

---

# **BenchBuild - Empirical Research Toolkit Documentation**

***Release 1.3.2***

**Andreas Simbürger**

**Jan 13, 2017**



---

## Contents:

---

<b>1</b>	<b>BenchBuild: Empirical-Research Toolkit</b>	<b>1</b>
1.1	Features . . . . .	1
1.2	Requirements . . . . .	1
1.3	Installation . . . . .	1
1.4	Configuration . . . . .	2
1.5	SLURM Configuration . . . . .	2
1.6	Gentoo Configuration . . . . .	3
1.7	Convert an automatic Gentoo project to a static one . . . . .	3
1.8	Documentation . . . . .	3
1.9	Misc . . . . .	4
1.9.1	benchbuild package . . . . .	4
1.9.1.1	Subpackages . . . . .	4
1.9.1.2	Submodules . . . . .	55
1.9.1.3	benchbuild.bootstrap module . . . . .	55
1.9.1.4	benchbuild.container module . . . . .	55
1.9.1.5	benchbuild.driver module . . . . .	57
1.9.1.6	benchbuild.experiment module . . . . .	57
1.9.1.7	benchbuild.likwid module . . . . .	60
1.9.1.8	benchbuild.log module . . . . .	61
1.9.1.9	benchbuild.project module . . . . .	62
1.9.1.10	benchbuild.report module . . . . .	63
1.9.1.11	benchbuild.run module . . . . .	63
1.9.1.12	benchbuild.settings module . . . . .	64
1.9.1.13	benchbuild.slurm module . . . . .	66
1.9.1.14	benchbuild.test module . . . . .	66
<b>2</b>	<b>Indices and tables</b>	<b>67</b>
	<b>Python Module Index</b>	<b>69</b>



---

## BenchBuild: Empirical-Research Toolkit

---

BenchBuild provides a lightweight toolkit to conduct empirical compile-time and run-time experiments. Striving to automate all tedious and error-prone tasks, it downloads, configure and builds all supported projects fully automatic and provides tools to wrap the compiler and any resulting binary with a customized measurement.

All results can be stored as the user desires. BenchBuild tracks the execution status of all its managed projects inside an own database.

### 1.1 Features

- Wrap compilation commands with arbitrary measurement functions written in python.
- Wrap binary commands with arbitrary measurement functions written in python.
- Parallel benchmarking using the SLURM cluster manager.
- Compile-time support for the gentoo portage tree using the `uchroot` command.

### 1.2 Requirements

You need a working PostgreSQL installation (There is no special reason for PostgreSQL, but the backend is not configurable at the moment). In addition to the PostgreSQL server, you need `libpqxx` available for the `psycpg2` package that benchbuild uses to connect.

### 1.3 Installation

After you have installed all necessary libraries, you can just clone this repo and install via pip.

```
$ pip install benchbuild
```

This will pull in all necessary python libraries into your local python installation. The installed program to control the study is called `benchbuild`.

## 1.4 Configuration

`benchbuild` can be configured in various ways: (1) command-line arguments, (2) configuration file in `.json` format, (3) environment variables.

You can dump the current active configuration with the command: .. code-block:: bash

```
$ benchbuild run -d
BB_BENCHBUILD_EBUILD="" BB_BENCHBUILD_PREFIX="/bench-build"
BB_BUILD_DIR="/tmp/benchbuild" ...
```

You can dump this information in `.json` format using the command:

```
$ benchbuild run -s
```

However, be careful. It dumps `_all_` configuration to `.json`, even those that are usually derived automatically (like UUIDs). In the future, this will be avoided automatically. For now, you should remove all ID related variables from the resulting `.json` file. The configuration file is searched from the current directory upwards automatically. Some key configuration variables:

BB_BUILD_DIR	The directory we place our temporary artifacts in.
BB_TMP_DIR	The directory we place our downloads in.
BB_SRC_DIR	The directory we pull additional artifacts from (e.g., patches)
BB_CLEAN	Should the build directory be cleaned after the run?
BB_CONFIG_FILE	Where is the config file? If you prefer an absolute location over automatic discovery.
BB_DB_HOST	Hostname of the database
BB_DB_NAME	Name of the database
BB_DB_USER	Username of the database
BB_DB_PASS	Password of the database
BB_DB_ROLLBACK	For testing Rollback all db actions after a run.
BB_JOBS	Number of threads to use for compiling / run-time testing.

You can set these in the `.json` config file or directly via environment variables. However, make sure that the values you pass in from the environment are valid JSON, or the configuration structure may ignore your input (or break).

## 1.5 SLURM Configuration

If you want to run experiments in parallel on a cluster managed by SLURM, you can use BenchBuild to generate a bash script that is compatible with SLURM's `sbatch` command. The following settings control SLURM's configuration:

BB_SLURM_ACCOUNT	The resource account log in to.
BB_SLURM_CPUS_PER_NODE	How many cores/threads should we request per node?
BB_SLURM_EXCLUSIVE	Should we request the node exclusively or share it with other tasks?
BB_SLURM_LOGS	Where do we put our logs (deprecated).
BB_SLURM_MAX_RUNNING	We generate array-Jobs. This parameter controls the number of array elements that are allowed to run in parallel.
BB_SLURM_MULTITHREADED	Should Hyper-Threading be enabled or not?
BB_SLURM_NICE	Adjust our priority on the cluster manually.
BB_SLURM_NICE_CLEAN	Adjust the priority of the clean jobs.
BB_SLURM_NODE_DIR	Where can we place our artifacts on the node?
BB_SLURM_PARTITION	Which partition should we run in?
BB_SLURM_SCRIPT	Base name of our resulting batch script.
BB_SLURM_TIMELIMIT	Enforce a timelimit on our batch jobs.

## 1.6 Gentoo Configuration

BenchBuild supports compile-time experiments on the complete portage tree of Gentoo Linux. You need to configure a few settings to make it work:

BB_GENTOO_AUTOTEST_LOC	A txt file that lists all gentoo package atoms that should be considered.
BB_GENTOO_AUTOTEST_FTP_PROXY	Proxy server for gentoo downloads.
BB_GENTOO_AUTOTEST_HTTP_PROXY	Proxy server for gentoo downloads.
BB_GENTOO_AUTOTEST_RSYNC_PROXY	Proxy server for gentoo downloads.

## 1.7 Convert an automatic Gentoo project to a static one

Gentoo projects are generated dynamically based on the `AutoPortage` class found in `pprof.gentoo.portage_gen`. If you want to define run-time tests for a dynamically generated project, you need to convert it to a static one, i.e., define a subclass of `AutoPortage` and add it to the configuration.

```
from pprof.projects.gentoo.portage_gen import AutoPortage

class BZip(AutoPortage):
    NAME = "app-arch"
    DOMAIN = "bzip2"

    def run_tests(self, experiment):
        """Add your custom test routines here."""
```

Now we just need to add this to the plugin registry via benchbuild's configuration file @ `CFG["plugins"]["projects"]`.

## 1.8 Documentation

For detailed API information please refer to the full [documentation](#):

## 1.9 Misc

### 1.9.1 benchbuild package

#### 1.9.1.1 Subpackages

##### benchbuild.experiments package

Experiments module.

By default, only experiments that are listed in the configuration are loaded automatically. See configuration variables:

`*_PLUGINS_AUTOLOAD *_PLUGINS_EXPERIMENTS`

`benchbuild.experiments.discover()`

Import all experiments listed in `PLUGINS_EXPERIMENTS`.

**Tests:**

```
>>> from benchbuild.settings import CFG
>>> from benchbuild.experiments import discover
>>> import logging as lg
>>> import sys
>>> l = lg.getLogger('benchbuild')
>>> lg.getLogger('benchbuild').setLevel(lg.DEBUG)
>>> lg.getLogger('benchbuild').handlers = [lg.StreamHandler(stream=sys.
↳ stdout)]
>>> CFG["plugins"]["experiments"] = ["benchbuild.non.existing", "benchbuild.
↳ experiments.raw"]
>>> discover()
Could not find 'benchbuild.non.existing'
ImportError: No module named 'benchbuild.non'
Found experiment: benchbuild.experiments.raw
```

#### Subpackages

##### benchbuild.experiments.polly package

#### Submodules

##### benchbuild.experiments.polly.openmp module

The ‘polly-openmp’ Experiment.

This experiment applies polly’s transformations with openmp code generation enabled to all projects and measures the runtime.

This forms the baseline numbers for the other experiments.

#### Measurements

**3 Metrics are generated during this experiment:** `time.user_s` - The time spent in user space in seconds (aka virtual time) `time.system_s` - The time spent in kernel space in seconds (aka system time) `time.real_s` - The time spent overall in seconds (aka Wall clock)



```
class benchbuild.experiments.polly.openmp.PollyOpenMP (projects=None, group=None)
    Bases: benchbuild.experiment.RuntimeExperiment

    Timing experiment with Polly & OpenMP support.

    NAME = 'polly-openmp'

    actions_for_project (p)
        Build & Run each project with Polly & OpenMP support.
```

### benchbuild.experiments.polly.openmpvect module

The 'polly-openmp-vectorize' Experiment.

This experiment applies polly's transformations with openmp code generation enabled to all projects and measures the runtime.

This forms the baseline numbers for the other experiments.

### Measurements

**3 Metrics are generated during this experiment:** time.user\_s - The time spent in user space in seconds (aka virtual time) time.system\_s - The time spent in kernel space in seconds (aka system time) time.real\_s - The time spent overall in seconds (aka Wall clock)

```
class benchbuild.experiments.polly.openmpvect.PollyOpenMPVectorizer (projects=None,
                                                                    group=None)
    Bases: benchbuild.experiment.RuntimeExperiment

    Timing experiment with Polly & OpenMP+Vectorizer support.

    NAME = 'polly-openmpvect'

    run_project (p)
```

### benchbuild.experiments.polly.polly module

#### The 'polly' Experiment

This experiment applies polly's transformations to all projects and measures the runtime.

This forms the baseline numbers for the other experiments.

### Measurements

**3 Metrics are generated during this experiment:** time.user\_s - The time spent in user space in seconds (aka virtual time) time.system\_s - The time spent in kernel space in seconds (aka system time) time.real\_s - The time spent overall in seconds (aka Wall clock)

```
class benchbuild.experiments.polly.polly.Polly (projects=None, group=None)
    Bases: benchbuild.experiment.RuntimeExperiment

    The polly experiment.

    NAME = 'polly'

    run_project (p)
```

## benchbuild.experiments.polly.pollyperformance module

### The ‘polly’ Experiment

This experiment applies polly’s transformations to all projects and measures the runtime.

This forms the baseline numbers for the other experiments.

### Measurements

**3 Metrics are generated during this experiment:** time.user\_s - The time spent in user space in seconds (aka virtual time) time.system\_s - The time spent in kernel space in seconds (aka system time) time.real\_s - The time spent overall in seconds (aka Wall clock)

**class** benchbuild.experiments.polly.pollyperformance.**PollyPerformance** (*projects=None, group=None*)

Bases: *benchbuild.experiment.RuntimeExperiment*

The polly performance experiment.

**NAME** = ‘pollyperformance’

**run\_project** (*p*)

**exception** benchbuild.experiments.polly.pollyperformance.**ShouldNotBeNone**

Bases: *RuntimeWarning*

User warning, if config var is null.

## benchbuild.experiments.polly.vectorize module

### The ‘polly-vectorize’ Experiment

This experiment applies polly’s transformations with stripmine vectorizer enabled to all projects and measures the runtime.

This forms the baseline numbers for the other experiments.

### Measurements

**3 Metrics are generated during this experiment:** time.user\_s - The time spent in user space in seconds (aka virtual time) time.system\_s - The time spent in kernel space in seconds (aka system time) time.real\_s - The time spent overall in seconds (aka Wall clock)

**class** benchbuild.experiments.polly.vectorize.**PollyVectorizer** (*projects=None, group=None*)

Bases: *benchbuild.experiment.RuntimeExperiment*

The polly experiment with vectorization enabled.

**NAME** = ‘polly-vectorize’

**run\_project** (*p*)

## Submodules

### benchbuild.experiments.compilestats module

The ‘compilestats’ Experiment.

This experiment is a basic experiment in the benchbuild study. It simply runs all projects after compiling it with -O3 and catches all statistics emitted by llvm

```
class benchbuild.experiments.compilestats.CompilestatsExperiment (projects=None,  
                                                                group=None)
```

Bases: *benchbuild.experiment.RuntimeExperiment*

The compilestats experiment.

**NAME** = ‘cs’

**actions\_for\_project** (*p*)

```
class benchbuild.experiments.compilestats.PollyCompilestatsExperiment (projects=None,  
                                                                group=None)
```

Bases: *benchbuild.experiment.RuntimeExperiment*

The compilestats experiment with polly enabled.

**NAME** = ‘p-cs’

**actions\_for\_project** (*p*)

```
benchbuild.experiments.compilestats.collect_compilestats (project, experiment, con-  
                                                            fig, clang, **kwargs)
```

Collect compilestats.

```
benchbuild.experiments.compilestats.get_compilestats (prog_out)
```

Get the LLVM compilation stats from :prog\_out:.

### benchbuild.experiments.compilestats\_ewpt module

The ‘compilestats\_ewpt’ Experiment.

Gathers compilation statistics for compiling the project with the EWPT alias analysis enabled.

```
class benchbuild.experiments.compilestats_ewpt.EWPTCompilestatsExperiment (projects=None,  
                                                                group=None)
```

Bases: *benchbuild.experiments.compilestats.CompilestatsExperiment*

**NAME** = ‘ewpt’

**extra\_cflags** ()

### benchbuild.experiments.empty module

The ‘empty’ Experiment.

This experiment is for debugging purposes. It only prepares the basic directories for benchbuild. No compilation & no run can be done with it.

```
class benchbuild.experiments.empty.Empty (projects=None, group=None)
```

Bases: *benchbuild.experiment.Experiment*

The empty experiment.

**NAME** = 'empty'

**actions\_for\_project** (*p*)  
Do nothing.

## benchbuild.experiments.papi module

PAPI based experiments.

These types of experiments (papi & papi-std) need to instrument the project with libbenchbuild support to work.

**class** benchbuild.experiments.papi.**PapiScopCoverage** (*projects=None, group=None*)  
Bases: *benchbuild.experiment.RuntimeExperiment*

PAPI-based dynamic SCoP coverage measurement.

**NAME** = 'papi'

**actions\_for\_project** (*p*)  
Create & Run a papi-instrumented version of the project.

This experiment uses the -jitable flag of libPolyJIT to generate dynamic SCoP coverage.

**run** ()  
Do the postprocessing, after all projects are done.

**class** benchbuild.experiments.papi.**PapiStandardScopCoverage** (*projects=None, group=None*)  
Bases: *benchbuild.experiments.papi.PapiScopCoverage*

PAPI Scop Coverage, without JIT.

**NAME** = 'papi-std'

**actions\_for\_project** (*p*)  
Create & Run a papi-instrumented version of the project.

This experiment uses the -jitable flag of libPolyJIT to generate dynamic SCoP coverage.

benchbuild.experiments.papi.**collect\_compilestats** (*project, experiment, clang, \*\*kwargs*)

Collect compilestats.

benchbuild.experiments.papi.**get\_compilestats** (*prog\_out*)  
Get the LLVM compilation stats from :prog\_out:.

## benchbuild.experiments.pjtest module

A test experiment for PolyJIT.

This experiment should only be used to test various features of PolyJIT. It provides only 1 configuration (maximum number of cores) and tests 2 run-time execution profiles of PolyJIT:

1. PolyJIT enabled, with specialization
2. PolyJIT enabled, without specialization

**class** benchbuild.experiments.pjtest.**Test** (*projects=None, group=None*)  
Bases: *benchbuild.experiments.polyjit.PolyJIT*

An experiment that executes all projects with PolyJIT support.

This is our default experiment for speedup measurements.

**NAME** = 'pj-test'

**actions\_for\_project** (*p*)

`benchbuild.experiments.pjtest.time_polyjit_and_polly` (*project, experiment, config, jobs, run\_f, args, \*\*kwargs*)

Run the given binary wrapped with time.

#### Parameters

- **project** – The `benchbuild.project`.
- **experiment** – The `benchbuild.experiment`.
- **config** – The `benchbuild.settings.config`.
- **jobs** – Number of cores we should use for this execution.
- **run\_f** – The file we want to execute.
- **args** – List of arguments that should be passed to the wrapped binary.
- **\*\*kwargs** – Dictionary with our keyword args. We support the following entries:
  - project\_name**: The real name of our project. This might not be the same as the configured project name, if we got wrapped with `::benchbuild.project.wrap_dynamic`
  - has\_stdin**: Signals whether we should take care of stdin.

## benchbuild.experiments.polyjit module

The 'polyjit' experiment.

This experiment uses `likwid` to measure the performance of all binaries when running with polyjit support enabled.

**class** `benchbuild.experiments.polyjit.Compilestats` (*projects=None, group=None*)

Bases: `benchbuild.experiments.polyjit.PolyJIT`

Gather compilestats, with enabled JIT.

**NAME** = 'pj-cs'

**actions\_for\_project** (*p*)

**class** `benchbuild.experiments.polyjit.PJITRaw` (*projects=None, group=None*)

Bases: `benchbuild.experiments.polyjit.PolyJIT`

An experiment that executes all projects with PolyJIT support.

This is our default experiment for speedup measurements.

**NAME** = 'pj-raw'

**actions\_for\_project** (*p*)

**class** `benchbuild.experiments.polyjit.PJITRegression` (*projects=None, group=None*)

Bases: `benchbuild.experiments.polyjit.PolyJIT`

This experiment will generate a series of regression tests.

This can be used every time a new revision is produced for PolyJIT, as it will automatically collect any new SCoPs detected, using the JIT.

The collection of the tests itself is integrated into the JIT, so this experiment looks a lot like a RAW experiment, except we don't run anything.

**NAME** = 'pj-collect'

**actions\_for\_project** (*p*)

**class** `benchbuild.experiments.polyjit.PJITlikwid` (*projects=None, group=None*)  
Bases: `benchbuild.experiments.polyjit.PolyJIT`

An experiment that uses likwid's instrumentation API for profiling.

This instruments all projects with likwid instrumentation API calls in key regions of the JIT.

This allows for arbitrary profiling of PolyJIT's overhead and run-time

**NAME** = 'pj-likwid'

**actions\_for\_project** (*p*)

**class** `benchbuild.experiments.polyjit.PJITpapi` (*projects=None, group=None*)  
Bases: `benchbuild.experiments.polyjit.PolyJIT`

Experiment that uses PolyJIT's instrumentation facilities.

This uses PolyJIT to instrument all projects with libPAPI based region measurements. In the end the region measurements are aggregated and metrics like the dynamic SCoP coverage are extracted.

This uses the same set of flags as all other PolyJIT based experiments.

**NAME** = 'pj-papi'

**actions\_for\_project** (*p*)

**run** ()

Do the postprocessing, after all projects are done.

**class** `benchbuild.experiments.polyjit.PJITperf` (*projects=None, group=None*)  
Bases: `benchbuild.experiments.polyjit.PolyJIT`

An experiment that uses linux perf tools to generate flamegraphs.

**NAME** = 'pj-perf'

**actions\_for\_project** (*p*)

**class** `benchbuild.experiments.polyjit.PolyJIT` (*projects=None, group=None*)  
Bases: `benchbuild.experiment.RuntimeExperiment`

The polyjit experiment.

**actions\_for\_project** (*p*)

**classmethod** **init\_project** (*project*)

Execute the benchbuild experiment.

**We perform this experiment in 2 steps:**

1. with likwid disabled.
2. with likwid enabled.

**Parameters** **project** – The project we initialize.

**Returns** The initialized project.

**class** `benchbuild.experiments.polyjit.PolyJITFull` (*projects=None, group=None*)  
Bases: `benchbuild.experiments.polyjit.PolyJIT`

An experiment that executes all projects with PolyJIT support.

This is our default experiment for speedup measurements.

**NAME** = 'pj'

**actions\_for\_project** (*p*)

`benchbuild.experiments.polyjit.run_raw` (*project, experiment, config, run\_f, args, \*\*kwargs*)

Run the given binary wrapped with nothing.

#### Parameters

- **project** – The `benchbuild.project`.
- **experiment** – The `benchbuild.experiment`.
- **config** – The `benchbuild.settings.config`.
- **run\_f** – The file we want to execute.
- **args** – List of arguments that should be passed to the wrapped binary.
- **\*\*kwargs** – Dictionary with our keyword args. We support the following entries:  
**project\_name**: The real name of our project. This might not be the same as the configured project name, if we got wrapped with `::benchbuild.project.wrap_dynamic`  
**has\_stdin**: Signals whether we should take care of stdin.

`benchbuild.experiments.polyjit.run_with_likwid` (*project, experiment, config, jobs, run\_f, args, \*\*kwargs*)

Run the given file wrapped by likwid.

#### Parameters

- **project** – The `benchbuild.project`.
- **experiment** – The `benchbuild.experiment`.
- **config** – The `benchbuild.settings.config`.
- **jobs** – Number of cores we should use for this execution.
- **run\_f** – The file we want to execute.
- **args** – List of arguments that should be passed to the wrapped binary.
- **\*\*kwargs** – Dictionary with our keyword args. We support the following entries:  
**project\_name**: The real name of our project. This might not be the same as the configured project name, if we got wrapped with `::benchbuild.project.wrap_dynamic`  
**has\_stdin**: Signals whether we should take care of stdin.

`benchbuild.experiments.polyjit.run_with_papi` (*project, experiment, config, jobs, run\_f, args, \*\*kwargs*)

Run the given file with PAPI support.

This just runs the project as PAPI support should be compiled in already. If not, this won't do a lot.

#### Parameters

- **project** – The `benchbuild.project`.
- **experiment** – The `benchbuild.experiment`.
- **config** – The `benchbuild.settings.config`.
- **jobs** – Number of cores we should use for this execution.
- **run\_f** – The file we want to execute.
- **args** – List of arguments that should be passed to the wrapped binary.

- **\*\*kwargs** – Dictionary with our keyword args. We support the following entries:  
**project\_name:** The real name of our project. This might not be the same as the configured project name, if we got wrapped with `::benchbuild.project.wrap_dynamic`  
**has\_stdin:** Signals whether we should take care of stdin.

```
benchbuild.experiments.polyjit.run_with_perf(project, experiment, config, jobs, run_f,  
                                             args, **kwargs)
```

Run the given binary wrapped with time.

#### Parameters

- **project** – The `benchbuild.project`.
- **experiment** – The `benchbuild.experiment`.
- **config** – The `benchbuild.settings.config`.
- **jobs** – Number of cores we should use for this execution.
- **run\_f** – The file we want to execute.
- **args** – List of arguments that should be passed to the wrapped binary.
- **\*\*kwargs** – Dictionary with our keyword args. We support the following entries:  
**project\_name:** The real name of our project. This might not be the same as the configured project name, if we got wrapped with `::benchbuild.project.wrap_dynamic`  
**has\_stdin:** Signals whether we should take care of stdin.

```
benchbuild.experiments.polyjit.run_with_time(project, experiment, config, jobs, run_f,  
                                             args, **kwargs)
```

Run the given binary wrapped with time.

#### Parameters

- **project** – The `benchbuild.project`.
- **experiment** – The `benchbuild.experiment`.
- **config** – The `benchbuild.settings.config`.
- **jobs** – Number of cores we should use for this execution.
- **run\_f** – The file we want to execute.
- **args** – List of arguments that should be passed to the wrapped binary.
- **\*\*kwargs** – Dictionary with our keyword args. We support the following entries:  
**project\_name:** The real name of our project. This might not be the same as the configured project name, if we got wrapped with `::benchbuild.project.wrap_dynamic`  
**has\_stdin:** Signals whether we should take care of stdin.

```
benchbuild.experiments.polyjit.run_without_recompile(project, experiment, config,  
                                                    jobs, run_f, args, **kwargs)
```

Run the given binary wrapped with time.

#### Parameters

- **project** – The `benchbuild.project`.
- **experiment** – The `benchbuild.experiment`.
- **config** – The `benchbuild.settings.config`.



- **jobs** – Number of cores we should use for this execution.
- **run\_f** – The file we want to execute.
- **args** – List of arguments that should be passed to the wrapped binary.
- **\*\*kwargs** – Dictionary with our keyword args. We support the following entries:  
**project\_name**: The real name of our project. This might not be the same as the configured project name, if we got wrapped with `::benchbuild.project.wrap_dynamic`  
**has\_stdin**: Signals whether we should take care of stdin.

## benchbuild.experiments.raw module

The ‘raw’ Experiment.

