Elliott Cable (prefer HTML? http://ell.io/tt/resume)
=============

“Move slowly and maintain things.”

Cell:	+1 919 636.4204	(SMS preferred; calls, FaceTime w/ prior notice)
E-mail:	hire@from.ec	(NOT CHECKED OFTEN; text or DM me first!)
Twitter:	twitter.com/ELLIOTTCABLE	(DMs welcome)
GitHub:	github.com/ELLIOTTCABLE	
Addt'l:	elliottcable.name	

I am a testing-, operations-, and documentation-obsessed software engineer. I am most motivated by projects involving:

- programming-language design and implementation,
- other software-tooling development,
- technical-debt paydown and maintainability refactors, or
- accessibility and diversity/inclusion.

My preferred process involves careful software development using powerful correctness tools like OCaml and ReasonML; or PLT and JIT/compiler design. I'm most experienced in: interpreter implementation; open-source library maintenance; modern JavaScript tooling, versioning, and deployment; JS library development; server-side development in Node.js or Ruby; POSIX shell and ISO C.

PROJECT EXPERIENCE

Here's a sampling of my work, selected primarily for breadth. Where legally possible, I've included links to source-code that I find to be particularly exemplary.

### 2018-2019: KidScript, an educational programming-language for kids
(as Senior Software Engineer at Codeverse)

Codeverse brought me onboard to draw on my open-source experience, improving on the stability of (and establishing engineering process for) their KidScript programming-language, as well as an iOS IDE. Responsibilities included:

- Establish strict Semantic Versioning procedure for the releases of, and dependencies between, internal packages and external products
- Rewrite organization's internal tooling for JavaScript and iOS compilation, deployment, etc. effectively from scratch, to enforce good practices
- Contribute features and handle fixes to the parsing, error-handling, and interpreter components of KidScript
- Manage contributions to the language's standard-library of components and objects
- Review other engineers' contributions to all of the above; providing guidance on maintainability and technical debt
- Rewrite critical portions of the IDE's game/graphics-engine in TypeScript and ML to improve stability

### 2010-2018: Pratchett, a programming language
My overarching work of the past ten years, Pratchett (née Paws) is the project that's driven much of my personal development as a programmer. From surveying prospective users' needs, staying on top of similar language work and PLT research, to organizing a (brief) community of interest and contributors and triaging design goals to try and keep a sense of constant progress, this has been my largest ongoing software project.

- ell.io/tt$Paws.js/tree/queueless+
- ell.io/tt$Paws.js/blob/d1a1f2/Source/datagraph.coffee#L27-L135
- ell.io/tt$Paws.js/blob/d1a1f2/Test/datagraph.tests.coffee#L246
- ell.io/tt$Paws.js/blob/546795/Source/reactor.coffee#L114-L196
- ell.io/tt$Paws.js/blob/d1a1f2/Scripts/test.sh#L3-L35

### 2018-now: Excmd.js, a performant and strict vi-mode parser for Tridactyl
I'm a huge fan of Firefox and Vim; and Tridactyl, a vi-mode for daily web-browsing. I offered to take over the rewrite of their Exmode (command-line) parser.

- ell.io/tt$excmd.js

Beyond just improving the parser, my *primary* goal here was to apply traditional OCaml parsing tooling to a front-end project. I have a lot of desire to 'shave the rough edges' off the user-experience of OCaml for use in the JavaScript community, and this project gave me a lot of opportunity to contribute back to the BuckleScript ecosystem and improve the overall experience of writing ML-for-the-web.

 - ell.io/tt$bs-sedlex
 - github.com/c-cube/gen/pull/17
 - github.com/ocaml/merlin/pull/771
 - github.com/ocaml-ppx/ppx_deriving/pull/204
 - github.com/ocamllabs/higher/pull/12

2017: Giraphe, a configurable JavaScript graph-walking library
Extracted from my work on Paws.js by necessity, this is an API designed to generate individual, optimized graph-walking iterators for various Pratchett-related tasks. (It currently only *implements* a generic, unoptimized walker.)

