NAME

kyua list - Lists test cases and their metadata

SYNOPSIS

kyua list [**--build-root** *path*] [**--kyuafile** *file*] [**--verbose**] *test_case1* [*.. test_caseN*]

DESCRIPTION

The **kyua list** command scans all the test programs and test cases in a test suite (as defined by a kyuafile(5)) and prints a list of all their names, optionally accompanied by any metadata properties they have.

The optional arguments to **kyua list** are used to select which test programs or test cases to run. These are filters and are described below in *Test filters*.

This command must be run within a test suite or a test suite must be provided with the **--kyuafile** flag.

The following subcommand options are recognized:

--build-root *path*
Specifies the build root in which to find the test programs referenced by the Kyuafile, if different from the Kyuafile's directory. See *Build directories* below for more information.

--kyuafile *path*, **-k** *path*
Specifies the Kyuafile to process. Defaults to a *Kyuafile* file in the current directory.

--verbose, **-v**
Prints metadata properties for every test case.

Build directories

Build directories (or object directories, target directories, product directories, etc.) is the concept that allows a developer to keep the source tree clean from build products by asking the build system to place such build products under a separate subtree.

Most build systems today support build directories. For example, the GNU Automake/Autoconf build system exposes such concept when invoked as follows:

```
$ cd my-project-1.0
$ mkdir build
$ cd build
$ ../configure
```

```
$ make
```

Under such invocation, all the results of the build are left in the *my-project-1.0/build/* subdirectory while maintaining the contents of *my-project-1.0/* intact.

Because build directories are an integral part of most build systems, and because they are a tool that developers use frequently, **kyua list** supports build directories too. This manifests in the form of **kyua list** being able to run tests from build directories while reading the (often immutable) test suite definition from the source tree.

One important property of build directories is that they follow (or need to follow) the exact same layout as the source tree. For example, consider the following directory listings:

```
src/Kyuafile
src/bin/ls/
src/bin/ls/Kyuafile
src/bin/ls/ls.c
src/bin/ls/ls_test.c
src/sbin/su/
src/sbin/su/Kyuafile
src/sbin/su/su.c
src/sbin/su/su_test.c

obj/bin/ls/
obj/bin/ls/ls*
obj/bin/ls/ls_test*
obj/sbin/su/
obj/sbin/su/su*
obj/sbin/su/su_test*
```

Note how the directory layout within *src/* matches that of *obj/*. The *src/* directory contains only source files and the definition of the test suite (the Kyuafiles), while the *obj/* directory contains only the binaries generated during a build.

All commands that deal with the workspace support the **--build-root** *path* option. When this option is provided, the directory specified by the option is considered to be the root of the build directory. For example, considering our previous fake tree layout, we could invoke **kyua list** as any of the following:

```
$ kyua list --kyuafile=src/Kyuafile --build-root=obj
$ cd src && kyua list --build-root=../obj
```

Test filters

A *test filter* is a string that is used to match test cases or test programs in a test suite. Filters have the following form:

test_program_name[:test_case_name]

Where ‘test_program_name’ is the name of a test program or a subdirectory in the test suite, and ‘test_case_name’ is the name of a test case.

EXIT STATUS

The **kyua list** command returns 0 on success or 1 if any of the given test case filters does not match any test case.

Additional exit codes may be returned as described in kyua(1).

SEE ALSO

kyua(1), kyuafile(5)