This experiment is the basic experiment in the benchbuild study. It simply runs all projects after compiling it with -O3. The binaries are wrapped with the time command and results are written to the database.

This forms the baseline numbers for the other experiments.

## Measurements

**3 Metrics are generated during this experiment:** `time.user_s` - The time spent in user space in seconds (aka virtual time) `time.system_s` - The time spent in kernel space in seconds (aka system time) `time.real_s` - The time spent overall in seconds (aka Wall clock)

**class** `benchbuild.experiments.raw.RawRuntime` (*projects=None, group=None*)  
 Bases: `benchbuild.experiment.RuntimeExperiment`

The polyjit experiment.

**NAME** = ‘raw’

**actions\_for\_project** (*project*)  
 Compile & Run the experiment with -O3 enabled.

`benchbuild.experiments.raw.run_with_time` (*project, experiment, config, jobs, run\_f, args, \*\*kwargs*)

Run the given binary wrapped with time.

### Parameters

- **project** – The benchbuild project that has called us.
- **experiment** – The benchbuild experiment which we operate under.
- **config** – The benchbuild configuration we are running with.
- **jobs** – The number of cores we are allowed to use. This may differ from the actual amount of available cores, obey it. We should enforce this from the outside. However, at the moment we do not do this.
- **run\_f** – The file we want to execute.
- **args** – List of arguments that should be passed to the wrapped binary.
- **\*\*kwargs** – Dictionary with our keyword args. We support the following entries:  
**project\_name**: The real name of our project. This might not be the same as the configured project name, if we got wrapped with `::benchbuild.project.wrap_dynamic`

has\_stdin: Signals whether we should take care of stdin. may\_wrap:

Project may signal that it they are not suitable for wrapping. Usually because they scan/parse the output, which may interfere with the output of the wrapper binary.

### benchbuild.projects package

Projects module.

By default, only projects that are listed in the configuration are loaded automatically. See configuration variables:

`*_PLUGINS_AUTOLOAD *_PLUGINS_PROJECTS`

`benchbuild.projects.discover()`

### Subpackages

#### benchbuild.projects.apollo package

#### Submodules

#### benchbuild.projects.apollo.group module

```
class benchbuild.projects.apollo.group.ApolloGroup(exp)
    Bases: benchbuild.project.Project
    GROUP = 'apollo'
    path_suffix = 'src'
```

#### benchbuild.projects.apollo.rodinia module

```
class benchbuild.projects.apollo.rodinia.Rodinia(exp)
    Bases: benchbuild.projects.apollo.group.ApolloGroup
    DOMAIN = 'scientific'
    NAME = 'rodinia'
    SRC_FILE = 'rodinia_3.1.tar.bz2'
    VERSION = '3.1'
    build()
    configure()
    download()
    prepare()
    run_tests(experiment)
    src_dir = 'rodinia_3.1'
    src_uri = 'http://www.cs.virginia.edu/~kw5na/lava/Rodinia/Packages/Current/rodinia_3.1.tar.bz2'
```

**benchbuild.projects.apollo.scimark module**

```
class benchbuild.projects.apollo.scimark.SciMark(exp)
    Bases: benchbuild.projects.apollo.group.ApolloGroup

    DOMAIN = 'scientific'
    NAME = 'scimark'
    SRC_FILE = 'scimark2_1c.zip'
    VERSION = '2.1c'

    build()
    configure()
    download()
    prepare()
    run_tests(experiment)
    src_uri = 'http://math.nist.gov/scimark2/scimark2_1c.zip'
```

**benchbuild.projects.benchbuild package****Submodules****benchbuild.projects.benchbuild.bzip2 module**

```
class benchbuild.projects.benchbuild.bzip2.Bzip2(exp)
    Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup

    DOMAIN = 'compression'
    NAME = 'bzip2'
    SRC_FILE = 'bzip2-1.0.6.tar.gz'
    VERSION = '1.0.6'

    build()
    configure()
    download()
    prepare()
    run_tests(experiment)
    src_dir = 'bzip2-1.0.6'
    src_uri = 'http://www.bzip.org/1.0.6/bzip2-1.0.6.tar.gz'
    testfiles = ['text.html', 'chicken.jpg', 'control', 'input.source', 'liberty.jpg']
```

### benchbuild.projects.benchbuild.ccrypt module

```
class benchbuild.projects.benchbuild.ccrypt.Ccrypt(exp)
    Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup
    ccrypt benchmark
    DOMAIN = 'encryption'
    NAME = 'ccrypt'
    SRC_FILE = 'ccrypt-1.10.tar.gz'
    VERSION = '1.10'
    build()
    configure()
    download()
    run_tests(experiment)
    src_dir = 'ccrypt-1.10'
    src_uri = 'http://ccrypt.sourceforge.net/download/ccrypt-1.10.tar.gz'
```

### benchbuild.projects.benchbuild.crafty module

```
class benchbuild.projects.benchbuild.crafty.Crafty(exp)
    Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup
    crafty benchmark
    DOMAIN = 'scientific'
    NAME = 'crafty'
    SRC_FILE = 'crafty-25.2.zip'
    VERSION = '25.2'
    build()
    configure()
    download()
    run_tests(experiment)
    src_dir = 'crafty-25.2'
    src_uri = 'http://www.craftychess.com/downloads/source/crafty-25.2.zip'
```

### benchbuild.projects.benchbuild.croccopat module

```
class benchbuild.projects.benchbuild.croccopat.Croccopat(exp)
    Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup
    croccopat benchmark
    DOMAIN = 'verification'
    NAME = 'croccopat'
```

```
SRC_FILE = 'crocopat-2.1.4.zip'
VERSION = '2.1.4'
build()
configure()
download()
run_tests(experiment)
src_dir = 'crocopat-2.1.4'
src_uri = 'http://crocopat.googlecode.com/files/crocopat-2.1.4.zip'
```

### benchbuild.projects.benchbuild.ffmpeg module

```
class benchbuild.projects.benchbuild.ffmpeg.LibAV(exp)
    Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup
    LibAV benchmark
    DOMAIN = 'multimedia'
    NAME = 'ffmpeg'
    SRC_FILE = 'ffmpeg-3.1.3.tar.bz2'
    VERSION = '3.1.3'
    build()
    configure()
    download()
    fate_dir = 'fate-samples'
    fate_uri = 'rsync://fate-suite.libav.org/fate-suite/'
    run_tests(experiment)
    src_dir = 'ffmpeg-3.1.3'
    src_uri = 'http://ffmpeg.org/releases/ffmpeg-3.1.3.tar.bz2'
```

### benchbuild.projects.benchbuild.group module

```
class benchbuild.projects.benchbuild.group.BenchBuildGroup(exp)
    Bases: benchbuild.project.Project
    GROUP = 'benchbuild'
    path_suffix = 'src'
```

### benchbuild.projects.benchbuild.gzip module

```
class benchbuild.projects.benchbuild.gzip.Gzip(exp)
    Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup
    DOMAIN = 'compression'
```

```
NAME = 'gzip'
SRC_FILE = 'gzip-1.6.tar.xz'
VERSION = '1.6'
build()
configure()
download()
prepare()
run_tests(experiment)
src_dir = 'gzip-1.6'
src_uri = 'http://ftpmirror.gnu.org/gzip/gzip-1.6.tar.xz'
testfiles = ['text.html', 'chicken.jpg', 'control', 'input.source', 'liberty.jpg']
```

### benchbuild.projects.benchbuild.js module

```
class benchbuild.projects.benchbuild.js.SpiderMonkey(exp)
    Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup
    SpiderMonkey requires a legacy version of autoconf: autoconf-2.13
    DOMAIN = 'compilation'
    NAME = 'js'
    VERSION = ''
    build()
    configure()
    download()
    run_tests(experiment)
    src_dir = 'gecko-dev.git'
    src_uri = 'https://github.com/mozilla/gecko-dev.git'
    version = ''
```

### benchbuild.projects.benchbuild.lammps module

```
class benchbuild.projects.benchbuild.lammps.Lammps(exp)
    Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup
    LAMMPS benchmark
    DOMAIN = 'scientific'
    NAME = 'lammps'
    SRC_FILE = 'lammps.git'
    build()
    configure()
```

```
download()
prepare()
run_tests(experiment)
src_dir = 'lammmps.git'
src_uri = 'https://github.com/lammmps/lammmps'
```

### benchbuild.projects.benchbuild.lapack module

```
class benchbuild.projects.benchbuild.lapack.Lapack(exp)
    Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup
    DOMAIN = 'scientific'
    NAME = 'lapack'
    SRC_FILE = 'clapack.tgz'
    VERSION = '3.2.1'
    build()
    configure()
    download()
    run_tests(experiment)
    src_dir = 'CLAPACK-3.2.1'
    src_uri = 'http://www.netlib.org/clapack/clapack.tgz'

class benchbuild.projects.benchbuild.lapack.OpenBlas(exp)
    Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup
    DOMAIN = 'scientific'
    NAME = 'openblas'
    SRC_FILE = 'OpenBLAS'
    build()
    configure()
    download()
    run_tests(experiment)
    src_uri = 'https://github.com/xianyi/OpenBLAS'
```

### benchbuild.projects.benchbuild.leveldb module

```
class benchbuild.projects.benchbuild.leveldb.LevelDB(exp)
    Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup
    DOMAIN = 'database'
    NAME = 'leveldb'
    SRC_FILE = 'leveldb.src'
```

```
build()
configure()
download()
run_tests(experiment)
    Execute LevelDB's runtime configuration.

    Parameters experiment – The experiment's run function.
src_uri = 'https://github.com/google/leveldb'
```

### **benchbuild.projects.benchbuild.linpack module**

```
class benchbuild.projects.benchbuild.linpack.Linpack(exp)
    Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup
    Linpack (C-Version)
    DOMAIN = 'scientific'
    NAME = 'linpack'
    build()
    configure()
    download()
    src_uri = 'http://www.netlib.org/benchmark/linpackc.new'
```

### **benchbuild.projects.benchbuild.lulesh module**

```
class benchbuild.projects.benchbuild.lulesh.Lulesh(exp)
    Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup
    DOMAIN = 'scientific'
    NAME = 'lulesh'
    SRC_FILE = 'LULESH.cc'
    build()
    configure()
    download()
    run_tests(experiment)
    src_uri = 'https://codesign.llnl.gov/lulesh/LULESH.cc'
```

### **benchbuild.projects.benchbuild.luleshomp module**

```
class benchbuild.projects.benchbuild.luleshomp.LuleshOMP(exp)
    Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup
    Lulesh-OMP
    DOMAIN = 'scientific'
```



```
NAME = 'lulesh-omp'
SRC_FILE = 'LULESH_OMP.cc'
build()
configure()
download()
run_tests(experiment)
src_uri = 'https://codesign.llnl.gov/lulesh/LULESH_OMP.cc'
```

### benchbuild.projects.benchbuild.mcrypt module

```
class benchbuild.projects.benchbuild.mcrypt.MCrypt(exp)
    Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup
    MCrypt benchmark
    DOMAIN = 'encryption'
    NAME = 'mcrypt'
    SRC_FILE = 'mcrypt-2.6.8.tar.gz'
    VERSION = '2.6.8'
    build()
    configure()
    download()
    libmccrypt_dir = 'libmccrypt-2.5.8'
    libmccrypt_file = 'libmccrypt-2.5.8.tar.gz'
    libmccrypt_uri = 'http://sourceforge.net/projects/mcrypt/files/Libmccrypt/2.5.8/libmccrypt-2.5.8.tar.gz'
    mhash_dir = 'mhash-0.9.9.9'
    mhash_file = 'mhash-0.9.9.9.tar.gz'
    mhash_uri = 'http://sourceforge.net/projects/mhash/files/mhash/0.9.9.9/mhash-0.9.9.9.tar.gz'
    run_tests(experiment)
    src_dir = 'mcrypt-2.6.8'
    src_uri = 'http://sourceforge.net/projects/mcrypt/files/MCrypt/2.6.8mcrypt-2.6.8.tar.gz'
```

### benchbuild.projects.benchbuild.minisat module

```
class benchbuild.projects.benchbuild.minisat.Minisat(exp)
    Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup
    minisat benchmark
    DOMAIN = 'verification'
    NAME = 'minisat'
    SRC_FILE = 'minisat.git'
```

```
build()
configure()
download()
run_tests(experiment)
src_uri = 'https://github.com/niklasso/minisat'
```

#### benchbuild.projects.benchbuild.openssl module

```
class benchbuild.projects.benchbuild.openssl.LibreSSL(exp)
    Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup
    OpenSSL
    DOMAIN = 'encryption'
    NAME = 'libressl'
    SRC_FILE = 'libressl-2.1.6.tar.gz'
    VERSION = '2.1.6'
    build()
    configure()
    download()
    run_tests(experiment)
    src_dir = 'libressl-2.1.6'
    src_uri = 'http://ftp.openbsd.org/pub/OpenBSD/LibreSSL/libressl-2.1.6.tar.gz'
```

#### benchbuild.projects.benchbuild.postgres module

```
class benchbuild.projects.benchbuild.postgres.Postgres(exp)
    Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup
    postgres benchmark
    DOMAIN = 'database'
    NAME = 'postgres'
    prepare()
    run_tests(experiment)
    testfiles = ['pg_ctl', 'dropdb', 'createdb', 'pgbench']
```

#### benchbuild.projects.benchbuild.povray module

```
class benchbuild.projects.benchbuild.povray.Povray(exp)
    Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup
    povray benchmark
    DOMAIN = 'multimedia'
```

```
NAME = 'povray'
SRC_FILE = 'povray.git'
boost_src_dir = 'boost_1_59_0'
boost_src_file = 'boost_1_59_0.tar.bz2'
boost_src_uri = 'http://sourceforge.net/projects/boost/files/boost/1.59.0/boost_1_59_0.tar.bz2'
build()
configure()
download()
prepare()
run_tests(experiment)
src_uri = 'https://github.com/POV-Ray/povray'
```

### benchbuild.projects.benchbuild.python module

```
class benchbuild.projects.benchbuild.python.Python(exp)
    Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup
    python benchmarks
    DOMAIN = 'compilation'
    NAME = 'python'
    SRC_FILE = 'Python-3.4.3.tar.xz'
    VERSION = '3.4.3'
    build()
    configure()
    download()
    run_tests(experiment)
    src_dir = 'Python-3.4.3'
    src_uri = 'https://www.python.org/ftp/python/3.4.3/Python-3.4.3.tar.xz'
```

### benchbuild.projects.benchbuild.rasdaman module

```
class benchbuild.projects.benchbuild.rasdaman.Rasdaman(exp)
    Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup
    DOMAIN = 'database'
    NAME = 'Rasdaman'
    SRC_FILE = 'rasdaman.git'
    build()
    configure()
    download()
```

```
gdal_dir = 'gdal'
gdal_uri = 'https://github.com/OSGeo/gdal'
run_tests(experiment)
src_uri = 'git://rasdaman.org/rasdaman.git'
```

### benchbuild.projects.benchbuild.ruby module

```
class benchbuild.projects.benchbuild.ruby.Ruby(exp)
  Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup

  DOMAIN = 'compilation'
  NAME = 'ruby'
  SRC_FILE = 'ruby-2.2.2.tar.gz'
  VERSION = '2.2.2'
  build()
  configure()
  download()
  run_tests(experiment)
  src_dir = 'ruby-2.2.2'
  src_uri = 'http://cache.ruby-lang.org/pub/ruby/2.2.2/ruby-2.2.2.tar.gz'
```

### benchbuild.projects.benchbuild.sdcc module

```
class benchbuild.projects.benchbuild.sdcc.SDCC(exp)
  Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup

  DOMAIN = 'compilation'
  NAME = 'sdcc'
  SRC_FILE = 'sdcc'
  build()
  configure()
  download()
  run_tests(experiment)
  src_uri = 'svn://svn.code.sf.net/p/sdcc/code/trunk/sdcc'
```

### benchbuild.projects.benchbuild.sevenz module

```
class benchbuild.projects.benchbuild.sevenz.SevenZip(exp)
  Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup

  7Zip
  DOMAIN = 'compression'
```

```
NAME = '7z'
SRC_FILE = 'p7zip_9.38.1_src_all.tar.bz2'
VERSION = '9.38.1'
build()
configure()
download()
run_tests(experiment)
src_dir = 'p7zip_9.38.1'
src_uri = 'http://downloads.sourceforge.net/project/p7zip/p7zip/9.38.1/p7zip_9.38.1_src_all.tar.bz2'
```

### benchbuild.projects.benchbuild.sqlite3 module

```
class benchbuild.projects.benchbuild.sqlite3.SQLite3(exp)
    Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup
    DOMAIN = 'database'
    NAME = 'sqlite3'
    SRC_FILE = 'sqlite-amalgamation-3080900.zip'
    build()
    build_leveldb()
    configure()
    download()
    fetch_leveldb()
    run_tests(experiment)
    src_dir = 'sqlite-amalgamation-3080900'
    src_uri = 'http://www.sqlite.org/2015/sqlite-amalgamation-3080900.zip'
```

### benchbuild.projects.benchbuild.tcc module

```
class benchbuild.projects.benchbuild.tcc.TCC(exp)
    Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup
    DOMAIN = 'compilation'
    NAME = 'tcc'
    SRC_FILE = 'tcc-0.9.26.tar.bz2'
    VERSION = '0.9.26'
    build()
    configure()
    download()
    run_tests(experiment)
```

```
src_dir = 'tcc-0.9.26'
src_uri = 'http://download-mirror.savannah.gnu.org/releases/tinycc/tcc-0.9.26.tar.bz2'
```

### benchbuild.projects.benchbuild.x264 module

```
class benchbuild.projects.benchbuild.x264.X264(exp)
    Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup
    x264
    DOMAIN = 'multimedia'
    NAME = 'x264'
    SRC_FILE = 'x264.git'
    build()
    configure()
    download()
    inputfiles = {'tbbt-small.y4m': [], 'Sintel.2010.720p.raw': ['-input-res', '1280x720']}
    prepare()
    run_tests(experiment)
    src_uri = 'git://git.videolan.org/x264.git'
```

### benchbuild.projects.benchbuild.xz module

```
class benchbuild.projects.benchbuild.xz.XZ(exp)
    Bases: benchbuild.projects.benchbuild.group.BenchBuildGroup
    DOMAIN = 'compression'
    NAME = 'xz'
    SRC_FILE = 'xz-5.2.1.tar.gz'
    VERSION = '5.2.1'
    build()
    configure()
    download()
    prepare()
    run_tests(experiment)
    src_dir = 'xz-5.2.1'
    src_uri = 'http://tukaani.org/xz/xz-5.2.1.tar.gz'
    testfiles = ['text.html', 'chicken.jpg', 'control', 'input.source', 'liberty.jpg']
```

## benchbuild.projects.gentoo package

Import all gentoo based modules.

All manually entered modules can be placed in the following import section. Portage\_Gen based projects will be generated automatically as soon as we can find an index generated by portage info.

## Submodules

### benchbuild.projects.gentoo.autoportage module

```
class benchbuild.projects.gentoo.autoportage.AutoPortage (exp)
    Bases: benchbuild.projects.gentoo.gentoo.GentooGroup
    Generic portage experiment.
    build()
    run_tests(_)
```

### benchbuild.projects.gentoo.bzip2 module

bzip2 experiment within gentoo chroot.

```
class benchbuild.projects.gentoo.bzip2.BZip2 (exp)
    Bases: benchbuild.projects.gentoo.gentoo.GentooGroup
    app-arch/bzip2
    DOMAIN = 'app-arch'
    NAME = 'gentoo-bzip2'
    VERSION = '1.0.6'
    build()
    prepare()
    run_tests (experiment)
    test_archive = 'compression.tar.gz'
    test_url = 'http://lairosiel.de/dist/'
    testfiles = ['text.html', 'chicken.jpg', 'control', 'input.source', 'liberty.jpg']
```

### benchbuild.projects.gentoo.crafty module

crafty experiment within gentoo chroot.

```
class benchbuild.projects.gentoo.crafty.Crafty (exp)
    Bases: benchbuild.projects.gentoo.gentoo.GentooGroup
    games-board/crafty
    DOMAIN = 'games-board'
    NAME = 'gentoo-crafty'
```

```
build()
download()
run_tests(experiment)
```

### **benchbuild.projects.gentoo.eix module**

eix experiment within gentoo chroot

```
class benchbuild.projects.gentoo.eix.Eix(exp)
    Bases: benchbuild.projects.gentoo.gentoo.GentooGroup

    Represents the package eix from the portage tree.

    DOMAIN = 'app-portage'
    NAME = 'eix'

    build()
        Compiles and installes eix within gentoo chroot

    run_tests(experiment)
        Runs runtime tests for eix
```

### **benchbuild.projects.gentoo.gentoo module**

The Gentoo module for running tests on builds from the portage tree.

This will install a stage3 image of gentoo together with a recent snapshot of the portage tree. For building / executing arbitrary projects successfully it is necessary to keep the installed image as close to the host system as possible. In order to speed up your experience, you can replace the stage3 image that we pull from the distfiles mirror with a new image that contains all necessary dependencies for your experiments. Make sure you update the hash alongside the gentoo image in benchbuild's source directory.

```
class benchbuild.projects.gentoo.gentoo.GentooGroup(exp)
    Bases: benchbuild.project.Project

    Gentoo ProjectGroup is the base class for every portage build.

    CONTAINER = <benchbuild.utils.container.Gentoo object>
    GROUP = 'gentoo'
    SRC_FILE = None

    build()
    configure()
    download()
    write_bashrc(path)
    write_layout(path)
    write_makeconfig(path)
    write_wgetrc(path)
```



### benchbuild.projects.gentoo.gzip module

gzip experiment within gentoo chroot.

```
class benchbuild.projects.gentoo.gzip.GZip(exp)
    Bases: benchbuild.projects.gentoo.gentoo.GentooGroup
    app-arch/gzip
    DOMAIN = 'app-arch'
    NAME = 'gentoo-gzip'
    build()
    prepare()
    run_tests(experiment)
    test_archive = 'compression.tar.gz'
    test_url = 'http://lairosiel.de/dist/'
    testfiles = ['text.html', 'chicken.jpg', 'control', 'input.source', 'liberty.jpg']
```

### benchbuild.projects.gentoo.info module

Get package infos, e.g., specific ebuids for given languages, from gentoo chroot.

```
class benchbuild.projects.gentoo.info.Info(exp)
    Bases: benchbuild.projects.gentoo.autoportage.AutoPortage
    Info experiment to retrieve package information from portage.
    DOMAIN = 'debug'
    NAME = 'gentoo-info'
    build()
```

```
benchbuild.projects.gentoo.info.get_string_for_language(language_name)
    Maps language names to the corresponding string for qgrep.
```

### benchbuild.projects.gentoo.lammps module

LAMMPS (sci-physics/lammps) project within gentoo chroot.

```
class benchbuild.projects.gentoo.lammps.Lammps(exp)
    Bases: benchbuild.projects.gentoo.gentoo.GentooGroup
    sci-physics/lammps
    DOMAIN = 'sci-physics'
    NAME = 'gentoo-lammps'
    build()
    prepare()
    run_tests(experiment)
    test_archive = 'lammps.tar.gz'
```

```
test_url = 'http://lairosiel.de/dist/'
```

## benchbuild.projects.gentoo.portage\_gen module

Generic experiment to test portage packages within gentoo chroot.

```
benchbuild.projects.gentoo.portage_gen.PortageFactory(name, NAME, DOMAIN,
                                                       BaseClass=<class 'bench-
                                                       build.projects.gentoo.autoportage.AutoPortage'>)
```

Create a new dynamic portage project.

Auto-Generated projects can only be used for compile-time experiments, because there simply is no run-time test defined for it. Therefore, we implement the run symbol as a noop (with minor logging).

This way we avoid the default implementation for run() that all projects inherit.

### Parameters

- **name** – Name of the dynamic class.
- **NAME** – NAME property of the dynamic class.
- **DOMAIN** – DOMAIN property of the dynamic class.
- **BaseClass** – Base class to use for the dynamic class.

**Returns** A new class with NAME,DOMAIN properties set, unable to perform run-time tests.

## Examples

```
>>> from benchbuild.projects.gentoo.portage_gen import PortageFactory
>>> from benchbuild.experiments.empty import Empty
>>> c = PortageFactory("test", "NAME", "DOMAIN")
>>> c
<class '__main__.test'>
>>> i = c(Empty())
>>> i.NAME
'NAME'
>>> i.DOMAIN
'DOMAIN'
```

## benchbuild.projects.gentoo.postgresql module

postgresql experiment within gentoo chroot.

```
class benchbuild.projects.gentoo.postgresql.Postgresql(exp)
    Bases: benchbuild.projects.gentoo.gentoo.GentooGroup
```

dev-db/postgresql

**DOMAIN** = 'dev-db/postgresql'

**NAME** = 'gentoo-postgresql'

**build**()

**outside**(chroot\_path)

Return the path with the outside prefix.

