Chrome OS Files App

Navigation UI Refresh

Public Document

lucmult@chromium.org, sashab@chromium.org - Last updated: 2018-06-07

Tracking bug/841640 - Launch bug/831945

Status: Implemented and enabled on M-69

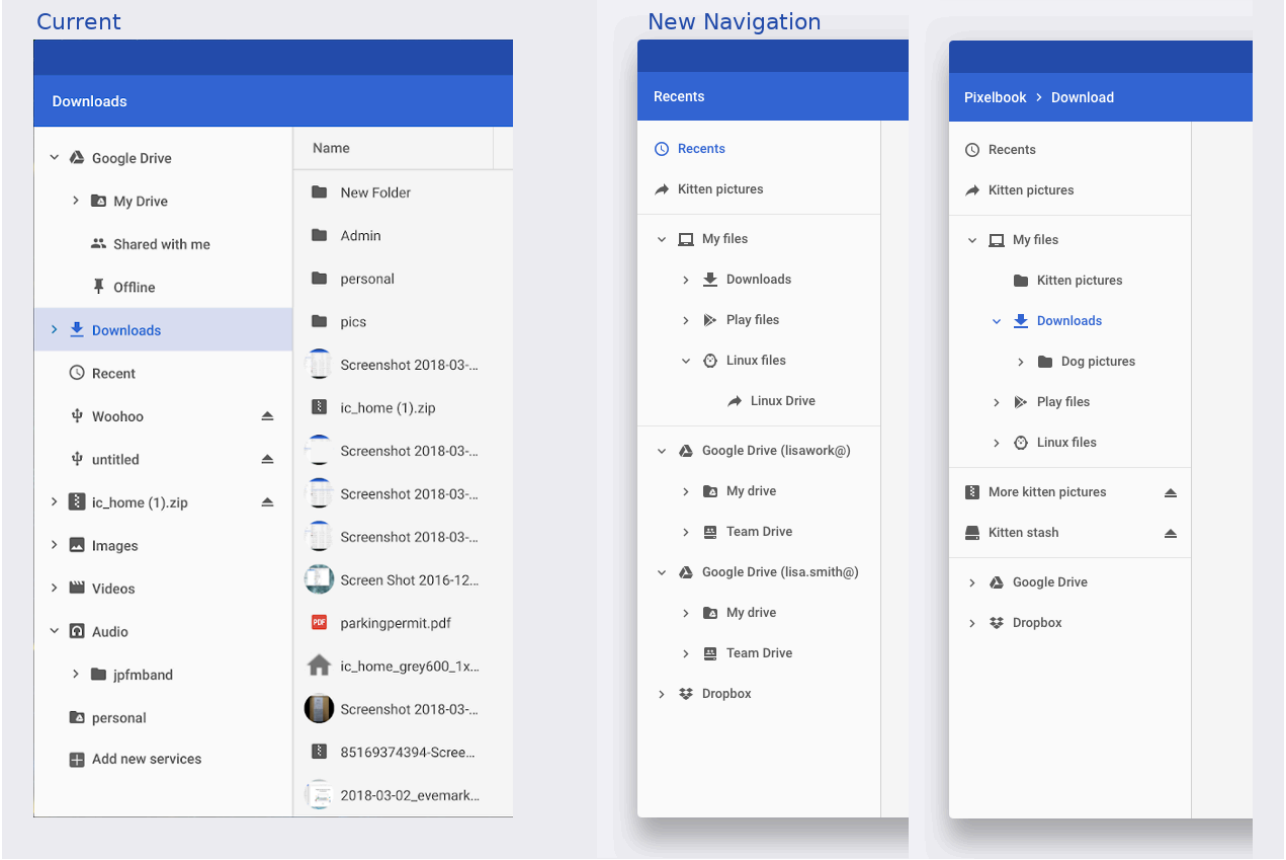
Reviewers: slangley@, weifangsun@

Background

As part of integrating ARC++/Android and Crostini/Linux on Chrome OS, we want to refresh the left-hand side directory navigation on Files app to conform with new UX mocks (go/filesapp-2018-mocks).

PRD (Product Requirements Doc): go/filesapp-home

Expected final state



Note: Some of the entries (multiple Google Drive entries, etc) is out of scope, and just shown to demonstrate the flexibility of the UI.

