Lacros: chromeos source code directory migration
jamescook@google.com
2020-10-19
go/lacros-directory-migration
Tracking bug: crbug.com/1164001

Reviewers

erikchen@ (lacros, browser)	Approved	2020-10-29
hidehiko@ (lacros, chromeos)	Approved	2020-11-08
oshima@ (chromeos)	Approved	2020-11-09
yusukes@ (chromeos small council)	Approved	2020-11-30

TL;DR: We want to mechanically move a large number of files "chromeos" directories to "ash" directories, to make it clear which code runs in the ash system UI vs. the lacros browser.

Overview

The Lacros project will extract a separate browser binary from the existing system UI + browser monolith on Chrome OS. We call the two binaries "lacros-chrome" and "ash-chrome", or sometimes just "lacros" and "ash".

Both binaries are built out of chromium git. Both binaries include //content and many parts of //chrome/browser. This is confusing. Given a path or file, developers need to reason about which binary it runs in.

Chrome solves this problem for its multiprocess architecture by encoding information in path names. For example, code in //chrome/renderer runs in the renderer process, whereas code in //chrome/common could run in any process.

Lacros wants to enforce similar naming conventions for paths:

1. "ash" implies code that runs in ash.
2. "lacros" implies code that runs in lacros.
3. "chromeos" implies code that can be used by both.

Unfortunately, there is a lot of code in //chromeos and //chrome/browser/chromeos that is ash-only and hence violates (3). This isn't a trivial problem - we've seen confusion in lacros-related code reviews, even in code written by people on the lacros teams. It's a very reasonable assumption that a binary running on Chrome OS can use code with "chromeos" in the name.

We propose to move/rename a large amount of code to make (3) true. Specifically, almost all of //chrome/browser/chromeos moves to //chrome/browser/ash, and a large subset of //chromeos subdirectories move under //ash. See subdirectory audit sheet (internal only, sorry) for details.

Note that there is other code that is ash-only, but has neither "ash" nor "chromeos" in the path. For example, //components/exo. We don't propose to change this.

For more background see the original doc Pre/Post-Lacros code layout (internal only, sorry).

Proposed migrations

We **will not** be changing the number of code modules or the dependency graph. Even small changes to dependencies impose very large refactoring costs.

We **will** rewrite namespaces. Code moving into //ash will be rewritten to namespace ash. Code moving into //chrome/browser/ash will use the ash:: namespace (see below). We will need temporary "using" declarations during the transition to avoid churn. For example, many classes in namespace chromeos refer to each other. When a class moves into namespace ash, we can add a temporary "namespace chromeos { using ::ash::MyClass; }" to its header. This is an intentional style guide violation that will be deleted after all the referring classes migrate into namespace ash.

We propose to do the migration in stages (e.g. not freeze the tree and do a single giant commit). We plan to use tools like tools/git/mass-rename.py for #include rewrites. We will use the large scale changes global owners-override review policy to avoid having to ping all subdirectory owners for simple mechanical changes.

Overall, the cost isn't really moving the files. It's the namespace rewrites and test suite refactors. I suspect the cost is roughly linear in the number of files. My guess is this is ~4 SWE quarters worth of work, but we will have a much better estimate after the first few moves.

//chrome/browser/chromeos => //chrome/browser/ash

- **Added 2020-11-11.** We previously considered //chrome/browser/chromeos => //ash/browser, but after feedback from Chrome Eng Review (jam, darin) we decided on this approach. See Appendix.
- Continues to depend on //content and //chrome/browser (otherwise it could have just moved into //ash).
- Continues to depend on //ash.
- 4200 files, 520K lines of code
- Keep existing BUILD.gn file during transition, use "../ash/foo.h" relative paths, move BUILD.gn last.
- ~~Remove namespace chromeos, so code is in the global namespace. Code in //chrome is not supposed to be in a namespace (thread).~~
- **Updated 2020-10-02.** Migrate namespace from chromeos to ash. After moving several directories, we found namespace ash reads more naturally, avoids symbol collisions in the global namespace, and results in less namespace-related churn.
- Maintain existing unit_tests and browser_tests binaries.
- Files sitting directly in //c/b/chromeos (not in subdirectories) would move into newly created subdirectories (e.g. //c/b/chromeos/throttle_*.{h|cc} would move into //c/b/ash/throttle/).
- //c/b/chromeos/ui will move to //c/b/ash/notifications (not //c/b/ash/ui), since it mostly contains notifications. This reduces confusion with the existing //chrome/browser/ui/ash.
- Subdirectory audit sheet (internal only, sorry)
- Example CLs to move //c/b/chromeos/accessibility to //c/b/ash/ and the namespace rewrite.