**Parameters** `chroot_path` – the path inside the chroot.

**Returns** Absolute path outside this chroot.

`run_tests` (*experiment*)

### **benchbuild.projects.gentoo.sevenz module**

p7zip experiment within gentoo chroot.

```
class benchbuild.projects.gentoo.sevenz.SevenZip(exp)
    Bases: benchbuild.projects.gentoo.gentoo.GentooGroup
    app-arch/p7zip
    DOMAIN = 'app-arch'
    NAME = 'gentoo-p7zip'
    build()
    run_tests(experiment)
```

### **benchbuild.projects.gentoo.x264 module**

media-video/x264-encoder within gentoo chroot.

```
class benchbuild.projects.gentoo.x264.X264(exp)
    Bases: benchbuild.projects.gentoo.gentoo.GentooGroup
    media-video/x264-encoder
    DOMAIN = 'media-libs'
    NAME = 'gentoo-x264'
    build()
    inputfiles = {'tbbt-small.y4m': [], 'Sintel.2010.720p.raw': ['-input-res', '1280x720']}
    prepare()
    run_tests(experiment)
    test_url = 'http://lairosiel.de/dist/'
```

### **benchbuild.projects.gentoo.xz module**

xz experiment within gentoo chroot.

```
class benchbuild.projects.gentoo.xz.XZ(exp)
    Bases: benchbuild.projects.gentoo.gentoo.GentooGroup
    app-arch/xz
    DOMAIN = 'app-arch'
    NAME = 'gentoo-xz'
    build()
    prepare()
```

```
run_tests(experiment)

test_archive = 'compression.tar.gz'

test_url = 'http://lairosiel.de/dist/'

testfiles = ['text.html', 'chicken.jpg', 'control', 'input.source', 'liberty.jpg']
```

## benchbuild.projects.lnt package

### Submodules

#### benchbuild.projects.lnt.lnt module

LNT based measurements.

```
class benchbuild.projects.lnt.lnt.LNTGroup(exp)
    Bases: benchbuild.project.Project
    LNT ProjectGroup for running the lnt test suite.

    DOMAIN = 'lnt'
    GROUP = 'lnt'
    VERSION = '9.0.1.13'

    build()

    configure()

    download()

    src_dir = 'lnt'

    src_uri = 'http://llvm.org/git/lnt'

    test_suite_dir = 'test-suite'

    test_suite_uri = 'http://llvm.org/git/test-suite'

class benchbuild.projects.lnt.lnt.MultiSourceApplications(exp)
    Bases: benchbuild.projects.lnt.lnt.LNTGroup
    NAME = 'MultiSourceApplications'

    run_tests(experiment)

class benchbuild.projects.lnt.lnt.MultiSourceBenchmarks(exp)
    Bases: benchbuild.projects.lnt.lnt.LNTGroup
    NAME = 'MultiSourceBenchmarks'

    run_tests(experiment)

class benchbuild.projects.lnt.lnt.Povray(exp)
    Bases: benchbuild.projects.lnt.lnt.LNTGroup
    NAME = 'Povray'

    download()

    povray_src_dir = 'Povray'

    povray_url = 'https://github.com/POV-Ray/povray'
```

```
    run_tests (experiment)
class benchbuild.projects.lnt.lnt.SPEC2006 (exp)
    Bases: benchbuild.projects.lnt.lnt.LNTGroup
    NAME = 'SPEC2006'
    download ()
    run_tests (experiment)
class benchbuild.projects.lnt.lnt.SingleSourceBenchmarks (exp)
    Bases: benchbuild.projects.lnt.lnt.LNTGroup
    NAME = 'SingleSourceBenchmarks'
    run_tests (experiment)
```

## benchbuild.projects.polybench package

### Submodules

#### benchbuild.projects.polybench.polybench module

```
class benchbuild.projects.polybench.polybench.Adi (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup
    NAME = 'adi'
class benchbuild.projects.polybench.polybench.Atax (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup
    NAME = 'atax'
class benchbuild.projects.polybench.polybench.BicG (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup
    NAME = 'bicg'
class benchbuild.projects.polybench.polybench.Cholesky (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup
    NAME = 'cholesky'
class benchbuild.projects.polybench.polybench.Correlation (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup
    NAME = 'correlation'
class benchbuild.projects.polybench.polybench.Covariance (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup
    NAME = 'covariance'
class benchbuild.projects.polybench.polybench.Deriche (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup
    NAME = 'deriche'
class benchbuild.projects.polybench.polybench.Doitgen (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup
    NAME = 'doitgen'
```

```
class benchbuild.projects.polybench.polybench.Durbin (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup
    NAME = 'durbin'

class benchbuild.projects.polybench.polybench.FDTD2D (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup
    NAME = 'fddt-2d'

class benchbuild.projects.polybench.polybench.FloydWarshall (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup
    NAME = 'floyd-warshall'

class benchbuild.projects.polybench.polybench.Gemm (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup
    NAME = 'gemm'

class benchbuild.projects.polybench.polybench.Gemver (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup
    NAME = 'gemver'

class benchbuild.projects.polybench.polybench.Gesummv (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup
    NAME = 'gesummv'

class benchbuild.projects.polybench.polybench.Gramschmidt (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup
    NAME = 'gramschmidt'

class benchbuild.projects.polybench.polybench.Heat3D (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup
    NAME = 'heat-3d'

class benchbuild.projects.polybench.polybench.Jacobi1D (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup
    NAME = 'jacobi-1d'

class benchbuild.projects.polybench.polybench.Jacobi2Dimper (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup
    NAME = 'jacobi-2d'

class benchbuild.projects.polybench.polybench.Lu (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup
    NAME = 'lu'

class benchbuild.projects.polybench.polybench.LuDCMP (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup
    NAME = 'ludcmp'

class benchbuild.projects.polybench.polybench.Mvt (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup
    NAME = 'mvt'

class benchbuild.projects.polybench.polybench.Nussinov (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup
```

```
NAME = 'nussinov'

class benchbuild.projects.polybench.polybench.PolyBenchGroup (exp)
    Bases: benchbuild.project.Project

    DOMAIN = 'polybench'

    GROUP = 'polybench'

    SRC_FILE = 'polybench-c-4.2.tar.gz'

    VERSION = '4.2'

    build()

    configure()

    download()

    path_dict = {'correlation': 'datamining', 'covariance': 'datamining', '2mm': 'linear-algebra/kernels', '3mm': 'linear-a

    src_dir = 'polybench-c-4.2'

    src_uri = 'http://downloads.sourceforge.net/project/polybench/polybench-c-4.2.tar.gz'

class benchbuild.projects.polybench.polybench.Seidel2D (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup

    NAME = 'seidel-2d'

class benchbuild.projects.polybench.polybench.Symm (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup

    NAME = 'symm'

class benchbuild.projects.polybench.polybench.Syr2k (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup

    NAME = 'syr2k'

class benchbuild.projects.polybench.polybench.Syrk (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup

    NAME = 'syrk'

class benchbuild.projects.polybench.polybench.ThreeMM (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup

    NAME = '3mm'

class benchbuild.projects.polybench.polybench.Trisolv (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup

    NAME = 'trisolv'

class benchbuild.projects.polybench.polybench.Trmm (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup

    NAME = 'trmm'

class benchbuild.projects.polybench.polybench.TwoMM (exp)
    Bases: benchbuild.projects.polybench.polybench.PolyBenchGroup

    NAME = '2mm'
```

## benchbuild.reports package

Register reports for an experiment

```
class benchbuild.reports.Report (exp_ids, out_path)
    Bases: object

    SUPPORTED_EXPERIMENTS = None

class benchbuild.reports.ReportRegistry (name, bases, dict)
    Bases: type

    reports = {'raw': [<class 'benchbuild.reports.raw.RawReport'>]}
```

```
benchbuild.reports.discover()
    Import all experiments listed in *_PLUGINS_REPORTS.
```

**Tests:**

```
>>> from benchbuild.settings import CFG
>>> from benchbuild.reports import discover
>>> import logging as lg
>>> import sys
>>> l = lg.getLogger('benchbuild')
>>> lg.getLogger('benchbuild').setLevel(lg.DEBUG)
>>> lg.getLogger('benchbuild').handlers = [lg.StreamHandler(stream=sys.
↳ stdout)]
>>> CFG["plugins"]["reports"] = ["benchbuild.non.existing", "benchbuild.
↳ reports.raw"]
>>> discover()
Could not find 'benchbuild.non.existing'
Found report: benchbuild.reports.raw
```

## Submodules

### benchbuild.reports.raw module

```
class benchbuild.reports.raw.RawReport (exp_ids, outfile)
    Bases: benchbuild.reports.Report

    SUPPORTED_EXPERIMENTS = ['raw']

    generate()

    get_exp_ids()

    report()

    session = <sqlalchemy.orm.session.Session object>
```

## benchbuild.utils package

## Submodules

### benchbuild.utils.actions module

This defines classes that can be used to implement a series of Actions.



```
class benchbuild.utils.actions.Any(actions)
    Bases: benchbuild.utils.actions.Step
    DESCRIPTION = 'Just run all actions, no questions asked.'
    NAME = 'ANY'

class benchbuild.utils.actions.Build(project)
    Bases: benchbuild.utils.actions.Step
    DESCRIPTION = 'Build the project'
    NAME = 'BUILD'

class benchbuild.utils.actions.Clean(project_or_experiment, action_fn=None,
                                     check_empty=False)
    Bases: benchbuild.utils.actions.Step
    DESCRIPTION = 'Cleans the build directory'
    NAME = 'CLEAN'

class benchbuild.utils.actions.CleanExtra(project_or_experiment, action_fn=None)
    Bases: benchbuild.utils.actions.Step
    DESCRIPTION = 'Cleans the extra directories.'
    NAME = 'CLEAN EXTRA'

class benchbuild.utils.actions.Configure(project)
    Bases: benchbuild.utils.actions.Step
    DESCRIPTION = 'Configure project source files'
    NAME = 'CONFIGURE'

class benchbuild.utils.actions.Download(project)
    Bases: benchbuild.utils.actions.Step
    DESCRIPTION = 'Download project source files'
    NAME = 'DOWNLOAD'

class benchbuild.utils.actions.Echo(message)
    Bases: benchbuild.utils.actions.Step
    DESCRIPTION = 'Print a message.'
    NAME = 'ECHO'

class benchbuild.utils.actions.Experiment(experiment, actions)
    Bases: benchbuild.utils.actions.Any
    DESCRIPTION = 'Run a experiment, wrapped in a db transaction'
    NAME = 'EXPERIMENT'
    begin_transaction()
    end_transaction(experiment, session)

class benchbuild.utils.actions.MakeBuildDir(project_or_experiment, action_fn=None)
    Bases: benchbuild.utils.actions.Step
    DESCRIPTION = 'Create the build directory'
    NAME = 'MKDIR'
```

```
class benchbuild.utils.actions.Prepare (project)
    Bases: benchbuild.utils.actions.Step

    DESCRIPTION = 'Prepare project build folder'

    NAME = 'PREPARE'

class benchbuild.utils.actions.RequireAll (actions)
    Bases: benchbuild.utils.actions.Step

class benchbuild.utils.actions.Run (project)
    Bases: benchbuild.utils.actions.Step

    DESCRIPTION = 'Execute the run action'

    NAME = 'RUN'

class benchbuild.utils.actions.Step (project_or_experiment, action_fn=None)
    Bases: object

    DESCRIPTION = None

    NAME = None

    onerror ()

class benchbuild.utils.actions.StepClass
    Bases: abc.ABCMeta

class benchbuild.utils.actions.StepResult
    Bases: enum.Enum

    An enumeration.

    ERROR = 2

    OK = (1,)

benchbuild.utils.actions.log_before_after (name, desc)

benchbuild.utils.actions.to_step_result (f)
```

### **benchbuild.utils.bootstrap module**

Helper functions for bootstrapping external dependencies.

```
benchbuild.utils.bootstrap.check_uchroot_config()

benchbuild.utils.bootstrap.find_package (binary)

benchbuild.utils.bootstrap.install_package (pkg_name)

benchbuild.utils.bootstrap.install_uchroot ()

benchbuild.utils.bootstrap.linux_distribution_major ()

benchbuild.utils.bootstrap.provide_package (pkg_name)

benchbuild.utils.bootstrap.provide_packages (pkg_names)
```

## benchbuild.utils.compiler module

Helper functions for dealing with compiler replacement.

This provides a few key functions to deal with varying/measuring the compilers used inside the benchbuild study. From a high-level view, there are 2 interesting functions:

- `lt_clang(cflags, ldflags, func)`
- `lt_clang_cxx(cflags, ldflags, func)`

These generate a wrapped clang/clang++ in the current working directory and hide the given cflags/ldflags from the calling build system. Both will give you a working plumbum command and call a python script that redirects to the real clang/clang++ given the additional cflags&ldflags.

**The wrapper-script generated for both functions can be found inside:**

- `wrap_cc()`

**The remaining methods:**

- `llvm()`
- `llvm_libs()`
- `clang()`
- `clang_cxx()`

Are just convenience methods that can be used when interacting with the configured llvm/clang source directories.

`benchbuild.utils.compiler.clang()`

Get a usable clang plumbum command.

This searches for a usable clang in the llvm binary path (See `llvm()`) and returns a plumbum command to call it.

**Returns** plumbum Command that executes clang++

`benchbuild.utils.compiler.clang_cxx()`

Get a usable clang++ plumbum command.

This searches for a usable clang++ in the llvm binary path (See `llvm()`) and returns a plumbum command to call it.

**Returns** plumbum Command that executes clang++

`benchbuild.utils.compiler.llvm()`

Get the path where all llvm binaries can be found.

**Environment variable:** `BB_LLVM_DIR`

**Returns** LLVM binary path.

`benchbuild.utils.compiler.llvm_libs()`

Get the path where all llvm libraries can be found.

**Environment variable:** `BB_LLVM_DIR`

**Returns** LLVM library path.

`benchbuild.utils.compiler.lt_clang(cflags, ldflags, func=None)`

Return a clang that hides CFLAGS and LDFLAGS.

This will generate a wrapper script in the current directory and return a complete plumbum command to it.

**Parameters**

- **cflags** – The CFLAGS we want to hide.
- **ldflags** – The LDFLAGS we want to hide.
- **func** (*optional*) – A function that will be pickled alongside the compiler. It will be called before the actual compilation took place. This way you can intercept the compilation process with arbitrary python code.

**Returns (benchbuild.utils.cmd):** Path to the new clang command.

```
benchbuild.utils.compiler.lt_clang_cxx(cflags, ldflags, func=None)
```

Return a clang++ that hides CFLAGS and LDFLAGS.

This will generate a wrapper script in the current directory and return a complete plumbum command to it.

**Parameters**

- **cflags** – The CFLAGS we want to hide.
- **ldflags** – The LDFLAGS we want to hide.
- **func** (*optional*) – A function that will be pickled alongside the compiler. It will be called before the actual compilation took place. This way you can intercept the compilation process with arbitrary python code.

**Returns (benchbuild.utils.cmd):** Path to the new clang command.

```
benchbuild.utils.compiler.wrap_cc_in_uchroot(cflags, ldflags, func=None,
                                             cc_name='clang')
```

Generate a clang wrapper that may be called from within a uchroot.

This basically does the same as lt\_clang/lt\_clang\_cxx. However, we do not create a valid plumbum command. The generated script will only work inside a uchroot environment that has its root at the current working directory, when calling this function.

**Parameters**

- **cflags** – The CFLAGS we want to hide
- **ldflags** – The LDFLAGS we want to hide
- **func** (*optional*) – A function that will be pickled alongside the compiler. It will be called before the actual compilation took place. This way you can intercept the compilation process with arbitrary python code.
- **uchroot\_path** – Prefix path of the compiler inside the uchroot.
- **cc\_name** – Name of the generated script.

```
benchbuild.utils.compiler.wrap_cxx_in_uchroot(cflags, ldflags, func=None)
```

Delegate to wrap\_cc\_in\_uchroot).

**benchbuild.utils.container module**

Container utilities.

```
class benchbuild.utils.container.Container
```

Bases: object

**filename**

**local**

Finds the current location of a container. Also unpacks the project if necessary.

**Returns** The path, where the container lies in the end.

**Return type** target

**remote**

**class** `benchbuild.utils.container.Gentoo`

Bases: `benchbuild.utils.container.Container`

**latest\_src\_uri** (\*args, \*\*kwargs)

**name** = 'gentoo'

**remote**

Get a remote URL of the requested container.

**class** `benchbuild.utils.container.Ubuntu`

Bases: `benchbuild.utils.container.Container`

**name** = 'ubuntu'

**remote**

Get a remote URL of the requested container.

`benchbuild.utils.container.cached` (func)

`benchbuild.utils.container.is_valid_container` (container, path)

Checks if a container exists and is unpacked.

**Parameters** **path** – The location where the container is expected.

**Returns** True if the container is valid, False if the container needs to be unpacked or if the path does not exist yet.

`benchbuild.utils.container.unpack_container` (container, path)

Method that checks if a directory for the container exists, checks if erlent support is needed and then unpacks the container accordingly.

**Parameters** **path** – The location where the container is, that needs to be unpacked.

**benchbuild.utils.db module**

Database support module for the benchbuild study.

`benchbuild.utils.db.create_run` (cmd, prj, exp, grp)

Create a new 'run' in the database.

This creates a new transaction in the database and creates a new run in this transaction. Afterwards we return both the transaction as well as the run itself. The user is responsible for committing it when the time comes.

**Parameters**

- **cmd** – The command that has been executed.
- **prj** – The project this run belongs to.
- **exp** – The experiment this run belongs to.
- **grp** – The run\_group (uuid) we belong to.

**Returns** The inserted tuple representing the run and the session opened with the new run. Don't forget to commit it at some point.

`benchbuild.utils.db.create_run_group(prj)`

Create a new 'run\_group' in the database.

This creates a new transaction in the database and creates a new run\_group within this transaction. Afterwards we return both the transaction as well as the run\_group itself. The user is responsible for committing it when the time comes.

**Parameters** – The project for which we open the run\_group. (*prj*) –

**Returns** A tuple (group, session) containing both the newly created run\_group and the transaction object.

`benchbuild.utils.db.persist_compilestats(run, session, stats)`

Persist the run results in the database.

**Parameters**

- **run** – The run we attach the compilestats to.
- **session** – The db transaction we belong to.
- **stats** – The stats we want to store in the database.

`benchbuild.utils.db.persist_config(run, session, cfg)`

Persist the configuration in as key-value pairs.

**Parameters**

- **run** – The run we attach the config to.
- **session** – The db transaction we belong to.
- **cfg** – The configuration we want to persist.

`benchbuild.utils.db.persist_experiment(experiment)`

Persist this experiment in the benchbuild database.

**Parameters** **experiment** – The experiment we want to persist.

`benchbuild.utils.db.persist_likwid(run, session, measurements)`

Persist all likwid results.

**Parameters**

- **run** – The run we attach our measurements to.
- **session** – The db transaction we belong to.
- **measurements** – The likwid measurements we want to store.

`benchbuild.utils.db.persist_perf(run, session, svg_path)`

Persist the flamegraph in the database.

The flamegraph exists as a SVG image on disk until we persist it in the database.

**Parameters**

- **run** – The run we attach these perf measurements to.
- **session** – The db transaction we belong to.
- **svg\_path** – The path to the SVG file we want to store.

`benchbuild.utils.db.persist_project(project)`

Persist this project in the benchbuild database.

**Parameters** **project** – The project we want to persist.

`benchbuild.utils.db.persist_time(run, session, timings)`

Persist the run results in the database.

#### Parameters

- **run** – The run we attach this timing results to.
- **session** – The db transaction we belong to.
- **timings** – The timing measurements we want to store.

### benchbuild.utils.downloader module

Downloading helper functions for benchbuild.

The helpers defined in this module provide access to some common Downloading methods for the source code of benchbuild projects. All downloads will be cached in BB\_TMP\_DIR and locked down with a hash that is generated after the first download. If the hash matches the file/folder found in BB\_TMP\_DIR, nothing will be downloaded at all.

**Supported methods:** Copy, CopyNoFail, Wget, Git, Svn, Rsync

`benchbuild.utils.downloader.Copy(From, To)`

Small copy wrapper.

#### Parameters

- **From** (*str*) – Path to the SOURCE.
- **To** (*str*) – Path to the TARGET.

`benchbuild.utils.downloader.CopyNoFail(src, root=None)`

Just copy fName into the current working directory, if it exists.

No action is executed, if fName does not exist. No Hash is checked.

#### Parameters

- **src** – The filename we want to copy to ‘.’.
- **root** – The optional source dir we should pull fName from. Defaults to benchbuild.settings.CFG[“tmpdir”].

**Returns** True, if we copied something.

`benchbuild.utils.downloader.Git(src_url, tgt_name, tgt_root=None)`

Get a shallow clone of the given repo

#### Parameters

- **src\_url** (*str*) – Git URL of the SOURCE repo.
- **tgt\_name** (*str*) – Name of the repo folder on disk.
- **tgt\_root** (*str*) – TARGET folder for the git repo. Defaults to CFG[“tmpdir”]

`benchbuild.utils.downloader.Rsync(url, tgt_name, tgt_root=None)`

RSync a folder.

#### Parameters

- **url** (*str*) – The url of the SOURCE location.
- **fname** (*str*) – The name of the TARGET.
- **to** (*str*) – Path of the target location. Defaults to CFG[“tmpdir”].

`benchbuild.utils.downloader.Svn(url, fname, to=None)`

Checkout the SVN repo.

### Parameters

- **url** (*str*) – The SVN SOURCE repo.
- **fname** (*str*) – The name of the repo on disk.
- **to** (*str*) – The name of the TARGET folder on disk. Defaults to `CFG["tmpdir"]`

`benchbuild.utils.downloader.Wget(src_url, tgt_name, tgt_root=None)`

Download url, if required.

### Parameters

- **src\_url** (*str*) – Our SOURCE url.
- **tgt\_name** (*str*) – The filename we want to have on disk.
- **tgt\_root** (*str*) – The TARGET directory for the download. Defaults to `CFG["tmpdir"]`.

`benchbuild.utils.downloader.get_hash_of_dirs(directory)`

Recursively hash the contents of the given directory.

**Parameters** **directory** (*str*) – The root directory we want to hash.

**Returns** A hash of all the contents in the directory.

`benchbuild.utils.downloader.source_required(src_file, src_root)`

Check, if a download is required.

### Parameters

- **src\_file** – The filename to check for.
- **src\_root** – The path we find the file in.

**Returns** True, if we need to download something, False otherwise.

`benchbuild.utils.downloader.update_hash(src, root)`

Update the hash for the given file.

### Parameters

- **src** – The file name.
- **root** – The path of the given file.

## benchbuild.utils.log module

`benchbuild.utils.log.configure()`

Load logging configuration from our own defaults.

`benchbuild.utils.log.set_defaults()`

Configure the loggers default settings.

## benchbuild.utils.path module

Path utilities for benchbuild.



`benchbuild.utils.path.determine_path()`

Borrowed from wxglade.py

`benchbuild.utils.path.list_to_path(pathlist)`

Convert a list of path elements to a path string.

`benchbuild.utils.path.mkdir_uchroot(dirpath, root='.')`

Create a file inside a uchroot env.

You will want to use this when you need to create a file with appropriate rights inside a uchroot container with subuid/subgid handling enabled.

#### Parameters

- **dirpath** – The dirpath that should be created. Absolute inside the uchroot container.
- **root** – The root PATH of the container filesystem as seen outside of the container.

`benchbuild.utils.path.mkfile_uchroot(filepath, root='.')`

Create a file inside a uchroot env.

You will want to use this when you need to create a file with appropriate rights inside a uchroot container with subuid/subgid handling enabled.

#### Parameters

- **filepath** – The filepath that should be created. Absolute inside the uchroot container.
- **root** – The root PATH of the container filesystem as seen outside of the container.

`benchbuild.utils.path.path_to_list(pathstr)`

Conver a path string to a list of path elements.

`benchbuild.utils.path.template_str(template)`

Read a template file from the resources and return it as str.

## benchbuild.utils.run module

Experiment helpers.

**exception** `benchbuild.utils.run.GuardedRunException(what, db_run, session)`

Bases: `Exception`

BB Run exception.

Contains an exception that occurred during execution of a benchbuild experiment.

