group of companies that brings together top talent based around the globe to produce **the highest quality results**

2012
established

500+
projects completed

150+
talent employed

1
company flag planted atop Mt.Elbrus

CSSSR

Some of our clients

BCS FINANCIAL GROUP

QuickPay

kaspersky

Alfa·Bank

CIAN.RU

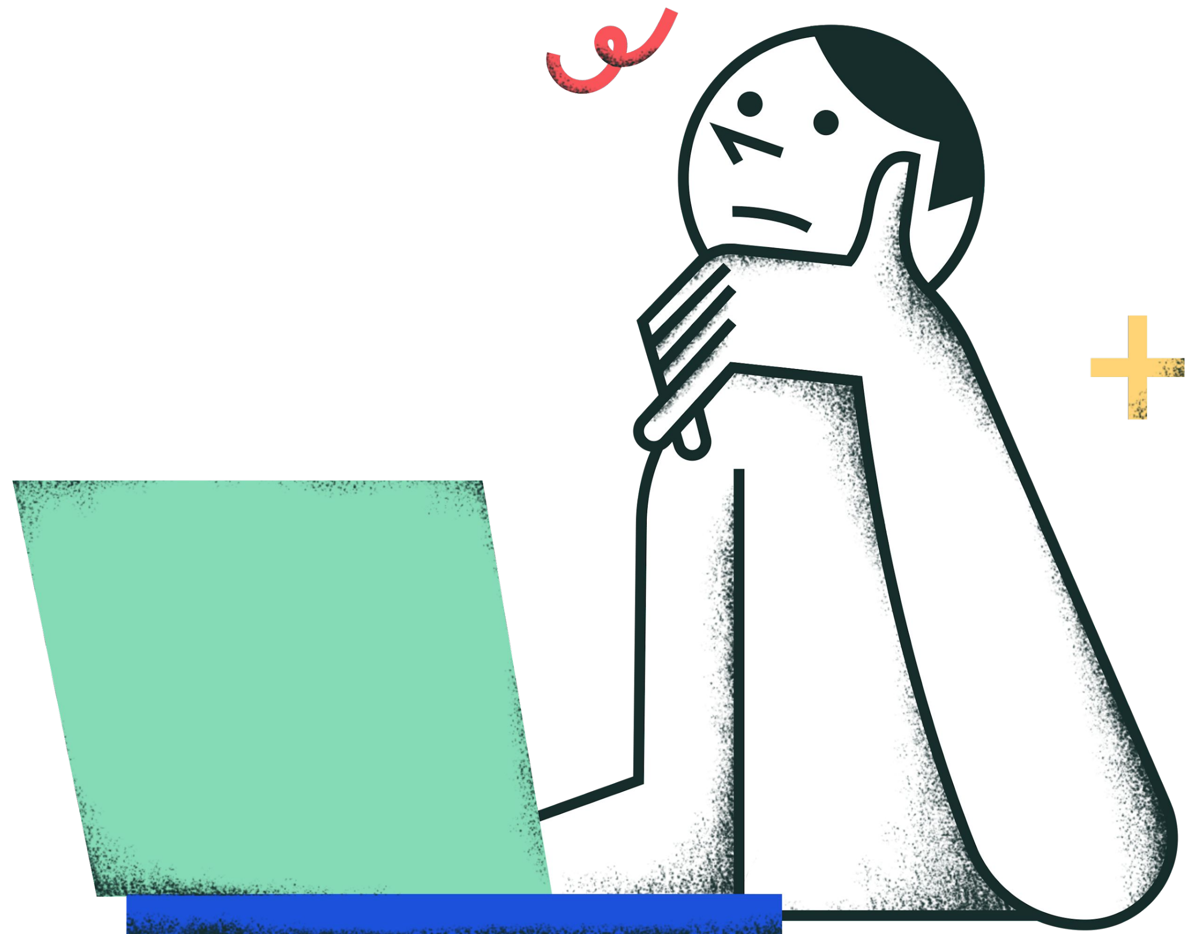
S7 Airlines

Tinkoff Bank

TOYOTA

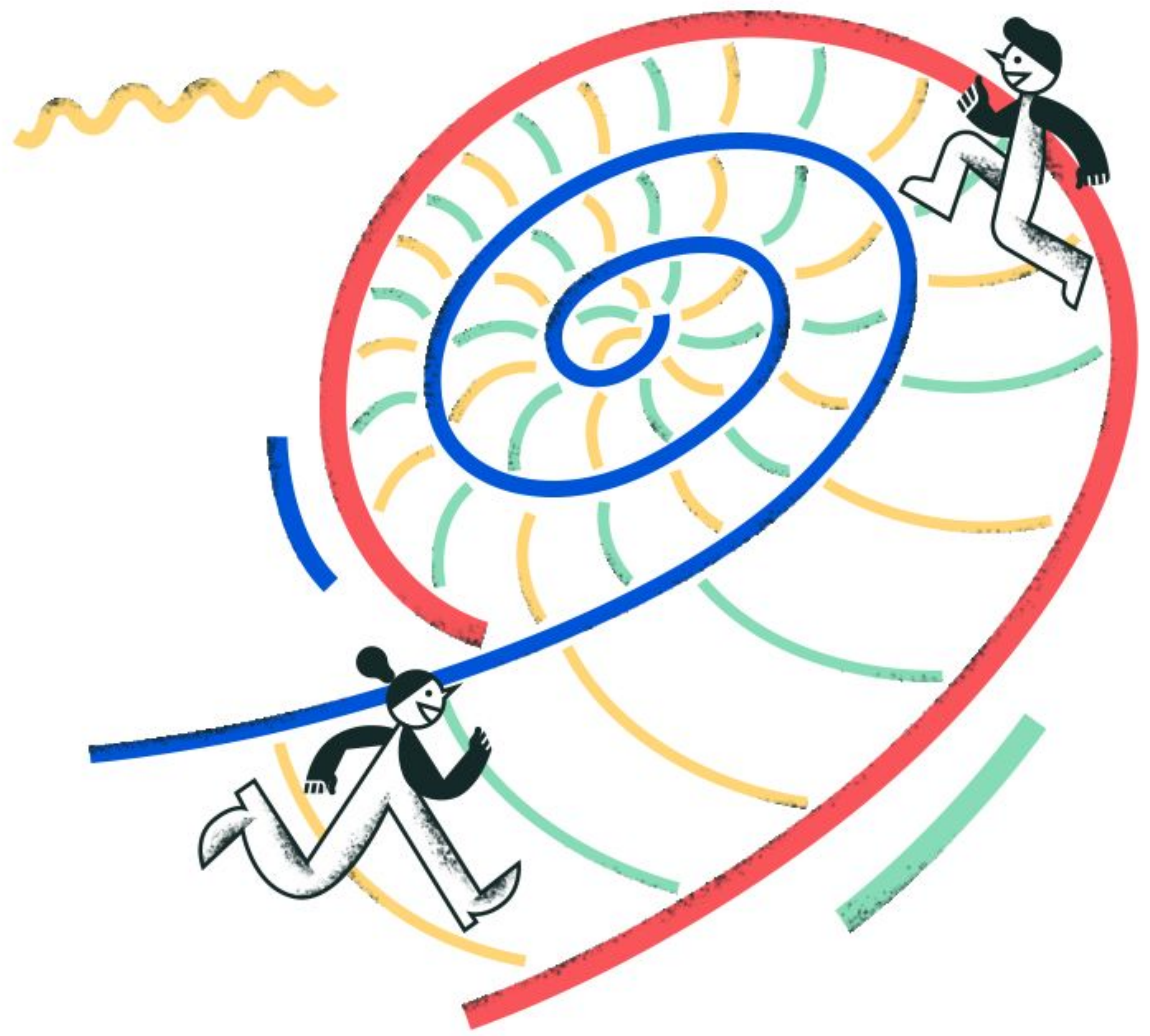
citilink

О чём этот доклад?



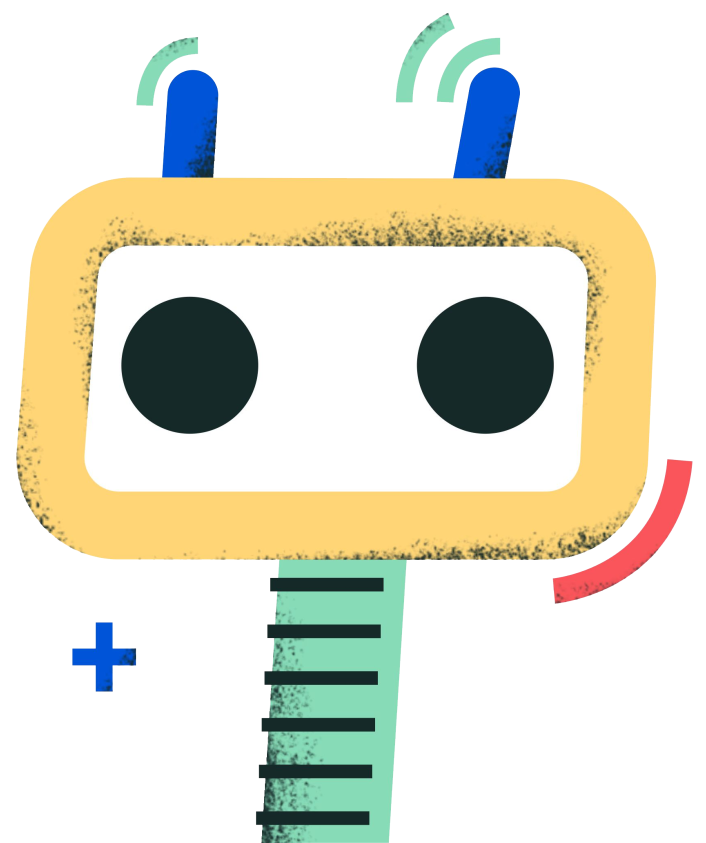
CSSSR

Прежде чем говорить о будущем, заглянем в прошлое



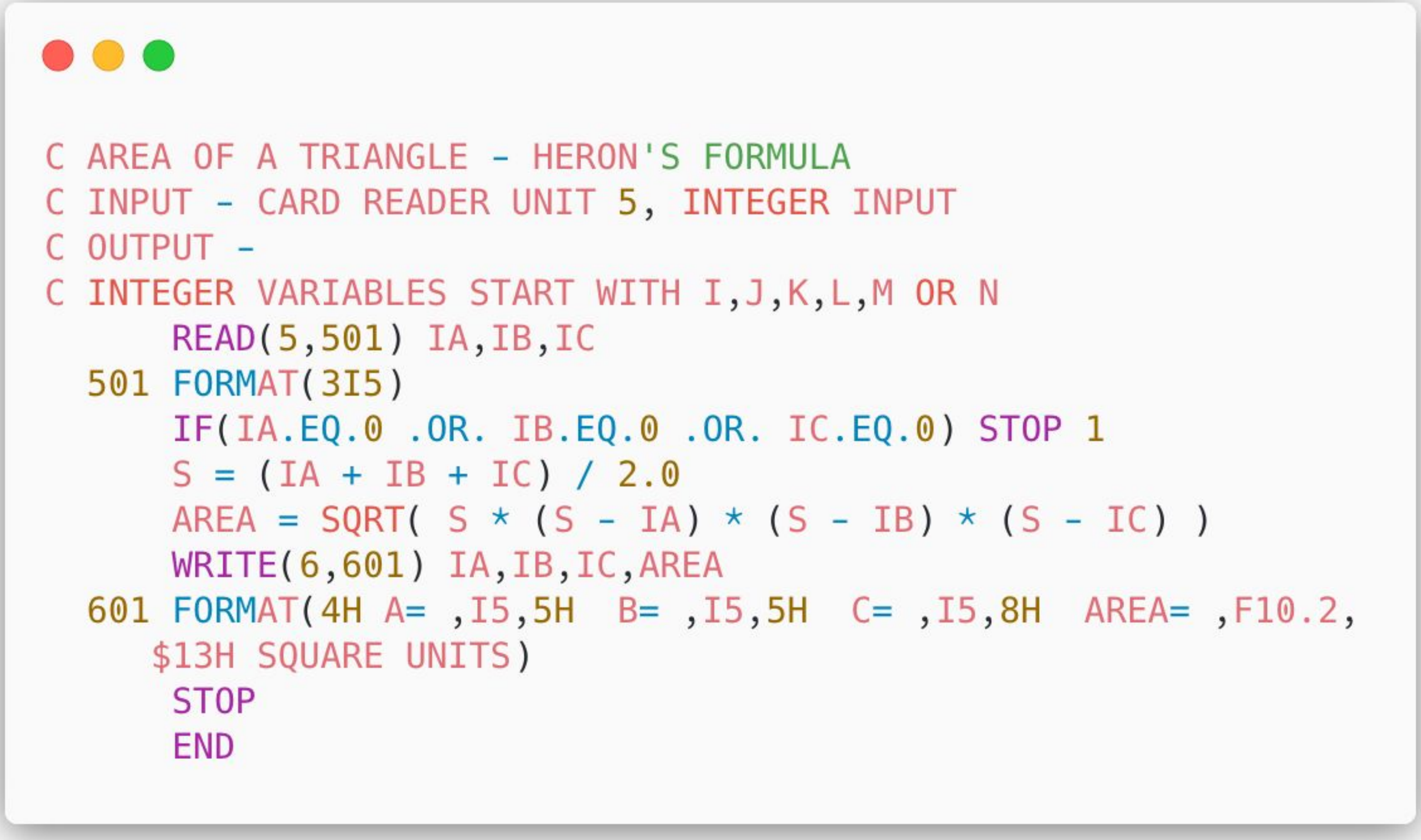
CSSSR

Программирование развивалось по пути наращивания ограничений



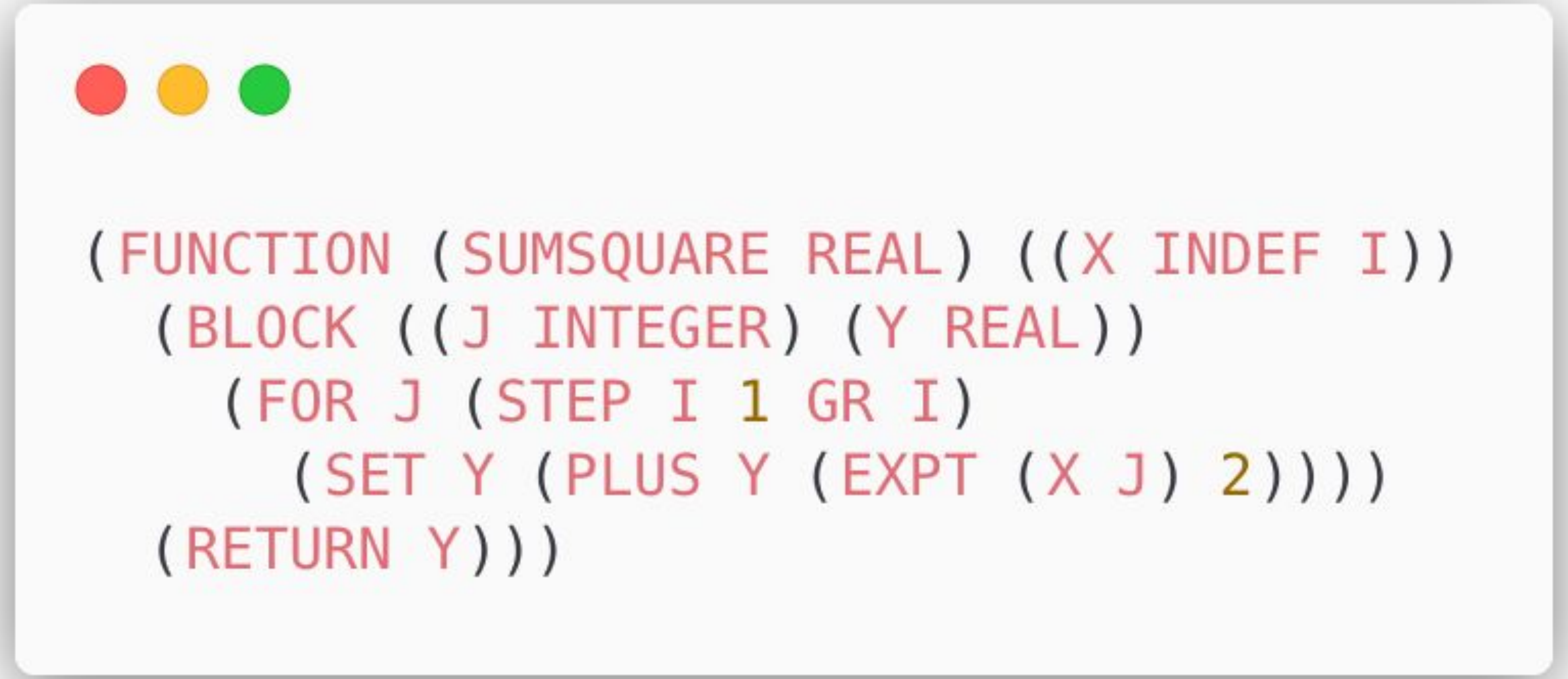
CSSSR

Fortran 66



```
C AREA OF A TRIANGLE - HERON'S FORMULA
C INPUT - CARD READER UNIT 5, INTEGER INPUT
C OUTPUT -
C INTEGER VARIABLES START WITH I,J,K,L,M OR N
      READ(5,501) IA,IB,IC
  501 FORMAT(3I5)
      IF(IA.EQ.0 .OR. IB.EQ.0 .OR. IC.EQ.0) STOP 1
      S = (IA + IB + IC) / 2.0
      AREA = SQRT( S * (S - IA) * (S - IB) * (S - IC) )
      WRITE(6,601) IA,IB,IC,AREA
  601 FORMAT(4H A= ,I5,5H  B= ,I5,5H  C= ,I5,8H  AREA= ,F10.2,
     $13H SQUARE UNITS)
      STOP
      END
```

Lisp 2



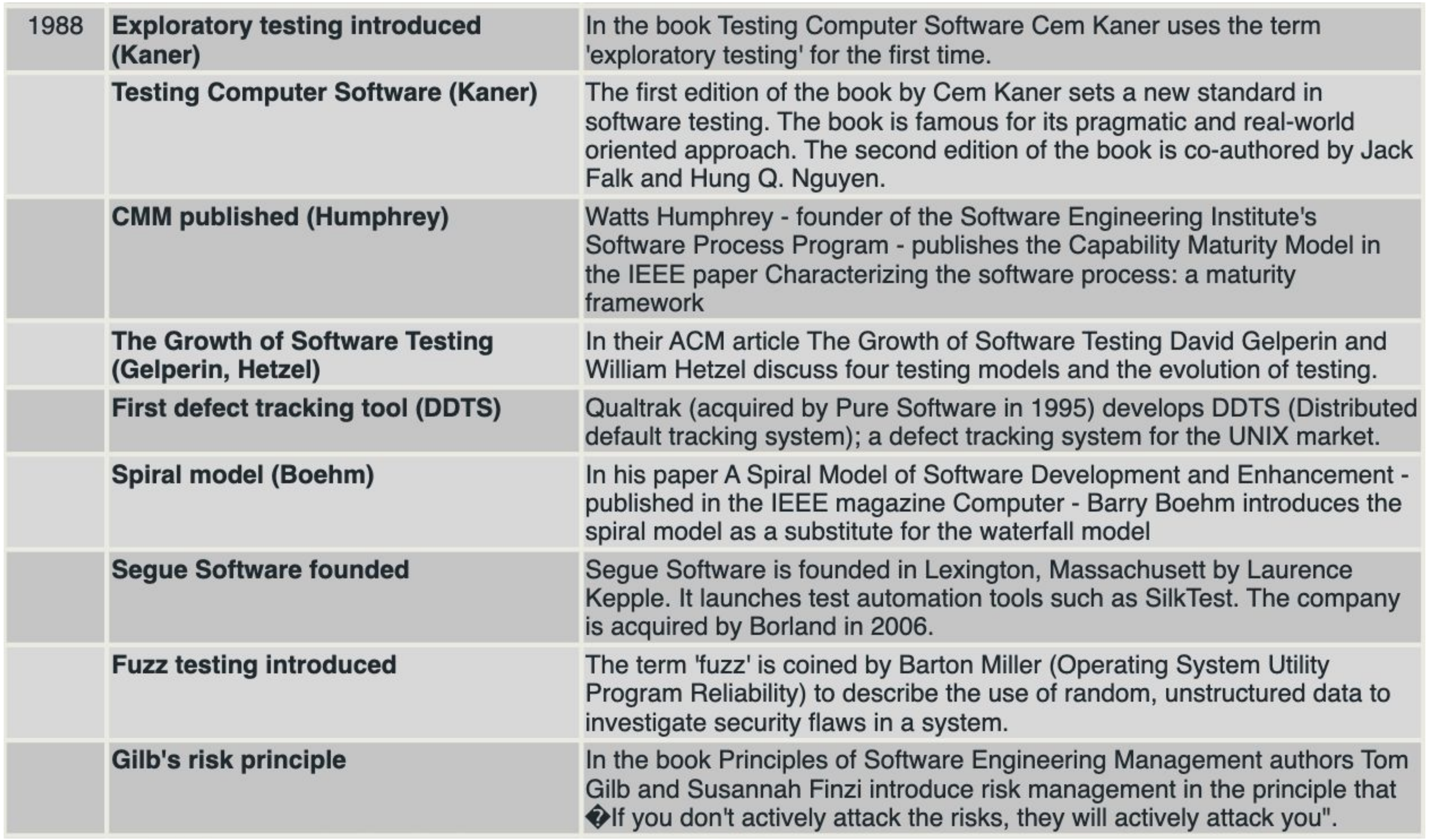
```
(FUNCTION (SUMSQUARE REAL) ((X INDEF I))
  (BLOCK ((J INTEGER) (Y REAL))
    (FOR J (STEP I 1 GR I)
      (SET Y (PLUS Y (EXPT (X J) 2))))
  (RETURN Y)))
```

CSSSR

Появлялись ограничения не только на уровне языка

CSSSR

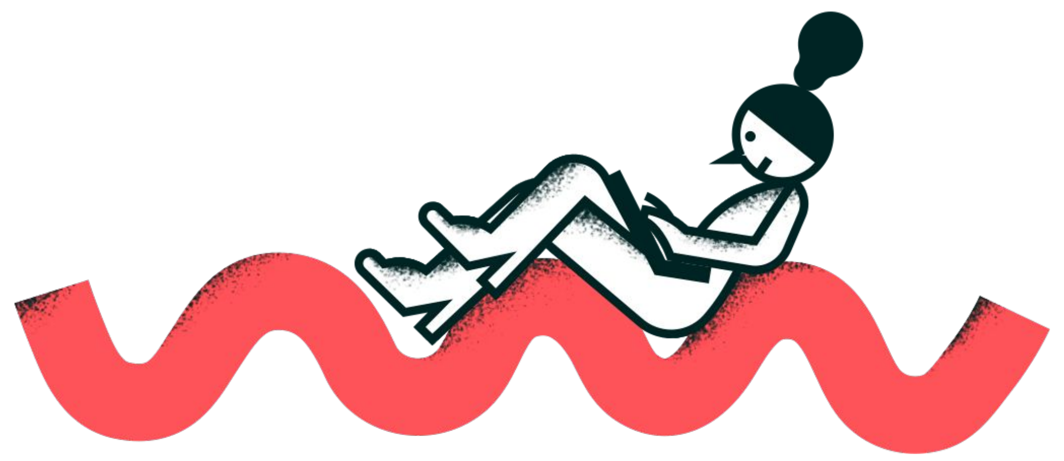
TestingReferences.com



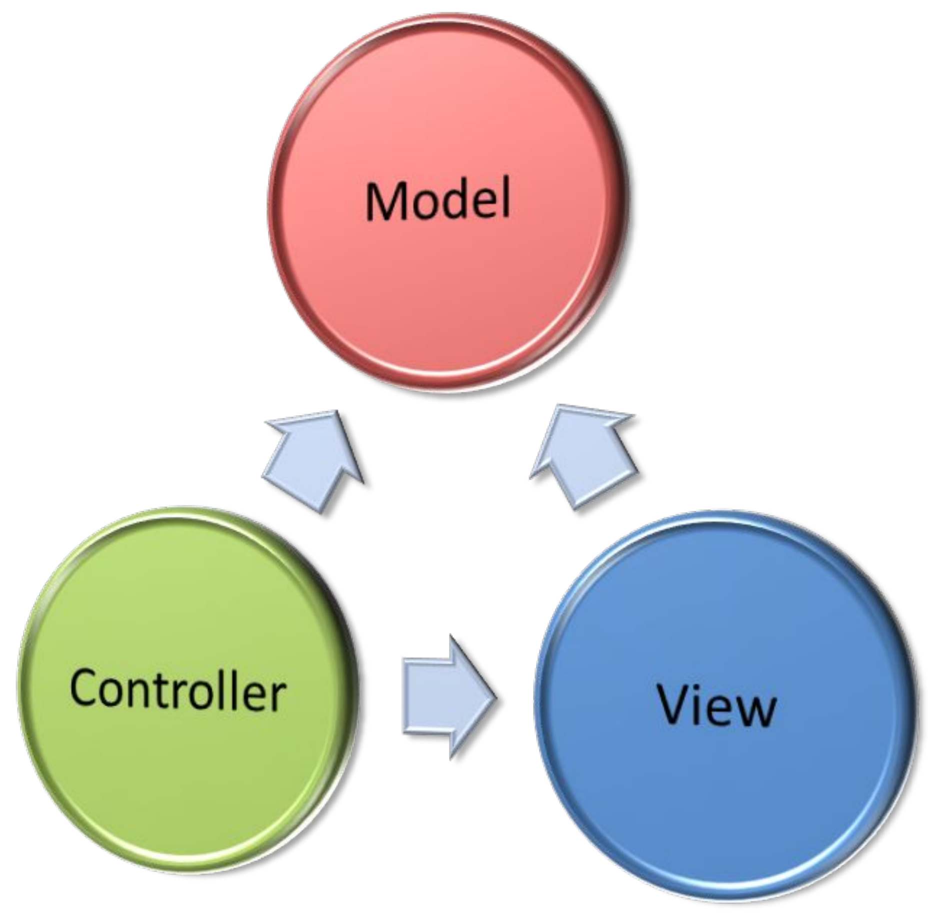
1988	**Exploratory testing introduced (Kaner)**	In the book Testing Computer Software Cem Kaner uses the term 'exploratory testing' for the first time.
	Testing Computer Software (Kaner)	The first edition of the book by Cem Kaner sets a new standard in software testing. The book is famous for its pragmatic and real-world oriented approach. The second edition of the book is co-authored by Jack Falk and Hung Q. Nguyen.
	CMM published (Humphrey)	Watts Humphrey - founder of the Software Engineering Institute's Software Process Program - publishes the Capability Maturity Model in the IEEE paper Characterizing the software process: a maturity framework
	The Growth of Software Testing (Gelperin, Hetzel)	In their ACM article The Growth of Software Testing David Gelperin and William Hetzel discuss four testing models and the evolution of testing.
	First defect tracking tool (DDTS)	Qualtrak (acquired by Pure Software in 1995) develops DDTS (Distributed default tracking system); a defect tracking system for the UNIX market.
	Spiral model (Boehm)	In his paper A Spiral Model of Software Development and Enhancement - published in the IEEE magazine Computer - Barry Boehm introduces the spiral model as a substitute for the waterfall model
	Segue Software founded	Segue Software is founded in Lexington, Massachusett by Laurence Kepple. It launches test automation tools such as SilkTest. The company is acquired by Borland in 2006.
	Fuzz testing introduced	The term 'fuzz' is coined by Barton Miller (Operating System Utility Program Reliability) to describe the use of random, unstructured data to investigate security flaws in a system.
	Gilb's risk principle	In the book Principles of Software Engineering Management authors Tom Gilb and Susannah Finzi introduce risk management in the principle that �If you don't actively attack the risks, they will actively attack you".