~~//chromeos/components => //ash/components + //ash/webui~~

//chromeos/components => //chromeos/ash/components/

- **Updated 2022-04-04.** Previously we moved //chromeos/components => //ash/components + //ash/webui. However, this was confusing for ash UI developers, who assume //ash is only UI code. Therefore we decided to move //chromeos/components => //chromeos/ash/components.
- Most of these components are ash-only (e.g. account_manager, drivefs, multidevice, power, etc.)

- Some components have //content deps because they are WebUIs. Move those to //ash/webui to make this clear.
    - This also provides a path for future refactors of //chrome/browser/ash. As browser dependencies are eliminated, code can move to //ash/webui. As content dependencies are eliminated, code can move to //ash/shell. However, this refactor is out-of-scope for lacros.
- The other components (the majority) move to //chromeos/ash/components.
- 90K LoC
- Each has its own BUILD.gn file and DEPS. Moves should be easy.
- Namespace rewrite chromeos:: to ash::
- See subdirectory audit sheet (internal only, sorry)

//components/arc => //ash/components/arc

- **Added** **2020-11-15**
- This code is ash-chrome only.
- 40 KLoC, no namespace changes (it will stay "arc")
- b/129295708
- **Update:** 2022-08-10
- This move was done independent from Lacros, so moving to //chromeos/ash/components/arc is out of scope now.

~~//chromeos/services => //ash/services~~

//chromeos/services => //chromeos/ash/services

- **Updated 2022-04-04.** Previously we moved //chromeos/services => //ash/services. However, this was confusing for ash UI developers, who assume //ash is only UI code. Therefore we decided to move //chromeos/services => //chromeos/ash/services.
- Many //chromeos/components depend on //chromeos/services, but all this code runs in ash-chrome
- Continues to depend on //components, //device, //services, etc.
- 110K LoC
- Each directory has its own BUILD.gn and DEPS. Moves should be easy.
- Namespace rewrite chromeos:: to ash::
- See subdirectory audit sheet (internal only, sorry)

//chromeos/constants => //ash/constants

- Very low-level module with few DEPS, mostly constants, switches and features.
- Depends on //base and //third_party/icu.
- Has its own BUILD.gn and DEPS
- Namespace rewrite to ash::
- We might want to move some constants from //ash/public into //ash/constants, but that's not a blocking issue for lacros.
- crbug.com/1157625

Other //chromeos subdirectories

- Less mechanical than above, see audit sheet for //chromeos (internal only, sorry) for details
- Most will move to //chromeos/ash/components, e.g. //chromeos/audio moves to //chromeos/ash/components/audio

- Some low-level platform utils will stay in place, e.g. //chromeos/hugepage_text, because they may be used by both ash and lacros.
- Most subdirectories are isolated and have their own BUILD.gn files
- A few subdirectories need to be extracted into modules
- //chromeos/dbus needs further auditing. Ash-only D-Bus client wrappers will move into //ash/dbus (after the existing //ash/dbus moves to //ash/shell/dbus). Wrappers shared by Ash and Lacros should stay in place, but this may require refactor of the DBusThreadManager.

//ash/public => no change

- //ash/public was introduced when we were trying to "servicify" ash for the mustash project. It represents code shared by //ash and code outside //ash, in particular //chrome/browser/chromeos.
- Long term we would like to eliminate most this (so //ash is self-contained), and maybe rename to //ash/base or //ash/core.
- However, this isn't causing problems now, so we'll revisit after lacros.

