BrowserThread::PostAfterStartupTask Deprecation

Attention: Shared externally

Author: carlscab@google.com

Last Updated: 2019-05-28

If you got here because you were assigned a patch review you might want to just jump to [I got a patch to review!]

Overview

We want to get rid of the [`content::BrowserThread::PostAfterStartupTask()`] method. There is no clear definition of what startup completed means, and most of the tasks being posted this way do just so to run after any critical work has completed.

From now on, code should post tasks via calls to base::PostTaskWithTraits with appropriate traits.

The current documentation for `BrowserThread::PostAfterStartupTask` states:

> *For use with scheduling non-critical tasks for execution after startup. The order or execution of tasks posted here is unspecified even when posting to a SequencedTaskRunner and* ***tasks are not guaranteed to be run prior to browser shutdown****. When called after the browser startup is complete, will post |task| to |task_runner| immediately.*

This is a good fit for `base::TaskPriority::BEST_EFFORT` annotated tasks given that they will only run after startup is completed. This will also make it clearer to clients that these tasks might never get a chance to run, moreover we will be able to test that code does not rely on these tasks via the existing `switches::kDisableBestEffortTasks` switch.
Initially this will only work for the IO and UI threads and we will soon support it on the ThreadPool too.

Proposed changes

Replace `BrowserThread::PostAfterStartupTask` with a call to

```
base::PostTaskWithTraits(
    FROM_HERE,
    {content::BrowserThread::UI, base::TaskPriority::BEST_EFFORT},
    task);
```

Or similarly for the IO thread:

```
base::PostTaskWithTraits(
    FROM_HERE,
    {content::BrowserThread::IO, base::TaskPriority::BEST_EFFORT},
    task);
```

We will also provide a helper function `content::PostBestEffortTask()` to post a task to run at `BEST_EFFORT` priority using an arbitrary `TaskRunner` for which we do not control the priority.

I got a patch to review!

Congratulations, you just got to review some code being refactored. Don't panic.

1. The patch you got should not change the current behaviour in any way
2. BUT you might want to double check whether your task being posted via `BrowserThread::PostAfterStartupTask` really cares about being run after startup or just wants to run after any critical work was completed. If you are fine with it just running after critical work has completed LGTM and keep on with your life.
    a. If that is not the case
        i. Add a comment to the patch explaining why this task must specifically run after startup, and you will get an updated patch.