State of the OCaml Platform

David Allsopp, Stephen Dolan, Louis Gesbert, Gemma Gordon, Anil Madhavapeddy (speaker), KC Sivaramakrishnan

September 2017

With contributions from Nicolas Assouad, Frédéric Bour, Daniel Buenzli, Romain Calascibetta, Amir Chaudhry, Enguerrand Decorne, Jeremie Dimino, Thomas Gazagnaire, Louis Gesbert, Armaël Guéneau, Daniel Hillerström, David Kaloper-Meršinjak, Théo Laurent, Qi Li, Jon Ludlam, Thomas Leonard, Maxime Lesourd, Hannes Mehnert, Richard Mortier, Olivier Nicole, Thomas Refis, David Scott, David Sheets, Mark Shinwell, Magnus Skjegstad, Liang Wang, Leo White, Jeremy Yallop, and members of the INRIA Galium team, OCamlPro and Jane Street.
tl;dr

• OCaml Labs is 5 years old today!

• This work is based on a huge collaborative effort

• First some **perspective on the past five years**

• Then a **status update on 2017's developments**

• And finally a **sneak preview of 2018 work**
What is the Platform?

The OCaml Platform combines the OCaml compiler toolchain with a coherent set of tools for build, documentation, testing and IDE integration.

The project is a collaborative effort across the OCaml community, tied together by the OCaml Labs group in Cambridge, and OCamlPro in Paris.

The requirements of the Platform are guided by large industrial users such as Jane Street, Citrix, Docker, Facebook, Microsoft and LexiFi, as well as accrued feedback from the opam project.
## Code Management Tools

<table>
<thead>
<tr>
<th>Year</th>
<th>Code Management Tools</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;2008</td>
<td>ocamlfind, omake</td>
</tr>
<tr>
<td>2009</td>
<td>ounit, odb</td>
</tr>
<tr>
<td>2010</td>
<td>odb</td>
</tr>
<tr>
<td>2011</td>
<td>mirage</td>
</tr>
<tr>
<td>2012</td>
<td>oasis</td>
</tr>
</tbody>
</table>

### Related Projects
- **The wilderness years**
- **Every project an island**
- **3 hour CUFP "tutorial"**
<table>
<thead>
<tr>
<th>Year</th>
<th>Code</th>
<th>Build</th>
<th>Test</th>
<th>Explain</th>
<th>Package</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;2008</td>
<td>ocamlfind</td>
<td>omake</td>
<td></td>
<td>ocamldoc</td>
<td>godi</td>
</tr>
<tr>
<td>2009</td>
<td></td>
<td></td>
<td></td>
<td>ounit</td>
<td></td>
</tr>
<tr>
<td>2010</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2011</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2012</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2013</td>
<td>merlin</td>
<td>irmin</td>
<td>assemblage</td>
<td>ocamlot</td>
<td>RWO codoc opam 1.0</td>
</tr>
<tr>
<td>2014</td>
<td>ctypes</td>
<td></td>
<td></td>
<td></td>
<td>ocaml.org</td>
</tr>
<tr>
<td>2015</td>
<td></td>
<td>docker</td>
<td>datakit-ci</td>
<td></td>
<td>AFP course</td>
</tr>
<tr>
<td>2016</td>
<td>git</td>
<td></td>
<td></td>
<td>ocamllabs.io</td>
<td>topkg</td>
</tr>
<tr>
<td>2017</td>
<td></td>
<td>jbuilder</td>
<td></td>
<td>odoc odig</td>
<td>opam 2.0</td>
</tr>
</tbody>
</table>

**OCaml Labs founded**

**A surge of development**

**Shift to smaller libs**
One in one out

Sustainable ecosystem

Consistent interfaces
camlp4 -> PPX

- PPX are extension points and annotations for OCaml.
- **Key PPX libraries now hosted in one place:**
  - [github.com/ocaml-ppx](https://github.com/ocaml-ppx)
  - more robust to OCaml compiler versions (ocaml-migrate-parsetree)
- **Getting rid of camlp4 is taking years.**
  - Steady education process required for community.
  - Migration guides on [discuss.ocaml.org](https://discuss.ocaml.org) would accelerate process.
    - e.g. pa_ulex -> sedlex
Merlin 3.0

• Major Merlin 3.0 developer tool release
  • Scalable protocol to communicate with IDEs
  • Robust Windows support.
  • Now promoted to https://github.com/ocaml/merlin

• Community now using it as a standard for IDEs
  • Visual Studio Code, Atom, Sublime Text
  • Facebook Reason syntax support also.
  • More sophisticated short paths algorithm than upstream.
**Documentation**

**odoc**: generate HTML for a group of libraries, with cross referencing.

![By name](image)

http://docs.mirage.io

```bash
$ opam install odoc odig
$ opam install mirage # and anything
$ odig odoc
```
**odoc**: generate HTML for a group of libraries, with cross referencing.