CSSSR

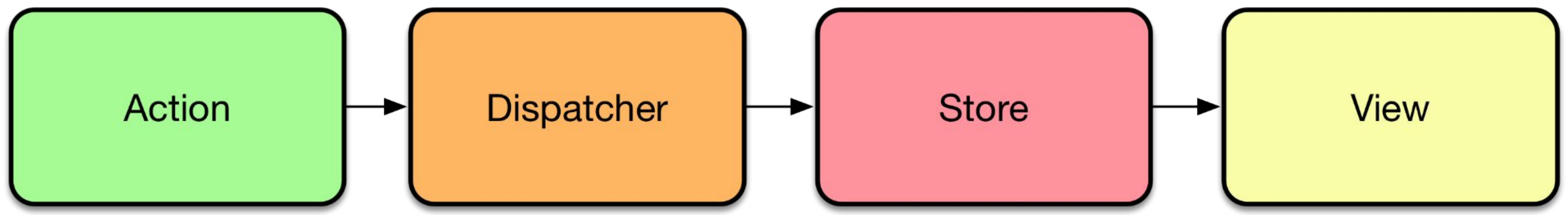
Как и многие другие технологии, автотесты далеко не сразу стали широко применяться



CSSSR



Архитектурные ограничения



CSSSR

ESLint

SENTRY

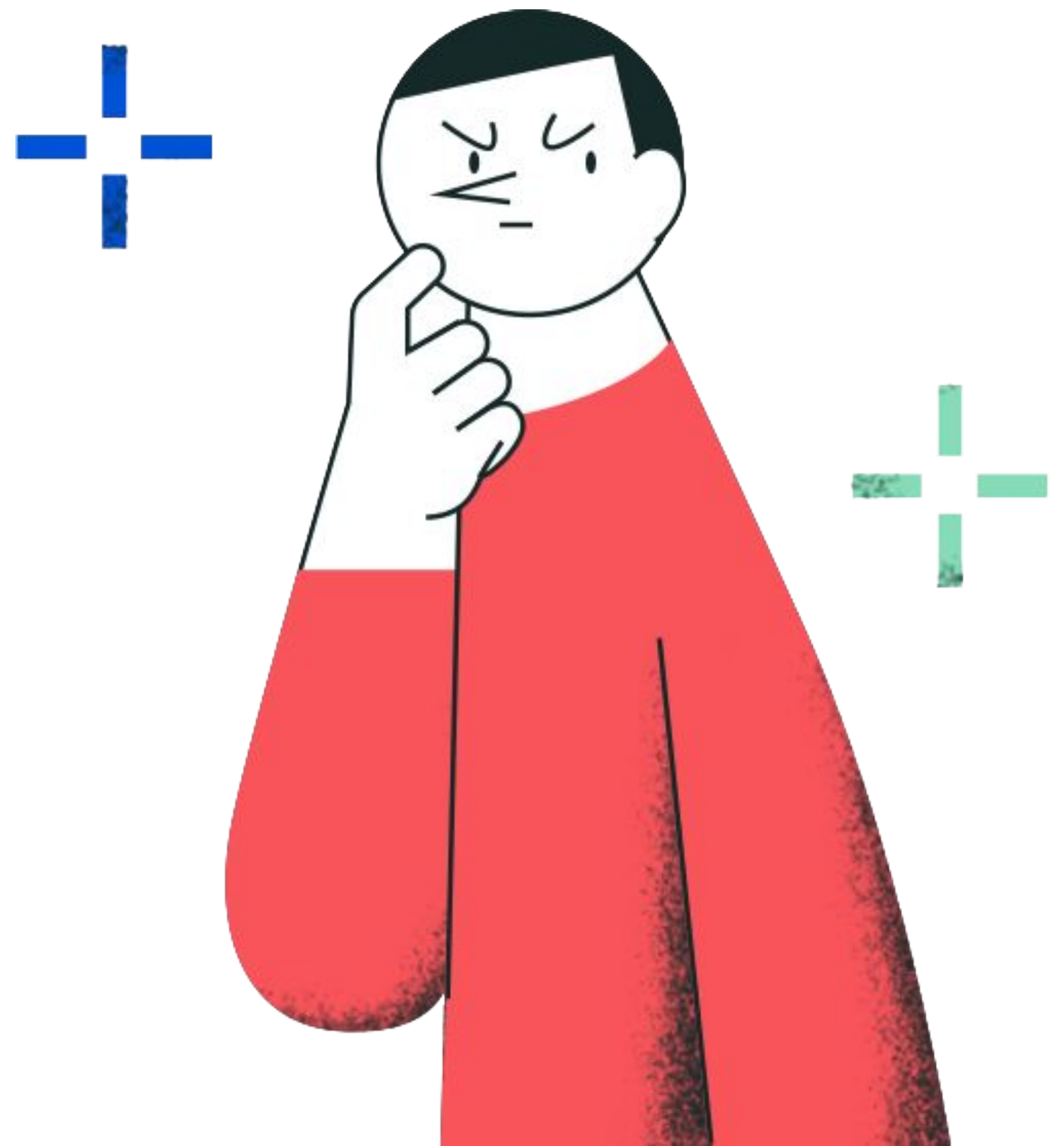
Grafana

И многое другое

SIZE
LIMIT
10
KB

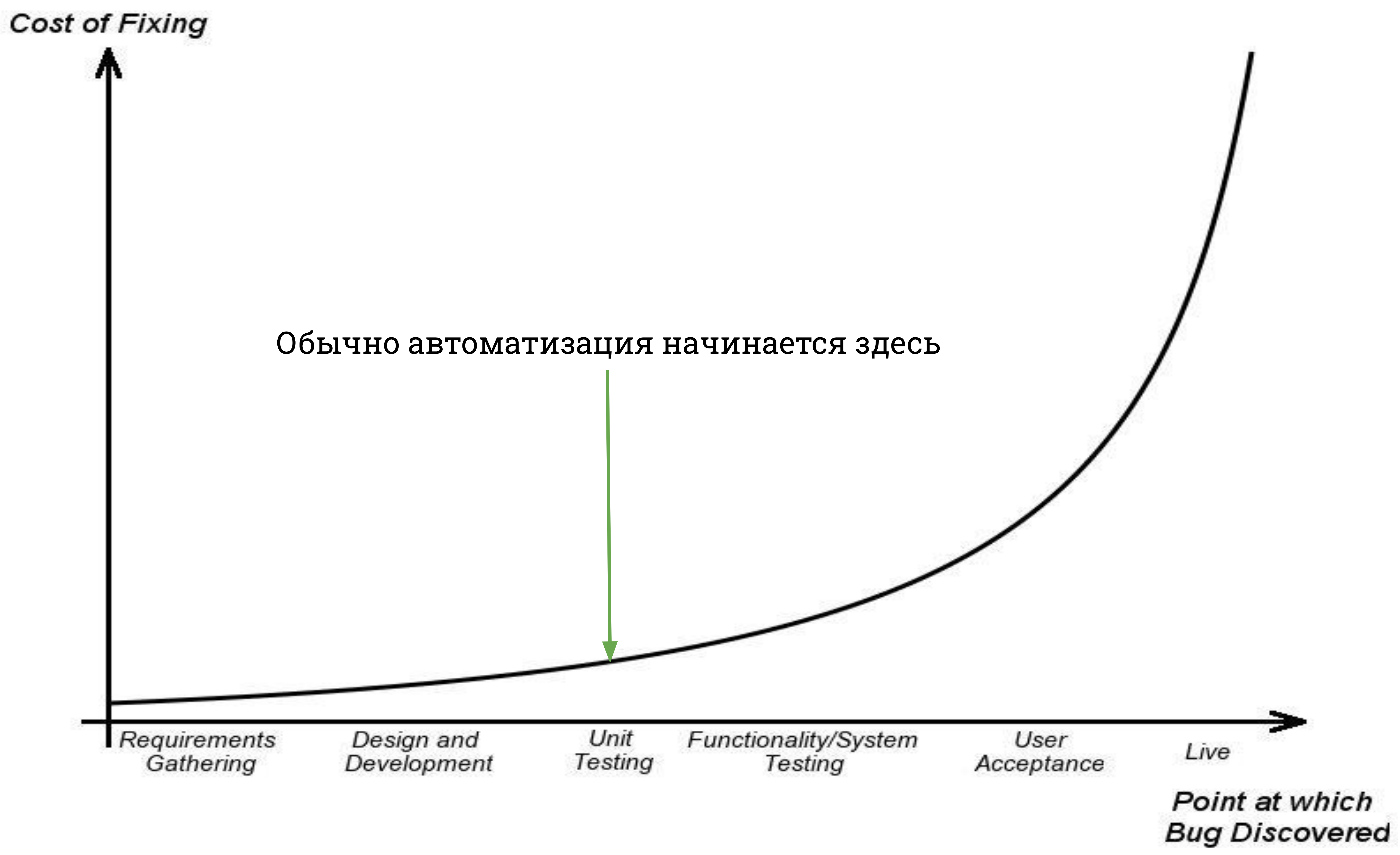
CSSSR

Зачем всё это нужно?



CSSSR

Fail fast!

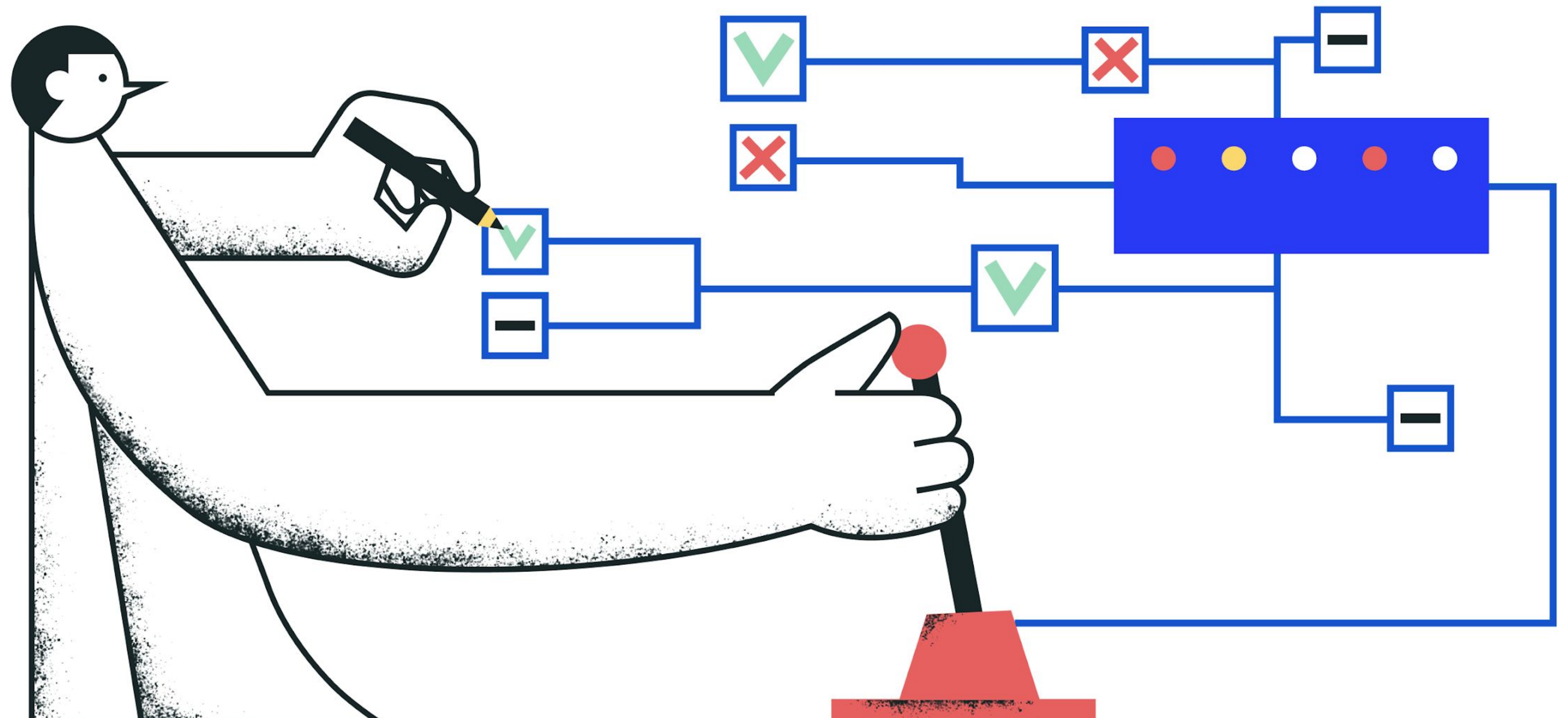


CSSSR

Чтобы избежать такого

CSSSR

А можно ли автоматически находить проблемы на более ранних этапах?

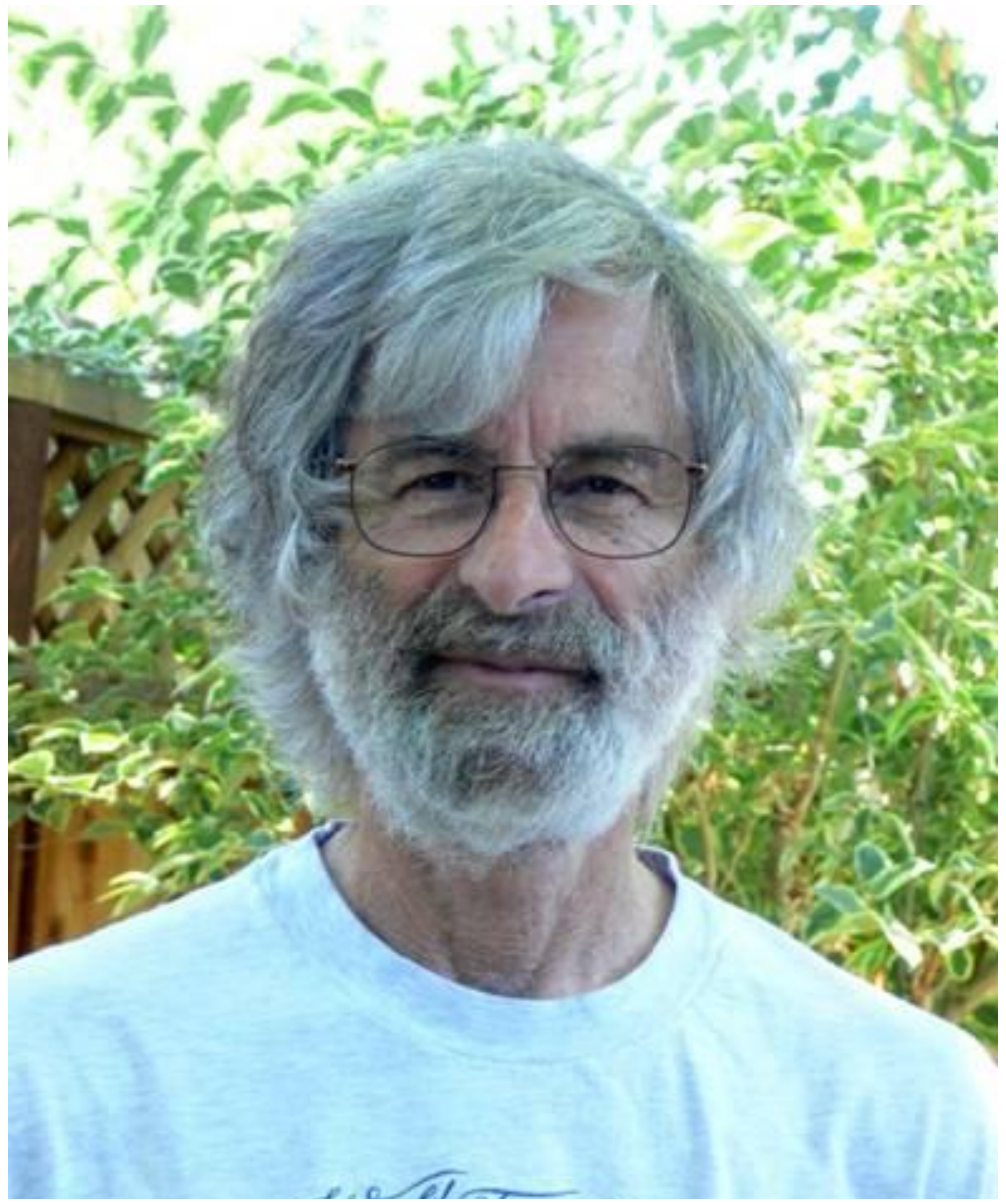


CSSSR

Model Checking

CSSSR

Leslie Lamport



CSSSR

Список наград

- National Academy of Engineering (1991)
- PODC Influential Paper Award (2000) (for paper [27])
- Honorary Doctorate, University of Rennes (2003)
- Honorary Doctorate, Christian Albrechts University, Kiel (2003)
- Honorary Doctorate, Ecole Polytechnique Fédérale de Lausanne (2004)
- IEEE Piore Award (2004)
- Edsger W. Dijkstra Prize in Distributed Computing (2005) (for paper [41])
- Honorary Doctorate, Università della Svizzera Italiana, Lugano (2006)
- ACM SIGOPS Hall of Fame Award (2007) (for paper [27])
- Honorary Doctorate, Université Henri Poincaré, Nancy (2007)
- LICS 1988 Test of Time Award (2008) (for paper [92])
- IEEE John von Neumann Medal (2008)
- National Academy of Sciences (2011)
- ACM SIGOPS Hall of Fame Award (2012) (for paper [123])
- Jean-Claude Laprie Award in Dependable Computing (2013) (for paper [46])
- ACM SIGOPS Hall of Fame Award (2013) (for paper [66])
- 2013 ACM Turing Award (2014)
- American Academy of Arts and Sciences (2014)
- Jean-Claude Laprie Award in Dependable Computing (2014) (for paper [30])
- Edsger W. Dijkstra Prize in Distributed Computing (2014) (for paper [66])
- Honorary Doctorate, Brandeis University (2017)
- Fellow of the Computer History Museum (2019)
- NEC C&C Prize (2019)