`benchbuild.utils.run.begin(command, pname, ename, group)`

Begin a run in the database log.

#### Parameters

- **command** – The command that will be executed.
- **pname** – The project name we belong to.
- **ename** – The experiment name we belong to.
- **group** – The run group we belong to.

**Returns** (run, session), where run is the generated run instance and session the associated transaction for later use.

`benchbuild.utils.run.begin_run_group(project)`

Begin a `run_group` in the database.

A `run_group` groups a set of runs for a given project. This models a series of runs that form a complete binary runtime test.

**Parameters** `project` – The project we begin a new `run_group` for.

**Returns** (`group`, `session`) where `group` is the created group in the database and `session` is the database session this group lives in.

`benchbuild.utils.run.end(db_run, session, stdout, stderr)`

End a run in the database log (Successfully).

This will persist the log information in the database and commit the transaction.

**Parameters**

- **db\_run** – The `run` schema object we belong to
- **session** – The db transaction we belong to.
- **stdout** – The stdout we captured of the run.
- **stderr** – The stderr we capture of the run.

`benchbuild.utils.run.end_run_group(group, session)`

End the `run_group` successfully.

**Parameters**

- **group** – The `run_group` we want to complete.
- **session** – The database transaction we will finish.

`benchbuild.utils.run.fail(db_run, session, retcode, stdout, stderr)`

End a run in the database log (Unsuccessfully).

This will persist the log information in the database and commit the transaction.

**Parameters**

- **db\_run** – The `run` schema object we belong to
- **session** – The db transaction we belong to.
- **retcode** – The return code we captured of the run.
- **stdout** – The stdout we captured of the run.
- **stderr** – The stderr we capture of the run.

`benchbuild.utils.run.fail_run_group(group, session)`

End the `run_group` unsuccessfully.

**Parameters**

- **group** – The `run_group` we want to complete.
- **session** – The database transaction we will finish.

`benchbuild.utils.run.fetch_time_output(marker, format_s, ins)`

Fetch the output `/usr/bin/time` from a.

**Parameters**

- **marker** – The marker that limits the time output
- **format\_s** – The format string used to parse the timings

- **ins** – A list of lines we look for the output.

**Returns** A list of timing tuples

`benchbuild.utils.run.guarded_exec(cmd, project, experiment, **kwargs)`

Guard the execution of the given command.

**Parameters**

- **cmd** – the command we guard.
- **pname** – the database we run under.
- **ename** – the database session this run belongs to.
- **run\_group** – the run group this execution will belong to.

**Raises** `RunException` – If the `cmd` encounters an error we wrap the exception in a `RunException` and re-raise. This ends the run unsuccessfully.

`benchbuild.utils.run.handle_stdin(cmd, kwargs)`

Handle stdin for wrapped runtime executors.

This little helper checks the `kwargs` for a `has_stdin` key containing a boolean value. If necessary it will pipe in the stdin of this process into the plumbum command.

**Parameters**

- **cmd** (`benchbuild.utils.cmd`) – Command to wrap a stdin handler around.
- **kwargs** – Dictionary containing the `kwargs`. We check for they key `has_stdin`

**Returns** A new plumbum command that deals with stdin redirection, if needed.

`benchbuild.utils.run.in_builddir(sub='.')`

Decorate a project phase with a local working directory change.

**Parameters** **sub** – An optional subdirectory to change into.

`benchbuild.utils.run.run(command, retcode=0)`

Execute a plumbum command, depending on the user's settings.

**Parameters** **command** – The plumbumb command to execute.

`benchbuild.utils.run.store_config(func)`

Decorator for storing the configuration in the project's builddir.

`benchbuild.utils.run.uchroot(*args, **kwargs)`

Return a customizable uchroot command.

**Parameters** **args** – List of additional arguments for uchroot (typical: mounts)

**Returns** `chroot_cmd`

`benchbuild.utils.run.uchroot_env(mounts)`

Compute the environment of the change root for the user.

**Parameters** **mounts** – The mountpoints of the current user.

**Returns** `paths ld_libs`

`benchbuild.utils.run.uchroot_mounts(prefix, mounts)`

Compute the mountpoints of the current user.

**Parameters**

- **prefix** – Define where to job was running if it ran on a cluster.

- **mounts** – All mounts the user currently uses in his file system.

**Returns** mntpoints

`benchbuild.utils.run.uchroot_no_args()`

Return the uchroot command without any customizations.

`benchbuild.utils.run.uchroot_no_llvm(*args, **kwargs)`

Return a customizable uchroot command.

The command will be executed inside a uchroot environment.

**Parameters** **args** – List of additional arguments for uchroot (typical: mounts)

**Returns** chroot\_cmd

`benchbuild.utils.run.uchroot_with_mounts(*args, **kwargs)`

Return a uchroot command with all mounts enabled.

`benchbuild.utils.run.unionfs(base_dir='./base', image_dir='./image', image_prefix=None, mountpoint='./union')`

Decorator for the UnionFS feature.

This configures a unionfs for projects. The given base\_dir and/or image\_dir are layered as follows:

image\_dir=RW;base\_dir=RO

All writes go to the image\_dir, while base\_dir delivers the (read-only) versions of the rest of the filesystem.

The unified version will be provided in the project's build\_dir. Unmounting is done as soon as the function completes.

**Parameters**

- **base\_dir** – The unpacked container of a project delivered by a method out of the container utils.
- **image\_dir** – Virtual image of the actual file system represented by the build\_dir of a project.
- **image\_prefix** – Useful prefix if the projects run on a cluster, to identify where the job came from and where it runs.
- **mountpoint** – Location where the filesystems merge, currently per default as './union'.

`benchbuild.utils.run.unionfs_set_up(ro_base, rw_image, mountpoint)`

Setup a unionfs via unionfs-fuse.

**Parameters**

- **ro\_base** – base\_directory of the project
- **rw\_image** – virtual image of actual file system
- **mountpoint** – location where ro\_base and rw\_image merge

`benchbuild.utils.run.unionfs_tear_down(mountpoint, tries=3)`

Tear down a unionfs mountpoint.

`benchbuild.utils.run.with_env_recursive(cmd, **envvars)`

Recursively updates the environment of cmd and all its subcommands.

**Parameters**

- – **A plumbum command-like object** (*cmd*) –
- – **The environment variables to update** (*\*\*envvars*) –

**Returns** The updated command.

### benchbuild.utils.schema module

Database schema for benchbuild.

`benchbuild.utils.schema.CONNECTION_MANAGER = <benchbuild.utils.schema.SessionManager object>`  
Import this session manager to create new database sessions as needed.

**class** `benchbuild.utils.schema.CompileStat (**kwargs)`

Bases: `sqlalchemy.ext.declarative.api.Base`

Store compilestats as given by LLVM's '-stats' commoand.

**component**

**id**

**name**

**run\_id**

**value**

**class** `benchbuild.utils.schema.Config (**kwargs)`

Bases: `sqlalchemy.ext.declarative.api.Base`

Store customized information about a run.

You can store arbitrary configuration information about a run here. Use it for extended filtering against the run table.

**name**

**run\_id**

**value**

**class** `benchbuild.utils.schema.Event (**kwargs)`

Bases: `sqlalchemy.ext.declarative.api.Base`

Store PAPI profiling based events.

**duration**

**id**

**name**

**run\_id**

**start**

**tid**

**type**

**class** `benchbuild.utils.schema.Experiment (**kwargs)`

Bases: `sqlalchemy.ext.declarative.api.Base`

Store metadata about experiments.

**begin**

**description**

**end**

```
id
name
class benchbuild.utils.schema.GlobalConfig(**kwargs)
    Bases: sqlalchemy.ext.declarative.api.Base
    Store customized information about a run.
    You can store arbitrary configuration information about a run here. Use it for extended filtering against the run
    table.
    experiment_group
    name
    value
class benchbuild.utils.schema.Likwid(**kwargs)
    Bases: sqlalchemy.ext.declarative.api.Base
    Store measurement results of likwid based experiments.
    core
    metric
    region
    run_id
    value
class benchbuild.utils.schema.Metadata(**kwargs)
    Bases: sqlalchemy.ext.declarative.api.Base
    Store metadata information for every run.
    If you happen to have some free-form data that belongs to the database, this is the place for it.
    name
    run_id
    value
class benchbuild.utils.schema.Metric(**kwargs)
    Bases: sqlalchemy.ext.declarative.api.Base
    Store default metrics, simple name value store.
    name
    run_id
    value
class benchbuild.utils.schema.PerfEvent(**kwargs)
    Bases: sqlalchemy.ext.declarative.api.Base
    Store PAPI profiling based events.
    duration
    id
    name
    run_id
```

```
start
tid
type
class benchbuild.utils.schema.Project (**kwargs)
    Bases: sqlalchemy.ext.declarative.api.Base
    Store project metadata.
    description
    domain
    group_name
    name
    src_url
    version
class benchbuild.utils.schema.RegressionTest (**kwargs)
    Bases: sqlalchemy.ext.declarative.api.Base
    Store regression tests for all projects.
    This relation is filled from the PolyJIT side, not from benchbuild-study. We collect all JIT SCoPs found by
    PolyJIT in this relation for regression-test generation.
    module
    name
    project_name
    run_id
class benchbuild.utils.schema.Run (**kwargs)
    Bases: sqlalchemy.ext.declarative.api.Base
    Store a run for each executed test binary.
    begin
    command
    end
    experiment_group
    experiment_name
    id
    project_name
    run_group
    status
class benchbuild.utils.schema.RunGroup (**kwargs)
    Bases: sqlalchemy.ext.declarative.api.Base
    Store information about a run group.
    begin
    end
```

```
    experiment
    id
    project
    status

class benchbuild.utils.schema.RunLog (**kwargs)
    Bases: sqlalchemy.ext.declarative.api.Base

    Store log information for every run.

    Properties like, start time, finish time, exit code, stderr, stdout are stored here.

    begin
    config
    end
    run_id
    status
    stderr
    stdout

class benchbuild.utils.schema.SessionManager
    Bases: object

    get ()
```

### **benchbuild.utils.slurm module**

SLURM support for the benchbuild study.

This module can be used to generate bash scripts that can be executed by the SLURM controller either as batch or interactive script.

`benchbuild.utils.slurm.dump_slurm_script (script_name, benchbuild, experiment, projects)`  
Dump a bash script that can be given to SLURM.

#### **Parameters**

- **script\_name** (*str*) – name of the bash script.
- **commands** (*list (benchbuild.utils.cmd)*) – List of plumbum commands to write to the bash script.
- **\*\*kwargs** – Dictionary with all environment variable bindings we should map in the bash script.

`benchbuild.utils.slurm.prepare_directories (dirs)`  
Make sure that the required directories exist.

#### **Parameters – the directories we want. (dirs) –**

`benchbuild.utils.slurm.prepare_slurm_script (experiment, projects)`  
Prepare a slurm script that executes the experiment for a given project.

#### **Parameters**

- **experiment** – The experiment we want to execute



- **projects** – All projects we generate an array job for.

### benchbuild.utils.user\_interface module

User interface helpers for benchbuild.

`benchbuild.utils.user_interface.ask(question, default_answer=False, fault_answer_str='no')`

`benchbuild.utils.user_interface.query_yes_no(question, default='yes')`

Ask a yes/no question via `raw_input()` and return their answer.

#### Parameters

- **question** (*str*) – Question hat is presented to the user.
- **default** (*str*) – The presumed answer, if the user just hits <Enter>. It must be “yes” (the default), “no” or None (meaning an answer is required of the user).

**Returns (boolean):** True, if ‘yes’, False otherwise.

### benchbuild.utils.versions module

Gather version information for BB.

`benchbuild.utils.versions.get_git_hash(from_url)`

Get the git commit hash of HEAD from `:from_url`.

**Parameters** **from\_url** – The file system url of our git repository.

**Returns** git commit hash of HEAD, or empty string.

`benchbuild.utils.versions.get_version_from_cache_dir(src_file)`

Creates a version for a project out of the hash.

The hash is taken from the directory of the source file.

**Parameters** **src\_file** – The source file of the project using this function.

**Returns** Either returns the first 8 digits of the hash as string, the entire hash as a string if the hash consists out of less than 7 digits or None if the path is incorrect.

### benchbuild.utils.wrapping module

`benchbuild.utils.wrapping.strip_path_prefix(ipath, prefix)`

Strip prefix from path.

#### Parameters

- **ipath** – input path
- **prefix** – the prefix to remove, if it is found in `:ipath`:

### Examples

```
>>> strip_path_prefix("/foo/bar", "/bar")
'/foo/bar'
>>> strip_path_prefix("/foo/bar", "/")
'foo/bar'
>>> strip_path_prefix("/foo/bar", "/foo")
'/bar'
>>> strip_path_prefix("/foo/bar", "None")
'/foo/bar'
```

`benchbuild.utils.wrapping.wrap(name, runner, sprefix=None, **template_vars)`

Wrap the binary :name: with the function :runner:.

This module generates a python tool that replaces :name: The function in runner only accepts the replaced binaries name as argument. We use the cloudpickle package to perform the serialization, make sure :runner: can be serialized with it and you're fine.

#### Parameters

- **name** – Binary we want to wrap
- **runner** – Function that should run instead of :name:

**Returns** A plumbum command, ready to launch.

`benchbuild.utils.wrapping.wrap_cc(filepath, cflags, ldflags, compiler, extension, compiler_ext_name=None, **template_vars)`

Substitute a compiler with a script that hides CFLAGS & LDFLAGS.

This will generate a wrapper script in the current directory and return a complete plumbum command to it.

#### Parameters

- **filepath** (*str*) – Path to the wrapper script.
- **cflags** (*list(str)*) – The CFLAGS we want to hide.
- **ldflags** (*list(str)*) – The LDFLAGS we want to hide.
- **compiler** (*benchbuild.utils.cmd*) – Real compiler command we should call in the script.
- **extension** – A function that will be pickled alongside the compiler. It will be called before the actual compilation took place. This way you can intercept the compilation process with arbitrary python code.
- **compiler\_ext\_name** – The name that we should give to the generated dill blob for :func:

**Returns** (*benchbuild.utils.cmd*): Command of the new compiler we can call.

`benchbuild.utils.wrapping.wrap_dynamic(self, name, runner, sprefix=None, **template_vars)`

Wrap the binary :name with the function :runner.

This module generates a python tool :name: that can replace a yet unspecified binary. It behaves similar to the :wrap: function. However, the first argument is the actual binary name.

#### Parameters

- **name** – name of the python module
- **runner** – Function that should run the real binary
- **base\_class** – The base\_class of our project.

- **base\_module** – The module of base\_class.

Returns: plumbum command, ready to launch.

`benchbuild.utils.wrapping.wrap_dynamic_in_uchroot` (*self, name, runner, sprefix=None*)

`benchbuild.utils.wrapping.wrap_in_uchroot` (*name, runner, sprefix=None*)

### 1.9.1.2 Submodules

#### 1.9.1.3 benchbuild.bootstrap module

**class** `benchbuild.bootstrap.BenchBuildBootstrap` (*executable*)

Bases: `plumbum.cli.application.Application`

Bootstrap benchbuild external dependencies, if possible.

**main** (*\*args*)

**store\_config**

Sets an attribute

#### 1.9.1.4 benchbuild.container module

**class** `benchbuild.container.BashStrategy`

Bases: `benchbuild.container.ContainerStrategy`

The user interface for setting up a bash inside the container.

**run** (*context*)

**class** `benchbuild.container.Container` (*exe*)

Bases: `plumbum.cli.application.Application`

Manage uchroot containers.

**VERSION** = '1.2.1'

**builddir** (*tmpdir*)

Set the current builddir of the container.

**input\_file** (*container*)

Find the input path of a uchroot container.

**main** (*\*args*)

**mounts** (*user\_mount*)

Save the current mount of the container into the settings.

**output\_file** (*container*)

Find and writes the output path of a chroot container.

**shell** (*custom\_shell*)

The command to run inside the container.

**verbosity**

Sets an attribute

**class** `benchbuild.container.ContainerBootstrap` (*executable*)

Bases: `plumbum.cli.application.Application`

Check for the needed files.

```
install_cmake_and_exit ()
```

Tell the user to install cmake and aborts the current process.

```
main (*args)
```

```
class benchbuild.container.ContainerCreate (executable)
```

Bases: `plumbum.cli.application.Application`

Create a new container with a predefined strategy.

We offer a variety of creation policies for a new container. By default a basic ‘spawn a bash’ policy is used. This just leaves you inside a bash that is started in the extracted container. After customization you can exit the bash and pack up the result.

```
main (*args)
```

```
strategy (strategy)
```

```
class benchbuild.container.ContainerList (executable)
```

Bases: `plumbum.cli.application.Application`

Prints a list of the known containers.

```
main (*args)
```

```
class benchbuild.container.ContainerRun (executable)
```

Bases: `plumbum.cli.application.Application`

Execute commands inside a prebuilt container.

```
main (*args)
```

```
class benchbuild.container.ContainerStrategy
```

Bases: `object`

Interfaces for the different containers chosen by the experiment.

```
run (context)
```

```
class benchbuild.container.MockObj (**kwargs)
```

Bases: `object`

```
class benchbuild.container.SetupPolyJITGentooStrategy
```

Bases: `benchbuild.container.ContainerStrategy`

Interface of using gentoo as a container for an experiment.

```
configure ()
```

Configure the gentoo container for a PolyJIT experiment.

```
run (context)
```

Setup a gentoo container suitable for PolyJIT.

```
write_bashrc (path)
```

Write inside a bash and update the shell if necessary.

```
write_layout (path)
```

Create a layout from the given path.

```
write_makeconfig (path)
```

Create the string to be written in the settings.

```
write_wgetrc (path)
```

Wget the project from a specified link.

`benchbuild.container.clean_directories` (*builddir*, *in\_dir=True*, *out\_dir=True*)

Remove the in and out of the container if confirmed by the user.

`benchbuild.container.find_hash` (*container\_db*, *key*)

Find the first container in the database with the given key.

`benchbuild.container.main` (*\*args*)

`benchbuild.container.pack_container` (*in\_container*, *out\_file*)

`benchbuild.container.run_in_container` (*command*, *container\_dir*, *mounts*)

Run a given command inside a container.

Mounts a directory as a container at the given mountpoint and tries to run the given command inside the new container.

`benchbuild.container.set_input_container` (*container*, *cfg*)

Save the input for the container in the configurations.

`benchbuild.container.setup_bash_in_container` (*builddir*, *container*, *outfile*, *mounts*, *shell*)

Setup a bash environment inside a container.

Creates a new chroot, which the user can use as a bash to run the wanted projects inside the mounted container, that also gets returned afterwards.

`benchbuild.container.setup_container` (*builddir*, *container*)

Prepare the container and returns the path where it can be found.

`benchbuild.container.setup_directories` (*builddir*)

Create the in and out directories of the container.

#### 1.9.1.5 benchbuild.driver module

`class benchbuild.driver.PollyProfiling` (*executable*)

Bases: `plumbum.cli.application.Application`

Frontend for running/building the benchbuild study framework.

**VERSION** = '1.2.1'

**main** (*\*args*)

**verbosity**

Sets an attribute

`benchbuild.driver.main` (*\*args*)

Main function.

#### 1.9.1.6 benchbuild.experiment module

BenchBuild's skeleton for experiments.

An `benchbuild.experiment` defines a series of phases that constitute a benchbuild compatible experiment. This is the default implementation of an experiment.

Clients can derive from class `class::benchbuild.experiment.Experiment` and override the methods relevant to their experiment.

An experiment can have a variable number of phases / steps / substeps.

## Phases / Steps / Substeps

All phases/steps/substeps support being used as a context manager. All 3 of them catch `ProcessExecutionErrors` that may be thrown from `plumbum`, without aborting the whole experiment. However, an error is tracked.

## Actions

An experiment performs the following actions in order:

1. **clean** - Clean any previous runs that collide with our directory
2. **prepare** - Prepare the experiment, this is a good place to copy relevant files over for testing.
3. **run (run\_tests)** - Run the experiment. The ‘meat’ lies here. Override This to perform all your experiment needs.

```
class benchbuild.experiment.Experiment (projects=None, group=None)
    Bases: object
```

A series of commands executed on a project that form an experiment.

The default implementation should provide a sane environment for all derivatives.

One important task executed by the basic implementation is setting up the default set of projects that belong to this project. As every project gets registered in the `ProjectFactory`, the experiment gets a list of experiment names that work as a filter.

**NAME** = None

**actions** ()

**actions\_for\_project** (project)

Get the actions a project wants to run.

**Parameters** **project** (*benchbuild.Project*) – the project we want to run.

**populate\_projects** (projects\_to\_filter=None, group=None)

Populate the list of projects that belong to this experiment.

**Parameters**

- **projects\_to\_filter** (*list*) – List of projects we want to assign to this experiment. We intersect the list of projects with the list of supported projects to get the list of projects that belong to this experiment.
- **group** (*str*) – In addition to the project filter, we provide a way to filter whole groups.

```
class benchbuild.experiment.ExperimentRegistry (name, bases, dict)
    Bases: type
```

Registry for benchbuild experiments.

**experiments** = {'cs': <class 'benchbuild.experiments.compilestats.CompilestatsExperiment'>, 'p-cs': <class 'benchbuild.experiments.compilestats.CompilestatsExperiment'>}

```
class benchbuild.experiment.RuntimeExperiment (projects=None, group=None)
    Bases: benchbuild.experiment.Experiment
```

Additional runtime only features for experiments.

**get\_papi\_calibration** (project, calibrate\_call)

Get calibration values for PAPI based measurements.

**Parameters**

- **project** (`Project`) – Unused (deprecated).
- **calibrate\_call** (`benchbuild.utils.cmd`) – The calibration command we will use.

**persist\_calibration** (`project, cmd, calibration`)

Persist the result of a calibration call.

#### Parameters

- **project** (`benchbuild.Project`) – The calibration values will be assigned to this project.
- **cmd** (`benchbuild.utils.cmd`) – The command we used to generate the calibration values.
- **calibration** (`int`) – The calibration time in nanoseconds.

`benchbuild.experiment.get_group_projects` (`group, experiment`)

Get a list of project names for the given group.

Filter the projects assigned to this experiment by group.

#### Parameters

- **group** (`str`) – The group.
- **experiment** (`benchbuild.Experiment`) – The experiment we draw our projects to filter from.

**Returns (list):** A list of project names for the group that are supported by this experiment.

`benchbuild.experiment.newline` (`ostream`)

Break the current line in the output stream.

Don't reuse the current line, if :o: is not attached to a tty.

**Parameters** **o** (`stream`) – The stream we insert a newline.

Returns (stream): The stream

`benchbuild.experiment.phase` (`name, pname='FIXME: Unset', cleaner=None`)

Introduce a new phase.

This just introduces a new (cosmetic) phase distinction between different experiment phases. This can be used as a contextmanager to distinguish different experiment phases.

#### Parameters

- **name** (`str`) – Name of the phase.
- **pname** (`str`) – Project Name this phase will be started for.
- **cleaner** (`callable`) – Cleaner callable that will be invoked as soon as the phase failed and we cannot recover.

`benchbuild.experiment.static_var` (`varname, value`)

Decorate something with a static variable.

### Example

```
@staticvar(bar, 0)
def foo():
    foo.bar = 1
    return foo.bar
```

**Parameters**

- **varname** (*str*) – The name of the static variable.
- **value** – The initial value of the static variable.

**Returns** A decorator that adds a new attribute to the given object.

`benchbuild.experiment.step` (*name*)

Introduce a new step.

This just introduces a new (cosmetic) step distinction between different experiment steps. This can be used as a contextmanager to distinguish different experiment steps.

**Parameters** **name** (*str*) – The name of the step

`benchbuild.experiment.substep` (*name*)

Introduce a new substep.

This just introduces a new (cosmetic) substep distinction between different experiment steps. This can be used as a contextmanager to distinguish different experiment steps.

**Parameters** **name** (*str*) – The name of the substep

### 1.9.1.7 benchbuild.likwid module

Likwid helper functions.

Extract information from likwid's CSV output.

`benchbuild.likwid.fetch_cols` (*fstream*, *split\_char*=',')

Fetch columns from likwid's output stream.

**Parameters**

- **fstream** – The filestream with likwid's output.
- **split\_car** (*str*) – The character we split on, default ','

**Returns** (**list(str)**): A list containing the elements of *fstream*, after splitting at *split\_char*.

`benchbuild.likwid.get_likwid_perfctr` (*infile*)

Get a complete list of all measurements.

**Parameters** **infile** – The filestream containing all likwid output.

**Returns** A list of all measurements extracted from likwid's file stream.

`benchbuild.likwid.get_measurements` (*region*, *core\_info*, *data*, *extra\_offset*=0)

Get the complete measurement info from likwid's region info.

