ocamlbuild, a tool for automatic compilation of OCaml projects

Berke Durak    Nicolas Pouillard

Berke.Durak@inria.fr    Nicolas.Pouillard@inria.fr

June 6, 2007
Outline

1 Introduction

2 Regular OCaml projects

3 Dealing with exceptions to standard rules

4 Writing an ocamlbuild plugin

5 General features

6 Conclusion
Why such a tool?

- To make our OCaml life easier
Why such a tool?

- To make our OCaml life easier
- To stop writing poor MakefileS
Why such a tool?

- To make our OCaml life easier
- To stop writing poor MakefileS
- To have a tool that Just works™
What does ocamlbuild handle?

**Regular OCaml projects of arbitrary size**

Trivially handled using the command line options.
What does ocamlbuild handle?

Regular OCaml projects of arbitrary size
Trivially handled using the command line options.

Mostly regular OCaml projects with common exceptions
Requires writing one tag file (_tags) that declares those exceptions.
What does ocamlbuild handle?

Regular OCaml projects of arbitrary size
Trivially handled using the command line options.

Mostly regular OCaml projects with common exceptions
Requires writing one tag file (_tags) that declares those exceptions.

Almost any project
Accomplished by writing an ocamlbuild plugin.
ocamlbuild, a tool for automatic compilation of OCaml projects

Introduction

What does ocamlbuild provide?

- Automated whole-project compilation
- Minimal recompilation
- Lots of useful targets (doc, debugging, profiling...)
- Supports multiple build directories
- Automatic and safe cleaning
- A source directory uncluttered by object files
- A portable tool shipped with OCaml
What does ocamllbuild provide?

- Automated whole-project compilation
- Minimal recompilation
- Lots of useful targets (doc, debugging, profiling...)
- Supports multiple build directories
- Automatic and safe cleaning
- A source directory uncluttered by object files
- A portable tool shipped with OCaml
- Saves time and money!
Outline

1. Introduction
2. Regular OCaml projects
3. Dealing with exceptions to standard rules
4. Writing an ocamlbuild plugin
5. General features
6. Conclusion
What’s a regular OCaml project?

It’s a project that needs no exceptions from the standard rules:

- Has compilation units (ml and mli files)
- May have parsers and lexers (mly and mll files)
- May use packages, libraries and toplevels (ml{pack,lib,top})
- May link with external libraries
- Has one main OCaml unit from which these units are reachable
How difficult is it to build regular projects by hand?

**OCaml** has subtle compilation rules

- Interfaces (.mli) can be absent, yet buildable (.mly)
- Native and bytecode suffixes and settings differ
- Native packages are difficult to do (-for-pack)
- Linkage order must be correctly computed
- Include directories must be ordered
- **ocamldep** gives partial information (too conservative)
How does ocamlbuild manage all that?

It has a lot of hand-crafted Ocaml-specific compilation logic!
How does `ocamlbuild` manage all that?

It has a lot of hand-crafted Ocaml-specific compilation logic!

A dynamic exploration approach

- Start from the given targets
- Attempt to discover dependencies using `ocamldep`
- `ocamldep` cannot always be trusted: backtrack if necessary
- Launch compilations and discover more dependencies
Many projects can be compiled with a single command:

- Menhir: `ocamlbuild -lib unix back.native`
- Hevea: `ocamlbuild latexmain.native`
- Ergo: `ocamlbuild main.native`
- Ocamlgraph: `ocamlbuild -cflags -for-pack,Occamlgraph demo.native`
- ...

To be fair...

Some of these projects require that a `version.ml` or `stdlib.ml` file be generated beforehand.
Outline

1 Introduction

2 Regular OCaml projects

3 Dealing with exceptions to standard rules

4 Writing an ocamlbuild plugin

5 General features

6 Conclusion
What’s an exception?

Files that need specific flags

- Warnings to be enabled or disabled
- Debugging (-g), profiling (-p), type annotation, recursive types, -linkall, -thread, -custom...

- Units that need external C libraries
- Binaries that need external OCaml libraries
- Directories that must be included or excluded
- Dependencies that cannot be discovered
Make and exceptions

- The make tool can’t handle exceptions very well
- Needs exceptions to be encoded as specific rules
- This generally makes rules and exceptions tightly bound by variables
- This creates non-modular makefiles that don’t scale
The tags, our way to specify exceptions

- The `_tags` file is made of lines
- Each line is made of a pattern and a list of signed tags
- A line adds or removes tags from matching files
- Patterns are boolean combinations of shell-like globbing expressions

```
"funny.ml": rectypes
<**/*.ml*>: warn_A, warn_error_A, debug, dtypes
<**/*.cmx>: inline(9)
"foo.ml" or "bar.ml": warn_v, warn_error_v
"vendor.ml": -warn_A, -warn_error_A
<main.{byte,native}>: use_unix
"main.byte": use_dynlink, linkall
"test": not_hygienic
<satsolver.cm[io]>: precious
```
ocamlbuild, a tool for automatic compilation of OCaml projects
Dealing with exceptions to standard rules