CSSSR

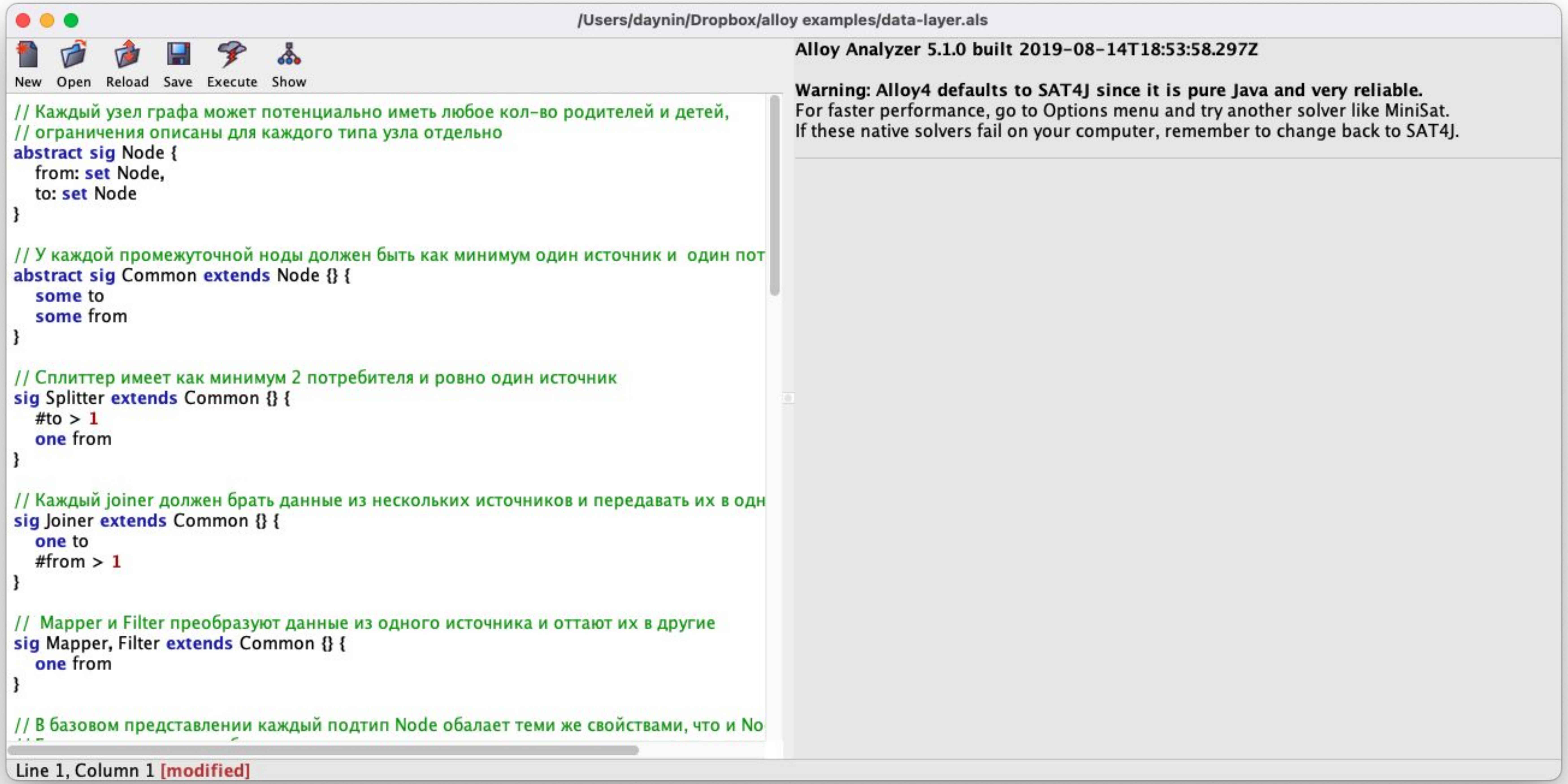
Некоторые инструменты

TLA+

alloy

CSSSR

Alloy



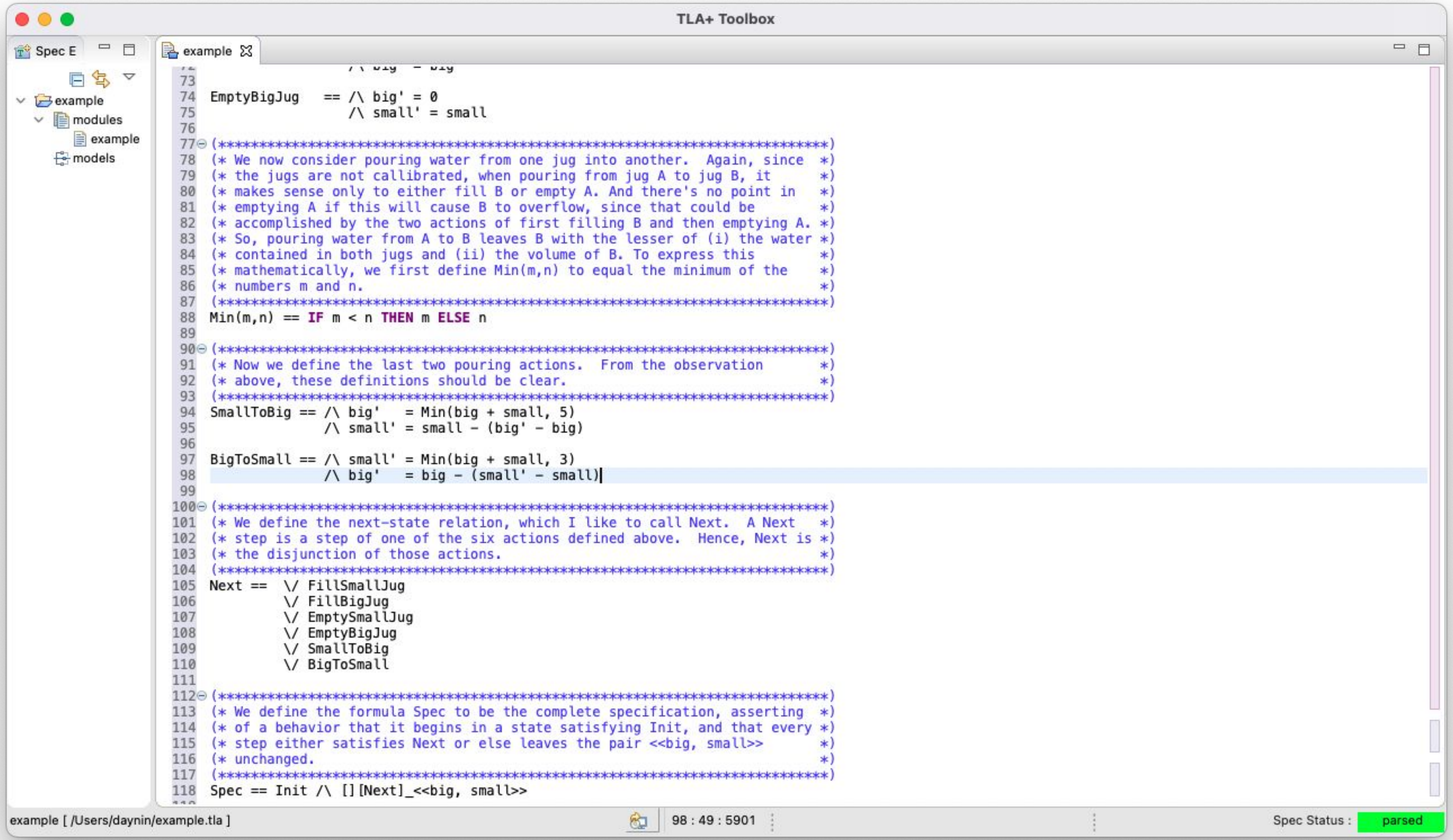
```
// Каждый узел графа может потенциально иметь любое кол-во родителей и детей,
// ограничения описаны для каждого типа узла отдельно
abstract sig Node {
    from: set Node,
    to: set Node
}

// У каждой промежуточной ноды должен быть как минимум один источник и  один пот
abstract sig Common extends Node {} {
    some to
    some from
}

// Сплиттер имеет как минимум 2 потребителя и ровно один источник
sig Splitter extends Common {} {
    #to > 1
    one from
}

// Каждый joiner должен брать данные из нескольких источников и передавать их в одн
sig Joiner extends Common {} {
    one to
    #from > 1
}

//  Mapper и Filter преобразуют данные из одного источника и оттают их в другие
sig Mapper, Filter extends Common {} {
    one from
}

// В базовом представлении каждый подтип Node обалает теми же свойствами, что и No
```

```
Alloy Analyzer 5.1.0 built 2019-08-14T18:53:58.297Z

Warning: Alloy4 defaults to SAT4J since it is pure Java and very reliable.
For faster performance, go to Options menu and try another solver like MiniSat.
If these native solvers fail on your computer, remember to change back to SAT4J.
```

Line 1, Column 1 [modified]

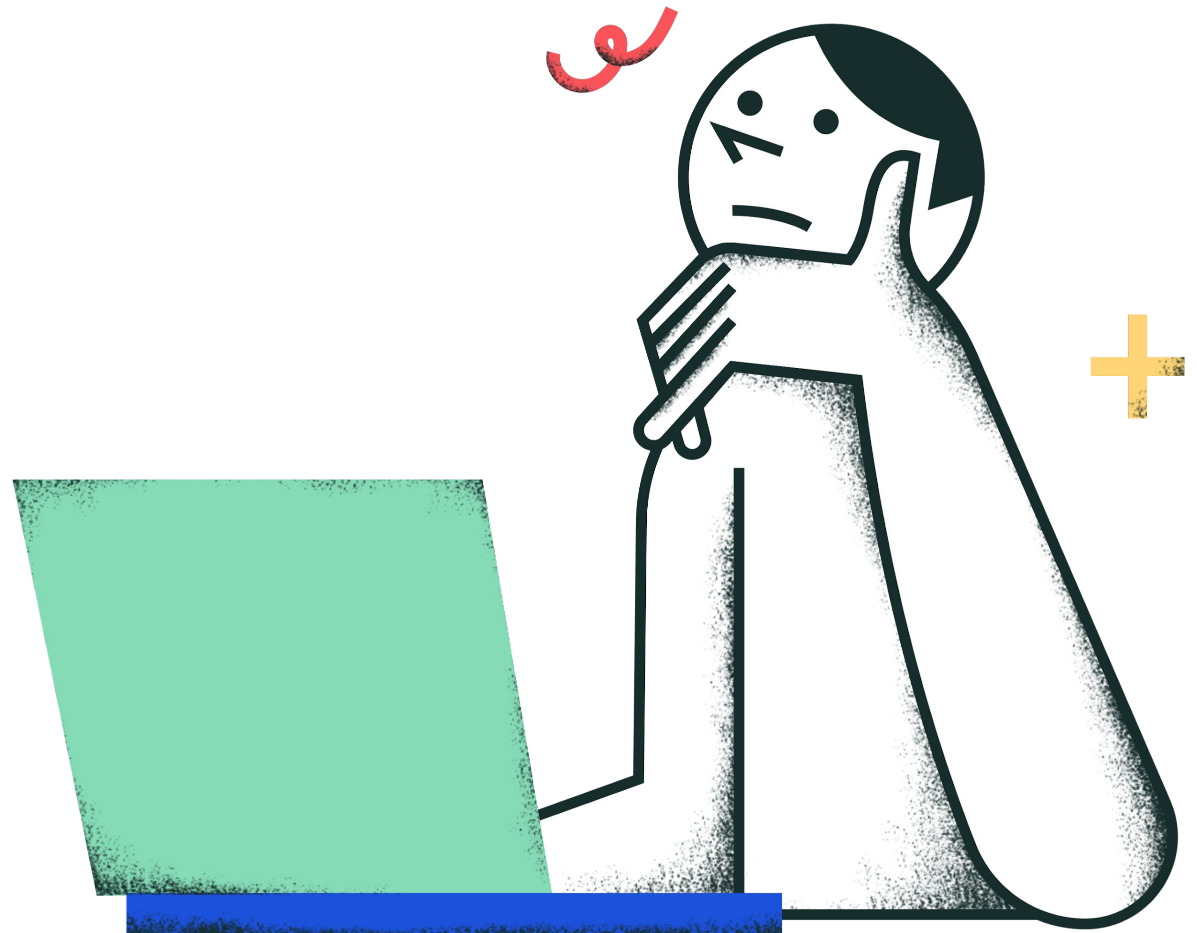
CSSSR

TLA+



```
EmptyBigJug   == /\ big' = 0
                 /\ small' = small

(***************************************************************************)
(* We now consider pouring water from one jug into another.  Again, since  *)
(* the jugs are not callibrated, when pouring from jug A to jug B, it      *)
(* makes sense only to either fill B or empty A. And there's no point in   *)
(* emptying A if this will cause B to overflow, since that could be        *)
(* accomplished by the two actions of first filling B and then emptying A. *)
(* So, pouring water from A to B leaves B with the lesser of (i) the water *)
(* contained in both jugs and (ii) the volume of B. To express this        *)
(* mathematically, we first define Min(m,n) to equal the minimum of the    *)
(* numbers m and n.                                                        *)
(***************************************************************************)
Min(m,n) == IF m < n THEN m ELSE n

(***************************************************************************)
(* Now we define the last two pouring actions.  From the observation       *)
(* above, these definitions should be clear.                               *)
(***************************************************************************)
SmallToBig == /\ big'   = Min(big + small, 5)
              /\ small' = small - (big' - big)

BigToSmall == /\ small' = Min(big + small, 3)
              /\ big'   = big - (small' - small)

(***************************************************************************)
(* We define the next-state relation, which I like to call Next.  A Next   *)
(* step is a step of one of the six actions defined above.  Hence, Next is *)
(* the disjunction of those actions.                                       *)
(***************************************************************************)
Next ==  \/ FillSmallJug
         \/ FillBigJug
         \/ EmptySmallJug
         \/ EmptyBigJug
         \/ SmallToBig
         \/ BigToSmall

(***************************************************************************)
(* We define the formula Spec to be the complete specification, asserting  *)
(* of a behavior that it begins in a state satisfying Init, and that every *)
(* step either satisfies Next or else leaves the pair <<big, small>>       *)
(* unchanged.                                                              *)
(***************************************************************************)
Spec == Init /\ [][Next]_<<big, small>>
```

Кто использует и для чего?



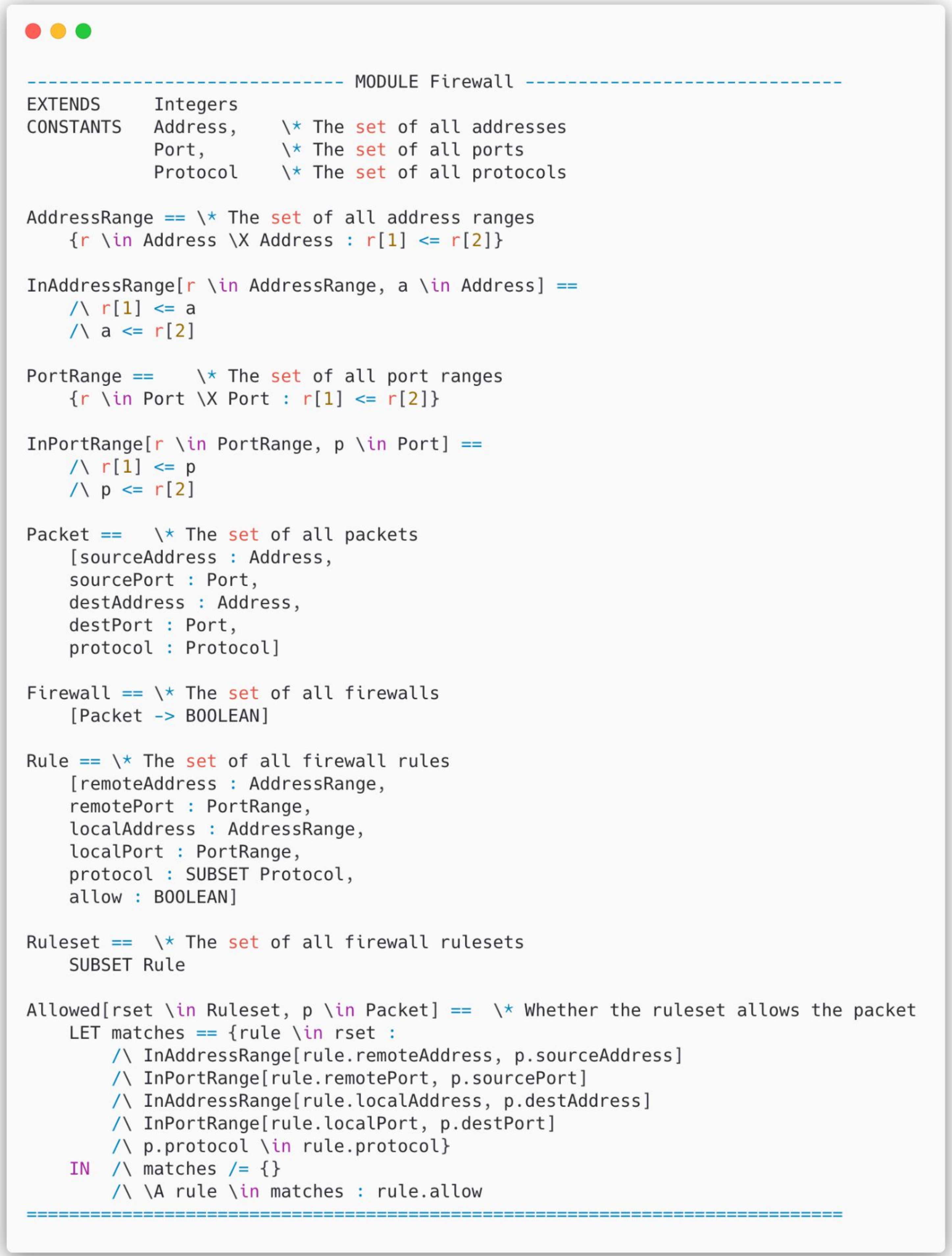
CSSSR

Success stories!

- Нашли баг в модуле памяти Xbox 360.
- Верифицировали Paxos.
- Amazon успешно находил и находит баги в DynamoDB, S3 и т. д.
- Microsoft использовал TLA+ для проектирования Cosmos DB.
- При помощи Alloy найдена ошибка в протоколе Chord.
- И многое другое.

CSSSR

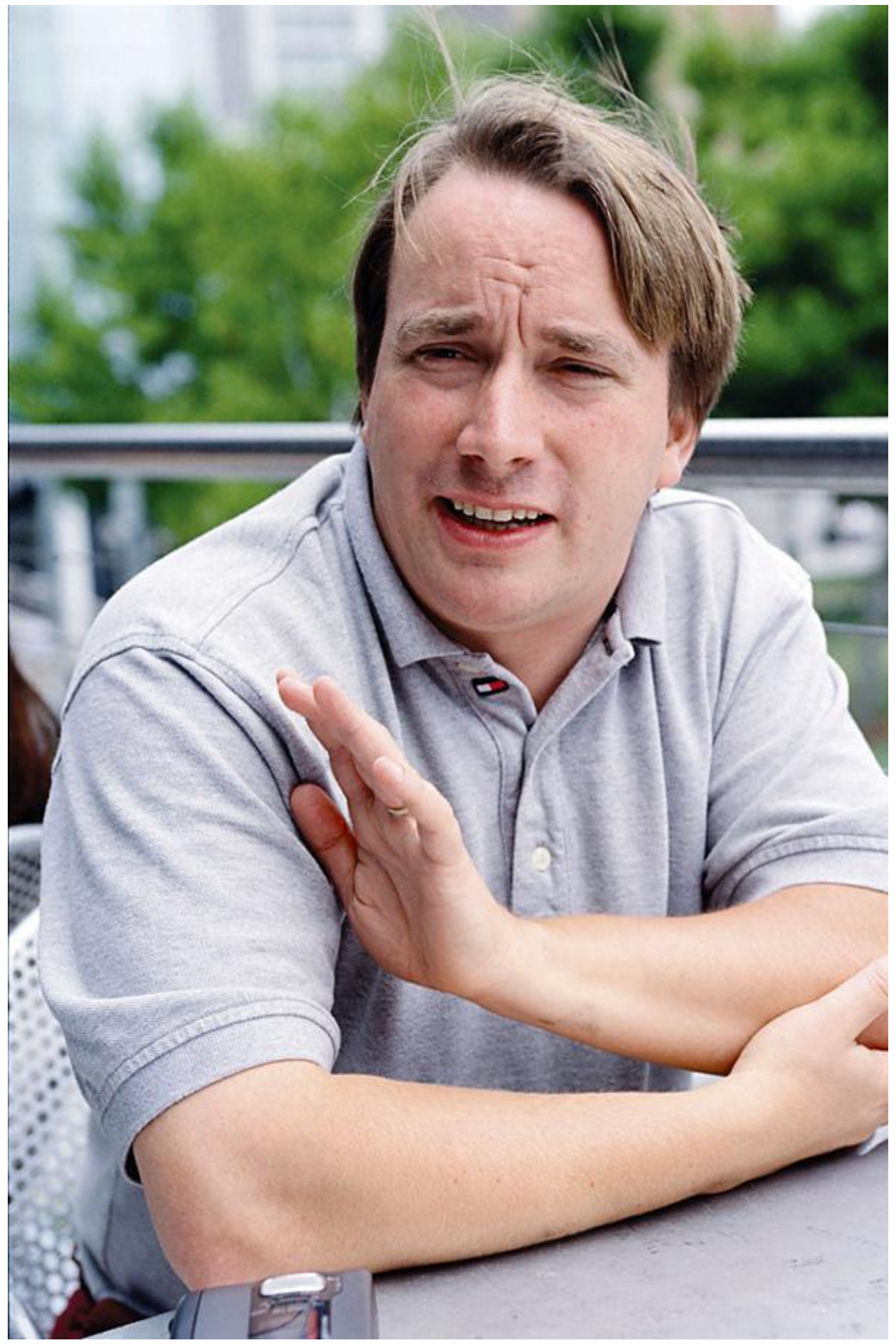
Пример Firewall на TLA+



```
------------------------------ MODULE Firewall ------------------------------
EXTENDS     Integers
CONSTANTS   Address,    \* The set of all addresses
            Port,       \* The set of all ports
            Protocol    \* The set of all protocols

AddressRange == \* The set of all address ranges
    {r \in Address \X Address : r[1] <= r[2]}

InAddressRange[r \in AddressRange, a \in Address] ==
    /\ r[1] <= a
    /\ a <= r[2]

PortRange ==    \* The set of all port ranges
    {r \in Port \X Port : r[1] <= r[2]}

InPortRange[r \in PortRange, p \in Port] ==
    /\ r[1] <= p
    /\ p <= r[2]

Packet ==   \* The set of all packets
    [sourceAddress : Address,
    sourcePort : Port,
    destAddress : Address,
    destPort : Port,
    protocol : Protocol]

Firewall == \* The set of all firewalls
    [Packet -> BOOLEAN]

Rule == \* The set of all firewall rules
    [remoteAddress : AddressRange,
    remotePort : PortRange,
    localAddress : AddressRange,
    localPort : PortRange,
    protocol : SUBSET Protocol,
    allow : BOOLEAN]

Ruleset ==  \* The set of all firewall rulesets
    SUBSET Rule

Allowed[rset \in Ruleset, p \in Packet] ==  \* Whether the ruleset allows the packet
    LET matches == {rule \in rset :
        /\ InAddressRange[rule.remoteAddress, p.sourceAddress]
        /\ InPortRange[rule.remotePort, p.sourcePort]
        /\ InAddressRange[rule.localAddress, p.destAddress]
        /\ InPortRange[rule.localPort, p.destPort]
        /\ p.protocol \in rule.protocol}
    IN  /\ matches /= {}
        /\ \A rule \in matches : rule.allow
=============================================================================
```

CSSSR

Don’t show me code...



