Effective in 2026, to align with our trunk stable development model and ensure platform stability for the ecosystem, we will publish source code to AOSP in Q2 and Q4. For building and contributing to AOSP, use `android-latest-release`. The `android-latest-release` manifest branch will always reference the most recent release pushed to AOSP. For more information, see Changes to AOSP (/docs/whatsnew/site-updates#aosp-changes).

Hello Rust example

The following is an example of constructing a Rust binary that depends on a Rust library.

Rust modules are currently confined to specific directories defined in build/soong/rust/config/allowed_list.go (https://android.googlesource.com/platform/build/soong/+/41461f36604948ef72855d56093d8b3e45f1d19c/rust/config/allowed_list.go). The following `helloWorld` module uses the `external/rust` directory to avoid modifying this list.

From the AOSP source root

```
$ mkdir -p external/rust/hello_rust/src/
$ mkdir -p external/rust/libsimple_printer/src/
```

Create `external/rust/hello_rust/src/hello_rust.rs` with the following contents:

```
use simple_printer;
fn main() {
 simple_printer::print_hello_rust();
}
```

Create `external/rust/libsimple_printer/src/lib.rs` file with the following contents:

```
pub fn print_hello_rust() {
  println!("Hello Rust!");
}
```

Create an `external/rust/hello_rust/Android.bp` file with the following contents:

```
rust_binary {
    name: "hello_rust",

    // srcs must contain a single source file, the entry point source.
    srcs: ["src/hello_rust.rs"],

    // rustlibs are Rust library dependencies. The type, rlib (static) or dylib
    // (dynamic), are chosen automatically based on module type and target
    // to avoid linkage issues.
    rustlibs: ["libsimple_printer"],

    // define any additional compilation flags
    flags: [
        "-C debug-assertions=yes",
    ],

    // cc libraries can be linked in to rust binaries and libraries through the
    // shared_libs and static_libs properties. These are not needed for this
    // simple example, so they are not included.
}
```

Create an `external/rust/libsimple_printer/Android.bp` file with the following contents:

```
// rust_library provides dylib and rlib variants.
rust_library {
    //name or stem properties must be of the form lib<crate_name><any_suffix>
    name: "libsimple_printer",

    //crate_name must match the name used in source (e.g. extern crate <name>)
    crate_name: "simple_printer",

    //srcs must contain a single source file, the entry point source
```

```
    srcs: ["src/lib.rs"],
}
```

Finally, build your `hello_rust` module:

```
$ source build/envsetup.sh
$ lunch aosp_arm64-eng
$ m hello_rust
```


Last updated 2026-06-17 UTC.