Implementation Plan

For M-69, the UI reorder has the following work:

1. **Done**: crbug.com/846585 Add flag --new-files-app-navigation
2. **Done**: crbug.com/846586 Add “Recents” view and “Shortcuts” at the top
3. **Done**: crbug.com/846587 Add support for nested roots and create “My Files” root
    a. crbug.com/846590 Add “Downloads” to “My Files” root
    b. crbug.com/846589 Add “Play Apps” to “My Files” root
    c. crbug.com/846588 Add “Linux Files” to “My Files” root

4. **Done**: crbug.com/846591 Add external media and ZIP sections
5. **Done**: crbug.com/846592 Add Google Drive and third-party cloud storage sections
6. **Done**: crbug.com/846593 Add separators and whitespace changes
7. **Done**: crbug.com/682356 Move “Add New Services” to gear menu
8. **Done**: crbug.com/848550 Disable Search for “My Files”, “Linux Files” and “Play Files” roots

The implementation should ensure the navigation is displayed with new volume icons, in the following order:

1. Recents
2. Shortcuts (if any)
3. Divider line
4. “My Files, with Play files and Linux volumes nested in “My Files” bottom
5. List of all removable medias and zip/archive volumes.
6. Google Drive
7. Any other File System Providers (FSP), e.g.: Dropbox.

A new flag called “**new-files-app-navigation**” will be used to enable or disable the new UI during M-69.

Rollout Plan

Waterfall. The flag “new-files-app-navigation” will be shipped on M-69 as enabled by default. It’s there for switching back in emergency or if the full feature isn’t completed on time.

Core principles and Privacy considerations

For M-69 it’s a UI-only change which we don’t expect any significant impact on **Speed**, **Security** nor **Privacy** for our users. This change doesn’t interact nor allow users to interact with any **new** sub-system or component of Chrome OS or external services. It’s implemented in the UI layer (HTML, JS and CSS).

Follow-up features such allowing new directories within “My Files” will require closer security review, as well as exposing Linux (Crostini) and Android (ARC++) files, but those have their own design doc and security reviews.

Implementation Specifics

Since this is a UI feature this will be implemented in the UI layer.

Current Files app state

It has several base models and types to represent the data that it displays and manages. The following are the relevant parts for this document.

Base Types: come from Web standard FileSystem API:

- FileSystemEntry (aka Entry)
- FileSystemFileEntry (aka FileEntry)
- FileSystemDirectoryEntry (aka DirectoryEntry)

Volumes:
- VolumeInfoList
- VolumeInfo
- VolumeManagerImpl
- VolumeManagerWrapper

Navigation Tree:
- In navigation_list_model.js:
  - NavigationListModel
  - NavigationModelItem
  - NavigationModelShortcutItem
  - NavigationModelVolumeItem
  - NavigationModelMenuItem
  - NavigationModelFakeItem
- UI elements in ui/directory_tree.js:
  - DirectoryItem
  - SubDirectoryItem
  - VolumeItem
  - DriveVolumeItem
  - ShortcutItem
  - MenuItem
  - FakeItem

File List: *(displayed as table or grid on the right-hand side)*
- Entry (either FileEntry or DirectoryEntry)
- MetadataModel
- UI elements:
  - FileGrid
  - FileTable
  - FileTableList

Breadcrumbs:
- EntryLocationImpl
- UI elements:
  - LocationLine
  - LocationLine.PathComponent

There are some limitations with the current implementation because some types are too specialized to navigation tree and breadcrumbs, whereas the file list only displays limited types. For example, navigation tree and file list can't display the same types, so most of the UI changes requires to customize several types, usually with similar code.

New Files app state

The long term intention is to make the 3 major components to operate on one common type (crbug.com/835203):

1. Navigation tree - Left Hand Side (LHS)
2. File list - Right Hand Side (RHS)
3. Breadcrumbs

However for this iteration (M-69) we will still maintain these components operating with different types.

At M-69, "**My Files**" is only a UI aggregation of independent volumes, "My Files" itself will **not** be a volume, instead it will be an artificial entry, which behaves similarly to a **DirectoryEntry** and contains multiple volumes, child entries are actually volumes wrapped in a new type VolumeEntry which also should behave similarly to **DirectoryEntry**. These new types will be called **EntryList** and **VolumeEntry**.

FilesAppEntry: An interface declaration to be implemented for other "Entry-like" types, future other Entry implementations should follow this interface.
VolumeEntry: Uses composition to make a volume behave like DirectoryEntry, so it can be displayed anywhere a DirectoryEntry this type can be displayed, in special RHS which currently can't display volumes.
EntryList: Hold a list of "entries" which for this iteration will only contain volumes: Downloads, Linux Files and Play Files. Also it will behave like a DirectoryEntry. so it can be displayed anywhere a DirectoryEntry this type can be displayed, in special LHS which can only display Volumes and some types dedicated to LHS.

For navigation menu

navigation_list_model.js is the data model for the navigation tree.

Create a new type "**NavigationModelDirectoryItem**" that should be able to display any object that behave like a DirectoryEntry, such as the new **EntryList** and **VolumeEntry** types.

