If we are going to transpile JavaScript, why not use ClojureScript?

medium.com/@shivekkhurana
github.com/shivekkhurana
twitter.com/shivek_khurana

Built an LMS from 2012 to 2016

Consisted of an API, Admin, Web and Native Mobile Interfaces

Long projects force you to eat your own dogfood

Followed all best practices

(linting, flux, folder structures, build workflows)

Ended up changing the build system thrice

You can argue that JS ecosystem was maturing

Either it’s still not mature or JS people like to reinvent the wheel

Even the syntax is still evolving

rollup.js

The JS community is blessed, with some amazing work like React, Lodash and date-fns

But there is a problem with the underlying language

That’s why nearly all prod apps are ES6, TypeScript or some other superset of JS

I wanted a language with:

Stable syntax and language guarantees

Rich standard library

First class functional transformations (map/reduce/filter)

Ability to easily understand, fork and update third party packages

A community with aversion to shiny things

Type Safety

Changing stacks is a high risk event

Need to keep prod systems functional

Train/ Hire Talent

Experiment, make mistakes, grow

And at the end, you might end up disliking it

Why Clojure/Script sucks

and 7 reasons why you should not try it, ever

Clojure is a hosted language

You need a JVM, Browser, V8, CLR or something else to run it

You also need to know the underlying language in case Clojure misbehaves (rare)

Clojure is a hosted language for good

Being hosted helps you tap into the existing ecosystem. Clojure gives you the power of and Java, JavaScript, .Net (C, Python, Erlang)

(It’s flooded (with (parenthesis)))

Matching parenthesis would be painful.
Resolving a merge conflict on a large feature would take forever.

(It’s flooded (with (parenthesis)))
for good

Parenthesis are added automatically using tools like paredit and parinfer

Tree based editing leads to terse code and allow compiler level optimizations

It’s hard to find or train engineers

Clojure community is small.

Even more so in India.

It’s hard to find or train engineers **not** for good

It is hard to find talent, that’s why I’m here.
My job is to convert and train talent.

The small community is close knit and attracts the best kind of thinkers.

It has a steep learning curve

Everything is immutable and functional. It’s not how humans think.

Humans think imperativelty.

It has a steep learning curve

Most of your tooling and thought process will need tweaking.

But the new can’t be born, until the old dies.

It’s hard to get started with

The community is already small.

Combine that with scarce documentation and little literature for new initiates.

It’s hard to get started with, but things are changing

People answer Clojure questions on StackOverflow. Docs are getting better.

And it’s really easy to read and interpret source code.

It’s Java at the end of the day

And Java has slow startup time, Null point bombs and a ton of bloat that I don’t need.

It’s Java at the end of the day

True, it suffers from slow startup time, but it covers for that once it gets started.

You rarely write Java. And the REPL is the answer to slow startup times.

The benefits outweigh the drawbacks

Everything is just data

TODO- Example between how OOP try to convert everything into DSL/Objects and end up losing power and increase complexity

Specs not types

- Static typing everywhere is an overkill
- Specs are optional defs of the shapes of expected data
- You can spec the language, not just the macros (example of server spec that finds dynamic issues)
- Generate stub data that can be used to

REPL

- REPL is a development time RUNTIME that lets you write and evalate code
- This leads to faster cycles
- Example of Syntax, followed by SPEC, followed by REPL

Conclusion