**Parameters**

- **region** – The region we took a measurement in.
- **core\_info** – The core information.



- **data** – The raw data.
- **extra\_offset** (*int*) – default = 0

**Returns** (**list**((**region**, **metric**, **core**, **value**))): A list of measurement tuples, a tuple contains the information about the region, the metric, the core and the actual value.

`benchbuild.likwid.read_struct (fstream)`

Read a likwid struct from the text stream.

**Parameters** **fstream** – Likwid’s filestream.

**Returns** (**dict**(**str**: **str**)): A dict containing all likwid’s struct info as key/value pairs.

`benchbuild.likwid.read_structs (fstream)`

Read all structs from likwid’s file stream.

**Parameters** **fstream** – Likwid’s output file stream.

**Returns** A generator that can be used to iterate over all structs in the fstream.

`benchbuild.likwid.read_table (fstream)`

Read a likwid table info from the text stream.

**Parameters** **fstream** – Likwid’s filestream.

**Returns** (**dict**(**str**: **str**)): A dict containing likwid’s table info as key/value pairs.

`benchbuild.likwid.read_tables (fstream)`

Read all tables from likwid’s file stream.

**Parameters** **fstream** – Likwid’s output file stream.

**Returns** A generator that can be used to iterate over all tables in the fstream.

### 1.9.1.8 benchbuild.log module

Analyze the BB database.

**class** `benchbuild.log.BenchBuildLog (executable)`

Bases: `plumbum.cli.application.Application`

Frontend command to the benchbuild database.

**experiment** (*experiments*)

Set the experiments to fetch the log for.

**experiment\_ids** (*experiment\_ids*)

Set the experiment ids to fetch the log for.

**log\_type** (*types*)

Set the output types to print.

**main** ()

Run the log command.

**project** (*projects*)

Set the projects to fetch the log for.

**project\_ids** (*project\_ids*)

Set the project ids to fetch the log for.

`benchbuild.log.print_logs(query, types=None)`  
Print status logs.

`benchbuild.log.print_runs(query)`  
Print all rows in this result query.

### 1.9.1.9 benchbuild.project module

Project handling for the benchbuild study.

**class** `benchbuild.project.Project(exp, group=None)`  
Bases: `object`

benchbuild's Project class.

**A project defines how run-time testing and cleaning is done for this IR project**

**CONTAINER = <benchbuild.utils.container.Gentoo object>**

**DOMAIN = None**

**GROUP = None**

**NAME = None**

**SRC\_FILE = None**

**VERSION = None**

**build()**  
Build the project.

**clean()**  
Clean the project build directory.

**compiler\_extension**  
Return the compiler extension registered to this project.

**configure()**  
Configure the project.

**download()**  
Download the input source for this project.

**prepare()**  
Prepare the build directory.

**run(experiment)**  
Run the tests of this project.

This method initializes the default environment and takes care of cleaning up the mess we made, after a successful run.

**Parameters** `experiment` – The experiment we run this project under

**run\_tests(experiment)**  
Run the tests of this project.

Clients override this method to provide customized run-time tests.

**Parameters** `experiment` – The experiment we run this project under

**run\_uuid**  
Get the UUID that groups all tests for one project run.

Parameters **create\_new** – Create a fresh UUID (Default: False)

**runtime\_extension**

Return the runtime extension registered for this project.

**setup\_derived\_filenames** ()

Construct all derived file names.

**class** `benchbuild.project.ProjectDecorator` (*name, bases, attrs*)

Bases: `benchbuild.project.ProjectRegistry`

Decorate the interface of a project with the `in_builddir` decorator.

This is just a small safety net for benchbuild users, because we make sure to run every project method in the project's build directory.

**decorated\_methods** = ['build', 'configure', 'download', 'prepare', 'run\_tests']

**class** `benchbuild.project.ProjectRegistry` (*name, bases, attrs*)

Bases: `type`

Registry for benchbuild projects.

**projects** = {'bzip2': <class 'benchbuild.projects.benchbuild.bzip2.Bzip2'>, 'ccrypt': <class 'benchbuild.projects.benchbuild.ccrypt.Ccrypt'>}

#### 1.9.1.10 benchbuild.report module

**class** `benchbuild.report.BenchBuildReport` (*executable*)

Bases: `plumbum.cli.application.Application`

Generate Reports from the benchbuild db.

**experiment\_ids** (*ids*)

**experiments** (*experiments*)

**main** (*\*args*)

**outfile** (*outfile*)

#### 1.9.1.11 benchbuild.run module

benchbuild's run command.

This subcommand executes experiments on a set of user-controlled projects. See the output of `benchbuild run --help` for more information.

**class** `benchbuild.run.BenchBuildRun` (*executable*)

Bases: `plumbum.cli.application.Application`

Frontend for running experiments in the benchbuild study framework.

**experiment\_tag** (*description*)

**experiments** (*experiments*)

**group** (*group*)

**list\_experiments** ()

**list\_projects** ()

**main** ()

Main entry point of benchbuild run.

**pretend**

Sets an attribute

**projects** (*projects*)**show\_config**

Sets an attribute

**store\_config**

Sets an attribute

`benchbuild.run.print_projects(exp)`

Print a list of projects registered for that experiment.

**Parameters** `exp` – The experiment to print all projects for.

### 1.9.1.12 benchbuild.settings module

Settings module for benchbuild.

All settings are stored in a simple dictionary. Each setting should be modifiable via environment variable.

**class** `benchbuild.settings.Configuration` (*parent\_key*, *node=None*, *parent=None*, *init=True*)Bases: `object`

Dictionary-like data structure to contain all configuration variables.

This serves as a configuration dictionary throughout benchbuild. You can use it to access all configuration options that are available. Whenever the structure is updated with a new subtree, all variables defined in the new subtree are updated from the environment.

**Environment variables are generated from the tree paths automatically.** `CFG["build_dir"]` becomes `BB_BUILD_DIR` `CFG["llvm"]["dir"]` becomes `BB_LLVM_DIR`

The configuration can be stored/loaded as JSON.

### Examples

```
>>> from benchbuild import settings as s
>>> c = s.Configuration('bb')
>>> c['test'] = 42
>>> c['test']
BB_TEST=42
>>> str(c['test'])
'42'
>>> type(c['test'])
<class 'benchbuild.settings.Configuration'>
```

**filter\_exports()****has\_default()**

Check, if the node contains a 'default' value.

**has\_value()**

Check, if the node contains a 'value'.

**init\_from\_env()**

Initialize this node from environment.

If we're a leaf node, i.e., a node containing a dictionary that consist of a 'default' key, compute our env variable and initialize our value from the environment. Otherwise, init our children.

**is\_leaf()**

Check, if the node is a 'leaf' node.

**load(\_from)**

Load the configuration dictionary from file.

**store(config\_file)**

Store the configuration dictionary to a file.

**update(cfg\_dict)**

Update the configuration dictionary with new content.

This just delegates the update down to the internal data structure. No validation is done on the format, be sure you know what you do.

**Parameters** `cfg_dict` – A configuration dictionary.

**value()**

Return the node value, if we're a leaf node.

## Examples

```
>>> from benchbuild import settings as s
>>> c = s.Configuration("test")
>>> c['x'] = { "y" : { "value" : None }, "z" : { "value" : 2 } }
>>> c['x']['y'].value() == None
True
>>> c['x']['z'].value()
2
>>> c['x'].value()
TEST_X_Y=null
TEST_X_Z=2
```

**exception** `benchbuild.settings.InvalidConfigKey`

Bases: `RuntimeWarning`

Warn, if you access a non-existing key benchbuild's configuration.

**class** `benchbuild.settings.UUIDEncoder` (\*, *skipkeys=False*, *ensure\_ascii=True*, *check\_circular=True*, *allow\_nan=True*, *sort\_keys=False*, *indent=None*, *separators=None*, *default=None*)

Bases: `json.encoder.JSONEncoder`

Encoder module for UUID objects.

**default(obj)**

Encode UUID objects as string.

`benchbuild.settings.available_cpu_count()`

Get the number of available CPUs.

Number of available virtual or physical CPUs on this system, i.e. user/real as output by `time(1)` when called with an optimally scaling userspace-only program.

**Returns** Number of available CPUs.

`benchbuild.settings.escape_json(raw_str)`

Shell-Escape a json input string.

**Parameters** `raw_str` – The unescaped string.

`benchbuild.settings.find_config (default='.benchbuild.json', root='.')`  
Find the path to the default config file.

We look at `:root:` for the `:default:` config file. If we can't find it there we start looking at the parent directory recursively until we find a file named `:default:` and return the absolute path to it. If we can't find anything, we return `None`.

**Parameters**

- **default** – The name of the config file we look for.
- **root** – The directory to start looking for.

**Returns** Path to the default config file, `None` if we can't find anything.

`benchbuild.settings.to_env_dict (config)`  
Convert configuration object to a flat dictionary.

`benchbuild.settings.update_env ()`

### 1.9.1.13 benchbuild.slurm module

Dump SLURM script that executes the selected experiment with all projects.

This basically provides the same as `benchbuild run`, except that it just dumps a slurm batch script that executes everything as an array job on a configurable SLURM cluster.

**class** `benchbuild.slurm.Slurm (executable)`  
Bases: `plumbum.cli.application.Application`  
Generate a SLURM script.  
**experiment** (*cfg\_experiment*)  
Specify experiments to run  
**experiment\_tag** (*description*)  
A description for this experiment run  
**group** (*groups*)  
Run a group of projects under the given experiments  
**main** ()  
Main entry point of benchbuild run.  
**projects** (*projects*)  
Specify projects to run

### 1.9.1.14 benchbuild.test module

**class** `benchbuild.test.BenchBuildTest (executable)`  
Bases: `plumbum.cli.application.Application`  
Create regression tests for polyjit from the measurements database.  
**get\_check\_line** (*name, module*)  
**main** ()  
**opt\_flags** ()  
**prefix** (*prefix*)

## CHAPTER 2

---

### Indices and tables

---

- `genindex`
- `modindex`
- `search`





### b

benchbuild, 4  
benchbuild.bootstrap, 55  
benchbuild.container, 55  
benchbuild.driver, 57  
benchbuild.experiment, 57  
benchbuild.experiments, 4  
benchbuild.experiments.compilestats, 7  
benchbuild.experiments.compilestats\_ewpt, 7  
benchbuild.experiments.empty, 7  
benchbuild.experiments.papi, 8  
benchbuild.experiments.pjtest, 8  
benchbuild.experiments.polly, 4  
benchbuild.experiments.polly.openmp, 4  
benchbuild.experiments.polly.openmpvect, 5  
benchbuild.experiments.polly.polly, 5  
benchbuild.experiments.polly.pollyperformance, 6  
benchbuild.experiments.polly.vectorize, 6  
benchbuild.experiments.polyjit, 9  
benchbuild.experiments.raw, 13  
benchbuild.likwid, 60  
benchbuild.log, 61  
benchbuild.project, 62  
benchbuild.projects, 14  
benchbuild.projects.apollo, 14  
benchbuild.projects.apollo.group, 14  
benchbuild.projects.apollo.rodinia, 14  
benchbuild.projects.apollo.scimark, 15  
benchbuild.projects.benchbuild, 15  
benchbuild.projects.benchbuild.bzip2, 15  
benchbuild.projects.benchbuild.ccrypt, 16  
benchbuild.projects.benchbuild.crafty, 16  
benchbuild.projects.benchbuild.croccopat, 16  
benchbuild.projects.benchbuild.ffmpeg, 17  
benchbuild.projects.benchbuild.group, 17  
benchbuild.projects.benchbuild.gzip, 17  
benchbuild.projects.benchbuild.js, 18  
benchbuild.projects.benchbuild.lammps, 18  
benchbuild.projects.benchbuild.lapack, 19  
benchbuild.projects.benchbuild.leveldb, 19  
benchbuild.projects.benchbuild.linpack, 20  
benchbuild.projects.benchbuild.lulesh, 20  
benchbuild.projects.benchbuild.luleshomp, 20  
benchbuild.projects.benchbuild.mcrypt, 21  
benchbuild.projects.benchbuild.minisat, 21  
benchbuild.projects.benchbuild.openssl, 22  
benchbuild.projects.benchbuild.postgres, 22  
benchbuild.projects.benchbuild.povray, 22  
benchbuild.projects.benchbuild.python, 23  
benchbuild.projects.benchbuild.rasdaman, 23  
benchbuild.projects.benchbuild.ruby, 24  
benchbuild.projects.benchbuild.sdcc, 24  
benchbuild.projects.benchbuild.sevenz, 24  
benchbuild.projects.benchbuild.sqlite3, 25

`benchbuild.projects.benchbuild.tcc`, 25  
`benchbuild.projects.benchbuild.x264`, 26  
`benchbuild.projects.benchbuild.xz`, 26  
`benchbuild.projects.gentoo`, 27  
`benchbuild.projects.gentoo.autoportage`,  
27  
`benchbuild.projects.gentoo.bzip2`, 27  
`benchbuild.projects.gentoo.crafty`, 27  
`benchbuild.projects.gentoo.eix`, 28  
`benchbuild.projects.gentoo.gentoo`, 28  
`benchbuild.projects.gentoo.gzip`, 29  
`benchbuild.projects.gentoo.info`, 29  
`benchbuild.projects.gentoo.lammps`, 29  
`benchbuild.projects.gentoo.portage_gen`,  
30  
`benchbuild.projects.gentoo.postgresql`,  
30  
`benchbuild.projects.gentoo.sevenz`, 31  
`benchbuild.projects.gentoo.x264`, 31  
`benchbuild.projects.gentoo.xz`, 31  
`benchbuild.projects.lnt`, 32  
`benchbuild.projects.lnt.lnt`, 32  
`benchbuild.projects.polybench`, 33  
`benchbuild.projects.polybench.polybench`,  
33  
`benchbuild.report`, 63  
`benchbuild.reports`, 36  
`benchbuild.reports.raw`, 36  
`benchbuild.run`, 63  
`benchbuild.settings`, 64  
`benchbuild.slurm`, 66  
`benchbuild.test`, 66  
`benchbuild.utils`, 36  
`benchbuild.utils.actions`, 36  
`benchbuild.utils.bootstrap`, 38  
`benchbuild.utils.compiler`, 39  
`benchbuild.utils.container`, 40  
`benchbuild.utils.db`, 41  
`benchbuild.utils.downloader`, 43  
`benchbuild.utils.log`, 44  
`benchbuild.utils.path`, 44  
`benchbuild.utils.run`, 45  
`benchbuild.utils.schema`, 49  
`benchbuild.utils.slurm`, 52  
`benchbuild.utils.user_interface`, 53  
`benchbuild.utils.versions`, 53  
`benchbuild.utils.wrapping`, 53

## A

- actions() (benchbuild.experiment.Experiment method), 58
- actions\_for\_project() (benchbuild.experiment.Experiment method), 58
- actions\_for\_project() (benchbuild.experiments.compilestats.CompilestatsExperiment method), 7
- actions\_for\_project() (benchbuild.experiments.compilestats.PollyCompilestatsExperiment method), 7
- actions\_for\_project() (benchbuild.experiments.empty.Empty method), 8
- actions\_for\_project() (benchbuild.experiments.papi.PapiScopCoverage method), 8
- actions\_for\_project() (benchbuild.experiments.papi.PapiStandardScopCoverage method), 8
- actions\_for\_project() (benchbuild.experiments.pjtest.Test method), 9
- actions\_for\_project() (benchbuild.experiments.polly.openmp.PollyOpenMP method), 5
- actions\_for\_project() (benchbuild.experiments.polyjit.Compilestats method), 9
- actions\_for\_project() (benchbuild.experiments.polyjit.PJITlikwid method), 10
- actions\_for\_project() (benchbuild.experiments.polyjit.PJITpapi method), 10
- actions\_for\_project() (benchbuild.experiments.polyjit.PJITperf method), 10
- actions\_for\_project() (benchbuild.experiments.polyjit.PJITraw method), 9
- actions\_for\_project() (benchbuild.experiments.polyjit.PJITRegression method), 10
- actions\_for\_project() (benchbuild.experiments.polyjit.PolyJIT method), 10
- actions\_for\_project() (benchbuild.experiments.polyjit.PolyJITFull method), 11
- actions\_for\_project() (benchbuild.experiments.raw.RawRuntime method), 13
- Adi (class in benchbuild.projects.polybench.polybench), 33
- Any (class in benchbuild.utils.actions), 36
- ApolloGroup (class in benchbuild.projects.apollo.group), 14
- ask() (in module benchbuild.utils.user\_interface), 53
- Atax (class in benchbuild.projects.polybench.polybench), 33
- AutoPortage (class in benchbuild.projects.gentoo.autoportage), 27
- available\_cpu\_count() (in module benchbuild.settings), 65

## B

- BashStrategy (class in benchbuild.container), 55
- begin (benchbuild.utils.schema.Experiment attribute), 49
- begin (benchbuild.utils.schema.Run attribute), 51
- begin (benchbuild.utils.schema.RunGroup attribute), 51
- begin (benchbuild.utils.schema.RunLog attribute), 52
- begin() (in module benchbuild.utils.run), 45
- begin\_run\_group() (in module benchbuild.utils.run), 45
- begin\_transaction() (benchbuild.utils.actions.Experiment method), 37
- benchbuild (module), 4
- benchbuild.bootstrap (module), 55
- benchbuild.container (module), 55
- benchbuild.driver (module), 57

- benchbuild.experiment (module), 57
- benchbuild.experiments (module), 4
- benchbuild.experiments.compilestats (module), 7
- benchbuild.experiments.compilestats\_ewpt (module), 7
- benchbuild.experiments.empty (module), 7
- benchbuild.experiments.papi (module), 8
- benchbuild.experiments.pjtest (module), 8
- benchbuild.experiments.polly (module), 4
- benchbuild.experiments.polly.openmp (module), 4
- benchbuild.experiments.polly.openmpvect (module), 5
- benchbuild.experiments.polly.polly (module), 5
- benchbuild.experiments.polly.pollyperformance (module), 6
- benchbuild.experiments.polly.vectorize (module), 6
- benchbuild.experiments.polyjit (module), 9
- benchbuild.experiments.raw (module), 13
- benchbuild.likwid (module), 60
- benchbuild.log (module), 61
- benchbuild.project (module), 62
- benchbuild.projects (module), 14
- benchbuild.projects.apollo (module), 14
- benchbuild.projects.apollo.group (module), 14
- benchbuild.projects.apollo.rodinia (module), 14
- benchbuild.projects.apollo.scimark (module), 15
- benchbuild.projects.benchbuild (module), 15
- benchbuild.projects.benchbuild.bzip2 (module), 15
- benchbuild.projects.benchbuild.ccrypt (module), 16
- benchbuild.projects.benchbuild.crafty (module), 16
- benchbuild.projects.benchbuild.crocpat (module), 16
- benchbuild.projects.benchbuild.ffmpeg (module), 17
- benchbuild.projects.benchbuild.group (module), 17
- benchbuild.projects.benchbuild.gzip (module), 17
- benchbuild.projects.benchbuild.js (module), 18
- benchbuild.projects.benchbuild.lammps (module), 18
- benchbuild.projects.benchbuild.lapack (module), 19
- benchbuild.projects.benchbuild.leveldb (module), 19
- benchbuild.projects.benchbuild.linpack (module), 20
- benchbuild.projects.benchbuild.lulesh (module), 20
- benchbuild.projects.benchbuild.luleshomp (module), 20
- benchbuild.projects.benchbuild.mcrypt (module), 21
- benchbuild.projects.benchbuild.minisat (module), 21
- benchbuild.projects.benchbuild.openssl (module), 22
- benchbuild.projects.benchbuild.postgres (module), 22
- benchbuild.projects.benchbuild.povray (module), 22
- benchbuild.projects.benchbuild.python (module), 23
- benchbuild.projects.benchbuild.rasdaman (module), 23
- benchbuild.projects.benchbuild.ruby (module), 24
- benchbuild.projects.benchbuild.sdcc (module), 24
- benchbuild.projects.benchbuild.sevenz (module), 24
- benchbuild.projects.benchbuild.sqlite3 (module), 25
- benchbuild.projects.benchbuild.tcc (module), 25
- benchbuild.projects.benchbuild.x264 (module), 26
- benchbuild.projects.benchbuild.xz (module), 26
- benchbuild.projects.gentoo (module), 27
- benchbuild.projects.gentoo.autoportage (module), 27
- benchbuild.projects.gentoo.bzip2 (module), 27
- benchbuild.projects.gentoo.crafty (module), 27
- benchbuild.projects.gentoo.eix (module), 28
- benchbuild.projects.gentoo.gentoo (module), 28
- benchbuild.projects.gentoo.gzip (module), 29
- benchbuild.projects.gentoo.info (module), 29
- benchbuild.projects.gentoo.lammps (module), 29
- benchbuild.projects.gentoo.portage\_gen (module), 30
- benchbuild.projects.gentoo.postgresql (module), 30
- benchbuild.projects.gentoo.sevenz (module), 31
- benchbuild.projects.gentoo.x264 (module), 31
- benchbuild.projects.gentoo.xz (module), 31
- benchbuild.projects.lnt (module), 32
- benchbuild.projects.lnt.lnt (module), 32
- benchbuild.projects.polybench (module), 33
- benchbuild.projects.polybench.polybench (module), 33
- benchbuild.report (module), 63
- benchbuild.reports (module), 36
- benchbuild.reports.raw (module), 36
- benchbuild.run (module), 63
- benchbuild.settings (module), 64
- benchbuild.slurm (module), 66
- benchbuild.test (module), 66
- benchbuild.utils (module), 36
- benchbuild.utils.actions (module), 36
- benchbuild.utils.bootstrap (module), 38
- benchbuild.utils.compiler (module), 39
- benchbuild.utils.container (module), 40
- benchbuild.utils.db (module), 41
- benchbuild.utils.downloader (module), 43
- benchbuild.utils.log (module), 44
- benchbuild.utils.path (module), 44
- benchbuild.utils.run (module), 45
- benchbuild.utils.schema (module), 49
- benchbuild.utils.slurm (module), 52
- benchbuild.utils.user\_interface (module), 53
- benchbuild.utils.versions (module), 53
- benchbuild.utils.wrapping (module), 53
- BenchBuildBootstrap (class in benchbuild.bootstrap), 55
- BenchBuildGroup (class in benchbuild.projects.benchbuild.group), 17
- BenchBuildLog (class in benchbuild.log), 61
- BenchBuildReport (class in benchbuild.report), 63
- BenchBuildRun (class in benchbuild.run), 63
- BenchBuildTest (class in benchbuild.test), 66
- BicG (class in benchbuild.projects.polybench.polybench), 33
- boost\_src\_dir (benchbuild.projects.benchbuild.povray.Povray attribute), 23
- boost\_src\_file (benchbuild.projects.benchbuild.povray.Povray attribute), 23
- boost\_src\_uri (benchbuild.projects.benchbuild.povray.Povray attribute), 23