How tags and rules give commands

Files are tagged using tagging rules

"foo/bar.ml": rectypes

Rules then produce commands with tagged holes

let tagged_hole =
tagged_for(ml)++"ocaml"++"compile"++"byte" in
Cmd(S[A"ocamlc";A"-c";T tagged_hole;P ml;A"-o";P cmo])

These holes are filled by command fragments (such as flags)

flag ["ocaml"; "compile"; "byte"; "rectypes"]
(A"-rectypes")
Tags and dependencies

One can define dependencies triggered by combinations of tags:

```ocaml
dep ["ocaml"; "link"; "byte"; "program"; "plugin:foo"]
["plugin/pluginlib.cma"; "plugin/plugin(foo).cmo"]
```

By tagging files we make things happen:

"test.byte": plugin:foo
Outline

1. Introduction
2. Regular OCaml projects
3. Dealing with exceptions to standard rules
4. Writing an ocamlbuild plugin
5. General features
6. Conclusion
ocamlbuild, a tool for automatic compilation of OCaml projects

Writing an ocamlbuild plugin

Not a specific language, but plain OCaml code

- Plugins are compiled on the fly
- Dynamic configuration is feasible

With a plugin one can:

- Extend rules (add new ones, override old ones)
- Add flags and dependencies based on tags
- Tag files
- Change options
- Define the directory structure precisely
- Help ocamldep
- Specify external libraries
A plugin example

Let’s read it in live...
Outline

1. Introduction
2. Regular OCaml projects
3. Dealing with exceptions to standard rules
4. Writing an ocamlbuild plugin
5. General features
6. Conclusion
Parallel execution where applicable

- You select the maximum number of jobs (-j N)
- Rules know how to ask for parallel targets
- The system keeps things scheduled correctly
- Example: Separate compilation of byte code
- (Optimal scheduling would require a static graph)
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

```
00:00:00 0  (0) STARTING
```

Berke Durak, Nicolas Pouillard
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

```
00:00:00 1   (0) back.ml.depends 0-------- |
```
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

```
00:00:00 8 (0) keyword.mli.depends 0-b---i- |
```
ocamlbuild, a tool for automatic compilation of OCaml projects

General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

```
00:00:00 16  (0) mark.cmi  O-B---I- /
```
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

```
00:00:00 20 (0) stringSet.cmi  0-B---I- /
```
ocamlbuild, a tool for automatic compilation of OCaml projects

General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

| 00:00:00 | 24  | (0) time.mli.depends | 0-b---i- / |
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

```
00:00:00 32  (0) stdlib.ml.depends  0-b---i- -
```
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:00 35 (0) stringSet.cmx

ONb---i--
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

```
00:00:00 37  (O) settings.cmx
```

0Nb---i--- -
ocamlbuild, a tool for automatic compilation of OCaml projects

General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:00 44  (0) lineCount.cmx  ONb---i-  |
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

```
00:00:00 45 (0) interface.ml.depends Onb---i- |
```
ocamlbuild, a tool for automatic compilation of OCaml projects
General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

```
00:00:00 45  (0) interface.ml.depends  Onb---i-  |
```
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:01 52 (0) stringMap.ml.depends Onb---i- |
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:01 53 (0) printer.cmx

ONb---i- |
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:01 53 (0) printer.cmx ONb---i- |
ocamlbuild, a tool for automatic compilation of OCaml projects
General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:01 57 (0) time.cmx ONb---i- /
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

```
00:00:01 64  (0) partialGrammar.cmi  OnB---I- /
```
ocamlbuild, a tool for automatic compilation of OCaml projects

General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:01 67  (0) parameters.ml.depends  Onb---i- /
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:01 72 (0) misc.ml.depends 0nb---i---
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:01 74 (0) keyword.ml.depends  Onb---i---
ocamlbuild, a tool for automatic compilation of OCaml projects
General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:01 77 (0) error.cmi  OnB---I- -
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:01 82  (0) parameters.cmx  ONb---i- |
General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:01 84 (0) action.cmx ONb---i- |
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:01 87  (0) parser.mli.depends       Onb---i- |
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:02 96 (0) parserAux.cmx ONb- ---i- |
ocamlbuild, a tool for automatic compilation of OCaml projects

General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

```
00:00:02 103 (0) tarjan.ml.depends Onb---i-- |  
```
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:02 106 (0) unionFind.cmx  ONb---i- |
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:02 108 (0) lexer.mll  Onb---i- /
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

```
00:00:02 108 (0) lexer.mll
```

ocamlbuild, a tool for automatic compilation of OCaml projects

General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:02 110 (0) lexer.cmo  OnB---i--
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

```
00:00:02 111 (0) parser.cmx ONb---i---
```
ocamlbuild, a tool for automatic compilation of OCaml projects