By name

A B C D E F G H I J L M N O P R S T U V W X Y Z

*alcoltest* 0.8.0  
*angstrom* 0.8.0  
*anycache* 0.8.0  
*arp* 0.2.0  
*aetmap* 0.8.1  
*asl* 0.11  
*asn1-combinators* 0.1.3  
*astring* 0.8.3  
*async* v0.9.0  
*async_extra* v0.9.0  
*async_find* v0.9.0

[http://docs.mirage.io](http://docs.mirage.io)

Now working on [docs.ocaml.org](http://docs.ocaml.org) - can we build this at a large scale for every package?
Documentation

Lwt API Reference

module Lwt

Asynchronous programming with promises.

Promises are placeholders for values that take a long time to compute. Promises are similar to refs – they can store one value. Here is how they differ from refs:

- A promise might not have a value yet. This is called a pending promise.
- Writing a value into a promise is called resolving it.
- It’s possible to attach callbacks to a pending promise. They will be run when the promise gets resolved.
- A promise can be resolved only once. Once a promise has a value, it’s immutable.

https://ocsigen.github.io/lwt/manual-draft/Lwt.html

A few remaining issues to fully deprecate ocamldoc: porting complex sites like Lwt
Documentation

Real World OCaml
FUNCTIONAL PROGRAMMING FOR THE MASSES
2nd Edition (in progress)

Homepage: dev.realworldocaml.org
OPAM 2.0beta

- OPAM is the source-based package manager for publishing OCaml code
- Focus this year has been on stabilising the upcoming 2.0
- Crossed PR #10000 in opam-repository and PR #3000 in opam!
OPAM 2.0beta

- OPAM is the source-based package manager for publishing OCaml code
- Focus this year has been on stabilising the upcoming 2.0
- Over 7000 packages now managed
OPAM 2.0beta

- OPAM is the source-based package manager for publishing OCaml code
- Focus this year has been on stabilising the upcoming 2.0
- Over 600 individual contributors to the repository
OPAM 2.0beta

• New features in recent betas:
  • more expressive package dependencies

```plaintext
depends: [ "foo" {>= "3.0" & < "4.0~" & os = "linux"} ]
```

```plaintext
depends: [ "ocaml" >= "4.05.0" ]
```
New features in recent betas:

- more expressive package dependencies

```opam
depends: [ "foo" {>= "3.0" & < "4.0~" & os = "linux"} ]
```

```opam
depends: [ "ocaml" >= "4.05.0" ]
```

```opam
depends: [
  ...
  "datakit-server" {>= "0.9.0"}
  "datakit-client" {with-test & >= "0.9.0"}
  "datakit-github" {with-test & >= "0.9.0"}
  "alcotest" {with-test & >= "0.7.0"}
]
```
OPAM 2.0beta

- New features in recent betas:
  - more expressive package dependencies
  - computed versions to make multiple packages easier

Simplifies the package repository by making it easier to have many small libraries
OPAM 2.0beta

- New features in recent betas:
  - more expressive package dependencies
  - computed versions to make multiple packages easier
  - local switches for use per-project

```
cd ~/src/project
opam switch create ./
...

opam list
```
OPAM 2.0beta

• New features in recent betas:
  • more expressive package dependencies
  • computed versions to make multiple packages easier
  • local switches for use per-project

• Solver integrated as a library, no external solver needed!
OPAM 2.0beta

- New features in recent betas:
  - more expressive package dependencies
  - computed versions to make multiple packages easier
  - local switches for use per-project
- Solver integrated as a library, no external solver needed!
- Windows support is being upstreamed!
OPAM 2.0beta

• New features in recent betas:
  • more expressive package dependencies
  • computed versions to make multiple packages easier
  • local switches for use per-project

• Solver integrated as a library, no external solver needed!

• Windows support is being upstreamed!

But how do we get a usable package repository?
Need a simpler build system

Luckily 2017 has been kind to build...
Jbuilder

A build system specialised to real world OCaml code. Provide a description of your project, and it will be built!
Jbuilder

A build system specialised to real world OCaml code. Provide a description of your project, and it will be built!

- **Compose multiple checkouts** in subdirs and it can be built in one pass

- **Integrates with opam** to locate external packages

- **Multiple workspaces** to support different OCaml versions or build options (e.g. afl or flambda)

- **Declarative model** encourages portable build rules, so it often "just works" on Windows.

- **Fast. Really fast.**
OPAM 2.0 release

Why isn't it released yet?

I want these cool features!
Continuous Integration

- 2017 has been a good year: big releases to opam, significantly improved CI to submitted packages

- Automated infrastructure is very efficient vs engineering time, so scripting everything we can!

