rustc-workspace-hack

This crate is a bit of a hack to make workspaces in rustc work a bit better. The rationale for this existence is a bit subtle, but the general idea is that we want commands like `./x.py build src/tools/{rls,clippy,cargo}` to share as many dependencies as possible.

Each invocation is a different invocation of Cargo, however. Each time Cargo runs a build it will re-resolve the dependency graph, notably selecting different features sometimes for each build.

For example, let's say there's a very deep dependency like `num-traits` in each of these builds. For Cargo the `num-traits`'s `default` feature is turned off. In RLS, however, the `default` feature is turned. This means that building Cargo and then the RLS will actually build Cargo twice (as a transitive dependency changed). This is bad!

The goal of this crate is to solve this problem and ensure that the resolved dependency graph for all of these tools is the same in the various subsets of each tool, notably enabling the same features of transitive dependencies.

All tools vendored here depend on the `rustc-workspace-hack` crate on crates.io. When on crates.io this crate is an empty crate that is just a noop. We override it, however, in this workspace to this crate here, which means we can control crates in the dependency graph for each of these tools.