General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:02 112 (0) partialGrammar.cmx ONb---i---
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

```
00:00:02 114  (0) lexer.cmx
```

ocamlbuild, a tool for automatic compilation of OCaml projects

General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:02 116 (0) codeBits.mli.depends  Onb---i- |
ocamlbuild, a tool for automatic compilation of OCaml projects

General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:03 118  (0) preFront.cmx ONb---i- |
ocamlbuild, a tool for automatic compilation of OCaml projects

General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:03 120 (0) tokenType.cmx ONb---i- |
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

```
00:00:03 123 (0) inliner.cmi OnB---I- |
```
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:03 126 (0) traverse.cmx ONb---i- /
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:03 126 (0) traverse.cmx   ONb---i- /
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:03 129 (0) code.cmi OnB---I- /
ocamlbuild, a tool for automatic compilation of OCaml projects

General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

<table>
<thead>
<tr>
<th>Time</th>
<th>Target Count</th>
<th>Target Name</th>
<th>Command Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:00:03</td>
<td>131</td>
<td>lr1.mli.depends</td>
<td>0nb---i- /</td>
</tr>
</tbody>
</table>
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

```
00:00:03 134 (0) lookahead.mli.depends Onb---i---
```
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:03 137 (0) gMap.ml.depends  Onb---i-- -
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:03 144 (0) lr1.cmi OnB---I- -
ocamlbuild, a tool for automatic compilation of OCaml projects

General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:03 146 (0) item.ml.depends 0nb----i----
ocamlbuild, a tool for automatic compilation of OCaml projects

General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:03 149  (0) patricia.cmi  OnB---I-  |
ocamlbuild, a tool for automatic compilation of OCaml projects

General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:03 151 (0) patricia.cmx ONb---i- |
ocamlbuild, a tool for automatic compilation of OCaml projects

General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

```
00:00:03 151 (0) patricia.cmx ONb---i- |
```
ocamlbuild, a tool for automatic compilation of OCaml projects
General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:04 154 (0) front.cmi OnB---I- |
ocamlbuild, a tool for automatic compilation of OCaml projects

General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:04 164 (0) listMonad.ml.deps |
ocamlbuild, a tool for automatic compilation of OCaml projects

General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:04 167 (0) listMonad.cmx ONb---i- |
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:04 170  (0) infer.cmi  OnB---I-  /
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

```
00:00:04 171 (0) lexmli.mll Onb---i- /
```
ocamlbuild, a tool for automatic compilation of OCaml projects

General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:04 172 (0) lexmli.ml.depends 0nb---i- /
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

```
00:00:04 174 (0) lexdep.mll
```

Berke Durak, Nicolas Pouillard
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:04 177 (0) interface.cmx ONb---i---
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

```
00:00:04 178  (0)  IO.ml.depends      Onb---i-  |
```
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

```
00:00:04 181 (0) lexmli.cmx ONb---i- |
```
ocamlbuild, a tool for automatic compilation of OCaml projects

General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:04 183 (0) IO.cmx

ONb---i- |
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:05 187 (0) infer.cmx ONb----i- |
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:05 190 (0) dot.cmi
ocamlbuild, a tool for automatic compilation of OCaml projects

General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:05 193 (0) compressedBitSet.cmi OnB---I- /
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:05 195 (0) dot.cmx ONb---i- /
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:05 197 (0) grammar.cmx ONb---i- /
ocamlbuild, a tool for automatic compilation of OCaml projects

General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:05 197 (0) grammar.cmx ONb-—i- /
ocamlbuild, a tool for automatic compilation of OCaml projects
General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:05 197 (0) grammar.cmx  ONb---i--
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:05 200  (0) infiniteArray.cmi  OnB---I--
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

```
00:00:05 201 (0) item.cmx
```
ocamlbuild, a tool for automatic compilation of OCaml projects

General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:05 204 (0) breadth.mli.depends Onb---i- |
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

```
00:00:05 208  (0) invariant.ml.depends Onb---i-  |
```
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:06 212 (0) invariant.cmx ONb---i- |
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:06 213  (0) inliner.cmx   ONb---i-  |
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

00:00:06 214 (0) code.cmx  ONb---i- /
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

```
00:00:06 216 (0) back.native ONbP--iL -
```
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

Finished, 216 targets (0 cached) in 00:00:06.
General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

Finished, 216 targets (0 cached) in 00:00:06.
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

Finished, 216 targets (0 cached) in 00:00:06.
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

Finished, 216 targets (0 cached) in 00:00:06.
A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

Finished, 216 targets (0 cached) in 00:00:06.
ocamlbuild, a tool for automatic compilation of OCaml projects