Create new UI elements that can display and manage **NavigationModelDirectoryItem**.

Create a new method "**orderAndNestItems_**", which effectively will replace "reorderNavigationItems_" when the flag "**new-files-app-navigation**" is enabled. This new method will:

1. iterate over *this*.**volumeList_**, which is a list of volumes as returned by VolumeManager API and generate "*this*.**navigationItems_**" attribute.
2. enforce the desired order for the items and volumes on the navigation tree.
3. append Downloads, Linux and Android volumes to "My Files".

For file list

When user selects “My Files” we want to display Downloads, Linux Files on the file list, however currently the file list doesn’t support displaying volumes. In fact file list can only display Entry, either FileEntry or Directory entry.

EntryList should implement the createReader API to return it’s children entities as **VolumeEntity** so the FileList will be able to display them. Since EntryList’s children is a simple list created during initialization, **createReader** API can be implemented in JS synchronously.

For breadcrumbs

Breadcrumbs are generated based on VolumeInfo instances and based on the current selected Entry. Individual entries don’t have a directly link to “My Files”, to be able to add “My Files” as prefix on the breadcrumbs, all VolumeInfo inserted in “My Files” EntryList will be called to inject “My Files” as an attribute to its VolumeInfo instance. With this injected prefix on VolumeInfo can generate the required prefix on the breadcrumbs component.

Other Details

Searching files within “My Files”

Initial implementation will hide “search” feature when the user has selected “My Files”. Another option is to search only within Downloads, however it can generate confusion to users.

For next iterations we intend to implement “full search”, searching across all volumes inside “My Files”.

Keep drag-n-drop support in the navigation tree

Ensure we don’t lose drag-and-drop functionality in the navigation tree. Currently Files app supports only “drop” action on navigation tree including copying and moving between different volumes.

The new section “My Files” should keep this ability of dragging and dropping files among its children volumes “Downloads”, “Linux Files” and “Play Files”, however create, move and delete of “My Files” children are not allowed on this iteration.

Testing Plan

- Existing integration tests should pass with the new flag on.
    - Until feature is ready to ship, we should run it in a temporary test fixture, so we should create a new class FilesExperimentalBrowserTest. We don’t want to use the term “FilesApp” to isolate those temporary tests from permanent Files App tests on flakiness dashboard.

    - This will guarantee current behaviour (with new flag off) will continue to be tested. We'll select some tests to run with flag on on the temporary fixture. In development environment the permanent test fixture should also be run with the flag on.
    - Some of the current tests might need to be updated to have their DOM selector/handling compatible with both navigation. It's important to maintain the current tests checking for the current navigation.
- Integration Tests for new "My Files" section:
    - Click and expand: Display children volumes on navigation and on file list.
    - Click on Downloads: Should display correct breadcrumbs and files.
    - Context menu should only show relevant menu enabled. **TBD**: What are the relevant menus.
    - When "My Files" is selected only relevant actions are visible on action bar, e.g.: search is hidden. **TBD**: Any other relevant actions should be visible/hidden?
    - When "My Files" is selected only relevant file tasks/commands are available. **TBD**: Anything needed here?
    - Downloads folder is displayed by default. **TBD:** pending confirmation if Recents should be displayed by default.
- Unittests for newly created JS types (VolumeEntry and EntryList).
- Ensure manual tests have good coverage
    - File operation (copy, paste, move) should work between Drive and Downloads.
    - User should not be able to create folder nor files within "My Files".

Metrics/UMA

- Use existing UMAs for Downloads, Linux Files and Play Files.
    - FileBrowser.ChangeDirectory.RootType
    - FileBrowser.Location.*
        - FileBrowser.Location.*.TopLevel
- Add new "RootType" "My Files".

Success metrics

- The change on this stage has only a subjective impact on user's interaction and perception of their own files, so there isn't a metric measuring this directly.
- However we expect the following metrics to **not** be negatively impacted (*to not have a drop in usage*):
    - FileBrowser.ChangeDirectory.**RootType**: mainly for **Downloads** root type, but also for all other root types.
    - FileBrowser.Location.* also for **Downloads** root type.
    - *Note:* a drop proportional to usage of new root type: "My Files" is still a valid outcome.
- Run a GCS (Google Consumer Survey) or equivalent while on Beta (or post-launch) to assess user sentiment regarding this update.

Future Work

- Remove the flag "new-files-app-navigation" crbug.com/850348

- Allowing users to create other directories above “Downloads”.
- Allowing multiple Drive profiles.
- Turning on the flag by default (and removing Media Views).
- Consolidate types for entries across the Files app crbug.com/835203.

Appendix

Next stages design doc.