Tools testing

Running end-to-end tests happens through the Self-Test. To run the tests:

```
./meteor self-test <regexp>
```

A very-very useful environment variable to set, in case you are running on a slow machine:

```
# set the multiplier for time-outs
set TIMEOUT_SCALE_FACTOR=3
```

Before you run your tests you can preview which ones will run and which ones will be skipped with `--preview` flag.

You can also use `--skip` or `--limit` . It's different from regex filter, we are not filtering anything, we are just skipping or limiting or both after the filter was already applied. This is helpful when you want to run in batches but you don't care about specific tests to be skipped or not.

Writing tests

All tests are currently stored at `/tools/tests/`, each JS file can register a self-test. Example:

```
selftest.define("mongo failover", [/* tags */], function () {
  var s = new Sandbox();
  s.set('METEOR_TEST_MULTIPLE_MONGOD_REPLSET', 't');
  s.createApp("failover-test", "failover-test");
  s.cd("failover-test");

  var run = s.run("--once", "--raw-logs");
  run.waitSecs(120);
  run.match("SUCCESS\n");
  run.expectEnd();
  run.expectExit(0);
});
```

The example above demonstrates how to define a test, create a Sandbox, create an app from a template and run the Meteor commands.

Templates for apps and packages are kept in `/tools/tests`, too.

Testing with Phantom/Browserstack

The sandbox has a `testWithAllClients` method that runs the clients like Phantom or Browserstack pointed to the page of the app (`localhost:3000` by default).

Tags

Tags are arbitrary. To make tags do anything, you should edit the `selftest.js` code.

Examples of some tags that exist today:

- `slow` - the test is skipped, unless the `--slow` flag is passed
- `windows` - the test is not run unless on Windows
- `net` - the test is talking to external Internet services, thus requires an Internet connection to run

There are others.

Self-test gotchas

- The docs for self-test is reading the code of self-test
- `run.forbid(regexp)` forbids the regexp from the entire output, not from the point it was called. It happens, because the output is matched asynchronously.