

Scheduling resource responses with right priority

Feb 16, 2018

This document discusses infrastructure needed to allow loading stack notify blink scheduler about the correct priorities for scheduling resource responses on the main thread.

WebURLLoader already takes `base::SingleThreadTaskRunner` to post task responses to. It is suggested to introduce a subclass of `SingleThreadTaskRunner` (`PrioritizableTaskRunner`) which allows setting priority for all tasks posted via this task runner.

```
class PrioritizableTaskRunner : SingleThreadTaskRunner {
    void SetPriority(Priority priority);

    // Implementation of SingleThreadTaskRunner
}
```

WebURLLoader obtains task runner via `FetchContext::GetLoadingTaskRunner`, which returns per-frame task runner already for `FrameFetchContext` and default task runner in all other use cases.

For `FrameFetchContext` use case task runner will be obtained via newly-introduced `scoped_refptr<PrioritizableTaskRunner> FrameScheduler::GetLoadingTaskRunner()` method. For all other endpoints a method to wrap any `SingleThreadTaskRunner` into a `PrioritizableTaskRunner` (with `SetPriority` method being no-op in this case) will be provided.

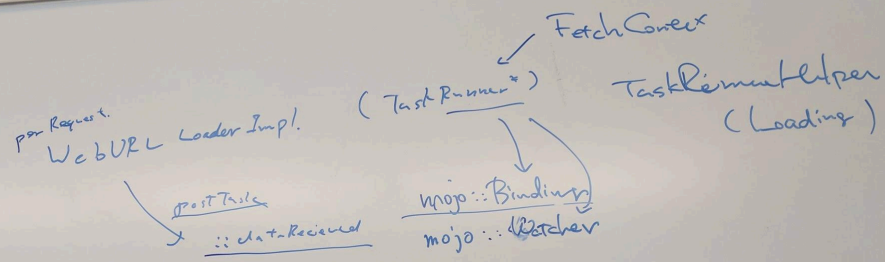
```
class PrioritizableTaskRunnerImpl : PrioritizableTaskRunner {
public:
    void SetPriority(Priority priority) { priority_ = priority; }

    void PostDelayedTask(...) override { task_runners_[priority_] -> PostDelayedTask(...); }

private:
    std::array<SingleThreadTaskRunner, Priority::kCount> task_runners_;
    Priority priority_;
}
```

For v2 a more complex implementation supporting changing priorities after posting a task can be provided.

Photo from Feb 14 meeting:



```

HTML Parser :: dataReceived() {
  postTask()
}
  
```

- Remove hack in loading context.
- Other consumers of loading task &
 - Script Runner
 - HTML Document Parser.
- ::dataReceived & other Web URL Loader call backs.