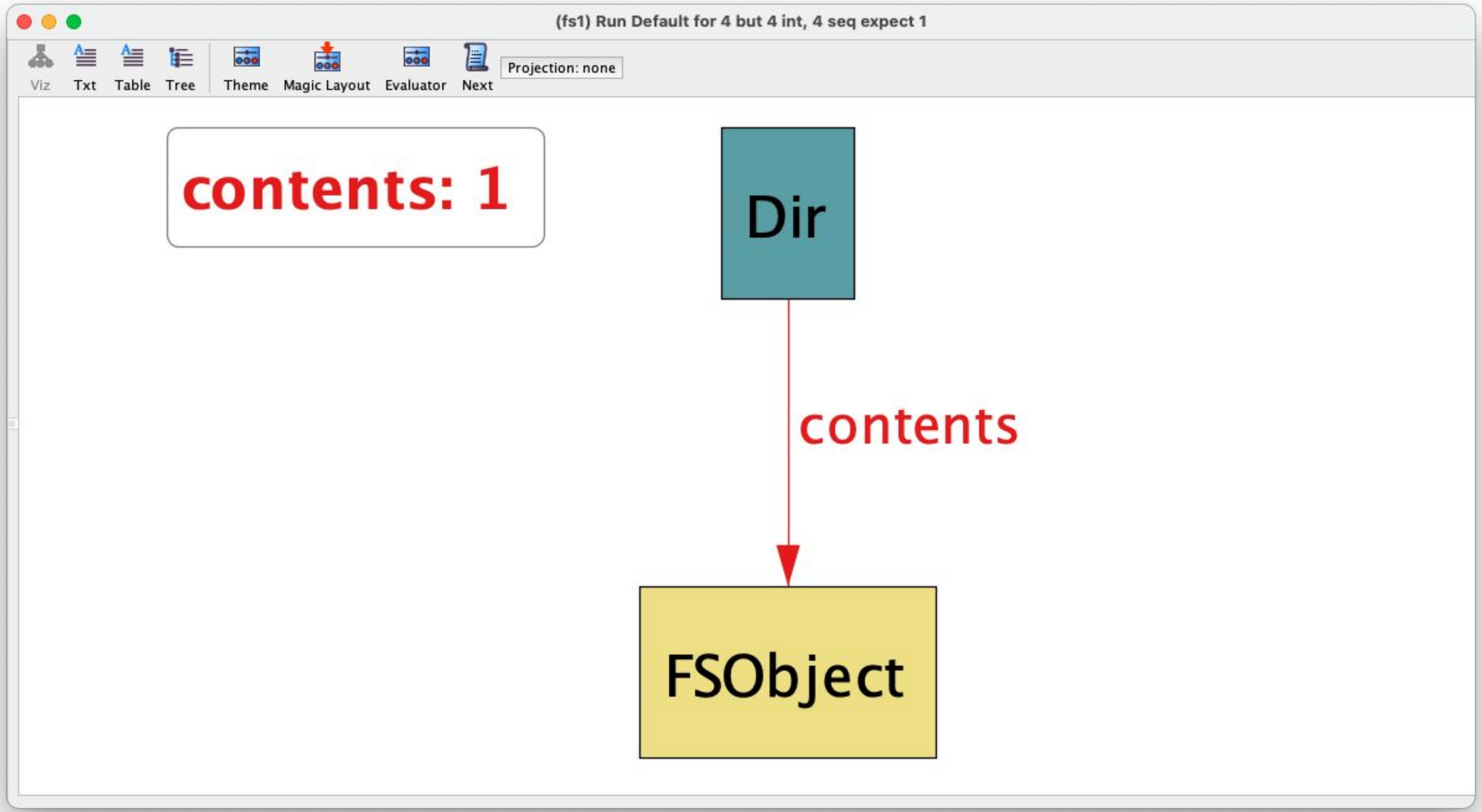
CSSSR

Интерактив для офлайн зрителей

https://github.com/AlloyTools/org.alloytools.alloy/releases

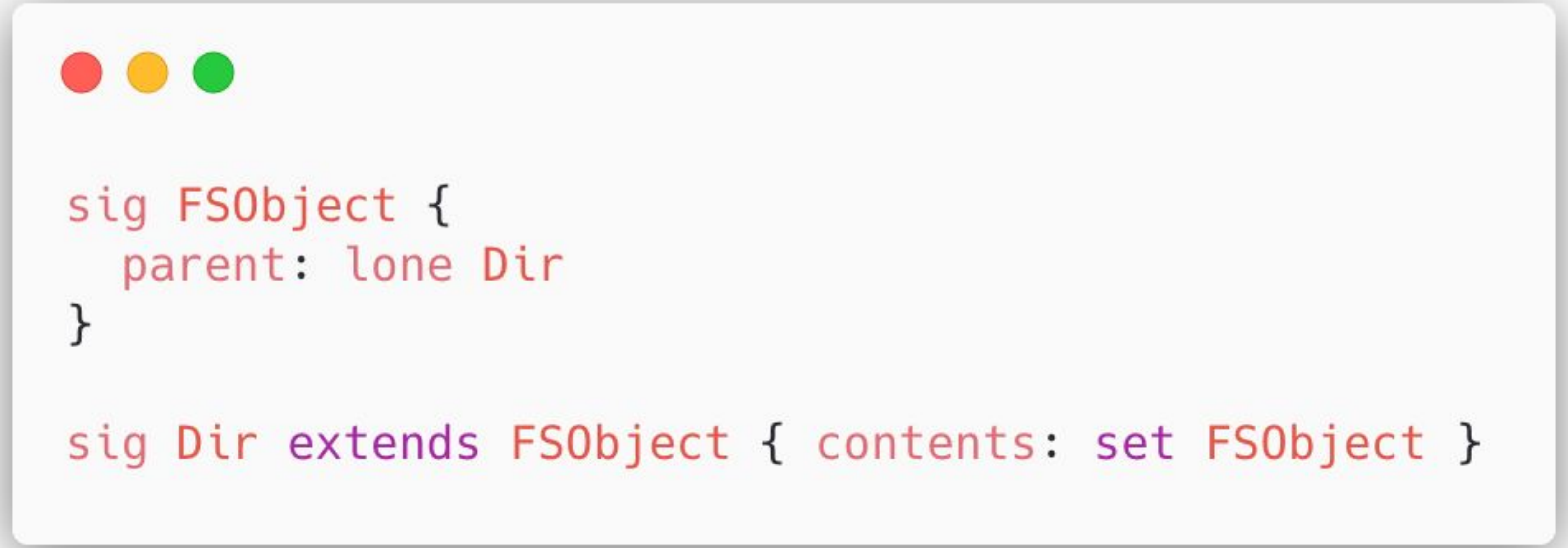
CSSSR

Модель файловой системы



CSSSR

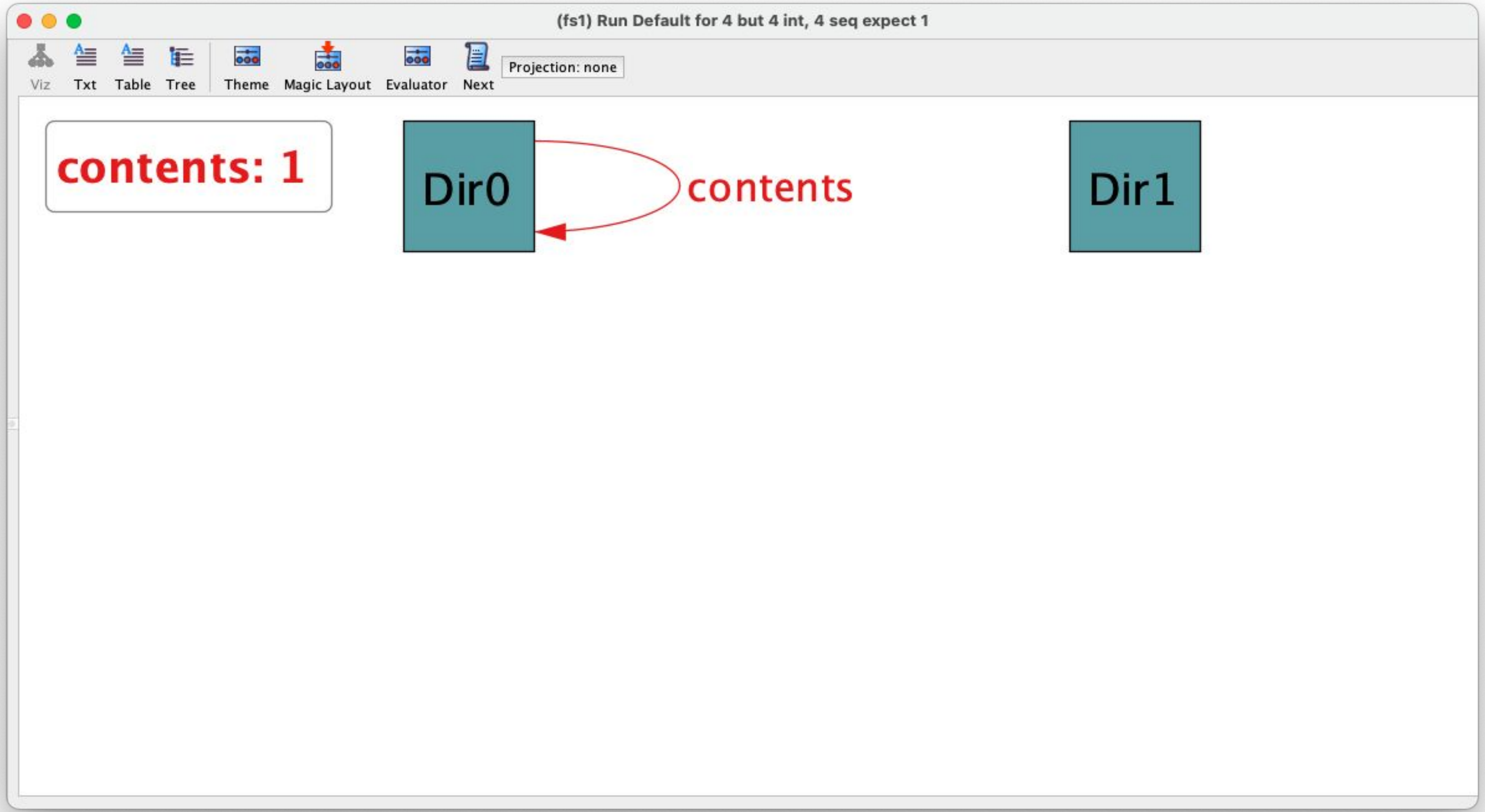
Модель файловой системы



```
sig FSObject {
  parent: lone Dir
}

sig Dir extends FSObject { contents: set FSObject }
```

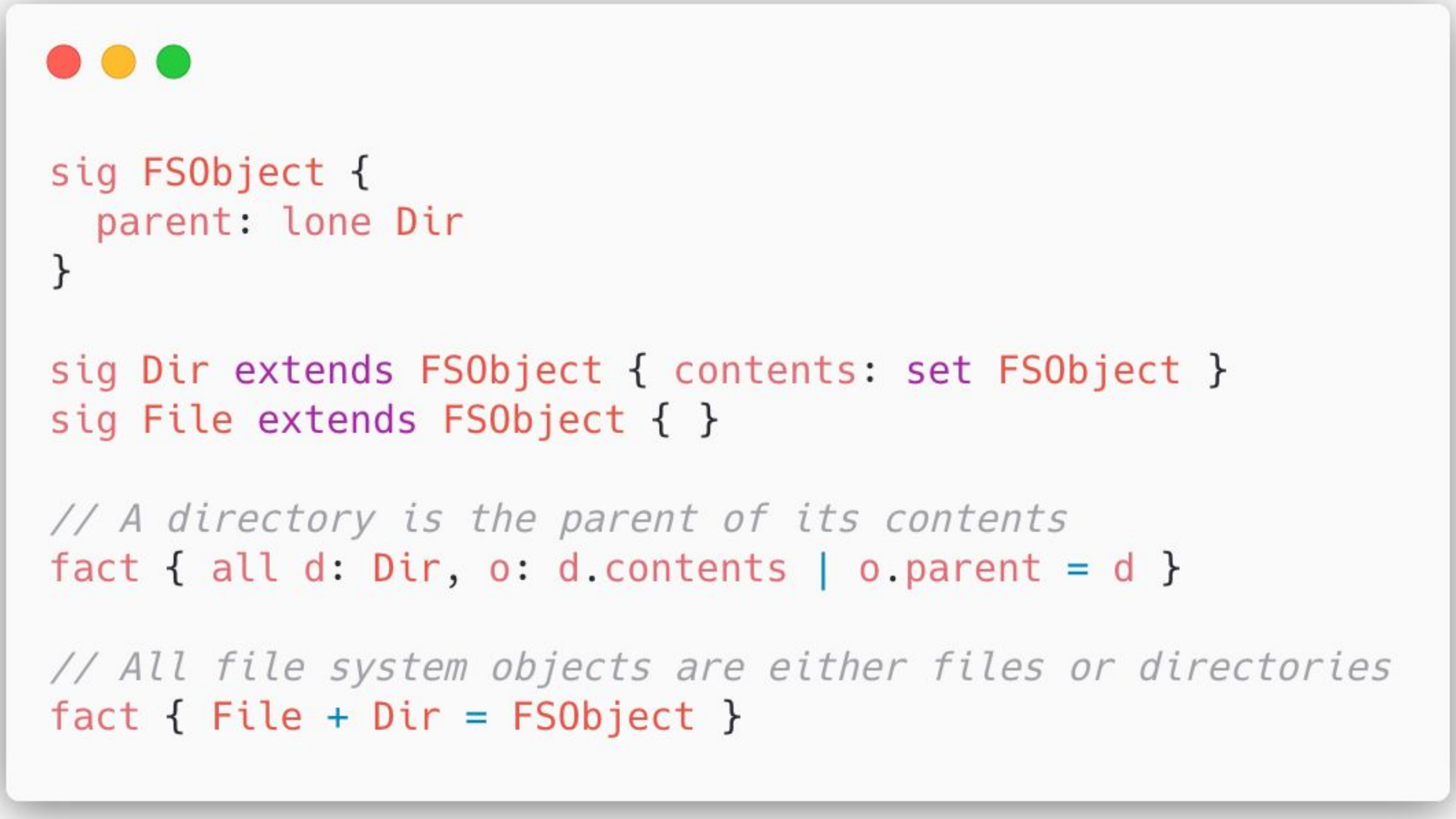
CSSSR

Что-то не то...


(fs1) Run Default for 4 but 4 int, 4 seq expect 1

CSSSR

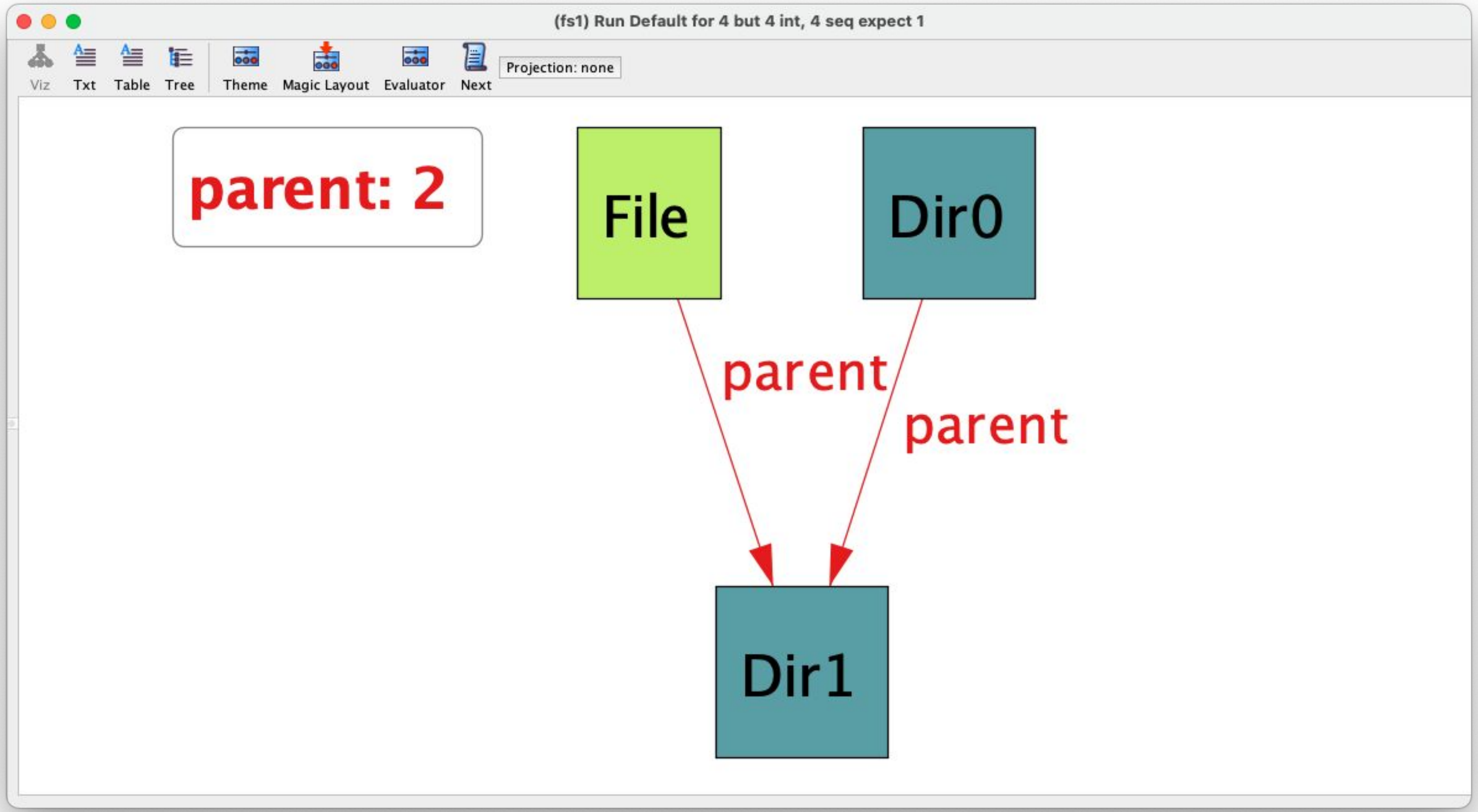
Дополняем модель



```
sig FSObject {
  parent: lone Dir
}

sig Dir extends FSObject { contents: set FSObject }
sig File extends FSObject { }

// A directory is the parent of its contents
fact { all d: Dir, o: d.contents | o.parent = d }

// All file system objects are either files or directories
fact { File + Dir = FSObject }
```

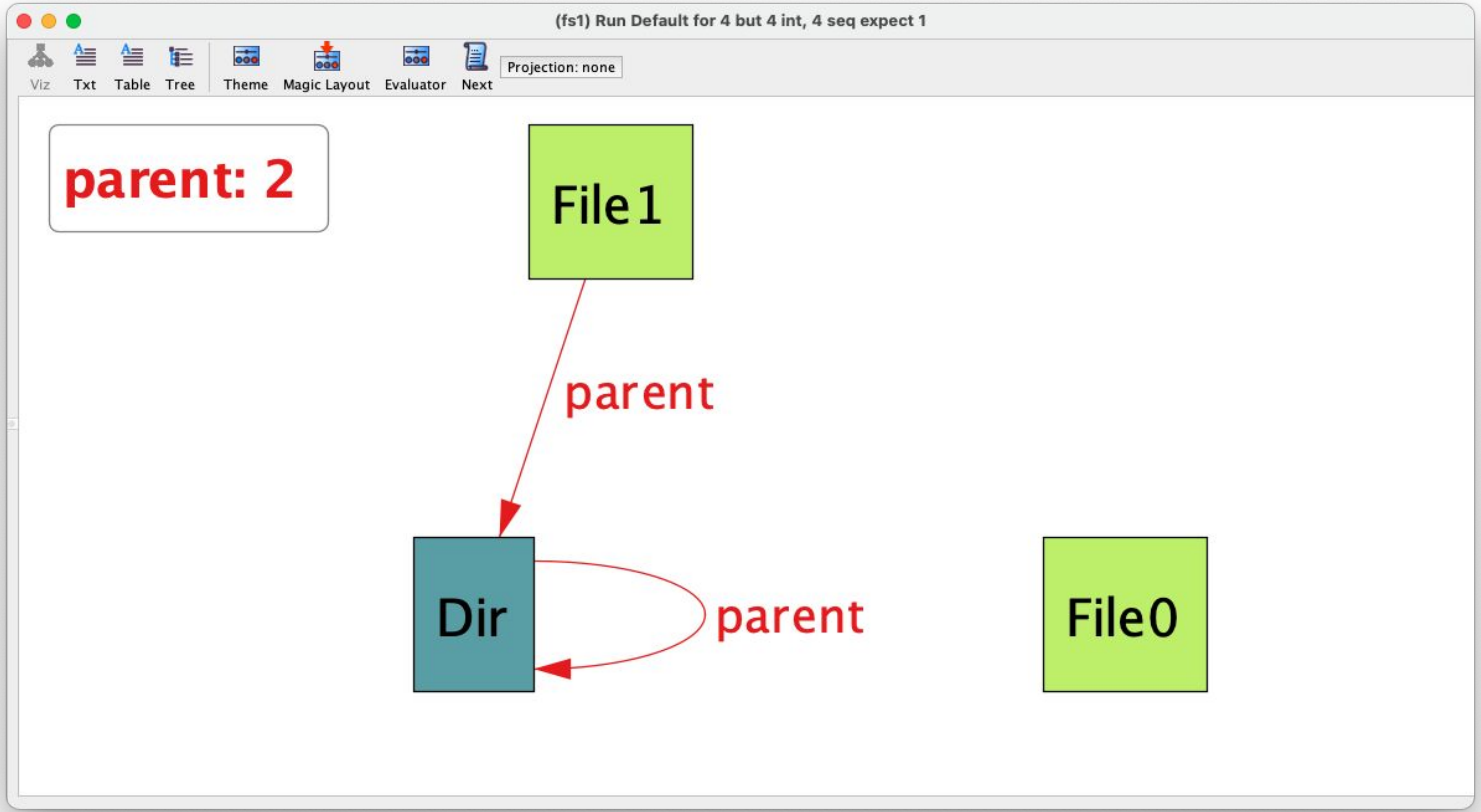
CSSSR

Дополняем модель


(fs1) Run Default for 4 but 4 int, 4 seq expect 1

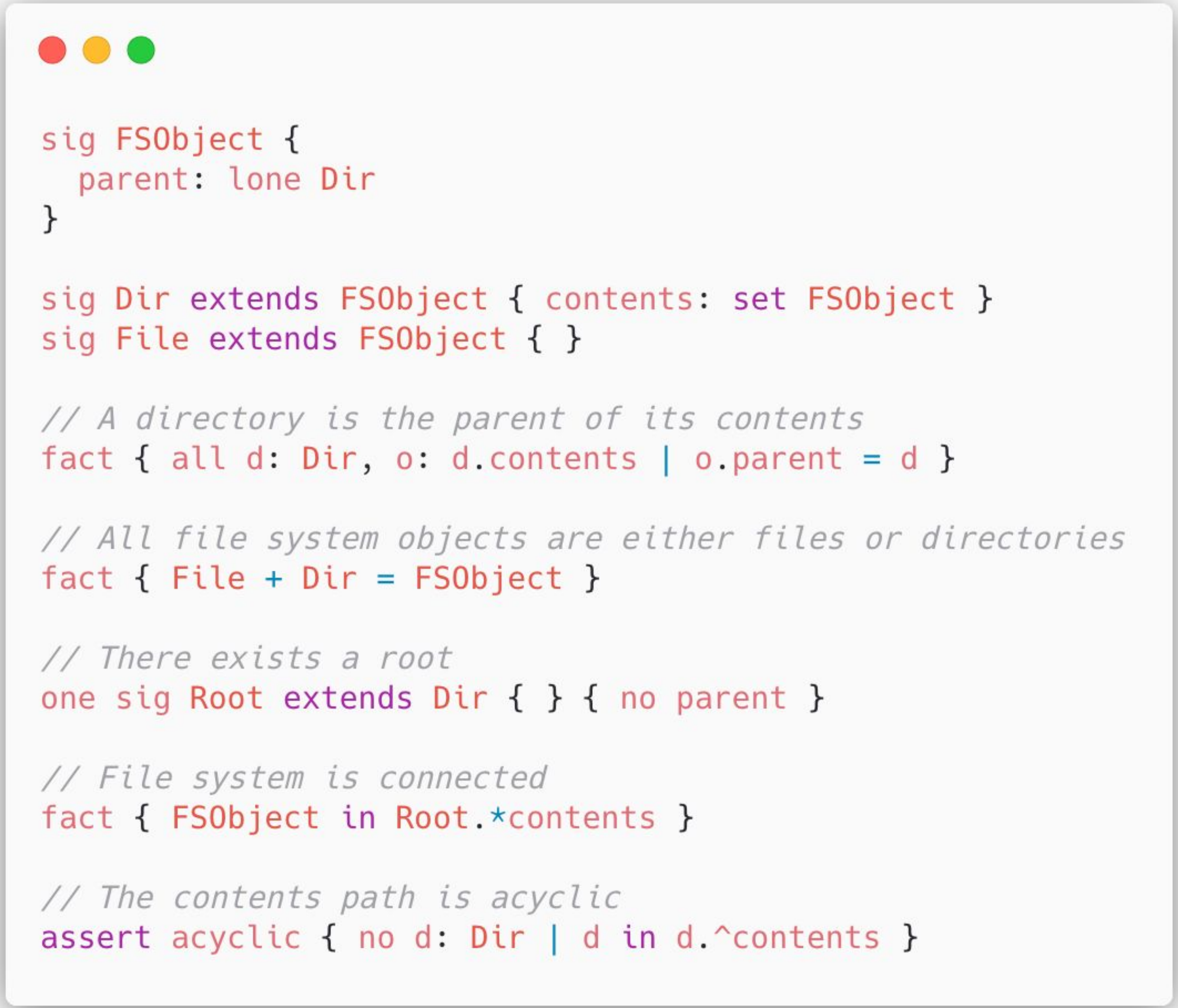
CSSSR

Уже лучше, но...


