RubyX

- Compile ruby to binary
- in 100% Ruby
- No external dependencies
- Fast (X times)
- Easy to understand
- Easy to modify own tool

Torsten

github.com/rubydesign

30+ years coding

Now coding as hobby

from Finland

Run a b&b

www.villataika.fi

Raisa

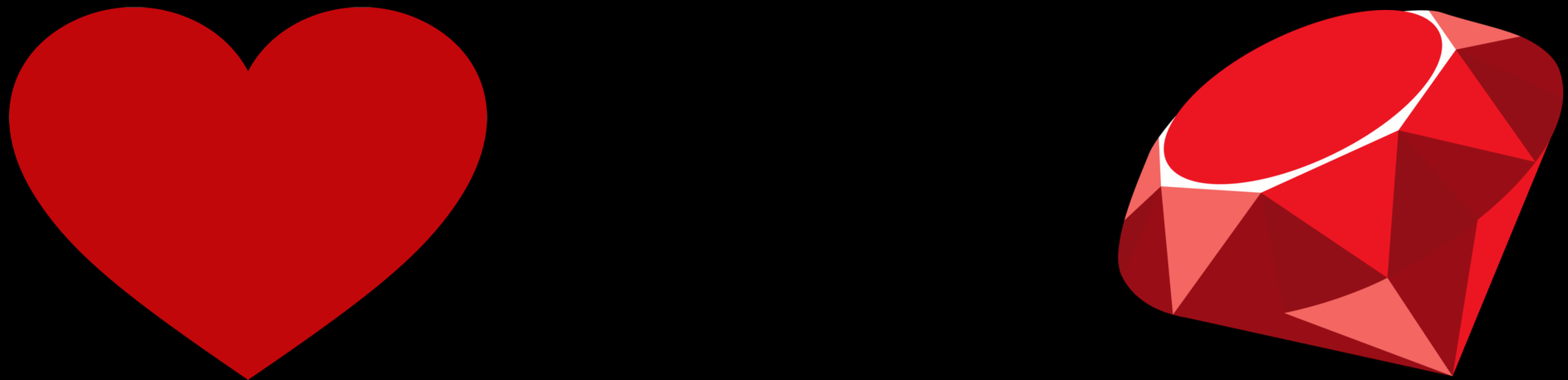
in south Goa 2017

RubyX

- Compile ruby to binary

- in 100% Ruby

- 30min: overview only

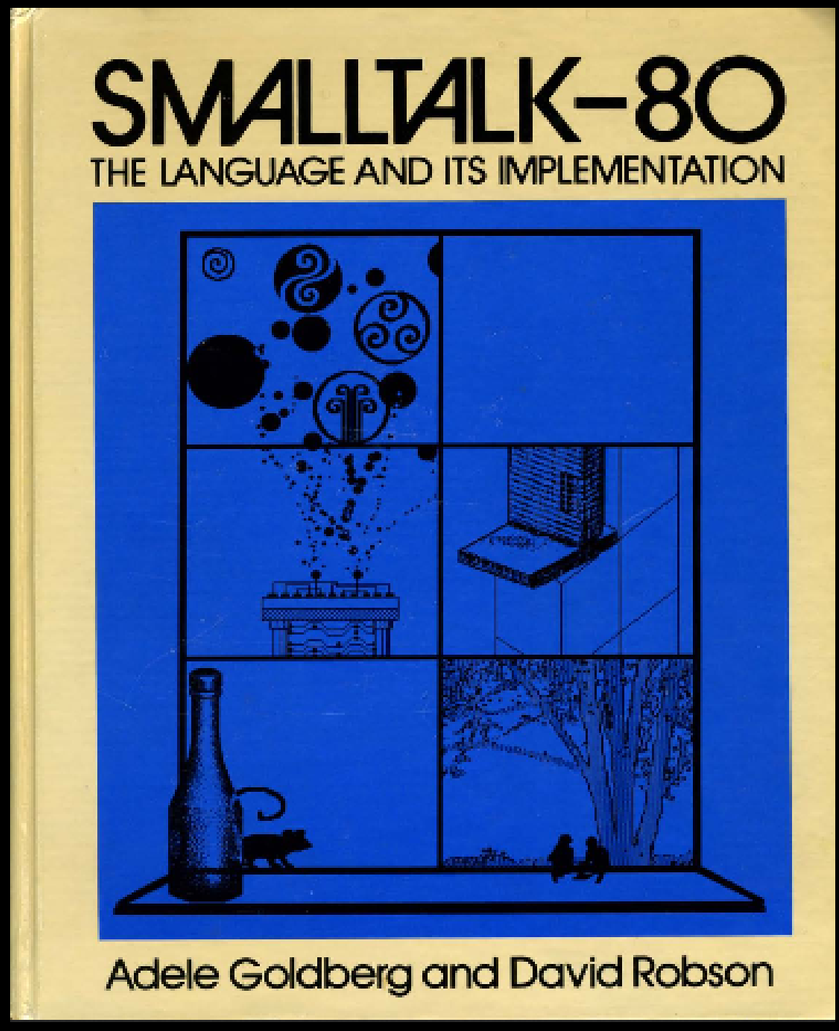


since 2001

mother Smalltalk

weird syntax

quite like ruby



Implementation Problems

- speed (laptop → pi , >10x)

- 2 language problem (nice OR fast)

- inaccessibility (written in C)

Boldy go where ...

compiling to binary

fix all 3 problems

(faster , one tool , in ruby)