General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

Finished, 216 targets (0 cached) in 00:00:06.
ocamlbuild, a tool for automatic compilation of OCaml projects

General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

Finished, 216 targets (0 cached) in 00:00:06.
ocamlbuild, a tool for automatic compilation of OCaml projects

General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

Finished, 216 targets (0 cached) in 00:00:06.
ocamlbuild, a tool for automatic compilation of OCaml projects

General features

A status bar for your visual comfort

- Compilation tools echo commands and their output
- This creates a long and boring output that scrolls too fast
- Here you can keep an eye on what is going on!
- It succinctly displays time, number of targets, and tags
- Command outputs are correctly multiplexed
- A trace of the commands executed is kept in a log file
- This log file can be used as the basis of a shell script

Example

Finished, 216 targets (0 cached) in 00:00:06.
Hygiene and sterilization

`ocamlbuild` has a Hygiene Squad (HS) that checks your source tree for cleanliness

It has preconceived but useful cleanliness notions

- Files dirty by default: `.cmi`, `.cmo`, `.cma`, `.cmx`
- `ocamllex/ocamlyacc` files: `.ml` if `.mll`, `.ml` & `.mli` if `.mly`
ocamlbuild has a Hygiene Squad (HS) that checks your source tree for cleanliness

It has preconceived but useful cleanliness notions

- Files dirty by default: `.cmi`, `.cmo`, `.cma`, `.cmx`
- `ocamllex/ocamlyacc` files: `.ml` if `.mll`, `.ml` & `.mli` if `.mly`

If unsatisfied, the HS produces a sterilization script

- Read it carefully (or work with versioning)
- Run at your own risks
Hygiene and sterilization

`ocamlbuild` has a Hygiene Squad (HS) that checks your source tree for cleanliness.

It has preconceived but useful cleanliness notions:
- Files dirty by default: `.cmi`, `.cmo`, `.cma`, `.cmx`...
- `ocamllex/ocamlyacc` files: `.ml` if `.mll`, `.ml`&`.mli` if `.mly`...

If unsatisfied, the HS produces a sterilization script:
- Read it carefully (or work with versioning)
- Run at your own risks

HS can be told of exceptions:
Files or directories tagged as `not_hygienic` or `precious`. 
Some supported tools

**Menhir as an ocamlyacc replacement**

- Enabled with the `use_menhir` global tag or the `-use-menhir` option
- Handles implicit dependencies using `-infer`
Some supported tools

**Menhir as an ocamlyacc replacement**
- Enabled with the `use_menhir` global tag or the `-use-menhir` option
- Handles implicit dependencies using `--infer`

**Ocamlproc to build your doc**
- Separated construction using `(-dump/-load)`
- Handles HTML, LaTeX, MAN, DOT, TeXi
Some supported tools

**Menhir as an ocamlyacc replacement**
- Enabled with the `use_menhir` global tag or the `-use-menhir` option
- Handles implicit dependencies using `--infer`

**Ocamldoc to build your doc**
- Separated construction using `(-dump/ -load)`
- Handles HTML, LaTeX, MAN, DOT, TeXI

**Camlp4 aware**
- Tags allow to setup any installed Camlp4 preprocessor
- Fine grained dependencies help a lot...
Outline

1 Introduction

2 Regular OCaml projects

3 Dealing with exceptions to standard rules

4 Writing an ocamlbuild plugin

5 General features

6 Conclusion
ocamlbuild, a tool for automatic compilation of OCaml projects

Conclusion

Resume

**ocamlbuild** can be used in three ways:

- With only command-line options for fully regular projects
- With the `_tags` file for intermediate projects
- With a plugin for the most complex projects
ocamlbuild, a tool for automatic compilation of OCaml projects

Conclusion

Resume

**ocamlbuild** can be used in three ways:

- With only command-line options for fully regular projects
- With the `_tags` file for intermediate projects
- With a plugin for the most complex projects

**ocamlbuild** saves your time by:

- Building your project gently
- Compiling only as necessary
- Running commands in parallel
- Keeping your house clean
- Letting you concentrate on your code!
Acknowledgments

For enlightening discussions about OCaml internals:

- Xavier Leroy
- Damien Doligez
Acknowledgments

For enlightening discussions about OCaml internals:

- Xavier Leroy
- Damien Doligez

For his insights about OCaml dependencies:

- Alain Frisch
Acknowledgments

For enlightening discussions about OCaml internals:
- Xavier Leroy
- Damien Doligez

For his insights about OCaml dependencies:
- Alain Frisch

For letting this happen:
- Michel Mauny
Conclusion

- **ocamlbuild** is not perfect but already damn useful
Conclusion

- **ocamlbuild** is not perfect but already damn useful
- Try it now! It’s in OCaml 3.10!