(fs1) Run Default for 4 but 4 int, 4 seq expect 1

CSSSR

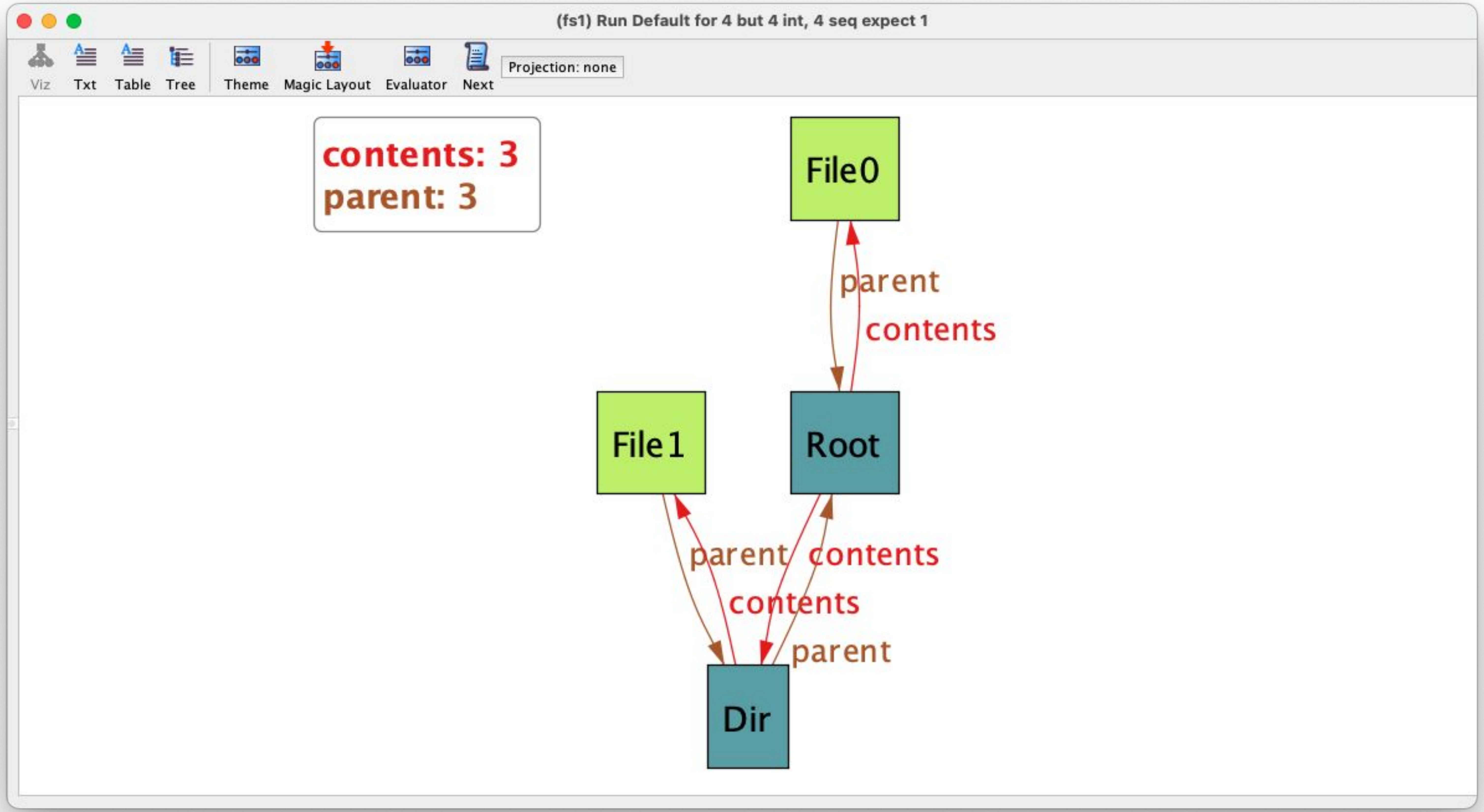
Уточним требования



```
sig FSObject {
  parent: lone Dir
}

sig Dir extends FSObject { contents: set FSObject }
sig File extends FSObject { }

// A directory is the parent of its contents
fact { all d: Dir, o: d.contents | o.parent = d }

// All file system objects are either files or directories
fact { File + Dir = FSObject }

// There exists a root
one sig Root extends Dir { } { no parent }

// File system is connected
fact { FSObject in Root.*contents }

// The contents path is acyclic
assert acyclic { no d: Dir | d in d.^contents }
```

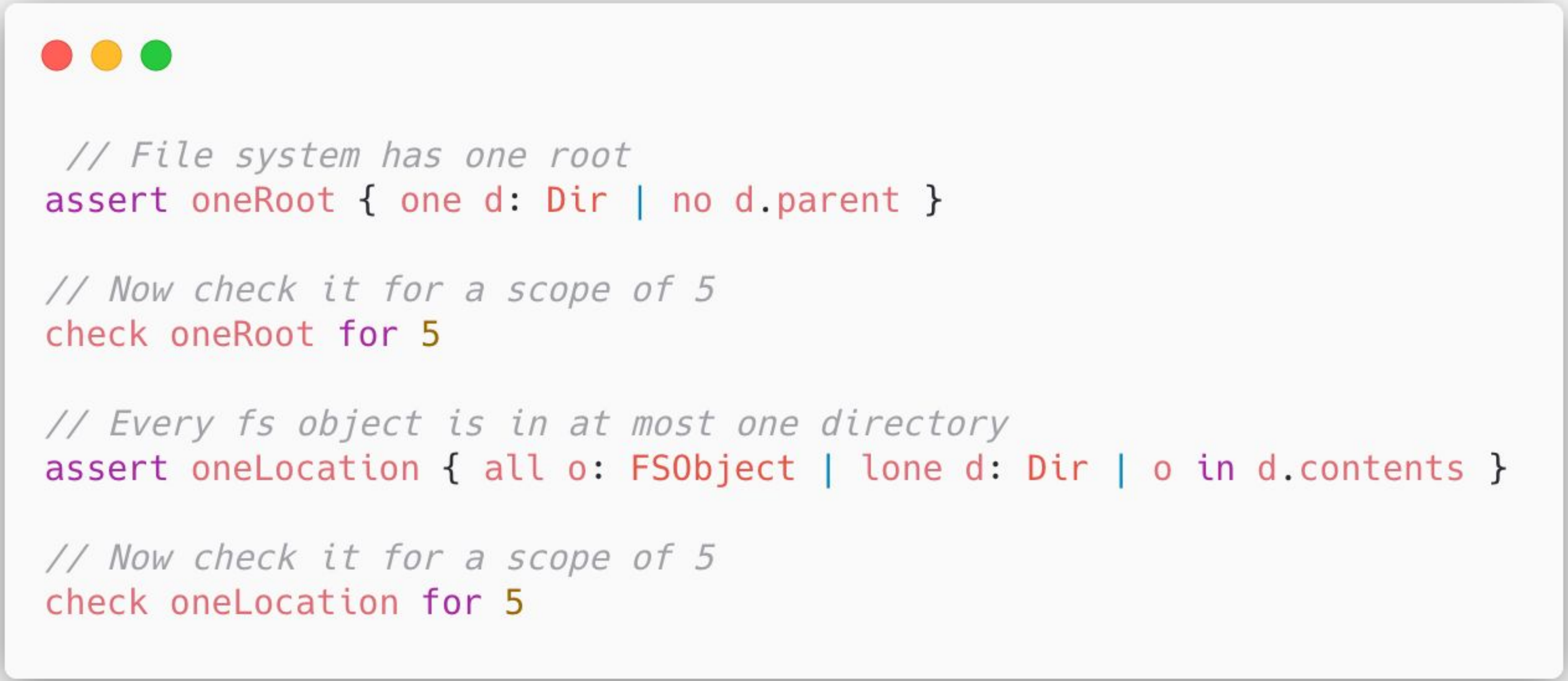
CSSSR

Уточним требования


(fs1) Run Default for 4 but 4 int, 4 seq expect 1

CSSSR

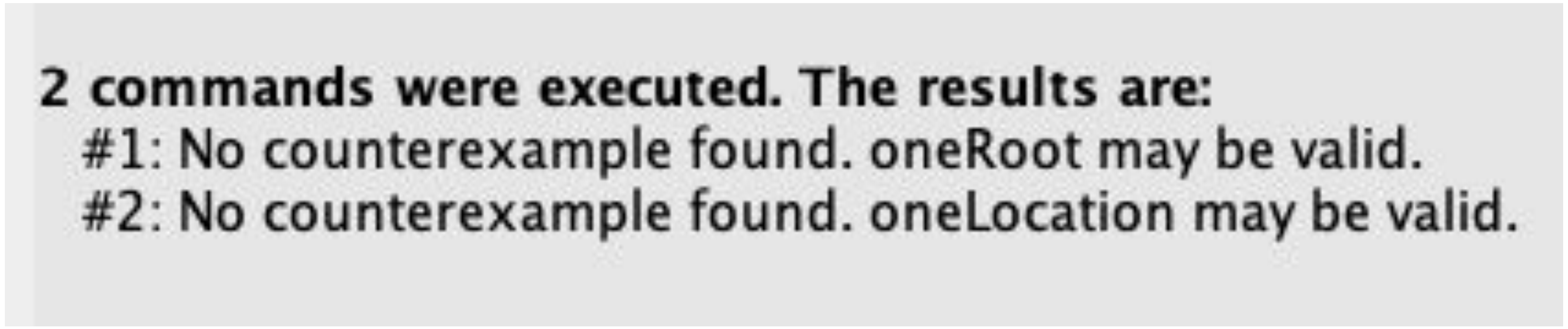
Теперь напишем тесты



```
 // File system has one root
assert oneRoot { one d: Dir | no d.parent }

// Now check it for a scope of 5
check oneRoot for 5

// Every fs object is in at most one directory
assert oneLocation { all o: FSObject | lone d: Dir | o in d.contents }

// Now check it for a scope of 5
check oneLocation for 5
```

CSSSR

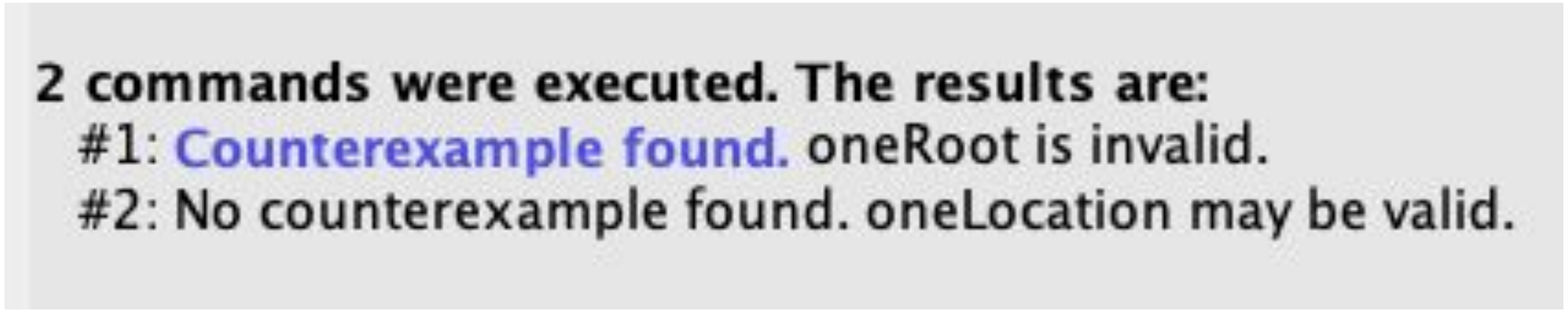
Теперь напишем тесты



```
2 commands were executed. The results are:
  #1: No counterexample found. oneRoot may be valid.
  #2: No counterexample found. oneLocation may be valid.
```

CSSSR

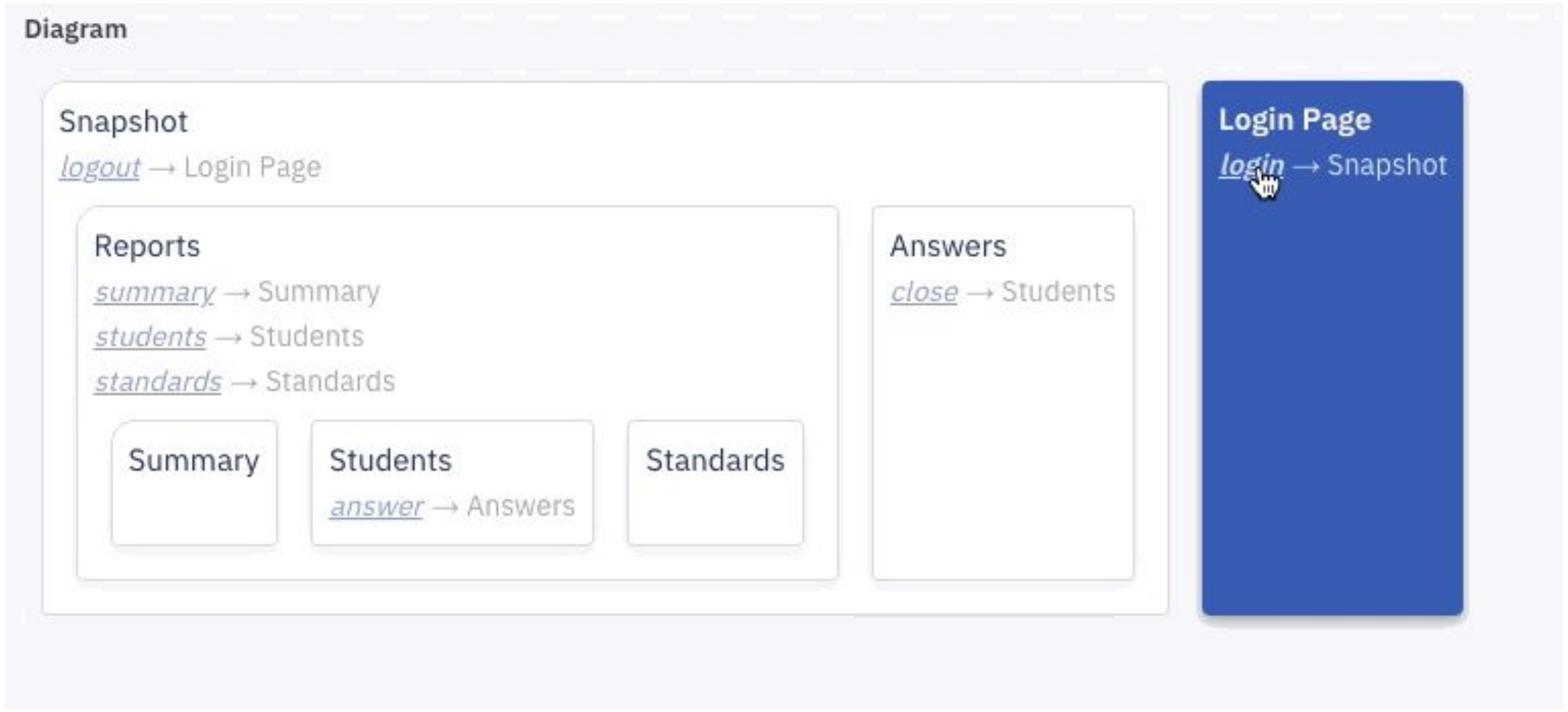
Сломаем модель



```
2 commands were executed. The results are:
  #1: Counterexample found. oneRoot is invalid.
  #2: No counterexample found. oneLocation may be valid.
```

CSSSR

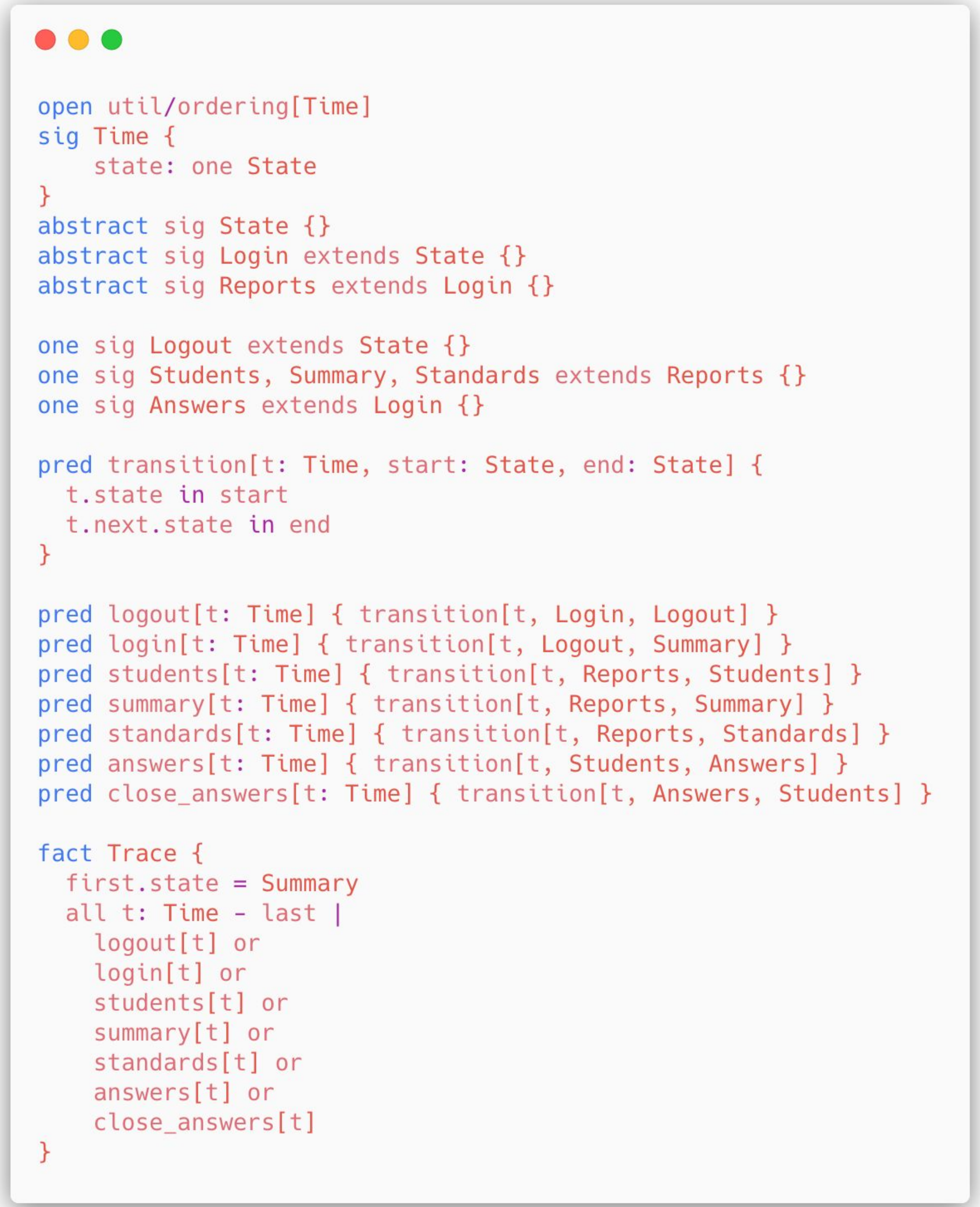
Ближе к вебу



Source: https://www.hillelwayne.com/post/formally-specifying-uis

CSSSR

Модель сайта

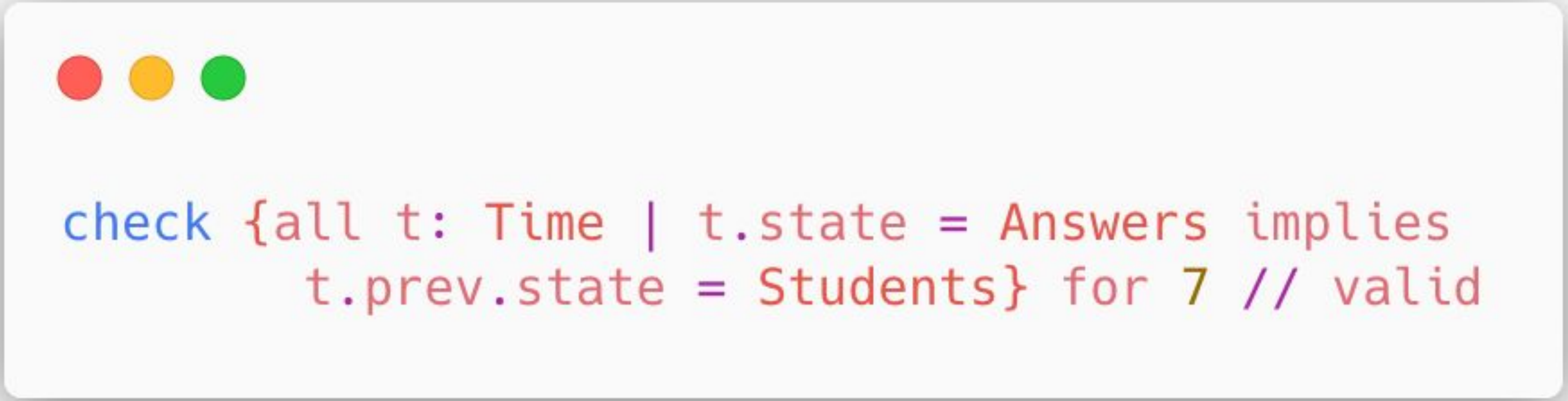


```
open util/ordering[Time]
sig Time {
    state: one State
}
abstract sig State {}
abstract sig Login extends State {}
abstract sig Reports extends Login {}

one sig Logout extends State {}
one sig Students, Summary, Standards extends Reports {}
one sig Answers extends Login {}

pred transition[t: Time, start: State, end: State] {
  t.state in start
  t.next.state in end
}

pred logout[t: Time] { transition[t, Login, Logout] }
pred login[t: Time] { transition[t, Logout, Summary] }
pred students[t: Time] { transition[t, Reports, Students] }
pred summary[t: Time] { transition[t, Reports, Summary] }
pred standards[t: Time] { transition[t, Reports, Standards] }
pred answers[t: Time] { transition[t, Students, Answers] }
pred close_answers[t: Time] { transition[t, Answers, Students] }

fact Trace {
  first.state = Summary
  all t: Time - last |
    logout[t] or
    login[t] or
    students[t] or
    summary[t] or
    standards[t] or
    answers[t] or
    close_answers[t]
}
```

Source: https://www.hillelwayne.com/post/formally-specifying-uis

CSSSR

Проверка факта



```
check {all t: Time | t.state = Answers implies
        t.prev.state = Students} for 7 // valid
```

Source: https://www.hillelwayne.com/post/formally-specifying-uis

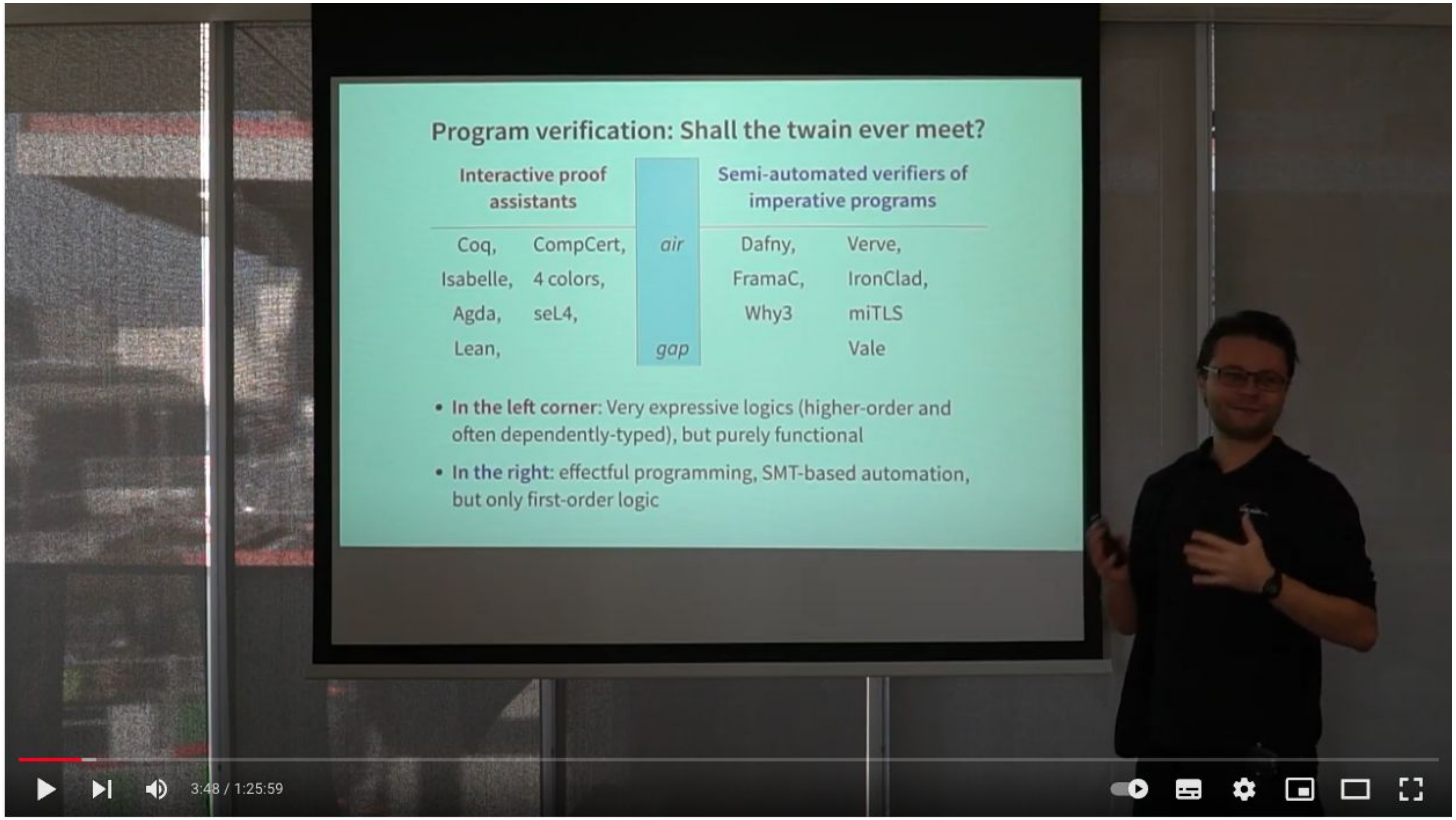
CSSSR

Почему разделение на разные технологии вредно?

CSSSR

F*

Ниша для таких языков



Program verification: Shall the twain ever meet?