Any Program

Input **Program** **Output**

files / stdin → **program** → **files / i**

Mri – runtime only

(none) → **ruby** → **“hello”**

↑

Source as Second Input

hello.rb

Compiling – compile and runtim

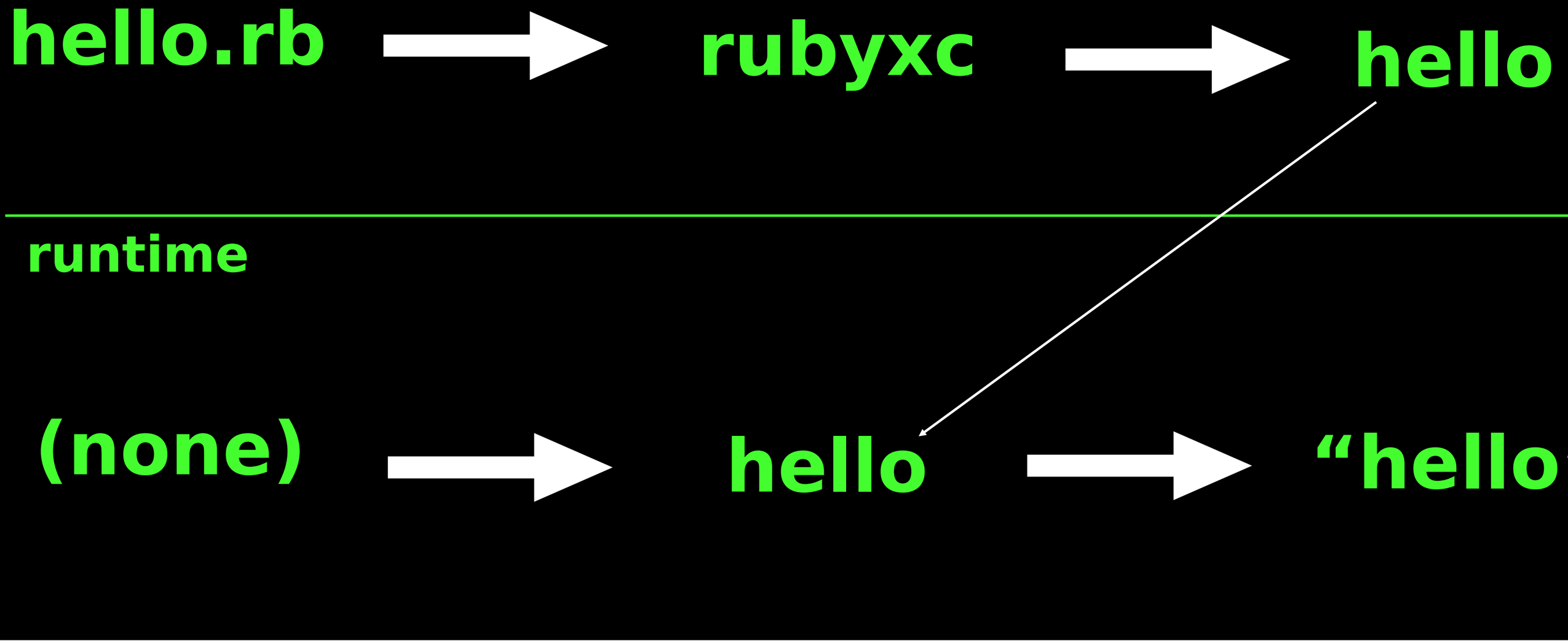
hello.c → **cc** → **hello**

---

later

(none) → **hello** → **“hello”**

RubyX compiling ruby



Mri – runtime only

(none) → **ruby** → **“hello”**

↑

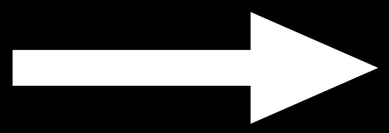
Source as Second Input

hello.rb

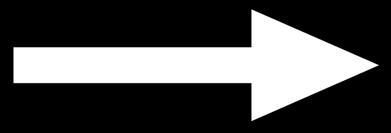
Boostrapping RubyX

ruby-x

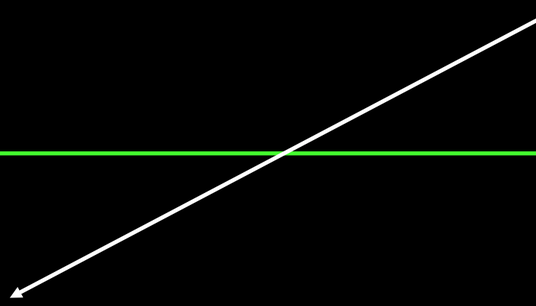
stdlib



rubyxc



rubyx



runtime (same as mri)

input

ruby

rubyx

Output

Currently working

- basic oo (classes/ objects)
- calling convention
- memory layout
- binary creation

Currently working

control structures

- if / else
- while (no break/continue)

assignment

- local
- ivar

Sending

static calling

argument passing

- local / ivar / expression

dynamic sending

- method resolution / caching

Blocks

- implicit block capture
- block passing as argument
- yield with arguments
- return (lambda style)

Lots not working

- eval
- procs / binding
- exceptions
- class methods / variables
- multi - assignment
- stdlib . . . . . .

Object Model

- Everything is Object
  - opaque data, ruby has no access
  - builtin functions to process data

Object has Type

- Types are immutable

- Type defines memory layout

- type reference may change

Many Types implement a Class

- Class has instance Type

- instance type may change

- instance type for new object

Compiler layers