- Build (class in benchbuild.utils.actions), 37
  - build() (benchbuild.project.Project method), 62
  - build() (benchbuild.projects.apollo.rodinia.Rodinia method), 14
  - build() (benchbuild.projects.apollo.scimark.SciMark method), 15
  - build() (benchbuild.projects.benchbuild.bzip2.Bzip2 method), 15
  - build() (benchbuild.projects.benchbuild.ccrypt.Ccrypt method), 16
  - build() (benchbuild.projects.benchbuild.crafty.Crafty method), 16
  - build() (benchbuild.projects.benchbuild.crocopat.Crocopat method), 17
  - build() (benchbuild.projects.benchbuild.ffmpeg.LibAV method), 17
  - build() (benchbuild.projects.benchbuild.gzip.Gzip method), 18
  - build() (benchbuild.projects.benchbuild.js.SpiderMonkey method), 18
  - build() (benchbuild.projects.benchbuild.lammps.Lammps method), 18
  - build() (benchbuild.projects.benchbuild.lapack.Lapack method), 19
  - build() (benchbuild.projects.benchbuild.lapack.OpenBlas method), 19
  - build() (benchbuild.projects.benchbuild.leveldb.LevelDB method), 19
  - build() (benchbuild.projects.benchbuild.linpack.Linpack method), 20
  - build() (benchbuild.projects.benchbuild.lulesh.Lulesh method), 20
  - build() (benchbuild.projects.benchbuild.luleshomp.LuleshOMP method), 21
  - build() (benchbuild.projects.benchbuild.mcrypt.MCrypt method), 21
  - build() (benchbuild.projects.benchbuild.minisat.Minisat method), 21
  - build() (benchbuild.projects.benchbuild.openssl.LibreSSL method), 22
  - build() (benchbuild.projects.benchbuild.povray.Povray method), 23
  - build() (benchbuild.projects.benchbuild.python.Python method), 23
  - build() (benchbuild.projects.benchbuild.rasdaman.Rasdaman method), 23
  - build() (benchbuild.projects.benchbuild.ruby.Ruby method), 24
  - build() (benchbuild.projects.benchbuild.sdcc.SDCC method), 24
  - build() (benchbuild.projects.benchbuild.sevenz.SevenZip method), 25
  - build() (benchbuild.projects.benchbuild.sqlite3.SQLite3 method), 25
  - build() (benchbuild.projects.benchbuild.tcc.TCC method), 25
  - build() (benchbuild.projects.benchbuild.x264.X264 method), 26
  - build() (benchbuild.projects.benchbuild.xz.XZ method), 26
  - build() (benchbuild.projects.gentoo.autoportage.AutoPortage method), 27
  - build() (benchbuild.projects.gentoo.bzip2.BZip2 method), 27
  - build() (benchbuild.projects.gentoo.crafty.Crafty method), 27
  - build() (benchbuild.projects.gentoo.eix.Eix method), 28
  - build() (benchbuild.projects.gentoo.gentoo.GentooGroup method), 28
  - build() (benchbuild.projects.gentoo.gzip.GZip method), 29
  - build() (benchbuild.projects.gentoo.info.Info method), 29
  - build() (benchbuild.projects.gentoo.lammps.Lammps method), 29
  - build() (benchbuild.projects.gentoo.postgresql.Postgresql method), 30
  - build() (benchbuild.projects.gentoo.sevenz.SevenZip method), 31
  - build() (benchbuild.projects.gentoo.x264.X264 method), 31
  - build() (benchbuild.projects.gentoo.xz.XZ method), 31
  - build() (benchbuild.projects.lnt.LNTGroup method), 32
  - build() (benchbuild.projects.polybench.polybench.PolyBenchGroup method), 35
  - build\_leveldb() (benchbuild.projects.benchbuild.sqlite3.SQLite3 method), 25
  - builddir() (benchbuild.container.Container method), 55
  - Bzip2 (class in benchbuild.projects.benchbuild.bzip2), 15
  - BZip2 (class in benchbuild.projects.gentoo.bzip2), 27
- ## C
- cached() (in module benchbuild.utils.container), 41
  - Ccrypt (class in benchbuild.projects.benchbuild.ccrypt), 16
  - check\_uchroot\_config() (in module benchbuild.utils.bootstrap), 38
  - Cholesky (class in benchbuild.projects.polybench.polybench), 33
  - clang() (in module benchbuild.utils.compiler), 39
  - clang\_cxx() (in module benchbuild.utils.compiler), 39
  - Clean (class in benchbuild.utils.actions), 37
  - clean() (benchbuild.project.Project method), 62
  - clean\_directories() (in module benchbuild.container), 56
  - CleanExtra (class in benchbuild.utils.actions), 37
  - collect\_compilestats() (in module benchbuild.experiments.compilestats), 7

[collect\\_compilestats\(\)](#) (in module `benchbuild.experiments.papi`), 8  
[command](#) (`benchbuild.utils.schema.Run` attribute), 51  
[compiler\\_extension](#) (`benchbuild.project.Project` attribute), 62  
[CompileStat](#) (class in `benchbuild.utils.schema`), 49  
[Compilestats](#) (class in `benchbuild.experiments.polyjit`), 9  
[CompilestatsExperiment](#) (class in `benchbuild.experiments.compilestats`), 7  
[component](#) (`benchbuild.utils.schema.CompileStat` attribute), 49  
[config](#) (`benchbuild.utils.schema.RunLog` attribute), 52  
[Config](#) (class in `benchbuild.utils.schema`), 49  
[Configuration](#) (class in `benchbuild.settings`), 64  
[Configure](#) (class in `benchbuild.utils.actions`), 37  
[configure\(\)](#) (`benchbuild.container.SetupPolyJITGentooStrategy` method), 56  
[configure\(\)](#) (`benchbuild.project.Project` method), 62  
[configure\(\)](#) (`benchbuild.projects.apollo.rodinia.Rodinia` method), 14  
[configure\(\)](#) (`benchbuild.projects.apollo.scimark.SciMark` method), 15  
[configure\(\)](#) (`benchbuild.projects.benchbuild.bzip2.Bzip2` method), 15  
[configure\(\)](#) (`benchbuild.projects.benchbuild.ccrypt.Ccrypt` method), 16  
[configure\(\)](#) (`benchbuild.projects.benchbuild.crafty.Crafty` method), 16  
[configure\(\)](#) (`benchbuild.projects.benchbuild.croccpat.Croccpat` method), 17  
[configure\(\)](#) (`benchbuild.projects.benchbuild.ffmpeg.LibAV` method), 17  
[configure\(\)](#) (`benchbuild.projects.benchbuild.gzip.Gzip` method), 18  
[configure\(\)](#) (`benchbuild.projects.benchbuild.js.SpiderMonkey` method), 18  
[configure\(\)](#) (`benchbuild.projects.benchbuild.lammps.Lammps` method), 18  
[configure\(\)](#) (`benchbuild.projects.benchbuild.lapack.Lapack` method), 19  
[configure\(\)](#) (`benchbuild.projects.benchbuild.lapack.OpenBLAS` method), 19  
[configure\(\)](#) (`benchbuild.projects.benchbuild.leveldb.LevelDB` method), 20  
[configure\(\)](#) (`benchbuild.projects.benchbuild.linpack.Linpack` method), 20  
[configure\(\)](#) (`benchbuild.projects.benchbuild.lulesh.Lulesh` method), 20  
[configure\(\)](#) (`benchbuild.projects.benchbuild.luleshomp.LuleshOMP` method), 21  
[configure\(\)](#) (`benchbuild.projects.benchbuild.mcrypt.MCrypt` method), 21  
[configure\(\)](#) (`benchbuild.projects.benchbuild.minisat.Minisat` create\_run() method), 22  
[configure\(\)](#) (`benchbuild.projects.benchbuild.openssl.LibreSSL` method), 22  
[configure\(\)](#) (`benchbuild.projects.benchbuild.povray.Povray` method), 23  
[configure\(\)](#) (`benchbuild.projects.benchbuild.python.Python` method), 23  
[configure\(\)](#) (`benchbuild.projects.benchbuild.rasdaman.Rasdaman` method), 23  
[configure\(\)](#) (`benchbuild.projects.benchbuild.ruby.Ruby` method), 24  
[configure\(\)](#) (`benchbuild.projects.benchbuild.sdcc.SDCC` method), 24  
[configure\(\)](#) (`benchbuild.projects.benchbuild.sevenz.SevenZip` method), 25  
[configure\(\)](#) (`benchbuild.projects.benchbuild.sqlite3.SQLite3` method), 25  
[configure\(\)](#) (`benchbuild.projects.benchbuild.tcc.TCC` method), 25  
[configure\(\)](#) (`benchbuild.projects.benchbuild.x264.X264` method), 26  
[configure\(\)](#) (`benchbuild.projects.benchbuild.xz.XZ` method), 26  
[configure\(\)](#) (`benchbuild.projects.gentoo.gentoo.GentooGroup` method), 28  
[configure\(\)](#) (`benchbuild.projects.lnt.lnt.LNTGroup` method), 32  
[configure\(\)](#) (`benchbuild.projects.polybench.polybench.PolyBenchGroup` method), 35  
[configure\(\)](#) (in module `benchbuild.utils.log`), 44  
[CONNECTION\\_MANAGER](#) (in module `benchbuild.utils.schema`), 49  
[CONTAINER](#) (`benchbuild.project.Project` attribute), 62  
[CONTAINER](#) (`benchbuild.projects.gentoo.gentoo.GentooGroup` attribute), 28  
[Container](#) (class in `benchbuild.container`), 55  
[Container](#) (class in `benchbuild.utils.container`), 40  
[ContainerBootstrap](#) (class in `benchbuild.container`), 55  
[ContainerCreate](#) (class in `benchbuild.container`), 56  
[ContainerList](#) (class in `benchbuild.container`), 56  
[ContainerRun](#) (class in `benchbuild.container`), 56  
[ContainerStrategy](#) (class in `benchbuild.container`), 56  
[Copy\(\)](#) (in module `benchbuild.utils.downloader`), 43  
[CopyNoFail\(\)](#) (in module `benchbuild.utils.downloader`), 43  
[core](#) (`benchbuild.utils.schema.Likwid` attribute), 50  
[Correlation](#) (class in `benchbuild.projects.polybench.polybench`), 33  
[Covariance](#) (class in `benchbuild.projects.polybench.polybench`), 33  
[Crafty](#) (class in `benchbuild.projects.benchbuild.crafty`), 16  
[Crafty](#) (class in `benchbuild.projects.gentoo.crafty`), 27  
[create\\_run\(\)](#) (in module `benchbuild.utils.db`), 41  
[create\\_run\\_group\(\)](#) (in module `benchbuild.utils.db`), 41



Crocopat (class in benchbuild.projects.benchbuild.crocopat), 16

## D

decorated\_methods (benchbuild.project.ProjectDecorator attribute), 63

default() (benchbuild.settings.UUIDEncoder method), 65

Deriche (class in benchbuild.projects.polybench.polybench), 33

DESCRIPTION (benchbuild.utils.actions.Any attribute), 37

DESCRIPTION (benchbuild.utils.actions.Build attribute), 37

DESCRIPTION (benchbuild.utils.actions.Clean attribute), 37

DESCRIPTION (benchbuild.utils.actions.CleanExtra attribute), 37

DESCRIPTION (benchbuild.utils.actions.Configure attribute), 37

DESCRIPTION (benchbuild.utils.actions.Download attribute), 37

DESCRIPTION (benchbuild.utils.actions.Echo attribute), 37

DESCRIPTION (benchbuild.utils.actions.Experiment attribute), 37

DESCRIPTION (benchbuild.utils.actions.MakeBuildDir attribute), 37

DESCRIPTION (benchbuild.utils.actions.Prepare attribute), 38

DESCRIPTION (benchbuild.utils.actions.Run attribute), 38

DESCRIPTION (benchbuild.utils.actions.Step attribute), 38

description (benchbuild.utils.schema.Experiment attribute), 49

description (benchbuild.utils.schema.Project attribute), 51

determine\_path() (in module benchbuild.utils.path), 44

discover() (in module benchbuild.experiments), 4

discover() (in module benchbuild.projects), 14

discover() (in module benchbuild.reports), 36

Doitgen (class in benchbuild.projects.polybench.polybench), 33

DOMAIN (benchbuild.project.Project attribute), 62

DOMAIN (benchbuild.projects.apollo.rodinia.Rodinia attribute), 14

DOMAIN (benchbuild.projects.apollo.scimark.SciMark attribute), 15

DOMAIN (benchbuild.projects.benchbuild.bzip2.Bzip2 attribute), 15

DOMAIN (benchbuild.projects.benchbuild.ccrypt.Ccrypt attribute), 16

DOMAIN (benchbuild.projects.benchbuild.crafty.Crafty attribute), 16

DOMAIN (benchbuild.projects.benchbuild.crocopat.Crocopat attribute), 16

DOMAIN (benchbuild.projects.benchbuild.ffmpeg.LibAV attribute), 17

DOMAIN (benchbuild.projects.benchbuild.gzip.Gzip attribute), 17

DOMAIN (benchbuild.projects.benchbuild.js.SpiderMonkey attribute), 18

DOMAIN (benchbuild.projects.benchbuild.lammps.Lammps attribute), 18

DOMAIN (benchbuild.projects.benchbuild.lapack.Lapack attribute), 19

DOMAIN (benchbuild.projects.benchbuild.lapack.OpenBlas attribute), 19

DOMAIN (benchbuild.projects.benchbuild.leveldb.LevelDB attribute), 19

DOMAIN (benchbuild.projects.benchbuild.linpack.Linpack attribute), 20

DOMAIN (benchbuild.projects.benchbuild.lulesh.Lulesh attribute), 20

DOMAIN (benchbuild.projects.benchbuild.luleshomp.LuleshOMP attribute), 20

DOMAIN (benchbuild.projects.benchbuild.mcrypt.MCrypt attribute), 21

DOMAIN (benchbuild.projects.benchbuild.minisat.Minisat attribute), 21

DOMAIN (benchbuild.projects.benchbuild.openssl.LibreSSL attribute), 22

DOMAIN (benchbuild.projects.benchbuild.postgres.Postgres attribute), 22

DOMAIN (benchbuild.projects.benchbuild.povray.Povray attribute), 22

DOMAIN (benchbuild.projects.benchbuild.python.Python attribute), 23

DOMAIN (benchbuild.projects.benchbuild.rasdaman.Rasdaman attribute), 23

DOMAIN (benchbuild.projects.benchbuild.ruby.Ruby attribute), 24

DOMAIN (benchbuild.projects.benchbuild.sdcc.SDCC attribute), 24

DOMAIN (benchbuild.projects.benchbuild.sevenz.SevenZip attribute), 24

DOMAIN (benchbuild.projects.benchbuild.sqlite3.SQLite3 attribute), 25

DOMAIN (benchbuild.projects.benchbuild.tcc.TCC attribute), 25

DOMAIN (benchbuild.projects.benchbuild.x264.X264 attribute), 26

DOMAIN (benchbuild.projects.benchbuild.xz.XZ attribute), 26

DOMAIN (benchbuild.projects.gentoo.bzip2.BZip2 attribute), 27

DOMAIN (benchbuild.projects.gentoo.crafty.Crafty attribute), 27

DOMAIN (benchbuild.projects.gentoo.eix.Eix attribute), 28

DOMAIN (benchbuild.projects.gentoo.gzip.GZip attribute), 29

DOMAIN (benchbuild.projects.gentoo.info.Info attribute), 29

DOMAIN (benchbuild.projects.gentoo.lammps.Lammps attribute), 29

DOMAIN (benchbuild.projects.gentoo.postgresql.Postgresql attribute), 30

DOMAIN (benchbuild.projects.gentoo.sevenz.SevenZip attribute), 31

DOMAIN (benchbuild.projects.gentoo.x264.X264 attribute), 31

DOMAIN (benchbuild.projects.gentoo.xz.XZ attribute), 31

DOMAIN (benchbuild.projects.Int.Int.LNTGroup attribute), 32

DOMAIN (benchbuild.projects.polybench.polybench.PolyBenchGroup attribute), 35

domain (benchbuild.utils.schema.Project attribute), 51

Download (class in benchbuild.utils.actions), 37

download() (benchbuild.project.Project method), 62

download() (benchbuild.projects.apollo.rodinia.Rodinia method), 14

download() (benchbuild.projects.apollo.scimark.SciMark method), 15

download() (benchbuild.projects.benchbuild.bzip2.Bzip2 method), 15

download() (benchbuild.projects.benchbuild.ccrypt.Ccrypt method), 16

download() (benchbuild.projects.benchbuild.crafty.Crafty method), 16

download() (benchbuild.projects.benchbuild.croccopat.Croccopat method), 17

download() (benchbuild.projects.benchbuild.ffmpeg.LibAV method), 17

download() (benchbuild.projects.benchbuild.gzip.Gzip method), 18

download() (benchbuild.projects.benchbuild.js.SpiderMonkey method), 18

download() (benchbuild.projects.benchbuild.lammps.Lammps method), 18

download() (benchbuild.projects.benchbuild.lapack.Lapack method), 19

download() (benchbuild.projects.benchbuild.lapack.OpenBlas method), 19

download() (benchbuild.projects.benchbuild.leveldb.LevelDB method), 20

download() (benchbuild.projects.benchbuild.linpack.Linpack method), 20

download() (benchbuild.projects.benchbuild.lulesh.Lulesh method), 20

download() (benchbuild.projects.benchbuild.luleshomp.Luleshomp method), 21

download() (benchbuild.projects.benchbuild.mcrypt.MCrypt method), 21

download() (benchbuild.projects.benchbuild.minisat.Minisat method), 22

download() (benchbuild.projects.benchbuild.openssl.LibreSSL method), 22

download() (benchbuild.projects.benchbuild.povray.Povray method), 23

download() (benchbuild.projects.benchbuild.python.Python method), 23

download() (benchbuild.projects.benchbuild.rasdaman.Rasdaman method), 23

download() (benchbuild.projects.benchbuild.ruby.Ruby method), 24

download() (benchbuild.projects.benchbuild.sdcc.SDCC method), 24

download() (benchbuild.projects.benchbuild.sevenz.SevenZip method), 25

download() (benchbuild.projects.benchbuild.sqlite3.SQLite3 method), 25

download() (benchbuild.projects.benchbuild.tcc.TCC method), 25

download() (benchbuild.projects.benchbuild.x264.X264 method), 26

download() (benchbuild.projects.benchbuild.xz.XZ method), 26

download() (benchbuild.projects.gentoo.crafty.Crafty method), 28

download() (benchbuild.projects.gentoo.gentoo.GentooGroup method), 28

download() (benchbuild.projects.Int.Int.LNTGroup method), 32

download() (benchbuild.projects.Int.Int.Povray method), 32

download() (benchbuild.projects.Int.Int.SPEC2006 method), 33

download() (benchbuild.projects.polybench.polybench.PolyBenchGroup method), 35

dump\_slurm\_script() (in module benchbuild.utils.slurm), 52

duration (benchbuild.utils.schema.Event attribute), 49

duration (benchbuild.utils.schema.PerfEvent attribute), 50

Durbin (class in benchbuild.projects.polybench.polybench), 33

**E**

Echo (class in benchbuild.utils.actions), 37

Eix (class in benchbuild.projects.gentoo.eix), 28

Empty (class in benchbuild.experiments.empty), 7

end (benchbuild.utils.schema.Experiment attribute), 49

end (benchbuild.utils.schema.Run attribute), 51

end (benchbuild.utils.schema.RunGroup attribute), 51



- end (benchbuild.utils.schema.RunLog attribute), 52  
 end() (in module benchbuild.utils.run), 46  
 end\_run\_group() (in module benchbuild.utils.run), 46  
 end\_transaction() (benchbuild.utils.actions.Experiment method), 37  
 ERROR (benchbuild.utils.actions.StepResult attribute), 38  
 escape\_json() (in module benchbuild.settings), 65  
 Event (class in benchbuild.utils.schema), 49  
 EWPTCompilestatsExperiment (class in benchbuild.experiments.compilestats\_ewpt), 7  
 experiment (benchbuild.utils.schema.RunGroup attribute), 51  
 Experiment (class in benchbuild.experiment), 58  
 Experiment (class in benchbuild.utils.actions), 37  
 Experiment (class in benchbuild.utils.schema), 49  
 experiment() (benchbuild.log.BenchBuildLog method), 61  
 experiment() (benchbuild.slurm.Slurm method), 66  
 experiment\_group (benchbuild.utils.schema.GlobalConfig attribute), 50  
 experiment\_group (benchbuild.utils.schema.Run attribute), 51  
 experiment\_ids() (benchbuild.log.BenchBuildLog method), 61  
 experiment\_ids() (benchbuild.report.BenchBuildReport method), 63  
 experiment\_name (benchbuild.utils.schema.Run attribute), 51  
 experiment\_tag() (benchbuild.run.BenchBuildRun method), 63  
 experiment\_tag() (benchbuild.slurm.Slurm method), 66  
 ExperimentRegistry (class in benchbuild.experiment), 58  
 experiments (benchbuild.experiment.ExperimentRegistry attribute), 58  
 experiments() (benchbuild.report.BenchBuildReport method), 63  
 experiments() (benchbuild.run.BenchBuildRun method), 63  
 extra\_flags() (benchbuild.experiments.compilestats\_ewpt.EWPTCompilestatsExperiment method), 7
- ## F
- fail() (in module benchbuild.utils.run), 46  
 fail\_run\_group() (in module benchbuild.utils.run), 46  
 fate\_dir (benchbuild.projects.benchbuild.ffmpeg.LibAV attribute), 17  
 fate\_uri (benchbuild.projects.benchbuild.ffmpeg.LibAV attribute), 17  
 FDTD2D (class in benchbuild.projects.polybench.polybench), 34  
 fetch\_cols() (in module benchbuild.likwid), 60  
 fetch\_leveladb() (benchbuild.projects.benchbuild.sqlite3.SQLite3 method), 25  
 fetch\_time\_output() (in module benchbuild.utils.run), 46  
 filename (benchbuild.utils.container.Container attribute), 40  
 filter\_exports() (benchbuild.settings.Configuration method), 64  
 find\_config() (in module benchbuild.settings), 66  
 find\_hash() (in module benchbuild.container), 57  
 find\_package() (in module benchbuild.utils.bootstrap), 38  
 FloydWarshall (class in benchbuild.projects.polybench.polybench), 34
- ## G
- gdal\_dir (benchbuild.projects.benchbuild.rasdaman.Rasdaman attribute), 23  
 gdal\_uri (benchbuild.projects.benchbuild.rasdaman.Rasdaman attribute), 24  
 Gemm (class in benchbuild.projects.polybench.polybench), 34  
 Gemver (class in benchbuild.projects.polybench.polybench), 34  
 generate() (benchbuild.reports.raw.RawReport method), 36  
 Gentoo (class in benchbuild.utils.container), 41  
 GentooGroup (class in benchbuild.projects.gentoo.gentoo), 28  
 Gesummv (class in benchbuild.projects.polybench.polybench), 34  
 get() (benchbuild.utils.schema.SessionManager method), 52  
 get\_check\_line() (benchbuild.test.BenchBuildTest method), 66  
 get\_compilestats() (in module benchbuild.experiments.compilestats), 7  
 get\_compilestats() (in module benchbuild.experiments.papi), 8  
 get\_exp\_ids() (benchbuild.reports.raw.RawReport method), 36  
 get\_group\_projects() (in module benchbuild.experiment), 59  
 get\_hash\_of\_dirs() (in module benchbuild.utils.downloader), 44  
 get\_likwid\_perfctr() (in module benchbuild.likwid), 60  
 get\_measurements() (in module benchbuild.likwid), 60  
 get\_papi\_calibration() (benchbuild.experiment.RuntimeExperiment method), 58  
 get\_string\_for\_language() (in module benchbuild.projects.gentoo.info), 29  
 get\_version\_from\_cache\_dir() (in module benchbuild.utils.versions), 53

