

What's Next for Teal

The typed dialect of Lua

Hisham Muhammad

@hisham_hm

{@twitter.com, @mastodon.social}

<https://hisham.hm>

FOSDEM 2021

About me

Hisham Muhammad

htop
GoboLinux
LuaRocks
Teal

@hisham_hm
{@twitter.com, @mastodon.social}
<https://hisham.hm>

About this talk

Part III of a **trilogy!**

Part I - Past

“Minimalism Versus Types”
FOSDEM 2019

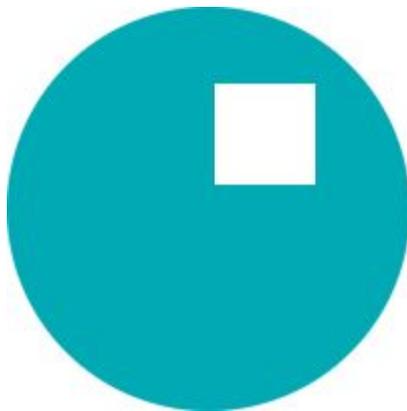
Part II - Present

“Minimalistic Typed Lua is Here”
FOSDEM 2020

Part III - Future

“What’s Next for Teal”
FOSDEM 2021

We have a name!



<https://teal-language.org>

Let's step back a bit

Lua

Let's step back a bit

Lua

a widely used niche language

Lua is everywhere

Game engines

big and small (too many to list)

IoT / Embedded systems

Networking software

Scripting for Applications
(often a good chunk of
entire applications)

LuaTeX

Wikipedia!

Why do people use Lua?

Why do people use Lua?

Embeddable (+small and fast)

Why do people use Lua?

Embeddable (+small and fast)

Extensible

Why do people use Lua?

Embeddable (+small and fast)

Extensible

Something they use
is using Lua

Lua is **minimalistic**
but not minimal

There's a pragmatism to it

The language grows, slowly

Features are added
often in a logical pattern

example: bitwise operators added
once integers were added

“Mechanisms, not policies”

pros:

fits well for an
embedded language
in an application

cons:

not ideal for
pure-Lua application programming
and ecosystem building

Several ecosystems

OpenResty

Love2D

Roblox

etc.

Teal

a new programming language:
a statically typed dialect of Lua

Why Teal?

Static types are good

I like Lua and I like compilers

people (myself included)
do write (large) applications
in Lua

minimalistic

a **single Lua module** (tl.lua)
implements a Teal-to-Lua compiler

can be loaded into any Lua 5.1-5.4
project
without any dependencies

pragmatic

not aiming for soundness:

it doesn't need to be perfect,

it needs to be

better than not having types at all

not afraid to add features

...when they follow a logical pattern:

Static types led to explicit globals

Union types led to the “is” operator

Where Teal is now

a small but growing **community!**

editor support (vim, vscode, etc.)

teal-**playground**

teal-types: repo with type definitions of
Lua libraries

language **improvements**

Language improvements

Union types

Tuple types

Metamethods

Integer division and bitwise operators
with metamethods
(working for all versions of Lua!)

Improved flow-based inference

Syntax tweaks and cleanups

What's Next for Teal

Time to **start using it!**

Use the experience to **mature** the language

Identify what is **missing**

Strike the balance between **minimalism and pragmatism**

Some early feedback

Integer type

Abstract interfaces for records

Optional/required function arity

Strict nil-safety?

Taking a page from Lua

Teal is embeddable

Taking a page from Lua

Teal is embeddable

It needs to be extensible

Extensibility

Lua extensibility:

metatables/methods at runtime

Lua/C API

Teal extensibility:

metaprogramming at compile time

Compiler API

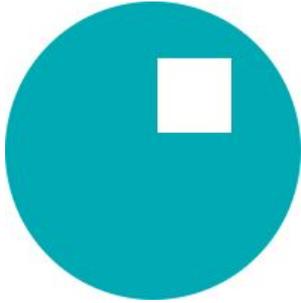
In short

Teal is usable today!

Embeddable static typing support
for Lua-based environments,
for programming in the large

A new language, evolving
with a welcoming community

Thank you!



Join the community via
Gitter or Matrix

Links at

<https://github.com/teal-language/tl>