

 **c-w-m / waf**
forked from waf-project/waf

The Waf build system <https://waf.io/>

Edit

Add topics

2,740 commits

12 branches

84 releases

84 contributors

Branch: master

New pull request

Create new file

Upload files


Find file

Clone or download

















This branch is even with waf-project:master.


Pull request

Compare

 **selsky** and **ita1024** python docs: check_python_version also defines PYTHONARCHDIR

Latest commit efea037 10 days ago

 build_system_kit	Remove the TaskBase class hierarchy level	a year ago
 demos	java demos: correct jaropts parameter example to an array as it is th...	2 months ago
 docs	docs	4 months ago
 playground	eclipse: generate an external builder when no CDT is used in the proj...	23 days ago
 tests	Update docs for 2018	5 months ago
 utils	Typos	3 months ago
 waflib	python docs: check_python_version also defines PYTHONARCHDIR	10 days ago
 zip	iso8859-1 -> latin-1	a year ago
 .gitignore	Ignore waf.bat (generated by Windows build).	a year ago
 ChangeLog	waf-2.0.8	12 days ago
 DEVEL	Docs	9 months ago
 README.md	docs	9 months ago
 TODO	waf-2.0.2	7 months ago
 configure	Update use_config - Issue 1608	3 years ago
 waf-light	waf-2.0.8	12 days ago
 wscript	waf-2.0.8	12 days ago

 **README.md**

ABOUT WAF

Waf is a Python-based framework for configuring, compiling and installing applications. Here are perhaps the most important features of Waf:

- Automatic build order*: the build order is computed from input and output files, among others
- Automatic dependencies*: tasks to execute are detected by hashing files and commands
- Performance*: tasks are executed in parallel automatically, the startup time is meant to be fast (separation between configuration and build)
- Flexibility*: new commands and tasks can be added very easily through subclassing, bottlenecks for specific builds can be eliminated through dynamic method replacement
- Extensibility*: though many programming languages and compilers are already supported by default, many others are available as extensions
- IDE support*: Eclipse, Visual Studio and Xcode project generators (waflib/extras/)
- Documentation*: the application is based on a robust model documented in [The Waf Book](#) and in the [API docs](#)
- Python compatibility*: cPython 2.5 to 3.4, Jython 2.5, IronPython, and Pypy

Waf is used in particular by innovative companies such as [Avalanche Studios](#) and by open-source projects such as [RTEMS](#). Learn more about Waf by reading [The Waf Book](#).

For researchers and build system writers, Waf also provides a framework for creating [custom build systems](#) and [package distribution systems](#).

Download the project from our page on [waf.io](#) or from a mirror on [freehackers.org](#), consult the [manual](#), the [API documentation](#) and the [showcases](#) and [experiments](#).

HOW TO CREATE THE WAF SCRIPT

Python >= 2.6 is required to generate the waf script, and the resulting file can then run on Python 2.5. Just run:

```
$ ./waf-light configure build
```

Or, if several python versions are installed:

```
$ python3 ./waf-light configure build
```

The Waf tools in `waf/lib/extras` are not added to the waf script. To add some of them, use the `--tools` switch. An absolute path can be passed if the module does not exist under the 'extras' folder:

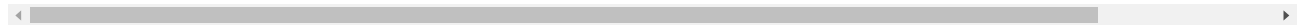
```
$ ./waf-light --tools=swig
```

To customize the initialization, pass the parameter 'prelude'. Here is for example how to create a waf file using the `compat15` module:

```
$ ./waf-light --tools=compat15 --prelude=$'\tfrom waf/lib/extras import compat15\n'
```

Although any kind of initialization is possible, using the build system kit may be easier (folder `build_system_kit`):

```
$ ./waf-light --make-waf --tools=compat15,/comp/waf/aba.py --prelude=$'\tfrom waf/lib/extras import compat15
```



To avoid regenerating the waf file all the time, just set the `WAFDIR` environment variable to the directory containing "waf/lib".

HOW TO RUN THE EXAMPLES

Try this:

```
cp waf demos/c/  
cd demos/c/  
./waf configure build
```