- Git() (in module benchbuild.utils.downloader), [43](#)
- GlobalConfig (class in benchbuild.utils.schema), [50](#)
- Gramschmidt (class in benchbuild.projects.polybench.polybench), [34](#)
- GROUP (benchbuild.project.Project attribute), [62](#)
- GROUP (benchbuild.projects.apollo.group.ApolloGroup attribute), [14](#)
- GROUP (benchbuild.projects.benchbuild.group.BenchBuildGroup attribute), [17](#)
- GROUP (benchbuild.projects.gentoo.gentoo.GentooGroup attribute), [28](#)
- GROUP (benchbuild.projects.Int.Int.LNTGroup attribute), [32](#)
- GROUP (benchbuild.projects.polybench.polybench.PolyBenchGroup attribute), [35](#)
- group() (benchbuild.run.BenchBuildRun method), [63](#)
- group() (benchbuild.slurm.Slurm method), [66](#)
- group\_name (benchbuild.utils.schema.Project attribute), [51](#)
- guarded\_exec() (in module benchbuild.utils.run), [47](#)
- GuardedRunException, [45](#)
- Gzip (class in benchbuild.projects.benchbuild.gzip), [17](#)
- GZip (class in benchbuild.projects.gentoo.gzip), [29](#)
- H**
- handle\_stdin() (in module benchbuild.utils.run), [47](#)
- has\_default() (benchbuild.settings.Configuration method), [64](#)
- has\_value() (benchbuild.settings.Configuration method), [64](#)
- Heat3D (class in benchbuild.projects.polybench.polybench), [34](#)
- I**
- id (benchbuild.utils.schema.CompileStat attribute), [49](#)
- id (benchbuild.utils.schema.Event attribute), [49](#)
- id (benchbuild.utils.schema.Experiment attribute), [49](#)
- id (benchbuild.utils.schema.PerfEvent attribute), [50](#)
- id (benchbuild.utils.schema.Run attribute), [51](#)
- id (benchbuild.utils.schema.RunGroup attribute), [52](#)
- in\_builddir() (in module benchbuild.utils.run), [47](#)
- Info (class in benchbuild.projects.gentoo.info), [29](#)
- init\_from\_env() (benchbuild.settings.Configuration method), [64](#)
- init\_project() (benchbuild.experiments.polyjit.PolyJIT class method), [10](#)
- input\_file() (benchbuild.container.Container method), [55](#)
- inputfiles (benchbuild.projects.benchbuild.x264.X264 attribute), [26](#)
- inputfiles (benchbuild.projects.gentoo.x264.X264 attribute), [31](#)
- install\_cmake\_and\_exit() (benchbuild.container.ContainerBootstrap method), [55](#)
- install\_package() (in module benchbuild.utils.bootstrap), [38](#)
- install\_uchroot() (in module benchbuild.utils.bootstrap), [38](#)
- InvalidConfigKey, [65](#)
- is\_leaf() (benchbuild.settings.Configuration method), [65](#)
- is\_valid\_container() (in module benchbuild.utils.container), [41](#)
- J**
- Jacobi1D (class in benchbuild.projects.polybench.polybench), [34](#)
- Jacobi2Dimper (class in benchbuild.projects.polybench.polybench), [34](#)
- L**
- Lammps (class in benchbuild.projects.benchbuild.lammps), [18](#)
- Lammps (class in benchbuild.projects.gentoo.lammps), [29](#)
- Lapack (class in benchbuild.projects.benchbuild.lapack), [19](#)
- latest\_src\_uri() (benchbuild.utils.container.Gentoo method), [41](#)
- LevelDB (class in benchbuild.projects.benchbuild.leveldb), [19](#)
- LibAV (class in benchbuild.projects.benchbuild.ffmpeg), [17](#)
- libmccrypt\_dir (benchbuild.projects.benchbuild.mccrypt.MCrypt attribute), [21](#)
- libmccrypt\_file (benchbuild.projects.benchbuild.mccrypt.MCrypt attribute), [21](#)
- libmccrypt\_uri (benchbuild.projects.benchbuild.mccrypt.MCrypt attribute), [21](#)
- LibreSSL (class in benchbuild.projects.benchbuild.openssl), [22](#)
- Likwid (class in benchbuild.utils.schema), [50](#)
- Linpack (class in benchbuild.projects.benchbuild.linpack), [20](#)
- linux\_distribution\_major() (in module benchbuild.utils.bootstrap), [38](#)
- list\_experiments() (benchbuild.run.BenchBuildRun method), [63](#)
- list\_projects() (benchbuild.run.BenchBuildRun method), [63](#)
- list\_to\_path() (in module benchbuild.utils.path), [45](#)
- llvm() (in module benchbuild.utils.compiler), [39](#)
- llvm\_libs() (in module benchbuild.utils.compiler), [39](#)
- LNTGroup (class in benchbuild.projects.Int.Int), [32](#)
- load() (benchbuild.settings.Configuration method), [65](#)
- local (benchbuild.utils.container.Container attribute), [40](#)
- log\_before\_after() (in module benchbuild.utils.actions), [38](#)
- log\_type() (benchbuild.log.BenchBuildLog method), [61](#)

lt\_clang() (in module benchbuild.utils.compiler), 39  
 lt\_clang\_cxx() (in module benchbuild.utils.compiler), 40  
 Lu (class in benchbuild.projects.polybench.polybench), 34

LuDCMP (class in benchbuild.projects.polybench.polybench), 34  
 Lulesh (class in benchbuild.projects.benchbuild.lulesh), 20  
 LuleshOMP (class in benchbuild.projects.benchbuild.luleshomp), 20

## M

main() (benchbuild.bootstrap.BenchBuildBootstrap method), 55  
 main() (benchbuild.container.Container method), 55  
 main() (benchbuild.container.ContainerBootstrap method), 56  
 main() (benchbuild.container.ContainerCreate method), 56  
 main() (benchbuild.container.ContainerList method), 56  
 main() (benchbuild.container.ContainerRun method), 56  
 main() (benchbuild.driver.PollyProfiling method), 57  
 main() (benchbuild.log.BenchBuildLog method), 61  
 main() (benchbuild.report.BenchBuildReport method), 63  
 main() (benchbuild.run.BenchBuildRun method), 63  
 main() (benchbuild.slurm.Slurm method), 66  
 main() (benchbuild.test.BenchBuildTest method), 66  
 main() (in module benchbuild.container), 57  
 main() (in module benchbuild.driver), 57  
 MakeBuildDir (class in benchbuild.utils.actions), 37  
 MCrypt (class in benchbuild.projects.benchbuild.mcrypt), 21  
 Metadata (class in benchbuild.utils.schema), 50  
 metric (benchbuild.utils.schema.Likwid attribute), 50  
 Metric (class in benchbuild.utils.schema), 50  
 mhash\_dir (benchbuild.projects.benchbuild.mcrypt.MCrypt attribute), 21  
 mhash\_file (benchbuild.projects.benchbuild.mcrypt.MCrypt attribute), 21  
 mhash\_uri (benchbuild.projects.benchbuild.mcrypt.MCrypt attribute), 21  
 Minisat (class in benchbuild.projects.benchbuild.minisat), 21  
 mkdir\_uchroot() (in module benchbuild.utils.path), 45  
 mkfile\_uchroot() (in module benchbuild.utils.path), 45  
 MockObj (class in benchbuild.container), 56  
 module (benchbuild.utils.schema.RegressionTest attribute), 51  
 mounts() (benchbuild.container.Container method), 55  
 MultiSourceApplications (class in benchbuild.projects.lnt.lnt), 32  
 MultiSourceBenchmarks (class in benchbuild.projects.lnt.lnt), 32

Mvt (class in benchbuild.projects.polybench.polybench), 34

## N

NAME (benchbuild.experiment.Experiment attribute), 58  
 NAME (benchbuild.experiments.compilestats.CompilestatsExperiment attribute), 7  
 NAME (benchbuild.experiments.compilestats.PollyCompilestatsExperiment attribute), 7  
 NAME (benchbuild.experiments.compilestats\_ewpt.EWPTCompilestatsExperiment attribute), 7  
 NAME (benchbuild.experiments.empty.Empty attribute), 7  
 NAME (benchbuild.experiments.papi.PapiScopCoverage attribute), 8  
 NAME (benchbuild.experiments.papi.PapiStandardScopCoverage attribute), 8  
 NAME (benchbuild.experiments.pjtest.Test attribute), 8  
 NAME (benchbuild.experiments.polly.openmp.PollyOpenMP attribute), 5  
 NAME (benchbuild.experiments.polly.openmpvect.PollyOpenMPVectorizer attribute), 5  
 NAME (benchbuild.experiments.polly.polly.Polly attribute), 5  
 NAME (benchbuild.experiments.polly.pollyperformance.PollyPerformance attribute), 6  
 NAME (benchbuild.experiments.polly.vectorize.PollyVectorizer attribute), 6  
 NAME (benchbuild.experiments.polyjit.Compilestats attribute), 9  
 NAME (benchbuild.experiments.polyjit.PJITlikwid attribute), 10  
 NAME (benchbuild.experiments.polyjit.PJITpapi attribute), 10  
 NAME (benchbuild.experiments.polyjit.PJITperf attribute), 10  
 NAME (benchbuild.experiments.polyjit.PJITRaw attribute), 9  
 NAME (benchbuild.experiments.polyjit.PJITRegression attribute), 9  
 NAME (benchbuild.experiments.polyjit.PolyJITFull attribute), 10  
 NAME (benchbuild.experiments.raw.RawRuntime attribute), 13  
 NAME (benchbuild.project.Project attribute), 62  
 NAME (benchbuild.projects.apollo.rodinia.Rodinia attribute), 14  
 NAME (benchbuild.projects.apollo.scimark.SciMark attribute), 15  
 NAME (benchbuild.projects.benchbuild.bzip2.Bzip2 attribute), 15  
 NAME (benchbuild.projects.benchbuild.ccrypt.Ccrypt attribute), 16

NAME (benchbuild.projects.benchbuild.crafty.Crafty attribute), 16	NAME (benchbuild.projects.gentoo.crafty.Crafty attribute), 27
NAME (benchbuild.projects.benchbuild.croccat.Croccat attribute), 16	NAME (benchbuild.projects.gentoo.eix.Eix attribute), 28
NAME (benchbuild.projects.benchbuild.ffmpeg.LibAV attribute), 17	NAME (benchbuild.projects.gentoo.gzip.GZip attribute), 29
NAME (benchbuild.projects.benchbuild.gzip.Gzip attribute), 17	NAME (benchbuild.projects.gentoo.info.Info attribute), 29
NAME (benchbuild.projects.benchbuild.js.SpiderMonkey attribute), 18	NAME (benchbuild.projects.gentoo.lammps.Lammps attribute), 29
NAME (benchbuild.projects.benchbuild.lammps.Lammps attribute), 18	NAME (benchbuild.projects.gentoo.postgresql.Postgresql attribute), 30
NAME (benchbuild.projects.benchbuild.lapack.Lapack attribute), 19	NAME (benchbuild.projects.gentoo.sevenz.SevenZip attribute), 31
NAME (benchbuild.projects.benchbuild.lapack.OpenBlas attribute), 19	NAME (benchbuild.projects.gentoo.x264.X264 attribute), 31
NAME (benchbuild.projects.benchbuild.leveldb.LevelDB attribute), 19	NAME (benchbuild.projects.gentoo.xz.XZ attribute), 31
NAME (benchbuild.projects.benchbuild.linpack.Linpack attribute), 20	NAME (benchbuild.projects.Int.Int.MultiSourceApplications attribute), 32
NAME (benchbuild.projects.benchbuild.lulesh.Lulesh attribute), 20	NAME (benchbuild.projects.Int.Int.MultiSourceBenchmarks attribute), 32
NAME (benchbuild.projects.benchbuild.luleshomp.LuleshOMP attribute), 20	NAME (benchbuild.projects.Int.Int.Povray attribute), 32
NAME (benchbuild.projects.benchbuild.mcrypt.MCrypt attribute), 21	NAME (benchbuild.projects.Int.Int.SingleSourceBenchmarks attribute), 33
NAME (benchbuild.projects.benchbuild.minisat.Minisat attribute), 21	NAME (benchbuild.projects.Int.Int.SPEC2006 attribute), 33
NAME (benchbuild.projects.benchbuild.openssl.LibSSL attribute), 22	NAME (benchbuild.projects.polybench.polybench.Adi attribute), 33
NAME (benchbuild.projects.benchbuild.postgres.Postgres attribute), 22	NAME (benchbuild.projects.polybench.polybench.Atax attribute), 33
NAME (benchbuild.projects.benchbuild.povray.Povray attribute), 22	NAME (benchbuild.projects.polybench.polybench.BicG attribute), 33
NAME (benchbuild.projects.benchbuild.python.Python attribute), 23	NAME (benchbuild.projects.polybench.polybench.Cholesky attribute), 33
NAME (benchbuild.projects.benchbuild.rasdaman.Rasdaman attribute), 23	NAME (benchbuild.projects.polybench.polybench.Correlation attribute), 33
NAME (benchbuild.projects.benchbuild.ruby.Ruby attribute), 24	NAME (benchbuild.projects.polybench.polybench.Covariance attribute), 33
NAME (benchbuild.projects.benchbuild.sdcc.SDCC attribute), 24	NAME (benchbuild.projects.polybench.polybench.Deriche attribute), 33
NAME (benchbuild.projects.benchbuild.sevenz.SevenZip attribute), 24	NAME (benchbuild.projects.polybench.polybench.Doitgen attribute), 33
NAME (benchbuild.projects.benchbuild.sqlite3.SQLite3 attribute), 25	NAME (benchbuild.projects.polybench.polybench.Durbin attribute), 34
NAME (benchbuild.projects.benchbuild.tcc.TCC attribute), 25	NAME (benchbuild.projects.polybench.polybench.FDTD2D attribute), 34
NAME (benchbuild.projects.benchbuild.x264.X264 attribute), 26	NAME (benchbuild.projects.polybench.polybench.FloydWarshall attribute), 34
NAME (benchbuild.projects.benchbuild.xz.XZ attribute), 26	NAME (benchbuild.projects.polybench.polybench.Gemm attribute), 34
NAME (benchbuild.projects.gentoo.bzip2.BZip2 attribute), 27	NAME (benchbuild.projects.polybench.polybench.Gemver attribute), 34
	NAME (benchbuild.projects.polybench.polybench.Gesummv attribute), 34
	NAME (benchbuild.projects.polybench.polybench.Gramschmidt attribute), 34

attribute), 34  
 NAME (benchbuild.projects.polybench.polybench.Heat3D attribute), 34  
 NAME (benchbuild.projects.polybench.polybench.Jacobi1D attribute), 34  
 NAME (benchbuild.projects.polybench.polybench.Jacobi2D attribute), 34  
 NAME (benchbuild.projects.polybench.polybench.Lu attribute), 34  
 NAME (benchbuild.projects.polybench.polybench.LuDCM attribute), 34  
 NAME (benchbuild.projects.polybench.polybench.Mvt attribute), 34  
 NAME (benchbuild.projects.polybench.polybench.Nussinov attribute), 34  
 NAME (benchbuild.projects.polybench.polybench.Seidel2D attribute), 35  
 NAME (benchbuild.projects.polybench.polybench.Symm attribute), 35  
 NAME (benchbuild.projects.polybench.polybench.Syr2k attribute), 35  
 NAME (benchbuild.projects.polybench.polybench.Syrk attribute), 35  
 NAME (benchbuild.projects.polybench.polybench.ThreeMM attribute), 35  
 NAME (benchbuild.projects.polybench.polybench.Trisolv attribute), 35  
 NAME (benchbuild.projects.polybench.polybench.Trm attribute), 35  
 NAME (benchbuild.projects.polybench.polybench.TwoMM attribute), 35  
 NAME (benchbuild.utils.actions.Any attribute), 37  
 NAME (benchbuild.utils.actions.Build attribute), 37  
 NAME (benchbuild.utils.actions.Clean attribute), 37  
 NAME (benchbuild.utils.actions.CleanExtra attribute), 37  
 NAME (benchbuild.utils.actions.Configure attribute), 37  
 NAME (benchbuild.utils.actions.Download attribute), 37  
 NAME (benchbuild.utils.actions.Echo attribute), 37  
 NAME (benchbuild.utils.actions.Experiment attribute), 37  
 NAME (benchbuild.utils.actions.MakeBuildDir attribute), 37  
 NAME (benchbuild.utils.actions.Prepare attribute), 38  
 NAME (benchbuild.utils.actions.Run attribute), 38  
 NAME (benchbuild.utils.actions.Step attribute), 38  
 name (benchbuild.utils.container.Gentoo attribute), 41  
 name (benchbuild.utils.container.Ubuntu attribute), 41  
 name (benchbuild.utils.schema.CompileStat attribute), 49  
 name (benchbuild.utils.schema.Config attribute), 49  
 name (benchbuild.utils.schema.Event attribute), 49  
 name (benchbuild.utils.schema.Experiment attribute), 50  
 name (benchbuild.utils.schema.GlobalConfig attribute), 50  
 name (benchbuild.utils.schema.Metadata attribute), 50  
 name (benchbuild.utils.schema.Metric attribute), 50  
 name (benchbuild.utils.schema.PerfEvent attribute), 50  
 name (benchbuild.utils.schema.Project attribute), 51  
 name (benchbuild.utils.schema.RegressionTest attribute), 51  
 discipline() (in module benchbuild.experiment), 59  
 Nussinov (class in benchbuild.projects.polybench.polybench), 34  
 OK (benchbuild.utils.actions.StepResult attribute), 38  
 onerror() (benchbuild.utils.actions.Step method), 38  
 OpenBlas (class in benchbuild.projects.benchbuild.lapack), 19  
 opt\_flags() (benchbuild.test.BenchBuildTest method), 66  
 outfile() (benchbuild.report.BenchBuildReport method), 63  
 output\_file() (benchbuild.container.Container method), 55  
 outside() (benchbuild.projects.gentoo.postgresql.Postgresql method), 30  
 P  
 pack\_container() (in module benchbuild.container), 57  
 PapiScopCoverage (class in benchbuild.experiments.papi), 8  
 PapiStandardScopCoverage (class in benchbuild.experiments.papi), 8  
 path\_dict (benchbuild.projects.polybench.polybench.PolyBenchGroup attribute), 35  
 path\_suffix (benchbuild.projects.apollo.group.ApolloGroup attribute), 14  
 path\_suffix (benchbuild.projects.benchbuild.group.BenchBuildGroup attribute), 17  
 path\_to\_list() (in module benchbuild.utils.path), 45  
 PerfEvent (class in benchbuild.utils.schema), 50  
 persist\_calibration() (benchbuild.experiment.RuntimeExperiment method), 59  
 persist\_compilestats() (in module benchbuild.utils.db), 42  
 persist\_config() (in module benchbuild.utils.db), 42  
 persist\_experiment() (in module benchbuild.utils.db), 42  
 persist\_likwid() (in module benchbuild.utils.db), 42  
 persist\_perf() (in module benchbuild.utils.db), 42  
 persist\_project() (in module benchbuild.utils.db), 42  
 persist\_time() (in module benchbuild.utils.db), 42  
 phase() (in module benchbuild.experiment), 59  
 PJITlikwid (class in benchbuild.experiments.polyjit), 10  
 PJITpapi (class in benchbuild.experiments.polyjit), 10  
 PJITperf (class in benchbuild.experiments.polyjit), 10  
 PJITRaw (class in benchbuild.experiments.polyjit), 9  
 PJITRegression (class in benchbuild.experiments.polyjit), 9  
 Polly (class in benchbuild.experiments.polly.polly), 5



- PollyCompilestatsExperiment (class in benchbuild.experiments.compilestats), 7
- PollyOpenMP (class in benchbuild.experiments.polly.openmp), 5
- PollyOpenMPVectorizer (class in benchbuild.experiments.polly.openmpvect), 5
- PollyPerformance (class in benchbuild.experiments.polly.pollyperformance), 6
- PollyProfiling (class in benchbuild.driver), 57
- PollyVectorizer (class in benchbuild.experiments.polly.vectorize), 6
- PolyBenchGroup (class in benchbuild.projects.polybench.polybench), 35
- PolyJIT (class in benchbuild.experiments.polyjit), 10
- PolyJITFull (class in benchbuild.experiments.polyjit), 10
- populate\_projects() (benchbuild.experiment.Experiment method), 58
- PortageFactory() (in module benchbuild.projects.gentoo.portage\_gen), 30
- Postgres (class in benchbuild.projects.benchbuild.postgres), 22
- Postgresql (class in benchbuild.projects.gentoo.postgresql), 30
- Povray (class in benchbuild.projects.benchbuild.povray), 22
- Povray (class in benchbuild.projects.lnt.lnt), 32
- povray\_src\_dir (benchbuild.projects.lnt.lnt.Povray attribute), 32
- povray\_url (benchbuild.projects.lnt.lnt.Povray attribute), 32
- prefix() (benchbuild.test.BenchBuildTest method), 66
- Prepare (class in benchbuild.utils.actions), 37
- prepare() (benchbuild.project.Project method), 62
- prepare() (benchbuild.projects.apollo.rodinia.Rodinia method), 14
- prepare() (benchbuild.projects.apollo.scimark.SciMark method), 15
- prepare() (benchbuild.projects.benchbuild.bzip2.Bzip2 method), 15
- prepare() (benchbuild.projects.benchbuild.gzip.Gzip method), 18
- prepare() (benchbuild.projects.benchbuild.lammps.Lammps method), 19
- prepare() (benchbuild.projects.benchbuild.postgres.Postgres method), 22
- prepare() (benchbuild.projects.benchbuild.povray.Povray method), 23
- prepare() (benchbuild.projects.benchbuild.x264.X264 method), 26
- prepare() (benchbuild.projects.benchbuild.xz.XZ method), 26
- prepare() (benchbuild.projects.gentoo.bzip2.BZip2 method), 27
- prepare() (benchbuild.projects.gentoo.gzip.GZip method), 29
- prepare() (benchbuild.projects.gentoo.lammps.Lammps method), 29
- prepare() (benchbuild.projects.gentoo.x264.X264 method), 31
- prepare() (benchbuild.projects.gentoo.xz.XZ method), 31
- prepare\_directories() (in module benchbuild.utils.slurm), 52
- prepare\_slurm\_script() (in module benchbuild.utils.slurm), 52
- pretend (benchbuild.run.BenchBuildRun attribute), 63
- print\_logs() (in module benchbuild.log), 61
- print\_projects() (in module benchbuild.run), 64
- print\_runs() (in module benchbuild.log), 62
- project (benchbuild.utils.schema.RunGroup attribute), 52
- Project (class in benchbuild.project), 62
- Project (class in benchbuild.utils.schema), 51
- project() (benchbuild.log.BenchBuildLog method), 61
- project\_ids() (benchbuild.log.BenchBuildLog method), 61
- project\_name (benchbuild.utils.schema.RegressionTest attribute), 51
- project\_name (benchbuild.utils.schema.Run attribute), 51
- ProjectDecorator (class in benchbuild.project), 63
- ProjectRegistry (class in benchbuild.project), 63
- projects (benchbuild.project.ProjectRegistry attribute), 63
- projects() (benchbuild.run.BenchBuildRun method), 64
- projects() (benchbuild.slurm.Slurm method), 66
- provide\_package() (in module benchbuild.utils.bootstrap), 38
- provide\_packages() (in module benchbuild.utils.bootstrap), 38
- Python (class in benchbuild.projects.benchbuild.python), 23
- ## Q
- query\_yes\_no() (in module benchbuild.utils.user\_interface), 53
- ## R
- Rasdaman (class in benchbuild.projects.benchbuild.rasdaman), 23
- RawReport (class in benchbuild.reports.raw), 36
- RawRuntime (class in benchbuild.experiments.raw), 13
- read\_struct() (in module benchbuild.likwid), 61
- read\_structs() (in module benchbuild.likwid), 61
- read\_table() (in module benchbuild.likwid), 61
- read\_tables() (in module benchbuild.likwid), 61
- region (benchbuild.utils.schema.Likwid attribute), 50
- RegressionTest (class in benchbuild.utils.schema), 51
- remote (benchbuild.utils.container.Container attribute), 41
- remote (benchbuild.utils.container.Gentoo attribute), 41

