Stable, nightly, and pinning a version

In PHP

Teams pin PHP so CI and every laptop agree. A `.php-version` file names the version for the local switcher. `composer.json` can set `config.platform.php` so Composer resolves packages as if that PHP version is installed. The Docker image tag (`php:8.3-cli` , for example) pins the version that actually runs in CI. None of these float to whatever PHP happens to be newest.

In Rust

`rust-toolchain.toml` at the repo root pins the toolchain for everyone who builds the project. rustup reads it and installs that toolchain automatically when someone runs `cargo` in the directory.

rust-toolchain.toml

```
 [toolchain]
channel = "1.94"
components = ["rustfmt", "clippy"]
```

Use the current stable at the time you pin (check `rustc --version`); `1.94` here is only an example, not a version to copy forever. A specific stable version is the usual choice for a team. The channel `"stable"` floats: rustup moves it forward every six weeks, so two laptops can be on different compilers until both update. `1.94` stays `1.94` until you edit the file. `components` installs `rustfmt` and `clippy` with that toolchain, so `cargo fmt` and `cargo clippy` work without a separate step.

If rustup auto-install is turned off (`RUSTUP_AUTO_INSTALL=0`), the pin will not download itself. Run `rustup toolchain install 1.94` (or whatever channel you pinned) once, then `cargo` again.

Nightly is only for an unstable feature a crate's docs explicitly need. Use `rustup override set nightly` in that project's directory, and `rustup override unset` when you

are done. Never run `rustup default nightly`. That changes the compiler for every project that does not pin its own toolchain. Async and procedural macros are stable (async since Rust 1.39, procedural macros since Rust 1.30). They are not a reason to leave stable.

> **Watch for:** `channel = "stable"` in a team repo. It floats, so CI and a laptop can disagree. `rustup default nightly` copied from a gist is worse: the next `cargo build` in an unrelated project follows nightly too.

> **Idiomatic Rust:** commit `rust-toolchain.toml` with a specific stable version, including `rustfmt` and `clippy`. Nightly only when a crate's docs name an unstable feature you have decided to use.

Side by side

	PHP	Rust
Pin the version	`.php-version`, or a Docker tag	`rust-toolchain.toml`
Float to newest	Whatever PHP is installed	the channel `"stable"`
A different build	another image	`rustup override set nightly` in one directory

Try it

In the hello-cargo project, add the `rust-toolchain.toml` above. Run `rustup show` inside that directory, then again from a directory outside it. Inside, rustup reports the pinned toolchain and installs `1.94` if it is missing. Outside, you still see your default toolchain.

Key takeaways

- Pin a specific stable version in `rust-toolchain.toml`. The channel `"stable"` floats.

- rustup installs that toolchain automatically for everyone who builds the project.
- Nightly is for one directory, and only when a crate explicitly needs an unstable feature. Never `rustup default nightly`.

What's next: A CLI calculator