- Currently expanding to meet new needs: **benchmarking**, **fuzzing**, and **portability**

- Also need to automate the opam1 -> opam2 transition

Please join the opam maintenance team!
<table>
<thead>
<tr>
<th>Year</th>
<th>CODE</th>
<th>BUILD</th>
<th>TEST</th>
<th>EXPLAIN</th>
<th>PACKAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;2008</td>
<td>ocamlfind</td>
<td>omake</td>
<td></td>
<td>ocamldoc</td>
<td>godi</td>
</tr>
<tr>
<td>2009</td>
<td></td>
<td></td>
<td>ounit</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2010</td>
<td></td>
<td></td>
<td>oasis</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2011</td>
<td>mirage</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2012</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2013</td>
<td>merlin</td>
<td></td>
<td>ocamlot</td>
<td>RWO</td>
<td>opam 1.0</td>
</tr>
<tr>
<td>2014</td>
<td></td>
<td></td>
<td></td>
<td>codoc</td>
<td></td>
</tr>
<tr>
<td>2015</td>
<td></td>
<td></td>
<td></td>
<td>topkg</td>
<td></td>
</tr>
<tr>
<td>2016</td>
<td></td>
<td></td>
<td></td>
<td>odoc</td>
<td></td>
</tr>
<tr>
<td>2017</td>
<td>&quot;One in one out&quot;</td>
<td>Sustainable ecosystem</td>
<td>Consistent interfaces</td>
<td>odoc</td>
<td>opam 2.0</td>
</tr>
<tr>
<td>Year</td>
<td>Code</td>
<td>Build</td>
<td>Test</td>
<td>Explain</td>
<td>Package</td>
</tr>
<tr>
<td>------</td>
<td>------</td>
<td>-------</td>
<td>------</td>
<td>---------</td>
<td>---------</td>
</tr>
<tr>
<td>2013</td>
<td>merlin</td>
<td>assemblage</td>
<td>ocamlot</td>
<td>RWO</td>
<td>codoc</td>
</tr>
<tr>
<td>2014</td>
<td>irmin</td>
<td>ctypes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2015</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2016</td>
<td></td>
<td>jbuilder</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2017</td>
<td>git</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2018</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2019</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2020</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2021</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2022</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Some common cargo commands are (see all commands with --list):

- **build**: Compile the current project
- **check**: Analyze the current project and report errors
- **clean**: Remove the target directory
- **doc**: Build this project's and its dependencies' docs
- **new**: Create a new cargo project
- **init**: Create a new cargo project in an existing directory
- **run**: Build and execute src/main.rs
- **test**: Run the tests
- **bench**: Run the benchmarks
- **update**: Update dependencies listed in Cargo.lock
- **search**: Search registry for crates
- **publish**: Package and upload this project to the registry
- **install**: Install a Rust binary

an aside: cargo/rust

Single binary, gives access to full Rust ecosystem
Can we adopt same approach for Platform?
Some common bob commands are (see all commands with --list):

- **build** Compile the current project
- **check** Analyze the current project and report errors
- **clean** Remove the target directory
- **doc** Build this project's and its dependencies' docs
- **new** Create a new OCaml project
- **init** Create a new OCaml project in an existing directory
- **run** Build and execute src/main.ml
- **test** Run the tests
- **fuzz** Run the fuzzer
- **compat** Check compatibility across different operating systems
- **bench** Run the benchmarks
- **update** Update dependencies listed in opam.lock
- **search** Search registry for opam packages
- **publish** Package and upload this project to the opam registry
- **install** Install an OCaml binary

**codename:** bob
• Single CLI tool for all OCaml platform activities.
  • Includes jbuilder, odoc, merlin, opam, alcotest, crowbar, topkg with a single CLI
  • Sensible defaults, with escape hatches for advanced users

• Access to offline/online infrastructure:
  • dedicated **continuous integration** access
  • **consistent OS support** for Linux, Windows, macOS
  • **search online** community (odoc, opam, github)

**Will be released in 2018 as the first OCaml Platform release, with associated tools bundled**
Bob is **not yet complete** or even fully designed.

We are starting an **open design process** this year, as we release opam 2.0.

Opam maintenance window is around ~3-5 years, so want to get this revision right.
Online Community

Unify the design of these sites, with **ocaml.org** as the main public site

- [ocaml.org](http://ocaml.org)
- [opam.ocaml.org](http://opam.ocaml.org)
- [realworldocaml.org](http://realworldocaml.org)
- [lists.ocaml.org](http://lists.ocaml.org)
- [discuss.ocaml.org](http://discuss.ocaml.org)
- [ci.ocaml.io](http://ci.ocaml.io)
- [docs.ocaml.org](http://docs.ocaml.org)
- [github.com/ocaml](http://github.com/ocaml)

Create searchable API for all online OCaml resources - used by **bob search**
What is the Platform?

The OCaml Platform combines the OCaml compiler toolchain with a coherent set of tools for build, documentation, testing and IDE integration.

The project is a collaborative effort across the OCaml community, tied together by the OCaml Labs group in Cambridge, and OCamlPro in Paris, and the whole OCaml community.

See you on discuss.ocaml.org :)

See you on discuss.ocaml.org :)

See you on discuss.ocaml.org :)