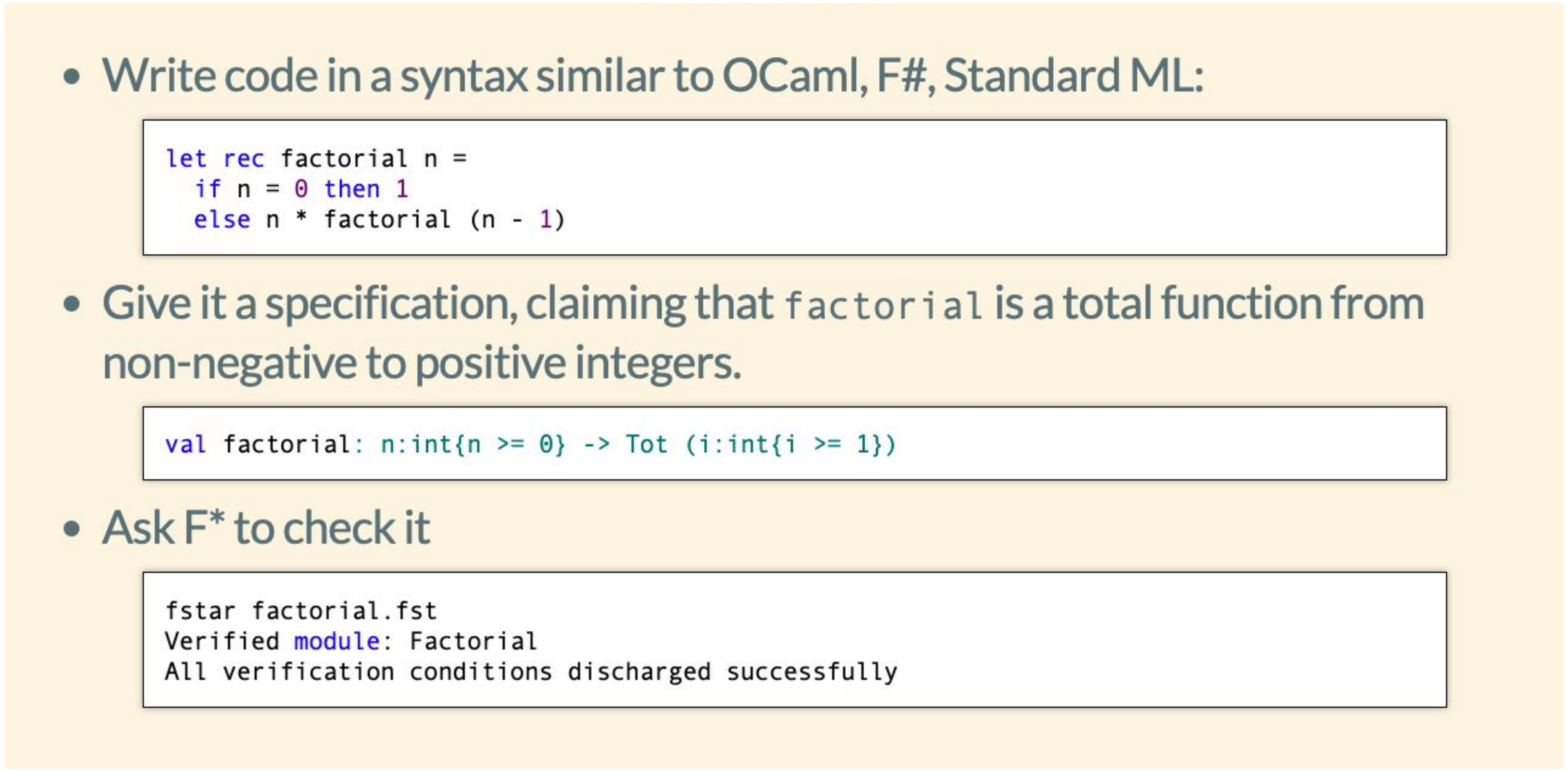
Interactive proof assistants		air gap	Semi-automated verifiers of imperative programs	
Coq,	CompCert,	air	Dafny,	Verve,
Isabelle,	4 colors,		FramaC,	IronClad,
Agda,	seL4,		Why3	miTLS
Lean,		gap		Vale

- **In the left corner**: Very expressive logics (higher-order and often dependently-typed), but purely functional
- **In the right**: effectful programming, SMT-based automation, but only first-order logic

Source: https://www.youtube.com/watch?v=SPCko8ACB0M&t=4449s&ab_channel=CatalinHritcu

CSSSR

A first taste



- Write code in a syntax similar to OCaml, F#, Standard ML:

```
let rec factorial n =
  if n = 0 then 1
  else n * factorial (n - 1)
```

- Give it a specification, claiming that `factorial` is a total function from non-negative to positive integers.

```
val factorial: n:int{n >= 0} -> Tot (i:int{i >= 1})
```

- Ask F* to check it

```
fstar factorial.fst
Verified module: Factorial
All verification conditions discharged successfully
```

Source: https://www.fstar-lang.org/gs2021/gs2021.html#/sec-a-first-taste

Почему не F*?

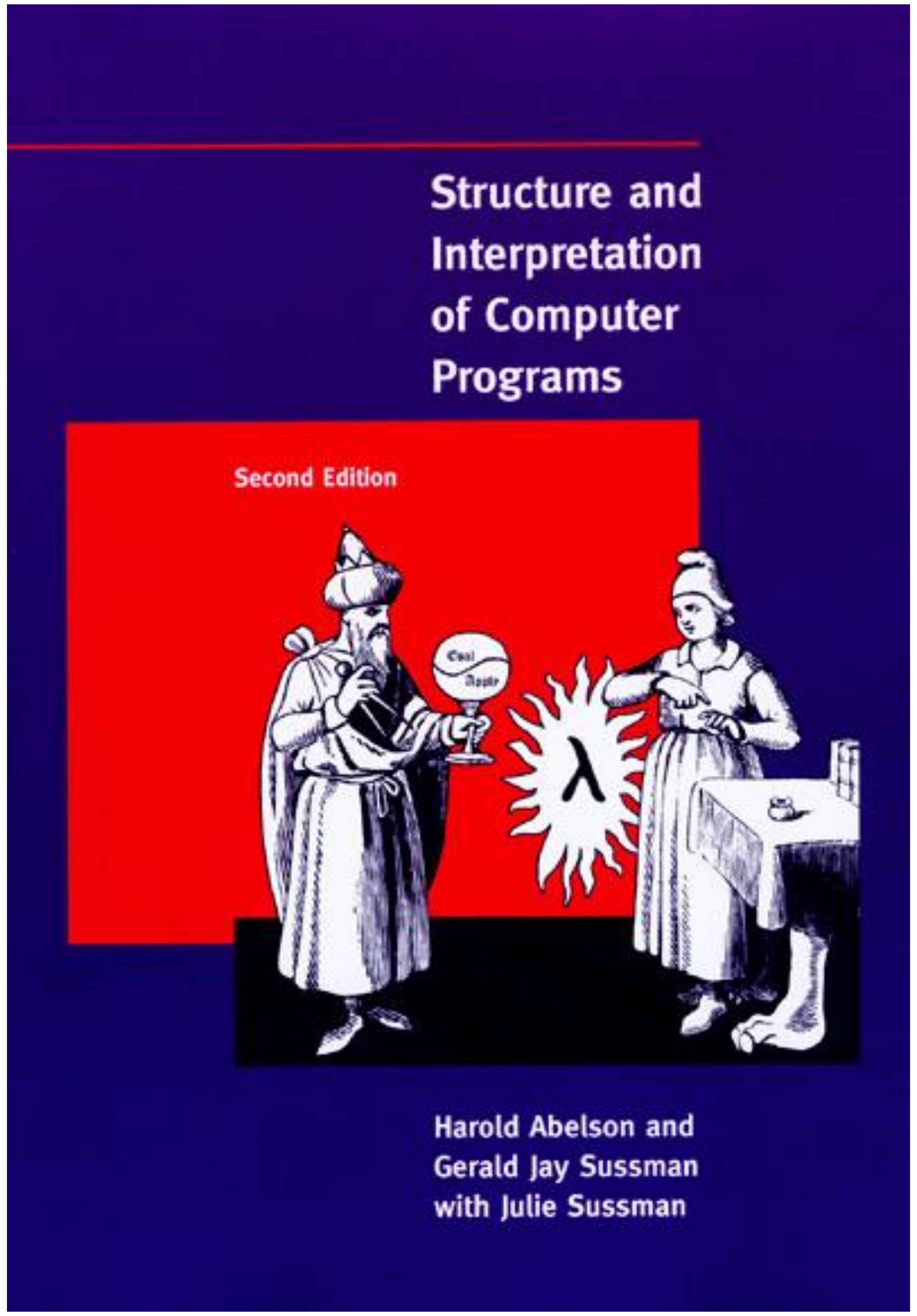
CSSSR

Писать что-то своё?

CSSSR

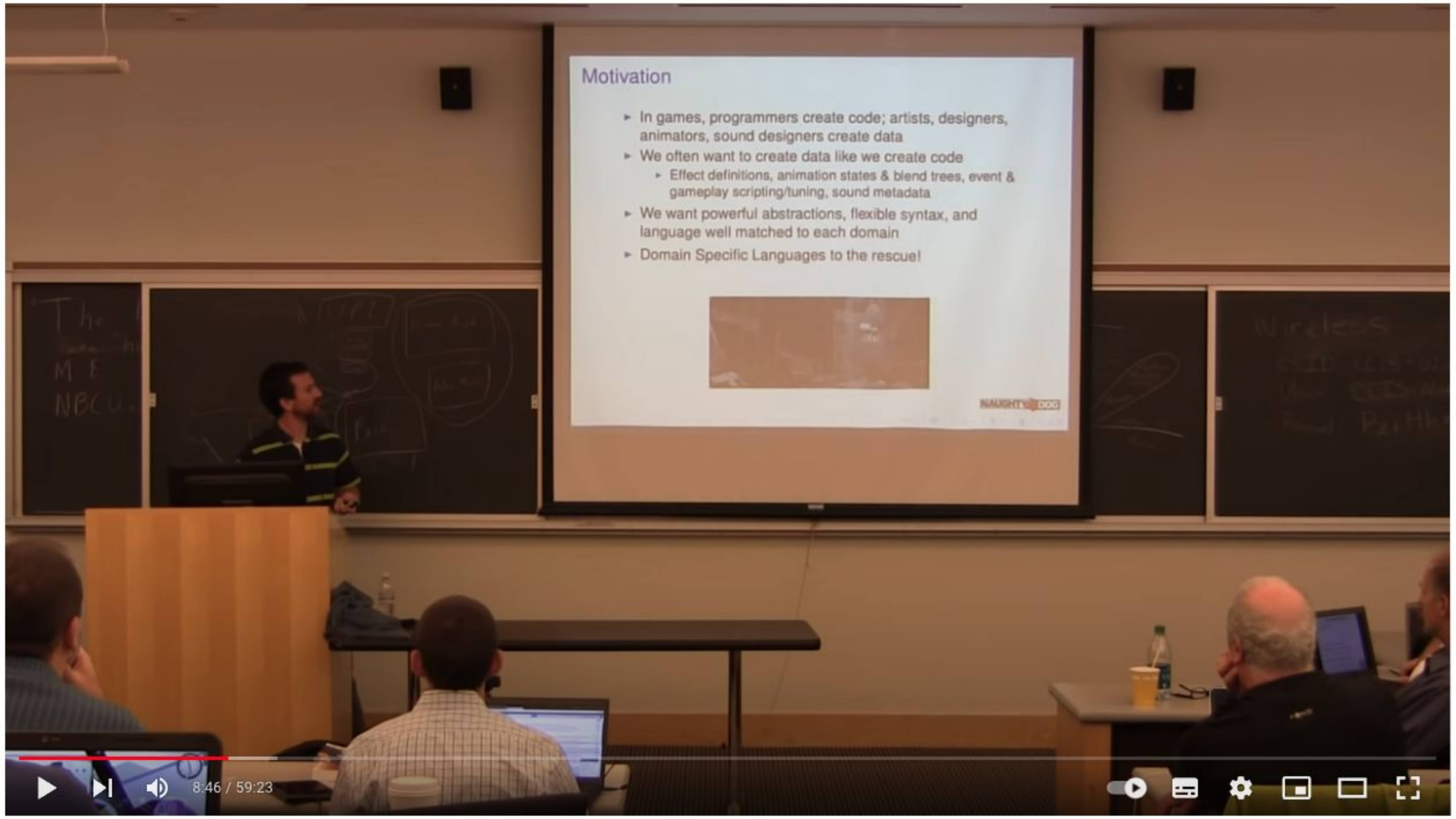
И тут на сцену выходит Racket!

SICP


Structure and Interpretation of Computer Programs
Second Edition
λ
Harold Abelson and Gerald Jay Sussman with Julie Sussman

Source: https://en.wikipedia.org/wiki/Structure_and_Interpretation_of_Computer_Programs#/media/File:SICP_cover.jpg

Naughty Dog



Motivation

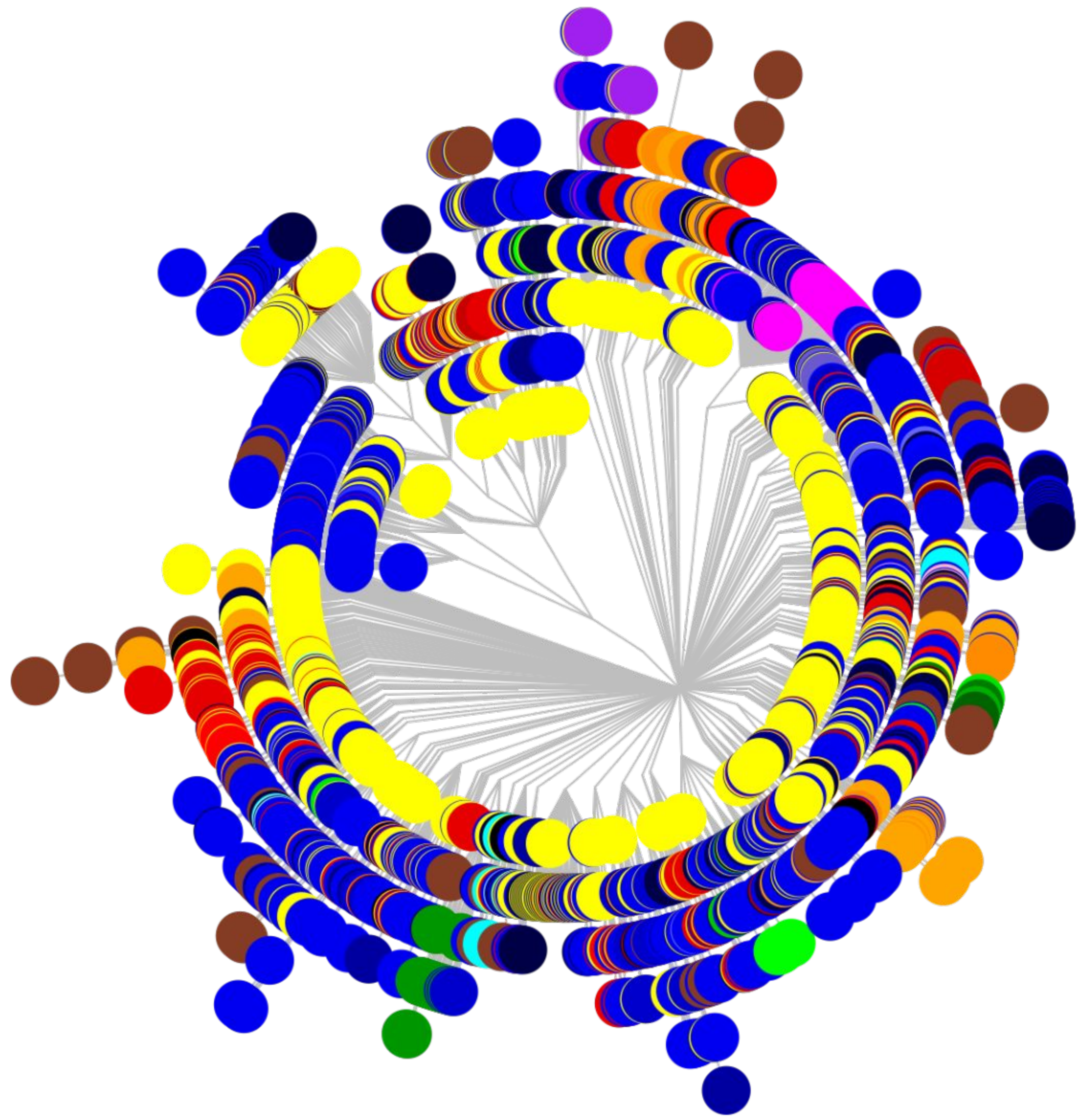
- In games, programmers create code; artists, designers, animators, sound designers create data
- We often want to create data like we create code
  - Effect definitions, animation states & blend trees, event & gameplay scripting/tuning, sound metadata
- We want powerful abstractions, flexible syntax, and language well matched to each domain
- Domain Specific Languages to the rescue!

RacketCon 2013: Dan Liebgold - Racket on the Playstation 3? It's Not What you Think!

Source: https://www.youtube.com/watch?v=oSmqbnhHp1c

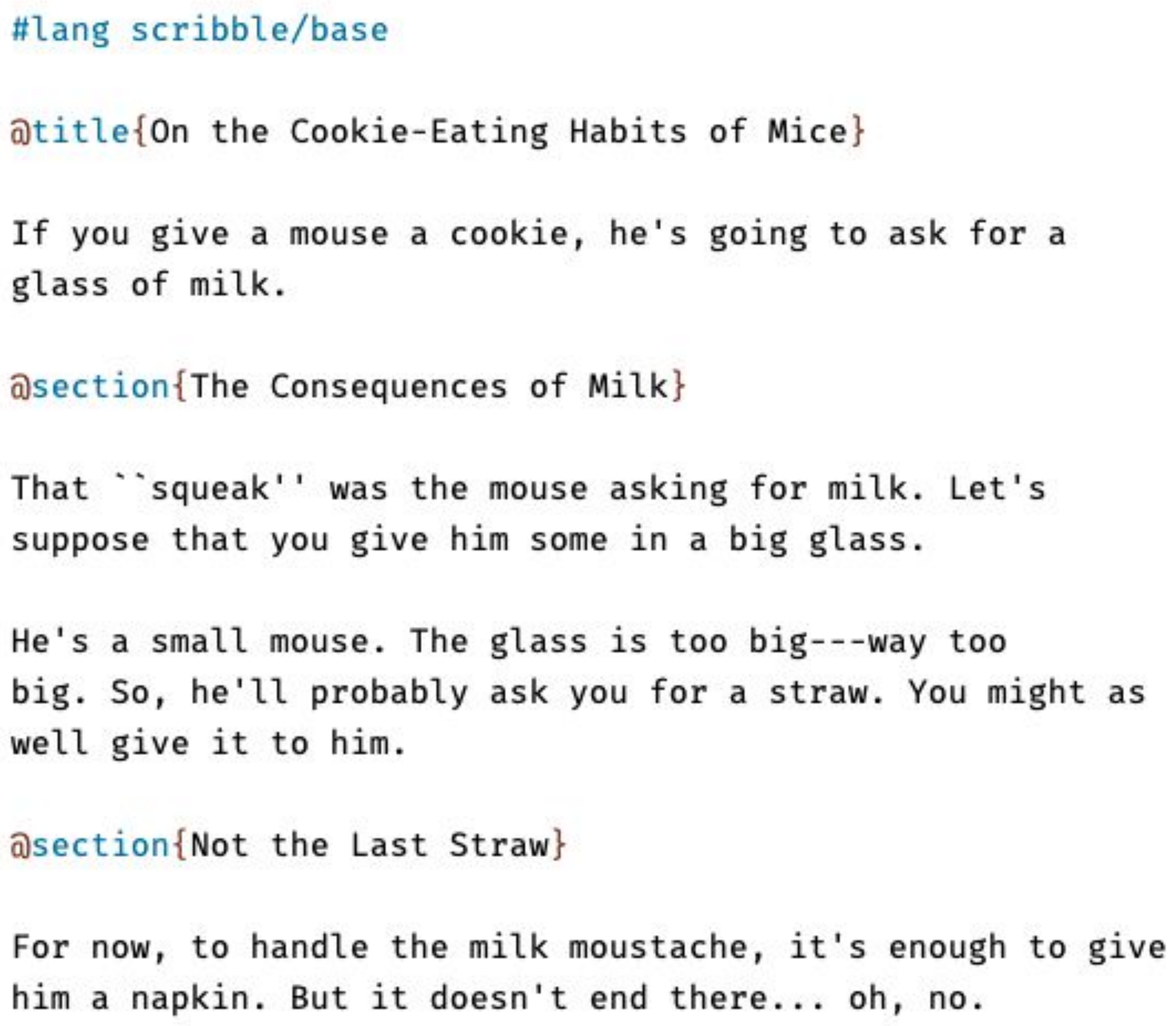
CSSSR

Language Oriented Programming Language



- ● racket
- ● mzscheme2.rkt
- ● racket/signature
- ● racket/gui
- ● racket/unit
- ● mzscheme
- ● racket/load
- ● racket/base
- ● 2d racket/base
- ● #%kernel
- ● racket/private/base
- ● pre-base.rkt
- ● racket/private/provider
- ● slideshow
- ● scribble/base
- ● at-exp racket
- ● at-exp scheme/base
- ● at-exp racket/base
- ● scribble/base/reader
- ● scribble/doc
- ● scribble/manual
- ● typed/racket
- ● typed-racket/extra-env-lang
- ● typed/racket/base
- ● typed-racket/minimal
- ● env-lang.rkt
- ● plai/collector
- ● plai/gc2/collector
- ● eopl
- ● plai/mutator
- ● htdp/isl+
- ● plai/gc2/mutator
- ● deinprogramm/DMdA
- ● plai
- ● htdp/bsl
- ● lazy/base
- ● srfi/provider
- ● r6rs
- ● r5rs
- ● racklog
- ● frtime/lang-utils
- ● frtime
- ● datalog/sexp
- ● swindle/turbo
- ● swindle/base
- ● frtime/frtime-lang-only
- ● datalog
- ● web-server/base
- ● web-server/insta
- ● web-server
- ● setup/infotab
- ● info
- ● wrapper.rkt
- ● wrap.rkt
- ● syntax/module-reader
- ● everything else

Scribble



```
#lang scribble/base

@title{On the Cookie-Eating Habits of Mice}

If you give a mouse a cookie, he's going to ask for a
glass of milk.

@section{The Consequences of Milk}

That ``squeak'' was the mouse asking for milk. Let's
suppose that you give him some in a big glass.

He's a small mouse. The glass is too big---way too
big. So, he'll probably ask you for a straw. You might as
well give it to him.

@section{Not the Last Straw}

For now, to handle the milk moustache, it's enough to give
him a napkin. But it doesn't end there... oh, no.
```