- remote (benchbuild.utils.container.Ubuntu attribute), 41
- Report (class in benchbuild.reports), 36
- report() (benchbuild.reports.raw.RawReport method), 36
- ReportRegistry (class in benchbuild.reports), 36
- reports (benchbuild.reports.ReportRegistry attribute), 36
- RequireAll (class in benchbuild.utils.actions), 38
- Rodinia (class in benchbuild.projects.apollo.rodinia), 14
- Rsync() (in module benchbuild.utils.downloader), 43
- Ruby (class in benchbuild.projects.benchbuild.ruby), 24
- Run (class in benchbuild.utils.actions), 38
- Run (class in benchbuild.utils.schema), 51
- run() (benchbuild.container.BashStrategy method), 55
- run() (benchbuild.container.ContainerStrategy method), 56
- run() (benchbuild.container.SetupPolyJITGentooStrategy method), 56
- run() (benchbuild.experiments.papi.PapiScopCoverage method), 8
- run() (benchbuild.experiments.polyjit.PJITpapi method), 10
- run() (benchbuild.project.Project method), 62
- run() (in module benchbuild.utils.run), 47
- run\_group (benchbuild.utils.schema.Run attribute), 51
- run\_id (benchbuild.utils.schema.CompileStat attribute), 49
- run\_id (benchbuild.utils.schema.Config attribute), 49
- run\_id (benchbuild.utils.schema.Event attribute), 49
- run\_id (benchbuild.utils.schema.Likwid attribute), 50
- run\_id (benchbuild.utils.schema.Metadata attribute), 50
- run\_id (benchbuild.utils.schema.Metric attribute), 50
- run\_id (benchbuild.utils.schema.PerfEvent attribute), 50
- run\_id (benchbuild.utils.schema.RegressionTest attribute), 51
- run\_id (benchbuild.utils.schema.RunLog attribute), 52
- run\_in\_container() (in module benchbuild.container), 57
- run\_project() (benchbuild.experiments.polly.openmpvect.PollyOpenMPVectorizer method), 5
- run\_project() (benchbuild.experiments.polly.polly.Polly method), 5
- run\_project() (benchbuild.experiments.polly.pollyperformance.PollyPerformance method), 6
- run\_project() (benchbuild.experiments.polly.vectorize.PollyVectorizer method), 6
- run\_raw() (in module benchbuild.experiments.polyjit), 11
- run\_tests() (benchbuild.project.Project method), 62
- run\_tests() (benchbuild.projects.apollo.rodinia.Rodinia method), 14
- run\_tests() (benchbuild.projects.apollo.scimark.SciMark method), 15
- run\_tests() (benchbuild.projects.benchbuild.bzip2.Bzip2 method), 15
- run\_tests() (benchbuild.projects.benchbuild.ccrypt.Ccrypt method), 16
- run\_tests() (benchbuild.projects.benchbuild.crafty.Crafty method), 16
- run\_tests() (benchbuild.projects.benchbuild.croccat.Croccat method), 17
- run\_tests() (benchbuild.projects.benchbuild.ffmpeg.LibAV method), 17
- run\_tests() (benchbuild.projects.benchbuild.gzip.Gzip method), 18
- run\_tests() (benchbuild.projects.benchbuild.js.SpiderMonkey method), 18
- run\_tests() (benchbuild.projects.benchbuild.lammps.Lammps method), 19
- run\_tests() (benchbuild.projects.benchbuild.lapack.Lapack method), 19
- run\_tests() (benchbuild.projects.benchbuild.lapack.OpenBlas method), 19
- run\_tests() (benchbuild.projects.benchbuild.leveldb.LevelDB method), 20
- run\_tests() (benchbuild.projects.benchbuild.lulesh.Lulesh method), 20
- run\_tests() (benchbuild.projects.benchbuild.luleshomp.LuleshOMP method), 21
- run\_tests() (benchbuild.projects.benchbuild.mcrypt.MCrypt method), 21
- run\_tests() (benchbuild.projects.benchbuild.minisat.Minisat method), 22
- run\_tests() (benchbuild.projects.benchbuild.openssl.LibreSSL method), 22
- run\_tests() (benchbuild.projects.benchbuild.postgres.Postgres method), 22
- run\_tests() (benchbuild.projects.benchbuild.povray.Povray method), 23
- run\_tests() (benchbuild.projects.benchbuild.python.Python method), 23
- run\_tests() (benchbuild.projects.benchbuild.rasdaman.Rasdaman method), 24
- run\_tests() (benchbuild.projects.benchbuild.ruby.Ruby method), 24
- run\_tests() (benchbuild.projects.benchbuild.sdcc.SDCC method), 24
- run\_tests() (benchbuild.projects.benchbuild.sevenz.SevenZip method), 25
- run\_tests() (benchbuild.projects.benchbuild.sqlite3.SQLite3 method), 25
- run\_tests() (benchbuild.projects.benchbuild.tcc.TCC method), 25
- run\_tests() (benchbuild.projects.benchbuild.x264.X264 method), 26
- run\_tests() (benchbuild.projects.benchbuild.xz.XZ method), 26
- run\_tests() (benchbuild.projects.gentoo.autoportage.AutoPortage method), 27
- run\_tests() (benchbuild.projects.gentoo.bzip2.BZip2 method), 27

[run\\_tests\(\)](#) (benchbuild.projects.gentoo.crafty.Crafty method), [28](#)  
[run\\_tests\(\)](#) (benchbuild.projects.gentoo.eix.Eix method), [28](#)  
[run\\_tests\(\)](#) (benchbuild.projects.gentoo.gzip.GZip method), [29](#)  
[run\\_tests\(\)](#) (benchbuild.projects.gentoo.lammps.Lammps method), [29](#)  
[run\\_tests\(\)](#) (benchbuild.projects.gentoo.postgresql.PostgreSQL method), [31](#)  
[run\\_tests\(\)](#) (benchbuild.projects.gentoo.sevenz.SevenZip method), [31](#)  
[run\\_tests\(\)](#) (benchbuild.projects.gentoo.x264.X264 method), [31](#)  
[run\\_tests\(\)](#) (benchbuild.projects.gentoo.xz.XZ method), [31](#)  
[run\\_tests\(\)](#) (benchbuild.projects.Int.Int.MultiSourceApplications method), [32](#)  
[run\\_tests\(\)](#) (benchbuild.projects.Int.Int.MultiSourceBenchmarks method), [32](#)  
[run\\_tests\(\)](#) (benchbuild.projects.Int.Int.Povray method), [32](#)  
[run\\_tests\(\)](#) (benchbuild.projects.Int.Int.SingleSourceBenchmarks method), [33](#)  
[run\\_tests\(\)](#) (benchbuild.projects.Int.Int.SPEC2006 method), [33](#)  
[run\\_uuid](#) (benchbuild.project.Project attribute), [62](#)  
[run\\_with\\_likwid\(\)](#) (in module benchbuild.experiments.polyjit), [11](#)  
[run\\_with\\_papi\(\)](#) (in module benchbuild.experiments.polyjit), [11](#)  
[run\\_with\\_perf\(\)](#) (in module benchbuild.experiments.polyjit), [12](#)  
[run\\_with\\_time\(\)](#) (in module benchbuild.experiments.polyjit), [12](#)  
[run\\_with\\_time\(\)](#) (in module benchbuild.experiments.raw), [13](#)  
[run\\_without\\_recompile\(\)](#) (in module benchbuild.experiments.polyjit), [12](#)  
[RunGroup](#) (class in benchbuild.utils.schema), [51](#)  
[RunLog](#) (class in benchbuild.utils.schema), [52](#)  
[runtime\\_extension](#) (benchbuild.project.Project attribute), [63](#)  
[RuntimeExperiment](#) (class in benchbuild.experiment), [58](#)

## S

[SciMark](#) (class in benchbuild.projects.apollo.scimark), [15](#)  
[SDCC](#) (class in benchbuild.projects.benchbuild.sdcc), [24](#)  
[Seidel2D](#) (class in benchbuild.projects.polybench.polybench), [35](#)  
[session](#) (benchbuild.reports.raw.RawReport attribute), [36](#)  
[SessionManager](#) (class in benchbuild.utils.schema), [52](#)  
[set\\_defaults\(\)](#) (in module benchbuild.utils.log), [44](#)  
[set\\_input\\_container\(\)](#) (in module benchbuild.container), [57](#)  
[setup\\_bash\\_in\\_container\(\)](#) (in module benchbuild.container), [57](#)  
[setup\\_container\(\)](#) (in module benchbuild.container), [57](#)  
[setup\\_derived\\_filenames\(\)](#) (benchbuild.project.Project method), [63](#)  
[setup\\_directories\(\)](#) (in module benchbuild.container), [57](#)  
[SetupPolyJITGentooStrategy](#) (class in benchbuild.container), [56](#)  
[SevenZip](#) (class in benchbuild.projects.benchbuild.sevenz), [24](#)  
[SevenZip](#) (class in benchbuild.projects.gentoo.sevenz), [31](#)  
[shell\(\)](#) (benchbuild.container.Container method), [55](#)  
[ShouldNotBeNone](#), [6](#)  
[show\\_config](#) (benchbuild.run.BenchBuildRun attribute), [64](#)  
[SingleSourceBenchmarks](#) (class in benchbuild.projects.Int.Int), [33](#)  
[Slurm](#) (class in benchbuild.slurm), [66](#)  
[source\\_required\(\)](#) (in module benchbuild.utils.downloader), [44](#)  
[SPEC2006](#) (class in benchbuild.projects.Int.Int), [33](#)  
[SpiderMonkey](#) (class in benchbuild.projects.benchbuild.js), [18](#)  
[SQLite3](#) (class in benchbuild.projects.benchbuild.sqlite3), [25](#)  
[src\\_dir](#) (benchbuild.projects.apollo.rodinia.Rodinia attribute), [14](#)  
[src\\_dir](#) (benchbuild.projects.benchbuild.bzip2.Bzip2 attribute), [15](#)  
[src\\_dir](#) (benchbuild.projects.benchbuild.ccrypt.Ccrypt attribute), [16](#)  
[src\\_dir](#) (benchbuild.projects.benchbuild.crafty.Crafty attribute), [16](#)  
[src\\_dir](#) (benchbuild.projects.benchbuild.croccpat.Croccpat attribute), [17](#)  
[src\\_dir](#) (benchbuild.projects.benchbuild.ffmpeg.LibAV attribute), [17](#)  
[src\\_dir](#) (benchbuild.projects.benchbuild.gzip.Gzip attribute), [18](#)  
[src\\_dir](#) (benchbuild.projects.benchbuild.js.SpiderMonkey attribute), [18](#)  
[src\\_dir](#) (benchbuild.projects.benchbuild.lammps.Lammps attribute), [19](#)  
[src\\_dir](#) (benchbuild.projects.benchbuild.lapack.Lapack attribute), [19](#)  
[src\\_dir](#) (benchbuild.projects.benchbuild.mcrypt.MCrypt attribute), [21](#)  
[src\\_dir](#) (benchbuild.projects.benchbuild.openssl.LibSSL attribute), [22](#)  
[src\\_dir](#) (benchbuild.projects.benchbuild.python.Python attribute), [23](#)  
[src\\_dir](#) (benchbuild.projects.benchbuild.ruby.Ruby



attribute), 24

src\_dir (benchbuild.projects.benchbuild.sevenz.SevenZip attribute), 25

src\_dir (benchbuild.projects.benchbuild.sqlite3.SQLite3 attribute), 25

src\_dir (benchbuild.projects.benchbuild.tcc.TCC attribute), 25

src\_dir (benchbuild.projects.benchbuild.xz.XZ attribute), 26

src\_dir (benchbuild.projects.lnt.lnt.LNTGroup attribute), 32

src\_dir (benchbuild.projects.polybench.polybench.PolyBenchGroup attribute), 35

SRC\_FILE (benchbuild.project.Project attribute), 62

SRC\_FILE (benchbuild.projects.apollo.rodinia.Rodinina attribute), 14

SRC\_FILE (benchbuild.projects.apollo.scimark.SciMark attribute), 15

SRC\_FILE (benchbuild.projects.benchbuild.bzip2.Bzip2 attribute), 15

SRC\_FILE (benchbuild.projects.benchbuild.ccrypt.Ccrypt attribute), 16

SRC\_FILE (benchbuild.projects.benchbuild.crafty.Crafty attribute), 16

SRC\_FILE (benchbuild.projects.benchbuild.croccat.Croccat attribute), 16

SRC\_FILE (benchbuild.projects.benchbuild.ffmpeg.LibAV attribute), 17

SRC\_FILE (benchbuild.projects.benchbuild.gzip.Gzip attribute), 18

SRC\_FILE (benchbuild.projects.benchbuild.lammps.Lammps attribute), 18

SRC\_FILE (benchbuild.projects.benchbuild.lapack.Lapack attribute), 19

SRC\_FILE (benchbuild.projects.benchbuild.lapack.OpenBlas attribute), 19

SRC\_FILE (benchbuild.projects.benchbuild.leveldb.LevelDB attribute), 19

SRC\_FILE (benchbuild.projects.benchbuild.lulesh.Lulesh attribute), 20

SRC\_FILE (benchbuild.projects.benchbuild.luleshomp.LuleshOMP attribute), 21

SRC\_FILE (benchbuild.projects.benchbuild.mcrypt.MCrypt attribute), 21

SRC\_FILE (benchbuild.projects.benchbuild.minisat.Minisat attribute), 21

SRC\_FILE (benchbuild.projects.benchbuild.openssl.LibreSSL attribute), 22

SRC\_FILE (benchbuild.projects.benchbuild.povray.Povray attribute), 23

SRC\_FILE (benchbuild.projects.benchbuild.python.Python attribute), 23

SRC\_FILE (benchbuild.projects.benchbuild.rasdaman.Rasdaman attribute), 23

SRC\_FILE (benchbuild.projects.benchbuild.ruby.Ruby attribute), 24

SRC\_FILE (benchbuild.projects.benchbuild.sdcc.SDCC attribute), 24

SRC\_FILE (benchbuild.projects.benchbuild.sevenz.SevenZip attribute), 25

SRC\_FILE (benchbuild.projects.benchbuild.sqlite3.SQLite3 attribute), 25

SRC\_FILE (benchbuild.projects.benchbuild.tcc.TCC attribute), 25

SRC\_FILE (benchbuild.projects.benchbuild.x264.X264 attribute), 26

SRC\_FILE (benchbuild.projects.benchbuild.xz.XZ attribute), 26

SRC\_FILE (benchbuild.projects.gentoo.gentoo.GentooGroup attribute), 28

SRC\_FILE (benchbuild.projects.polybench.polybench.PolyBenchGroup attribute), 35

src\_uri (benchbuild.projects.apollo.rodinia.Rodinina attribute), 14

src\_uri (benchbuild.projects.apollo.scimark.SciMark attribute), 15

src\_uri (benchbuild.projects.benchbuild.bzip2.Bzip2 attribute), 15

src\_uri (benchbuild.projects.benchbuild.ccrypt.Ccrypt attribute), 16

src\_uri (benchbuild.projects.benchbuild.crafty.Crafty attribute), 16

src\_uri (benchbuild.projects.benchbuild.croccat.Croccat attribute), 17

src\_uri (benchbuild.projects.benchbuild.ffmpeg.LibAV attribute), 17

src\_uri (benchbuild.projects.benchbuild.gzip.Gzip attribute), 18

src\_uri (benchbuild.projects.benchbuild.js.SpiderMonkey attribute), 18

src\_uri (benchbuild.projects.benchbuild.lammps.Lammps attribute), 19

src\_uri (benchbuild.projects.benchbuild.lapack.Lapack attribute), 19

src\_uri (benchbuild.projects.benchbuild.lapack.OpenBlas attribute), 19

src\_uri (benchbuild.projects.benchbuild.leveldb.LevelDB attribute), 20

src\_uri (benchbuild.projects.benchbuild.linpack.Linpack attribute), 20

src\_uri (benchbuild.projects.benchbuild.lulesh.Lulesh attribute), 20

src\_uri (benchbuild.projects.benchbuild.luleshomp.LuleshOMP attribute), 21

src\_uri (benchbuild.projects.benchbuild.mcrypt.MCrypt attribute), 21

src\_uri (benchbuild.projects.benchbuild.minisat.Minisat attribute), 22

src\_uri (benchbuild.projects.benchbuild.openssl.LibreSSL attribute), 22

src\_uri (benchbuild.projects.benchbuild.povray.Povray attribute), 23

src\_uri (benchbuild.projects.benchbuild.python.Python attribute), 23

src\_uri (benchbuild.projects.benchbuild.rasdaman.Rasdaman attribute), 24

src\_uri (benchbuild.projects.benchbuild.ruby.Ruby attribute), 24

src\_uri (benchbuild.projects.benchbuild.sdcc.SDCC attribute), 24

src\_uri (benchbuild.projects.benchbuild.sevenz.SevenZip attribute), 25

src\_uri (benchbuild.projects.benchbuild.sqlite3.SQLite3 attribute), 25

src\_uri (benchbuild.projects.benchbuild.tcc.TCC attribute), 26

src\_uri (benchbuild.projects.benchbuild.x264.X264 attribute), 26

src\_uri (benchbuild.projects.benchbuild.xz.XZ attribute), 26

src\_uri (benchbuild.projects.Int.Int.LNTGroup attribute), 32

src\_uri (benchbuild.projects.polybench.polybench.PolyBenchGroup attribute), 35

src\_url (benchbuild.utils.schema.Project attribute), 51

start (benchbuild.utils.schema.Event attribute), 49

start (benchbuild.utils.schema.PerfEvent attribute), 50

static\_var() (in module benchbuild.experiment), 59

status (benchbuild.utils.schema.Run attribute), 51

status (benchbuild.utils.schema.RunGroup attribute), 52

status (benchbuild.utils.schema.RunLog attribute), 52

stderr (benchbuild.utils.schema.RunLog attribute), 52

stdout (benchbuild.utils.schema.RunLog attribute), 52

Step (class in benchbuild.utils.actions), 38

step() (in module benchbuild.experiment), 60

StepClass (class in benchbuild.utils.actions), 38

StepResult (class in benchbuild.utils.actions), 38

store() (benchbuild.settings.Configuration method), 65

store\_config (benchbuild.bootstrap.BenchBuildBootstrap attribute), 55

store\_config (benchbuild.run.BenchBuildRun attribute), 64

store\_config() (in module benchbuild.utils.run), 47

strategy() (benchbuild.container.ContainerCreate method), 56

strip\_path\_prefix() (in module benchbuild.utils.wrapping), 53

substep() (in module benchbuild.experiment), 60

SUPPORTED\_EXPERIMENTS (benchbuild.reports.raw.RawReport attribute), 36

SUPPORTED\_EXPERIMENTS (benchbuild.reports.Report attribute), 36

Svn() (in module benchbuild.utils.downloader), 43

Symm (class in benchbuild.projects.polybench.polybench), 35

Syr2k (class in benchbuild.projects.polybench.polybench), 35

Syrk (class in benchbuild.projects.polybench.polybench), 35

## T

TCC (class in benchbuild.projects.benchbuild.tcc), 25

template\_str() (in module benchbuild.utils.path), 45

Test (class in benchbuild.experiments.pjtest), 8

test\_archive (benchbuild.projects.gentoo.bzip2.BZip2 attribute), 27

test\_archive (benchbuild.projects.gentoo.gzip.GZip attribute), 29

test\_archive (benchbuild.projects.gentoo.lammps.Lammps attribute), 29

test\_archive (benchbuild.projects.gentoo.xz.XZ attribute), 32

test\_suite\_dir (benchbuild.projects.Int.Int.LNTGroup attribute), 32

test\_suite\_uri (benchbuild.projects.Int.Int.LNTGroup attribute), 32

test\_group (benchbuild.projects.gentoo.bzip2.BZip2 attribute), 27

test\_url (benchbuild.projects.gentoo.gzip.GZip attribute), 29

test\_url (benchbuild.projects.gentoo.lammps.Lammps attribute), 29

test\_url (benchbuild.projects.gentoo.x264.X264 attribute), 31

test\_url (benchbuild.projects.gentoo.xz.XZ attribute), 32

testfiles (benchbuild.projects.benchbuild.bzip2.Bzip2 attribute), 15

testfiles (benchbuild.projects.benchbuild.gzip.Gzip attribute), 18

testfiles (benchbuild.projects.benchbuild.postgres.Postgres attribute), 22

testfiles (benchbuild.projects.benchbuild.xz.XZ attribute), 26

testfiles (benchbuild.projects.gentoo.bzip2.BZip2 attribute), 27

testfiles (benchbuild.projects.gentoo.gzip.GZip attribute), 29

testfiles (benchbuild.projects.gentoo.xz.XZ attribute), 32

ThreeMM (class in benchbuild.projects.polybench.polybench), 35

tid (benchbuild.utils.schema.Event attribute), 49

tid (benchbuild.utils.schema.PerfEvent attribute), 51

time\_polyjit\_and\_polly() (in module benchbuild.experiments.pjtest), 9

to\_env\_dict() (in module benchbuild.settings), 66

to\_step\_result() (in module benchbuild.utils.actions), 38

- Trisolv (class in benchbuild.projects.polybench.polybench), 35
  - Trmm (class in benchbuild.projects.polybench.polybench), 35
  - TwoMM (class in benchbuild.projects.polybench.polybench), 35
  - type (benchbuild.utils.schema.Event attribute), 49
  - type (benchbuild.utils.schema.PerfEvent attribute), 51
- ## U
- Ubuntu (class in benchbuild.utils.container), 41
  - uchroot() (in module benchbuild.utils.run), 47
  - uchroot\_env() (in module benchbuild.utils.run), 47
  - uchroot\_mounts() (in module benchbuild.utils.run), 47
  - uchroot\_no\_args() (in module benchbuild.utils.run), 48
  - uchroot\_no\_llvm() (in module benchbuild.utils.run), 48
  - uchroot\_with\_mounts() (in module benchbuild.utils.run), 48
  - unionfs() (in module benchbuild.utils.run), 48
  - unionfs\_set\_up() (in module benchbuild.utils.run), 48
  - unionfs\_tear\_down() (in module benchbuild.utils.run), 48
  - unpack\_container() (in module benchbuild.utils.container), 41
  - update() (benchbuild.settings.Configuration method), 65
  - update\_env() (in module benchbuild.settings), 66
  - update\_hash() (in module benchbuild.utils.downloader), 44
  - UUIDEncoder (class in benchbuild.settings), 65
- ## V
- value (benchbuild.utils.schema.CompileStat attribute), 49
  - value (benchbuild.utils.schema.Config attribute), 49
  - value (benchbuild.utils.schema.GlobalConfig attribute), 50
  - value (benchbuild.utils.schema.Likwid attribute), 50
  - value (benchbuild.utils.schema.Metadata attribute), 50
  - value (benchbuild.utils.schema.Metric attribute), 50
  - value() (benchbuild.settings.Configuration method), 65
  - verbosity (benchbuild.container.Container attribute), 55
  - verbosity (benchbuild.driver.PollyProfiling attribute), 57
  - VERSION (benchbuild.container.Container attribute), 55
  - VERSION (benchbuild.driver.PollyProfiling attribute), 57
  - VERSION (benchbuild.project.Project attribute), 62
  - VERSION (benchbuild.projects.apollo.rodinia.Rodinia attribute), 14
  - VERSION (benchbuild.projects.apollo.scimark.SciMark attribute), 15
  - VERSION (benchbuild.projects.benchbuild.bzip2.Bzip2 attribute), 15
  - VERSION (benchbuild.projects.benchbuild.ccrypt.Ccrypt attribute), 16
  - VERSION (benchbuild.projects.benchbuild.crafty.Crafty attribute), 16
  - VERSION (benchbuild.projects.benchbuild.crocpat.Crocpat attribute), 17
  - VERSION (benchbuild.projects.benchbuild.ffmpeg.LibAV attribute), 17
  - VERSION (benchbuild.projects.benchbuild.gzip.Gzip attribute), 18
  - VERSION (benchbuild.projects.benchbuild.js.SpiderMonkey attribute), 18
  - version (benchbuild.projects.benchbuild.js.SpiderMonkey attribute), 18
  - VERSION (benchbuild.projects.benchbuild.lapack.Lapack attribute), 19
  - VERSION (benchbuild.projects.benchbuild.mcrypt.MCrypt attribute), 21
  - VERSION (benchbuild.projects.benchbuild.openssl.LibreSSL attribute), 22
  - VERSION (benchbuild.projects.benchbuild.python.Python attribute), 23
  - VERSION (benchbuild.projects.benchbuild.ruby.Ruby attribute), 24
  - VERSION (benchbuild.projects.benchbuild.sevenz.SevenZip attribute), 25
  - VERSION (benchbuild.projects.benchbuild.tcc.TCC attribute), 25
  - VERSION (benchbuild.projects.benchbuild.xz.XZ attribute), 26
  - VERSION (benchbuild.projects.gentoo.bzip2.BZip2 attribute), 27
  - VERSION (benchbuild.projects.Int.Int.LNTGroup attribute), 32
  - VERSION (benchbuild.projects.polybench.polybench.PolyBenchGroup attribute), 35
  - version (benchbuild.utils.schema.Project attribute), 51
- ## W
- Wget() (in module benchbuild.utils.downloader), 44
  - with\_env\_recursive() (in module benchbuild.utils.run), 48
  - wrap() (in module benchbuild.utils.wrapping), 54
  - wrap\_cc() (in module benchbuild.utils.wrapping), 54
  - wrap\_cc\_in\_uchroot() (in module benchbuild.utils.compiler), 40
  - wrap\_cxx\_in\_uchroot() (in module benchbuild.utils.compiler), 40
  - wrap\_dynamic() (in module benchbuild.utils.wrapping), 54
  - wrap\_dynamic\_in\_uchroot() (in module benchbuild.utils.wrapping), 55
  - wrap\_in\_uchroot() (in module benchbuild.utils.wrapping), 55
  - write\_bashrc() (benchbuild.container.SetupPolyJITGentooStrategy method), 56
  - write\_bashrc() (benchbuild.projects.gentoo.gentoo.GentooGroup method), 28

`write_layout()` (`benchbuild.container.SetupPolyJITGentooStrategy`  
method), [56](#)  
`write_layout()` (`benchbuild.projects.gentoo.gentoo.GentooGroup`  
method), [28](#)  
`write_makeconfig()` (`bench-`  
`build.container.SetupPolyJITGentooStrategy`  
method), [56](#)  
`write_makeconfig()` (`bench-`  
`build.projects.gentoo.gentoo.GentooGroup`  
method), [28](#)  
`write_wgetrc()` (`benchbuild.container.SetupPolyJITGentooStrategy`  
method), [56](#)  
`write_wgetrc()` (`benchbuild.projects.gentoo.gentoo.GentooGroup`  
method), [28](#)

## X

`X264` (class in `benchbuild.projects.benchbuild.x264`), [26](#)  
`X264` (class in `benchbuild.projects.gentoo.x264`), [31](#)  
`XZ` (class in `benchbuild.projects.benchbuild.xz`), [26](#)  
`XZ` (class in `benchbuild.projects.gentoo.xz`), [31](#)