Lower-priority migrations

//ash => //ash/shell

- **2020-11-11:** Reduced priority after Chrome Eng Review feedback. This is less urgent if //c/b/chromeos does not move into //ash/browser.
- Goal is to ensure //ash has just a few subdirectories, and each one has clear deps
- 4200 files, 390K LoC (lines of code)
- //ash/shell will not have //content deps (just like //ash today)
- No namespace changes. No DEPS changes. Moves should be easy.
- Rename test suite to ash_shell_unittests
- Do not move //ash/public, see //ash/public below.
- Do not move ash_strings.grd, some chromeos strings will move into this file
- Example CL moving one subdir

//chrome/browser/ui/app_list => //chrome/browser/ash/app_list

- 40K LoC
- Moderately expensive
- Requires creation of new BUILD.gn file, with allow_circular_includes_from //chrome/browser/ui (for example, code in app_list uses //c/b/ui/browser.h, and code in //c/b/ui/views uses //c/b/ui/app_list).
- We don't propose to modularize the code. It would be expensive and would not help lacros.
- Keep existing namespace app_list (not worth the cost to migrate to ash)
- Keep existing unit_tests and browser_tests.

//chrome/browser/ui/webui/chromeos, //c/b/resources/settings/chromeos, etc.

- Wide variety of ash-specific Web UI
- Not currently a source of confusion. Most files are well named and it's usually obvious whether the top-level UI is ash or lacros.
- Migration may require changes to webui scripts/bundling
- Leave in place for now. Eventually move to //chrome/browser/ui/webui/ash.

Other //chrome/browser/<subdir>/chromeos

- Other than the webui directories mentioned above, this set is small.
- Some contain code that may be used by lacros (e.g. webshare)
- Move some to //chrome/browser/<subdir>/ash
- See tab in subdirectory audit sheet (internal only, sorry)

Final dependency structure

New subdirectories in //ash:

- //ash/components
- //ash/content
- //ash/services
- //ash/shell

Rough dependency stack (any higher-level component can depend on any same-level or lower-level component)

Dependency notes:

- //ash/content depends on //content
- Other ash subdirectories do not depend on //content, nor on components that depend on //content
    - This is the case with //ash pre-migration, and we propose to maintain it.

Lastly, I have some tricks for source migrations (internal only, sorry) I used for the test CLs. Engineers should read that before starting moves or namespace conversions.

Appendix

//chrome/browser/chromeos => //ash/browser

- **2020-11-11: Discarded in favor of //chrome/browser/chromeos => //chrome/browser/ash after Chrome Eng Review feedback**
- 520K LoC
- Most expensive, but most valuable

- The directory continues to depend on //chrome/browser
    - Unusual, but breaking those dependencies will be a huge amount of work
- May require BUILD.gn changes
- Namespace rewrite chromeos:: to ash::
- For existing unit_tests and browser_tests, introduce new ash_unittests and ash_browsertests
    - Test refactor may be expensive. Could defer.
    - erikchen: "Infra would like us to break tests into smaller binaries, this would be a big win."
    - This will move thousands of Chrome OS-only tests to new binaries.
    - New binaries will be roughly the size of existing unit_tests and browser_tests, since they include //content and some of //chrome.
- See subdirectory audit sheet (internal only, sorry)
- Files sitting directly in //c/b/chromeos (not in subdirectories) would move into newly created subdirectories (e.g. //c/b/chromeos/throttle_*.{h|cc} would move into //ash/browser/throttle/).
- //c/b/chromeos/ui will move to //ash/browser/notifications, since it mostly contains notifications and we need the //ash/browser/ui name for other moves below.
- Example CL moving //c/b/chromeos/accessibility to //ash/browser/accessibility

//chrome/browser/ui/ash => //ash/browser/ui

- **2020-11-11: Discarded after Chrome Eng Review feedback. We could move this into //chrome/browser/ash/ui, but I don't think there's much benefit.**
- 40K LoC
- Moderately expensive, will require creation of new BUILD.gn file, may have circular dependency issues
- Add namespace ash:: (most classes don't have a namespace)
- Opportunity to rename "launcher" directories to "shelf"
- Move tests into ash_unittests and ash_browsertests
- Could skip or defer.
- See subdirectory audit sheet (internal only, sorry)