Normally Flutter runs in JIT for faster compilation/debugging support in `debug` mode and AOT mode for better performance in `profile` and `release` mode. For platforms that Flutter cannot produce AOT artifacts for, such as Android x86 (32 bit), a JIT release build may be used instead. The advantage of this mode over a regular debug build is that it removes debugging support and disables assertions which makes the final artifact smaller and more performant, though less so than a full AOT build.

JIT release mode can be used with a local engine configuration. For example, to setup an Android x86 jit_release and host build you can use the GN command below. Both device and host artifacts need to be built, except in cases where they are the same such as the Desktop shells.

```
./flutter/tools/gn --runtime-mode=jit_release --android --android-cpu=x86
ninja -C out/android_jit_release_x86
./flutter/tools/gn --runtime-mode=jit_release
ninja -C out/host_jit_release
```

This can be used with the flutter tool via the `--local-engine` flag to produce a bundle containing the jit release artifacts using the `flutter assemble` command. By default, flutter.gradle does not know how to package this artifacts so it requires custom integration into a build pipeline. Nevertheless, the artifact structure should be identical to a debug build, but with asserts disabled and product mode enabled.

jit_release is not supported on iOS devices. Applications built in JIT mode cannot be distributed on the Apple App Store.