 - ell.io/tt$giraphe
 - ell.io/tt$giraphe/blob/fe85e6/giraphe.es6.js#L122-L257

As of December 2021, I'm busy re-writing this in TypeScript (sadly, not OCaml) and optimizing for performance.

 - ell.io/tt$giraphe/blob/33a5259f/src/edgeless-walker.ts

Various patches and contributions to others' open-source projects
I'm a repeat open-source developer and patcher. Although so many of my changes stay local and never see the light of day (a bad habit!), occasionally, I clean up and complete my explorations, and remember to submit a pull-request.

 - 2020, OCaml/JS: BuckleScript, Bloomberg's OCaml-to-JavaScript compiler

   My most recent sizable contribution, this patch was an on-and-off project throughout 2020. I would like to call attention to my documentation, clear explanatory git-log, and incremental changes, moreso than the actual code-changes I contributed.

   github.com/rescript-lang/rescript-compiler/pull/4116

 - 2018, Python/OCaml: Merlin, OCaml's analysis engine.
   ell.io/tt$merlin/blob/9b80dd/vim/merlin/autoload/merlin.py#L677-L734

 - 2017, ISO C: the XV6 operating-system.
   ell.io/tt$xv6/blob/efd94e/README#L10-L57,
   ell.io/tt$xv6/compare/08429c...379fc6

 - 2013, ISO C: Git. adding a history-reorganizing feature.
   ell.io/tt$git/compare/b5c267...author-order+

2016: `pin-cushion`, an API-client for Pinboard.in
A small, one-off, but complete and modern, command-line client for my bookmarking service of choice. Included simply because it's some of the only recent and pure JavaScript I've written, with no legacy cruft.

 - ell.io/tt$pin-cushion
 - ell.io/tt$pin-cushion/blob/36be70/pin-cushion#L78-L121

2008: ArchLinux image-builder for Amazon EC2
Arch, my preferred Linux distro, was unavailable on Amazon EC2 for years. I built a unnecessarily-complex, modular shell-script architecture for bundling custom builds of Arch and publishing them to your EC2 hosts. This involved poring through both Amazon's AWS documentation, as well as that of Arch's own build-system and packaging. This is the largest pure-POSIX-shell program I've written.

 - ell.io/tt$ArchLinux-AMIs/blob/a34646/bundle.sh

2006-2009: Assorted Ruby tools
I was heavily into metaprogramming Ruby for many years — these are some of the libraries and tools I was most proud of.

 - ell.io/tt$it/blob/b52441/lib/it/environmented_proc.rb#L63-L155
 - ell.io/tt$lobby/blob/8dc3f6/lib/lobby.rb
 - ell.io/tt$stringray/blob/117038/lib/stringray.rb#L3-L162
 - ell.io/tt$nfoiled/blob/d8e593/lib/nfoiled/window.rb#L3-L217

ADDITIONAL INFORMATION

I'm an avid maker/hacker, I hold a ‘General’-class gov't license for radio operation (KL4JC, monitoring!), and I'm an activist for the safety and inclusion of Black, trans, and other underrepresented folks in tech. I'm also a lifelong Eagle Scout.

Finally, a keyword-oriented list of additional related skills omitted from the above (at least, those with which I have at least **some** familiarity), follows:

    Practices: BDD & TDD, Agile; granular Git or git-flow; Agile, git-flow;
       Semantic Versioning
    Languages & platforms: CoffeeScript, TypeScript, BuckleScript, Flow; React,
       Rails, Python; Objective-C, Cocoa; Io, Lua, Potion; Racket, Guile, other
       R5RS; Tulip, Eff, MetaOCaml; VimScript; C++, Java
    Other: macOS, BSD, Linux, and the POSIX/UNIX APIs; Redis & MongoDB;
       firmware / RTOS development; Docker, Vagrant; PostgreSQL