Perl
Post Install Tests

April 10, 2013

Upgrading dependencies w/fear

1. Install a module named `Bear`
2. Install the module `Human`, which depends on the availability of `Bear`'s `ride()` method
3. (time passes...)
4. Upgrade to a newer version of `Bear`, which does NOT have the `ride()` method; The authhor of `Bear` learned it's not safe to ride bears.
5. *boom* `Human`'s attempt to `ride()` fails!
6. `Human` gets eaten by `Bear` and/or customer.

Sources of *upgrade fear*

- Don't assume CPAN authors have a commitment to their APIs. They may change at any time - even if the CPAN/Perl community has a history of preserving backwards compatibility!
- No *trivial* way of verifying if the installed Perl/CPAN dists still work.
- Modules might be used in ways unintended by the author.

Detecting an upgrade bug/bear early, is *good*.

Detecting breakage

We have the following options:

- Run your code and see
- Run integration/smoke tests for YOUR code
- Check cpantesters.org

We want more options for detecting breakage!

- Run tests for all packages found in `@INC`
- Run tests for the distributions that depend on the distribution you're about to upgrade
- Run tests in any runtime environment (development, test/CI, staging and if necessary, **production**)

Post Install Tests Requirements

- The tests for the installed modules must be available (installed)
- It must be possible to locate tests for a specific distribution and version
- In order to run the tests for only the dependent distributions it must be possible to do determine this distribution's reverse dependencies
- Distribution dependencies are saved and installed
- Writing a best practices guide for tests

Demo - Module::Build::PIT

- Extension of `Module::Build`
- Environment variable `PERL_INSTALL_TESTS` controls if tests should be installed when using `./Build install` (Also possible to use `./Build installtests` without the ENV)
- Tests is installed under `$install_base/auto/tests/$distname-$distversion/`
- Test files are written to the `.packlist` file
- Action `testinc` uses the `ExtUtils::Installed` distribution to retrieve the modules that would be loaded and their test files
- Action `testrdeps` is a fake action. The logic is hardcoded for demonstration purposes

Future Work / Considerations

- Permissions - Test that writes to devices, sockets(ports $<$ 1024), t/... and more
- Currently no trivial way of finding reverse dependencies locally
- Integration with packaging systems
- Upgrading `test_requires` to regular `requires`?
- Best practices doc

Code

- Code is on github!
- `git clone https://github.com/bannaN/Module-Build-PIT.git`
- `git clone https://github.com/bannaN/qa-hackathon-presentation.git`
- Fork it now!