- ruby source (parser gem)
- virtual oo language (vool)
  - Minimal oo Machine (mom)
  - risc abstraction
  - Arm / Elf / binary

Vool, virtual oo language

- virtual == no syntax
- oo == object model like ruby
- ruby without the fluff
- no splats
- no unless

Language vs Machine

- tree vs list (linear)
- abstract vs concrete
- control structures vs jumps
- variables vs memory
- next level is (abtract) machine

Mom, minimal obj. machine

- machine has instructions (list)
- 16 instructions
- higher level
- ease transition to next level
- 1 vool → 2-8 mom

MOM instructions

- ruby truth check
- identity check
- dynamic + simple call
- resolve method
- return sequence . . .

Risc abstraction

- arm without the fluff
- arm like registers
- 20 instructions (compare llvm)
- extensible (class hierarchy)
- last virtual layer (interpreter)
- visual debugger

Risc instructions

- mem/reg + reg/reg transfer
- reg load
- arithmetic operators
- tests on operation result
- jump , call , return
- syscall

calling convention

- linked list based (not stack)
- object oriented
- easy to understand
- exceptions easy
- binding easy

Message Object

- next / caller message
- frame (locals) + args
- return address + value
- method
- receiver (self)

Parfait

- minimal oo runtime (stdlib)
- about 20 classes
- with methods
- + additional builtin methods

Project

- 5 years
- 3k commits , 1300 tests
  - multi arch ready, arm working
  - basic executables (mini rt)
  - stable architecture

ruby-x.org

github.com/ruby-x

Demo time

http://ruby-x.org/

→ Architecture

→ Debugger