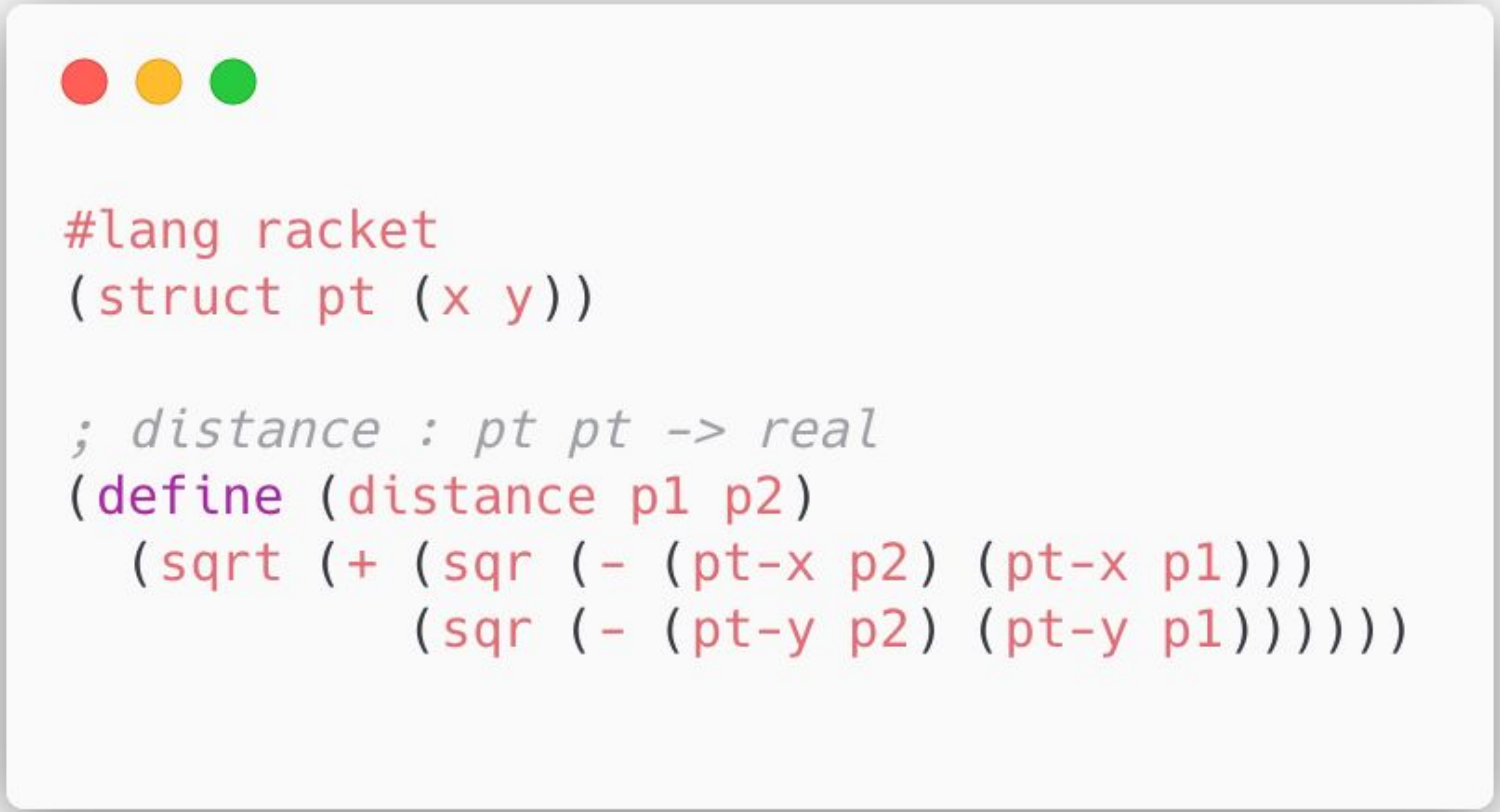
CSSSR

Fructure

```
escape \ → t a b → \ ↓ ↓ ↪ ...
```

CSSSR

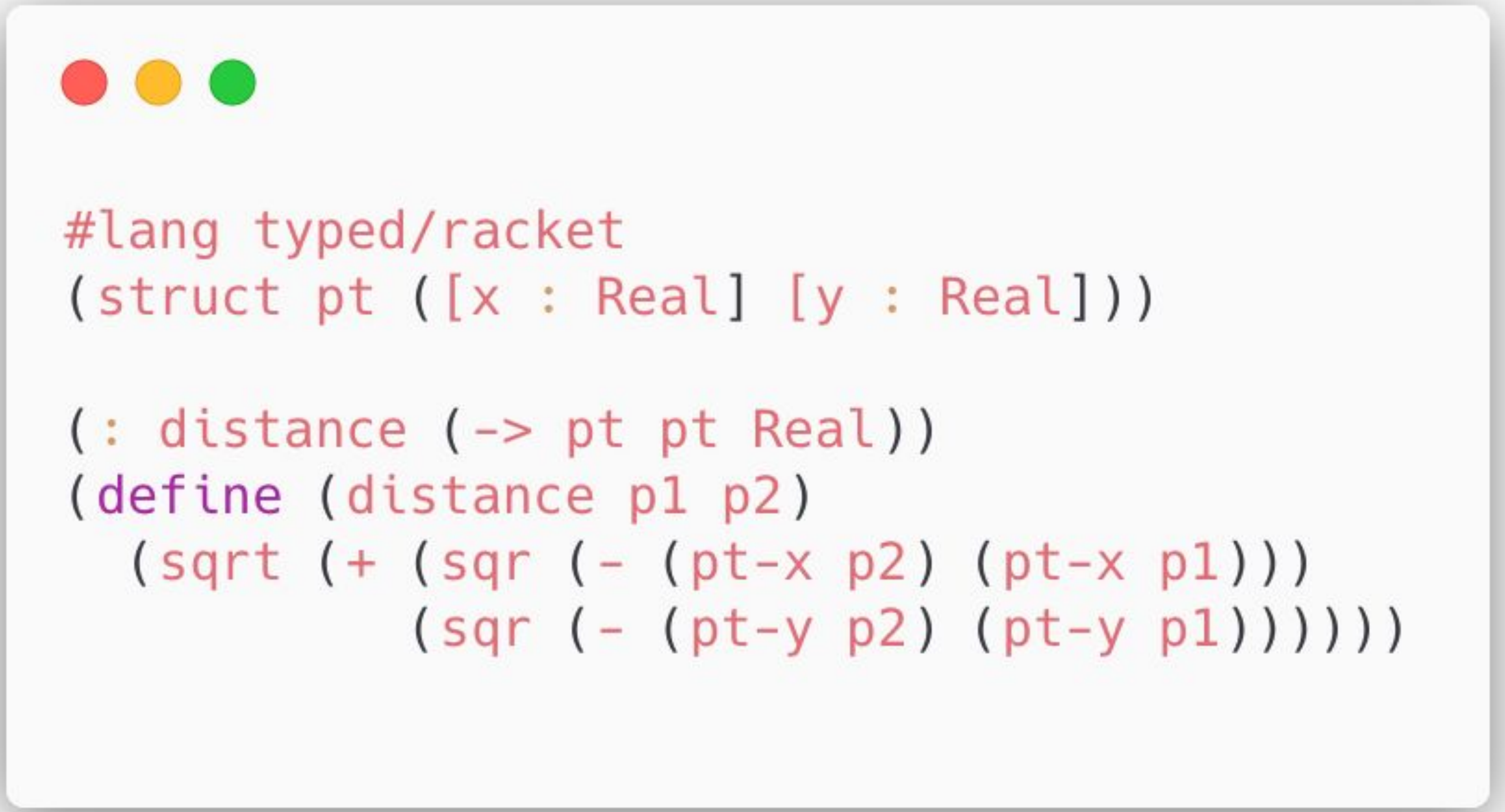
Racket без стат. типизации



```
#lang racket
(struct pt (x y))

; distance : pt pt -> real
(define (distance p1 p2)
  (sqrt (+ (sqr (- (pt-x p2) (pt-x p1)))
           (sqr (- (pt-y p2) (pt-y p1))))))
```

CSSSR

Racket со стат. типизацией

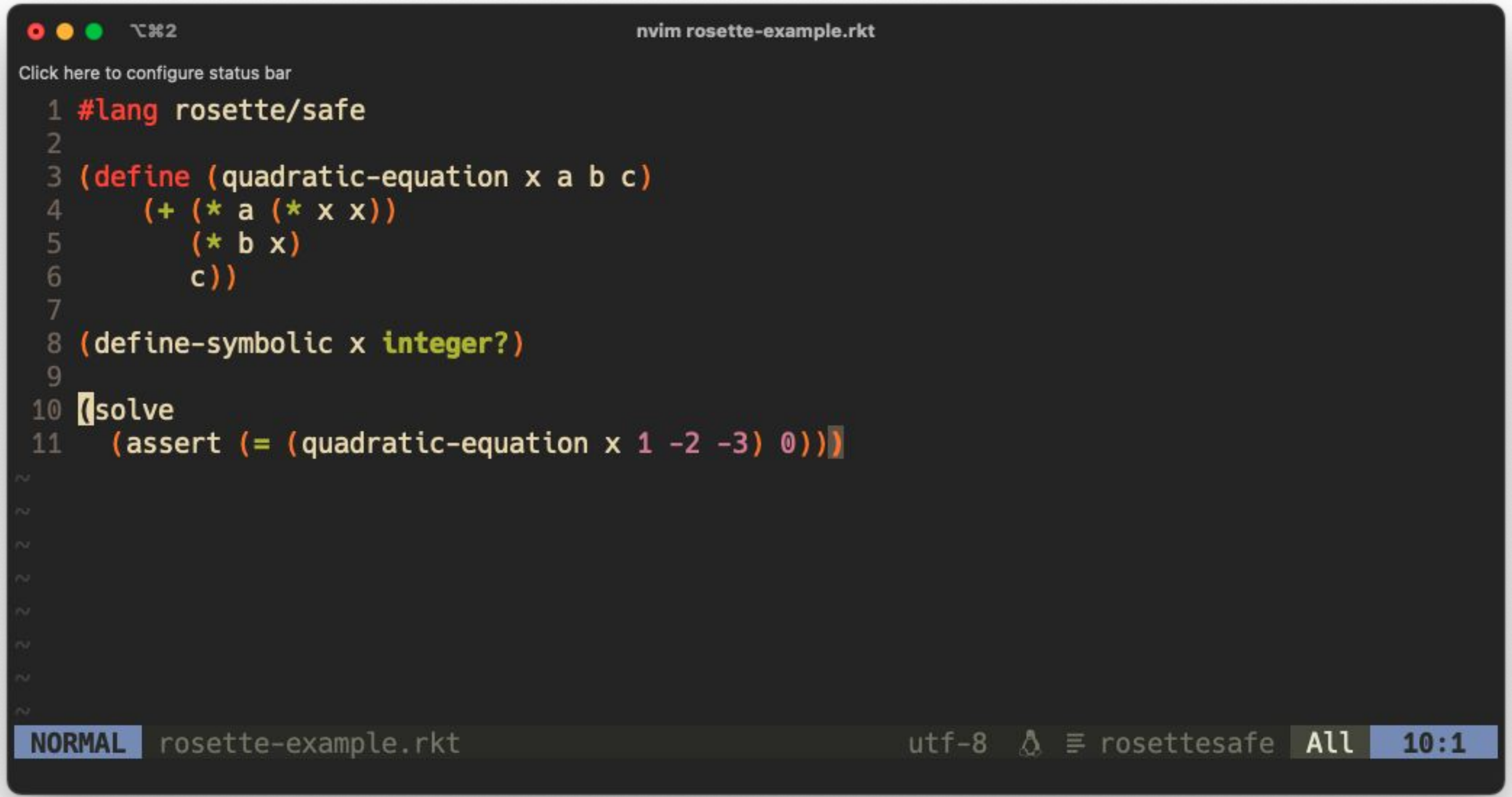


```
#lang typed/racket
(struct pt ([x : Real] [y : Real]))

(: distance (-> pt pt Real))
(define (distance p1 p2)
  (sqrt (+ (sqr (- (pt-x p2) (pt-x p1)))
           (sqr (- (pt-y p2) (pt-y p1))))))
```

А где Model Checking?

CSSSR

Rosette



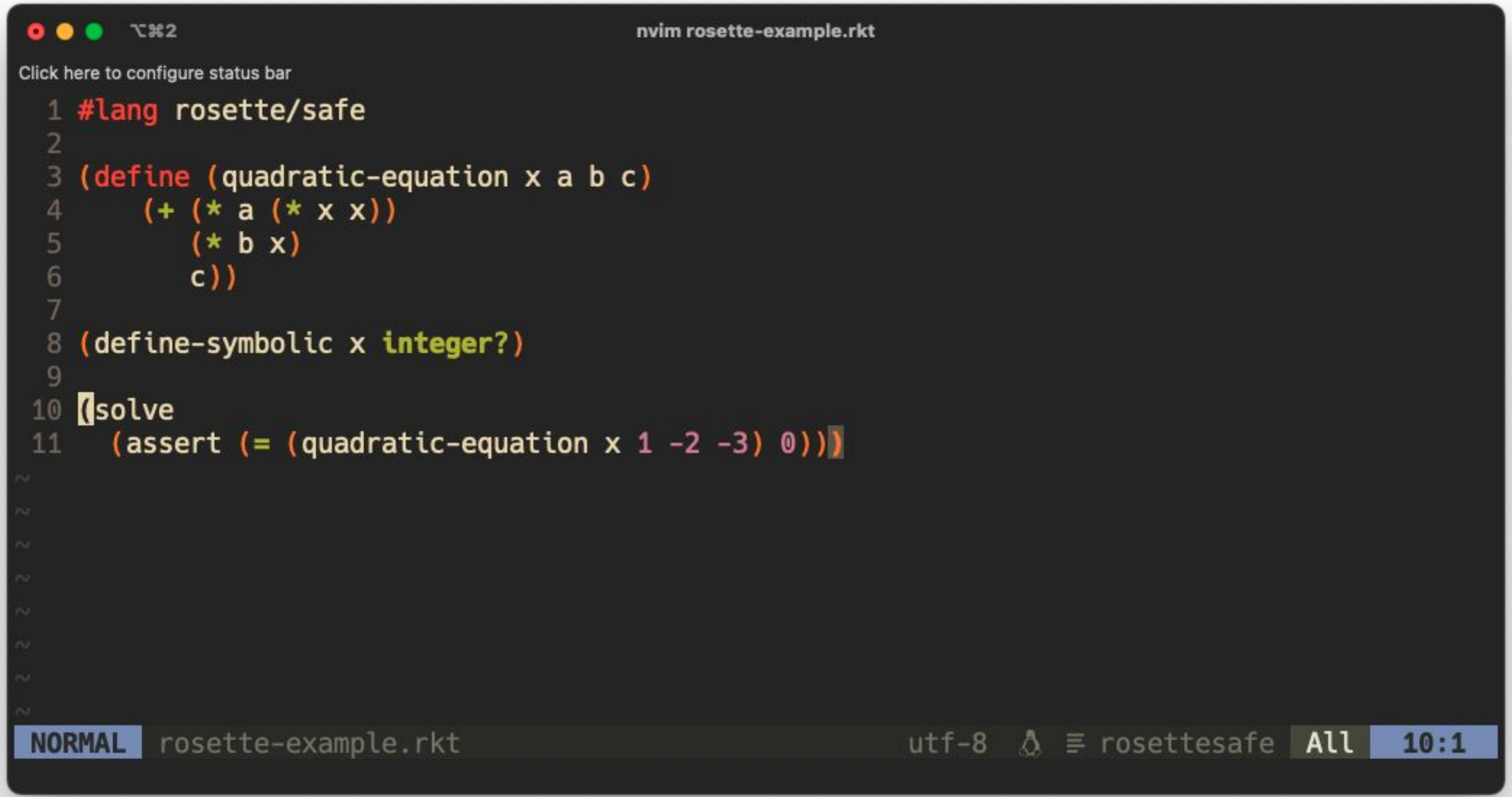
⌥⌘2
nvim rosette-example.rkt
Click here to configure status bar

```
#lang rosette/safe

(define (quadratic-equation x a b c)
    (+ (* a (* x x))
       (* b x)
       c))

(define-symbolic x integer?)

(solve
  (assert (= (quadratic-equation x 1 -2 -3) 0)))
```

NORMAL rosette-example.rkt utf-8 rosettesafe All 10:1

CSSSR

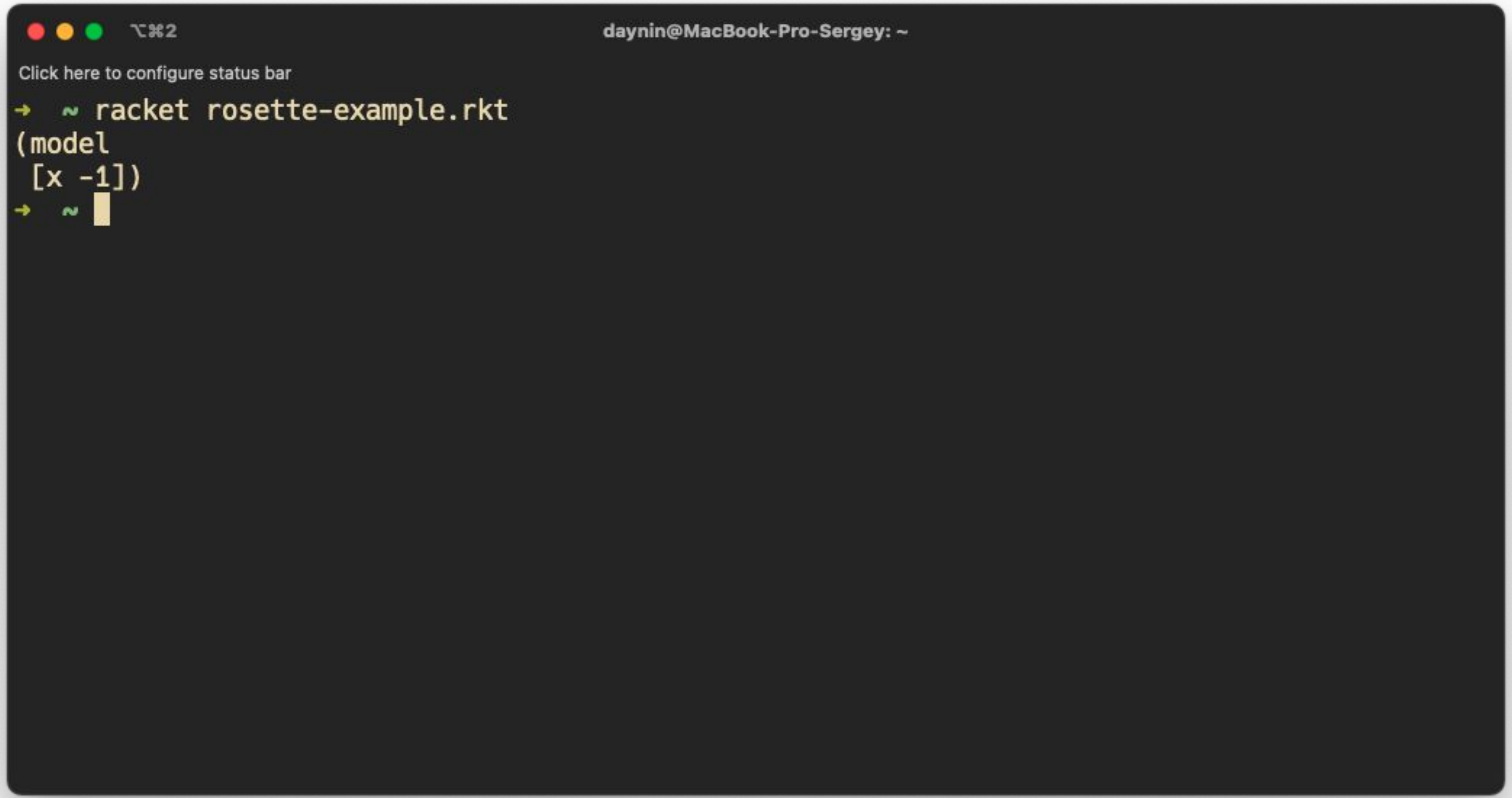
Rosette


⌥⌘2
nvim rosette-example.rkt
Click here to configure status bar

```
#lang rosette/safe

(define (quadratic-equation x a b c)
    (+ (* a (* x x))
       (* b x)
       c))

(define-symbolic x integer?)

(solve
  (assert (= (quadratic-equation x 1 -2 -3) 0)))
```

NORMAL rosette-example.rkt utf-8 rosettesafe All 10:1

Rosette


⌥⌘2
daynin@MacBook-Pro-Sergey: ~
Click here to configure status bar

```
➜  ~ racket rosette-example.rkt
(model
 [x -1])
➜  ~ 
```

CSSSR

Не хватает последнего звена!

CSSSR

RacketScript

CSSSR

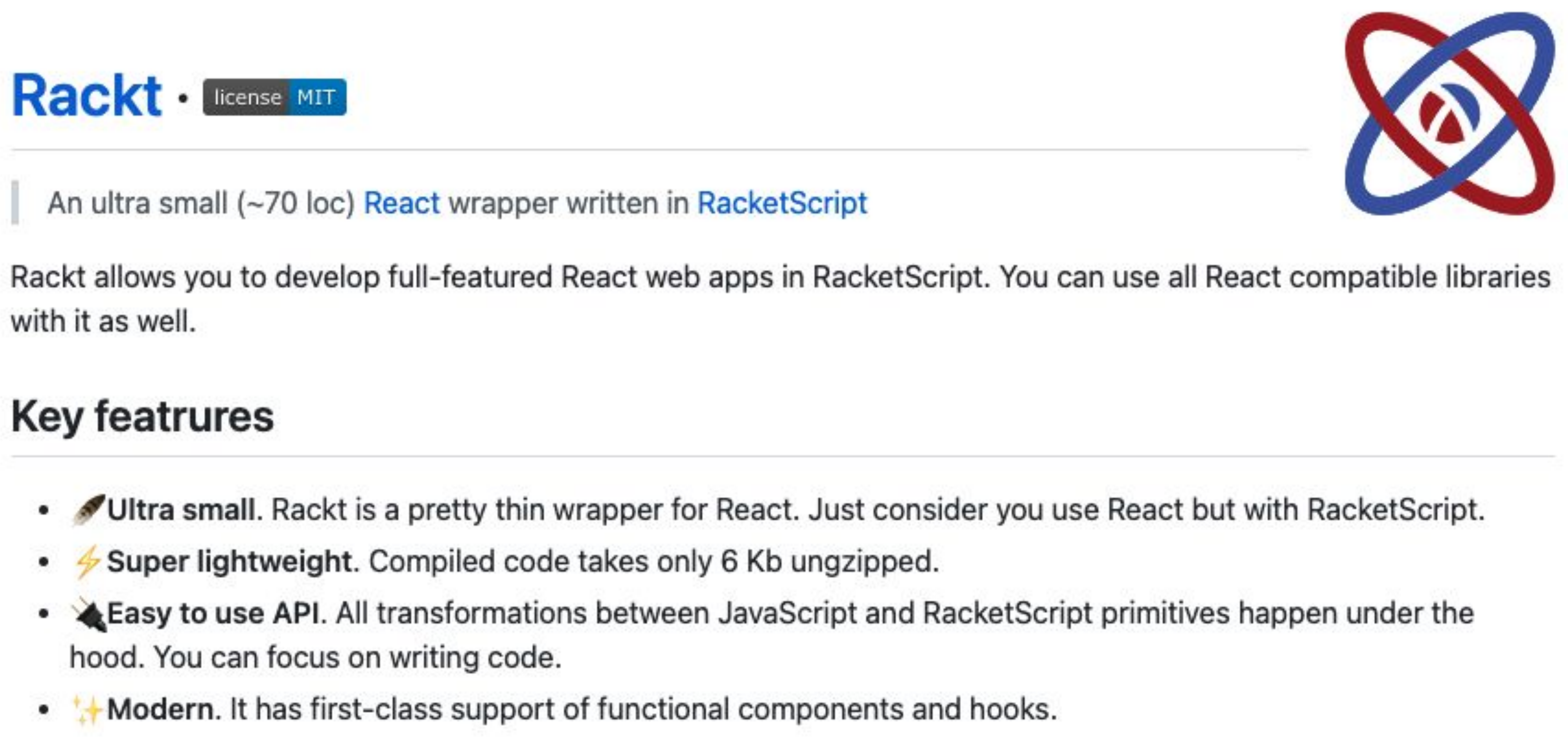
Автор — Vishesh Yadav

CSSSR

Уже production ready?

CSSSR

Rackt



Rackt • license MIT

> An ultra small (~70 loc) React wrapper written in RacketScript

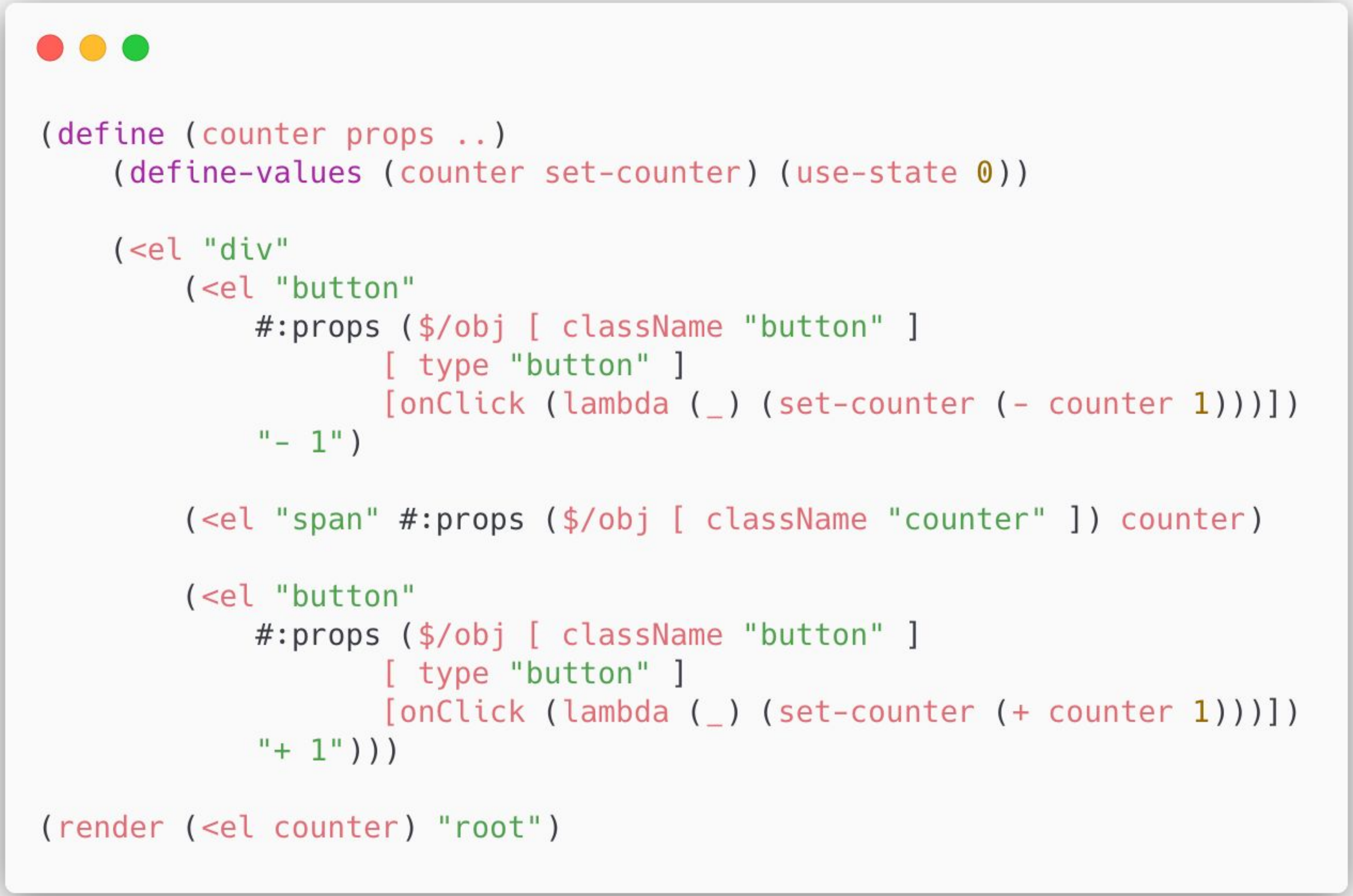
Rackt allows you to develop full-featured React web apps in RacketScript. You can use all React compatible libraries with it as well.

