GN Frequently Asked Questions

Contents

How will the build be converted?

We intend to build a second independent build in parallel to the GYP build. Previous efforts to generate GYP as an intermediate stage proved difficult. There will be some smaller set of bots compiling this build, and we'll keep the GN build running on these configurations.

What is unsupported in GN?

The main features not supported in GN yet are: * Mac/iOS bundles

Where is the GN documentation?

Rather than on a separate wiki, it is versioned with the tool. Run `gn help`. See also the quick start guide and the language and operation details.

What is likely to break?

Since common.gypi is not used for GN-generated GYP files, any rules there will no longer apply. There is a *lot* of logic in there for many build configurations and various conditions where certain flags should or should not be used. Some of these build configurations aren't even supported any more. Some are run on other waterfalls or are used by individuals manually setting GYP_DEFINES on their local system.

Will XCode/Visual Studio still be supported?

Visual Studio is supported. Visual Studio can be used as an IDE for code browsing or debugging but Ninja is used for building. Run `gn help gen` for more details.

XCode is not supported yet. We need help!

I'm weird. Will my uncommon build mode be supported?

One of the main benefits of the build changeover is that it will encourage us to refactor the build system. The project has generally not been as strict with build complexity and maintenance as we have with source code, and a lot of cruft has accumulated.

In most cases, we will want to rethink how build flags are supported. We want to be more modular rather than throwing everything in the common.gypi equivalent. The bar for additions at this level will be very high, and we will need to figure out how to design certain build features. If you are interested in some weird configurations, this will likely make your life more difficult in the short term, but will hopefully bring long-term benefits for everybody.

In some cases, we may decide that the overhead of supporting your build for BeOS running on a DEC Alpha is not in the interests of the project. There will likely be discussions about where to draw the line, and how to allow those who want to do weird things to do them cleanly without negatively affecting the rest of the Chromium community.

I'm only a little bit weird, will my development build flag be supported?

Only if you do it yourself!

Some features have build flags that turn on a debugging mode or switch between internal/external builds. This can be supported, but as with GYP, your team will have to add an maintain the support.

I use supplement.gypi, what's the GN equivalent?

Some projects use files called `supplement.gypi` to set build flags. GYP looks in each directory under `src` and adds merges these files into the build. The model is that then adding something to your gclient to add something to your build (e.g `src-internal`) automatically sets flags.

This behavior is fragile and mysterious. Some people get build flags and they don't know why. If you remove the entry from your `.gclient` and don't remove the directory you'll be stuck on an old version of the flags/code and not know why. You can't have builds in the same checkout with the corresponding flags on and off. Some people and projects were abusing this behavior.

In GN, such things should be done with build arguments (`gn args`) and configured on your build directory when you set it up. For some people, this will be an extra step. But it is explicit and clear, and you can have multiple builds in the same checkout with different flags.

How do I generate common build variants?

In GN, args go with a build directory rather than being global in the environment. To edit the args for your `out/Default` build directory:

```
gn args out/Default
```

You can set variables in that file:

- The default is a debug build. To do a release build add `is_debug = false`
- The default is a static build. To do a component build add `is_component_build = true`
- The default is a developer build. To do an official build, set `is_official_build = true`
- The default is Chromium branding. To do Chrome branding, set `is_chrome_branded = true`

How do I do cross-compiles?

GN has robust support for doing cross compiles and building things for multiple architectures in a single build.

See GNCrossCompiles for more info.