Key featrures

- 🪶**Ultra small**. Rackt is a pretty thin wrapper for React. Just consider you use React but with RacketScript.
- ⚡**Super lightweight**. Compiled code takes only 6 Kb ungzipped.
- 🔌**Easy to use API**. All transformations between JavaScript and RacketScript primitives happen under the hood. You can focus on writing code.
- ✨**Modern**. It has first-class support of functional components and hooks.

CSSSR

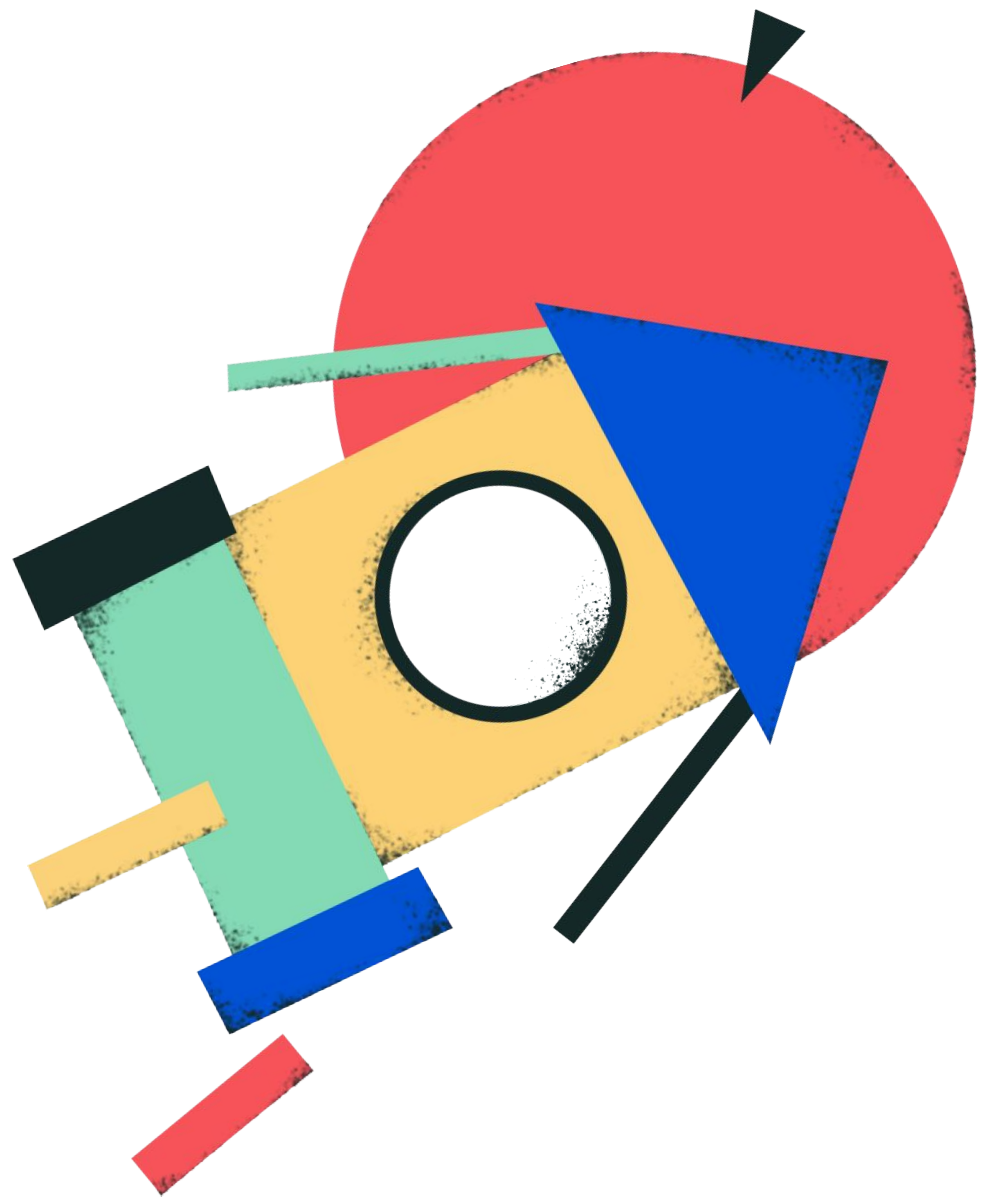
Rackt



```
(define (counter props ..)
    (define-values (counter set-counter) (use-state 0))

    (<el "div"
        (<el "button"
            #:props ($/obj [ className "button" ]
                           [ type "button" ]
                           [onClick (lambda (_) (set-counter (- counter 1)))])
            "- 1")

        (<el "span" #:props ($/obj [ className "counter" ]) counter)

        (<el "button"
            #:props ($/obj [ className "button" ]
                           [ type "button" ]
                           [onClick (lambda (_) (set-counter (+ counter 1)))])
            "+ 1")))

(render (<el counter) "root")
```

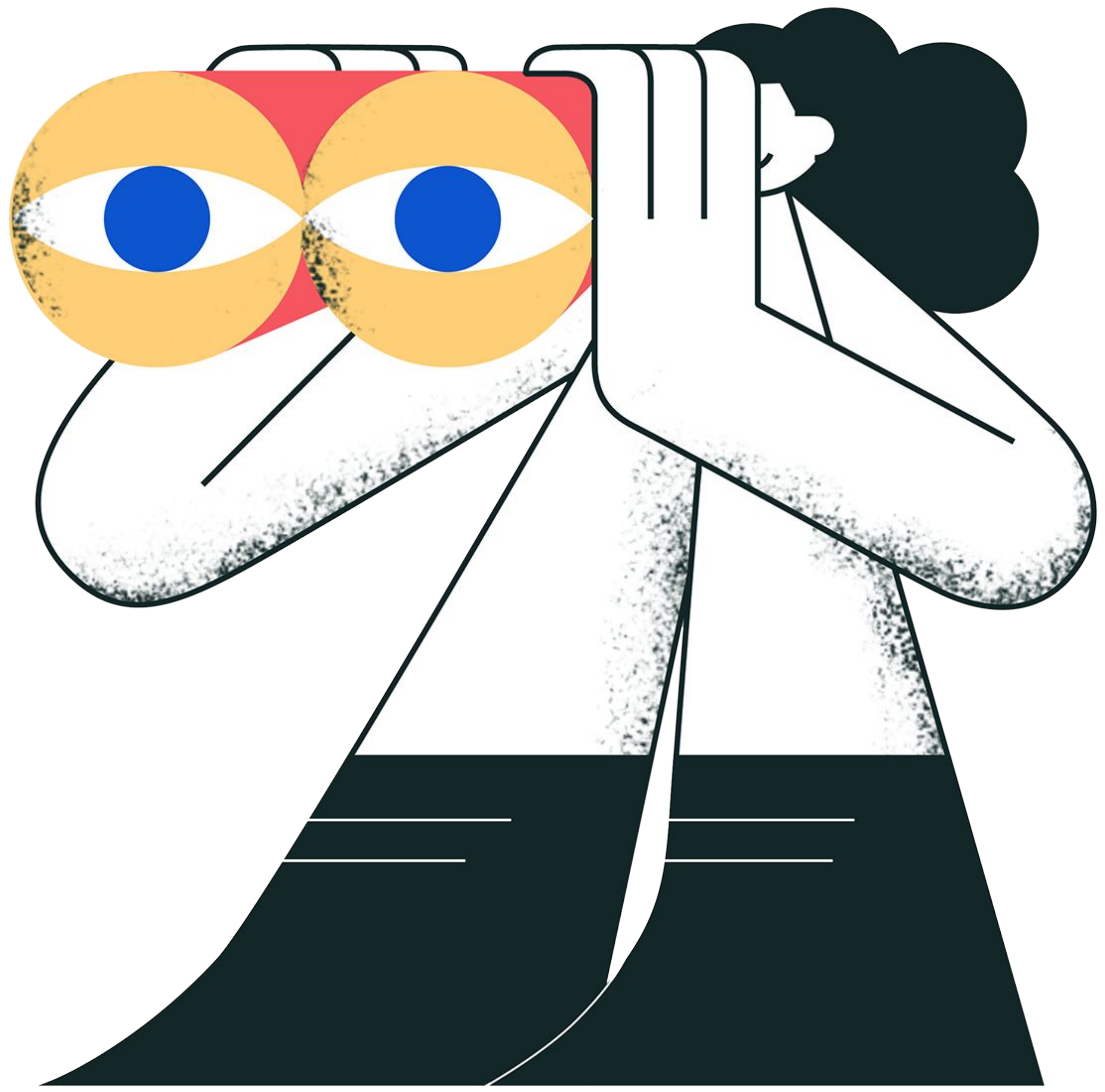
CSSSR

Demo time!



CSSSR

Что дальше?

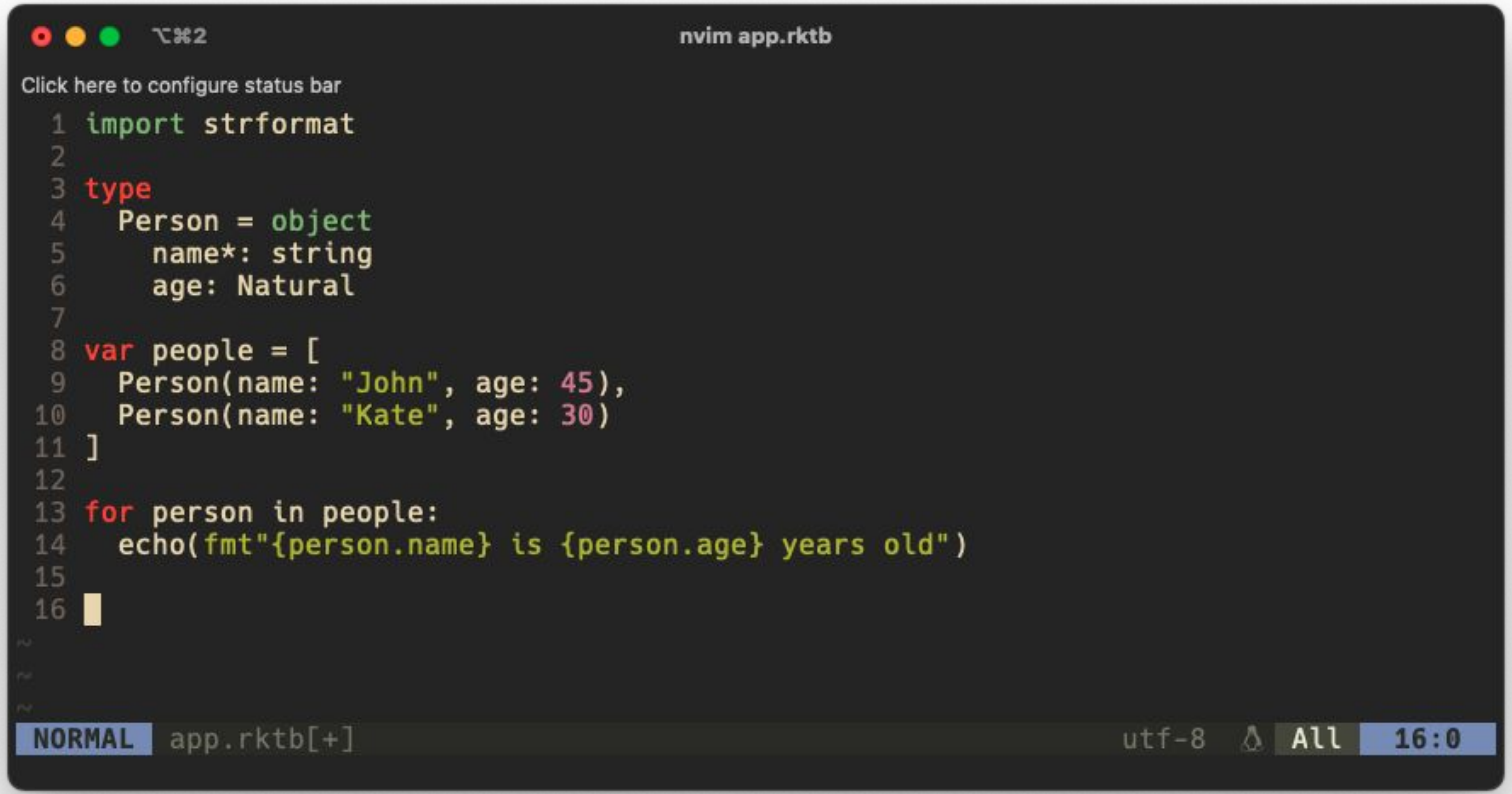


CSSSR

Немного пофантазируем

CSSSR

RacketScript based language



⌥⌘2 nvim app.rktb

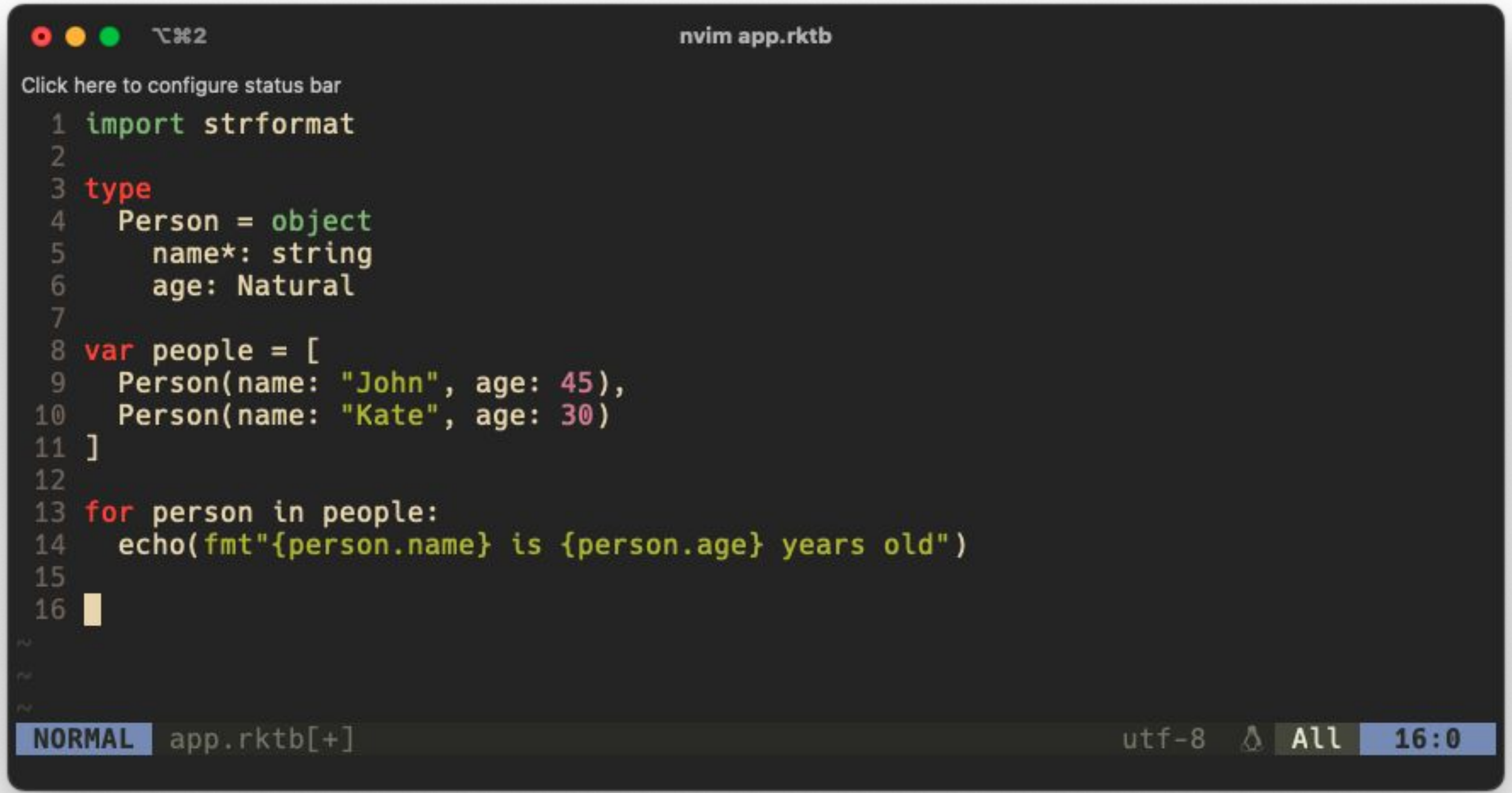
Click here to configure status bar

```
import strformat

type
  Person = object
    name*: string
    age: Natural

var people = [
  Person(name: "John", age: 45),
  Person(name: "Kate", age: 30)
]

for person in people:
  echo(fmt"{person.name} is {person.age} years old")

```

```
NORMAL  app.rktb[+]                                   utf-8  All  16:0
```

CSSSR

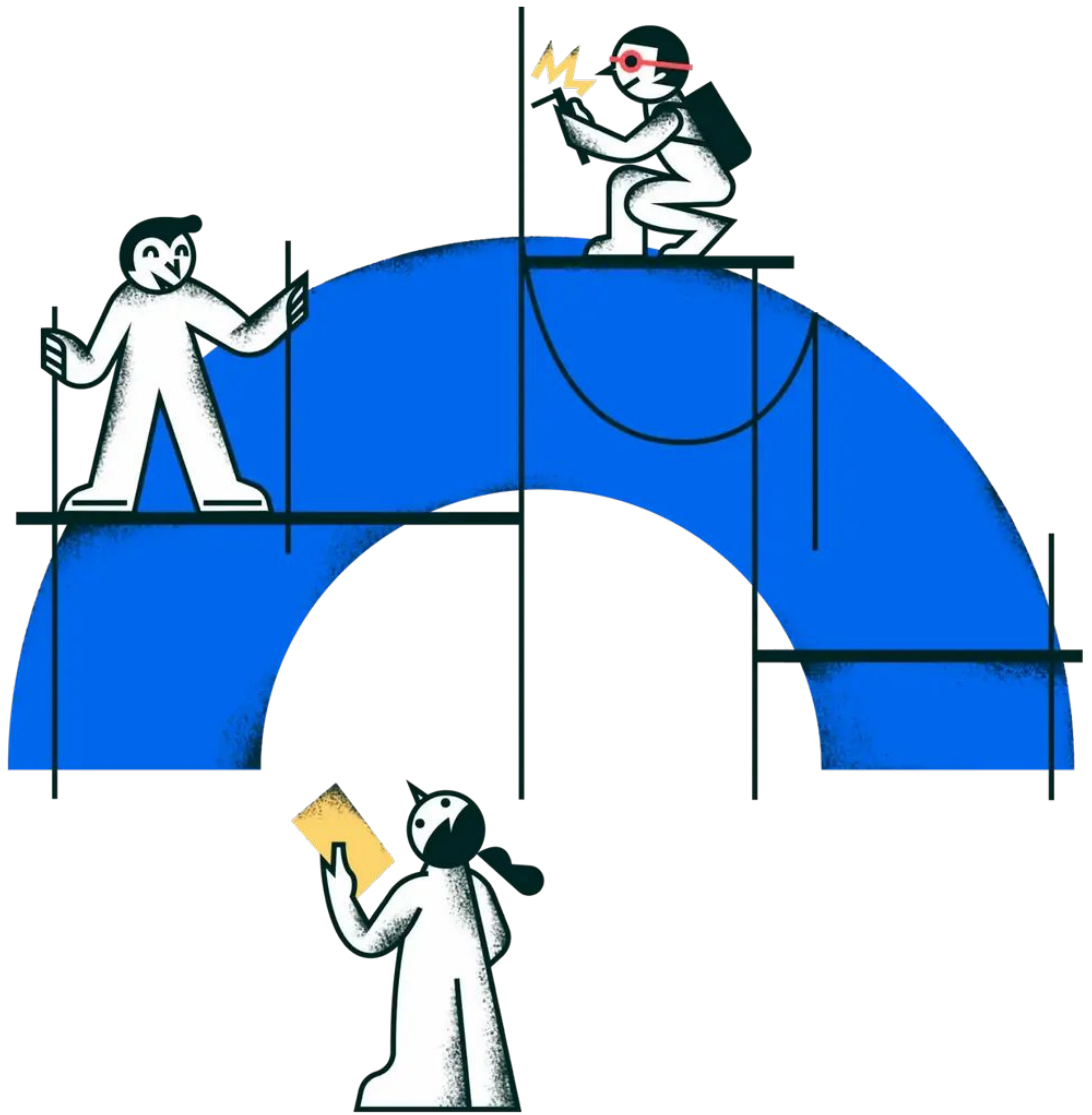
Nim



```nim
import strformat

type
  Person = object
    name*: string
    age: Natural

var people = [
  Person(name: "John", age: 45),
  Person(name: "Kate", age: 30)
]

for person in people:
  echo(fmt"{person.name} is {person.age} years old")

```

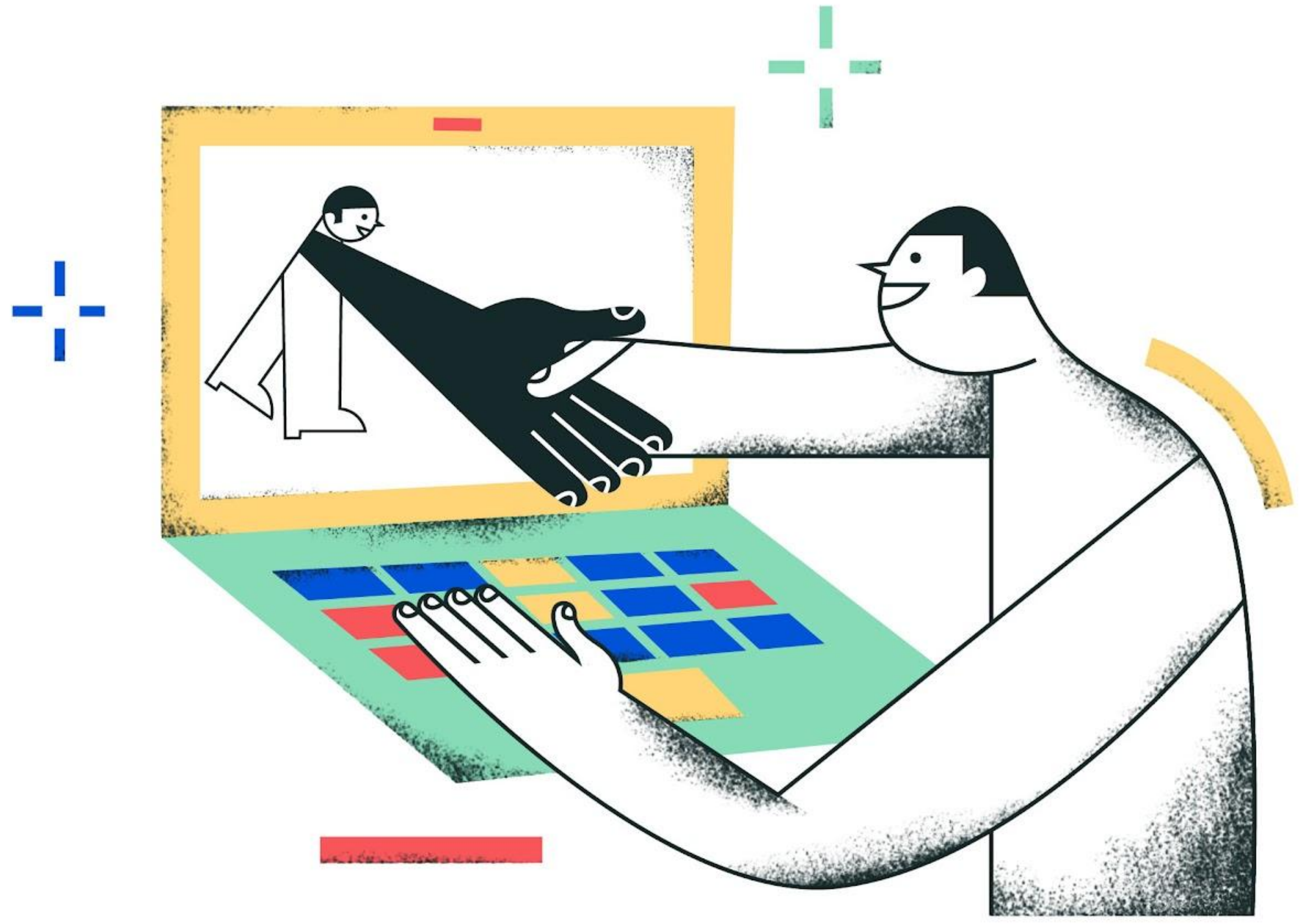
CSSSR

Я просто пишу фронтенд, зачем оно мне нужно?



CSSSR

Спасибо за внимание!



CSSSR

My contacts:

daynin

@_sgolovin

sgolovin

CSSSR contacts:

csssr.com

launch@csssr.com

CSSSR

Полезные ссылки

CSSSR