Refactor EventQueue

hajimehoshi@chromium.org
Last Updated: 2018-06-21
Status: Public

crbug.com/846618

tl;dr

EventQueue aggregates events and dispatches them but such thing should be done by task runners in scheduler. We aim to refactor EventQueue to use plain PostTask as much as possible. However, it is not doable to eliminate EventQueue since

- Connecting JavaScript callstacks is done by core::probes in EventQueue.
- Enqueued events can be cancelled later.

Instead of eliminating, we could keep it managing probes and cancellable events. There are similar classes like MediaElementEventQueue or NullEventQueue, so we should merge them for code health.

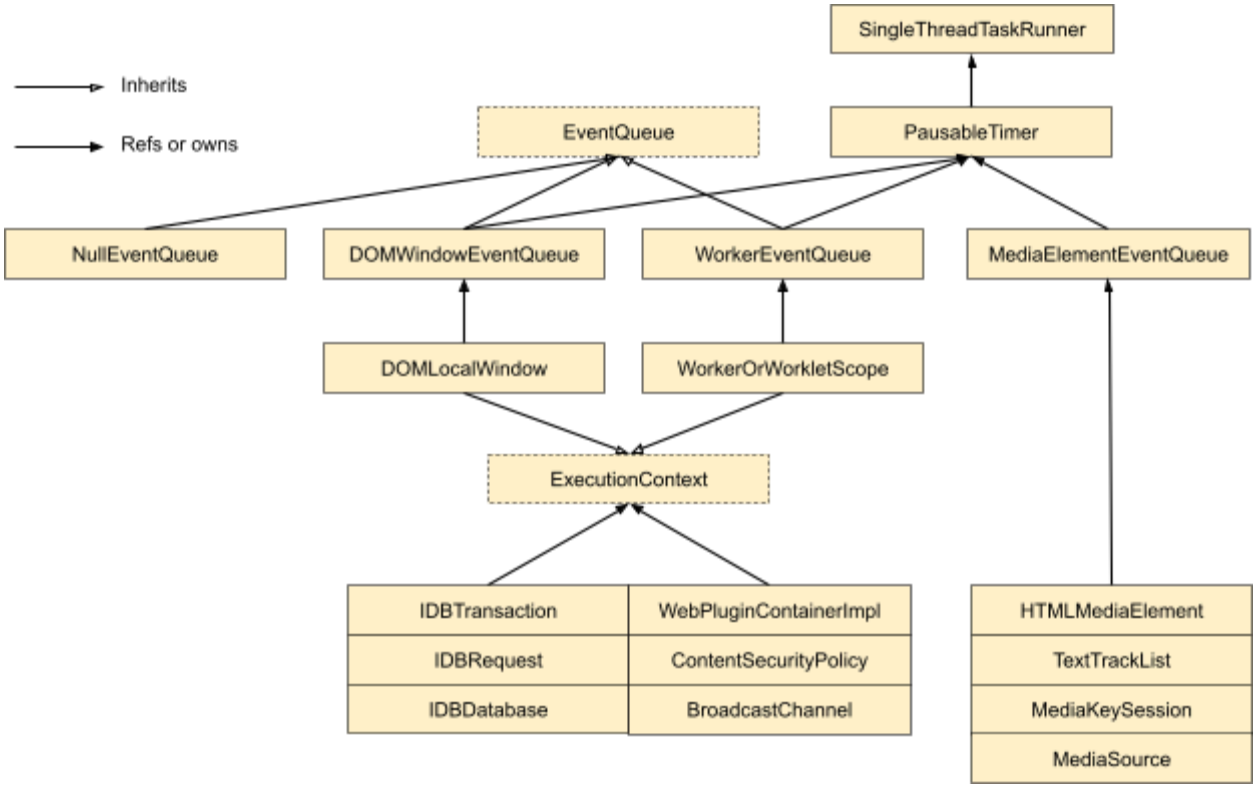
Current usage

There are three types of EventQueue:

- DOMWindowEventQueue (inheriting EventQueue)
- WorkerEventQueue (inheriting EventQueue)
- MediaElementEventQueue
- NullEventQueue (inheriting EventQueue)

We have already de-coulpled MediaElementEventQueue and EventQueue.

DOMWindowEventQueue and WorkerEventQueue are owned by ExecutionContext like DOMLocalWindow and WorkerOrWorletScope, and EventQueue users like IDBTransaction get an EventQueue from the current ExecutionContext. Each EventQueue has a PausableTimer that has a task runner, which is created from the passed ExecutionContext with a task type specified (e.g. kUnthrottled for DOMWindowEventQueue, kInternalWorker for WorkerEventQueue).

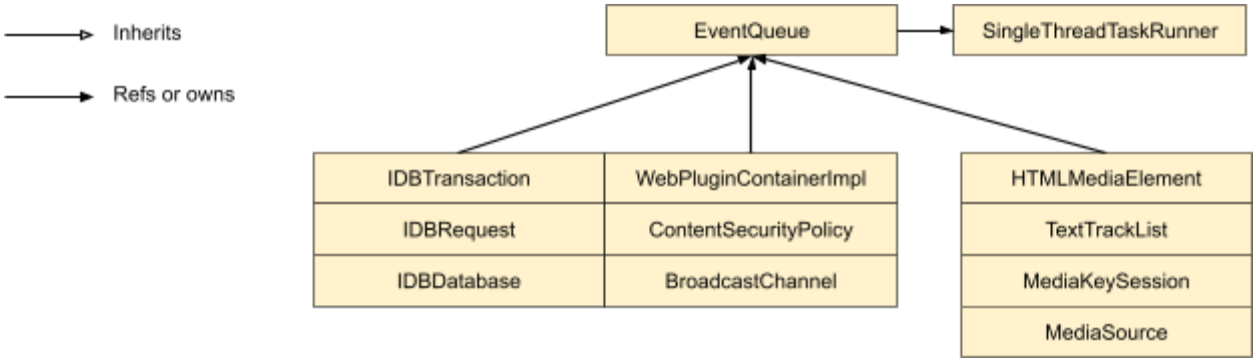


ExecutionContext owns EventQueue indirectly and its context is used to create a task runner to pass to internal PausableTimer. Objects like IDBTransaction get EventQueue from ExecutionContext. Whether DOMWindowEventQueue or WorkerEventQueue is used depends on the context.

NullEventQueue is for testing.

Suggestion

We suggest to simplify the relationship like this:



- PausableTimer will be gone

- Unify all the implementation of EventQueues
- EventQueue users own their EventDispatcher(Impl) instead of getting one from ExecutionContext
- Instead of ExecutionContext, pass a task runner to EventQueue
- One object owns one EventQueue

The pseudo EventQueue definition will be:

```
class EventQueue {
public:
      EventDispatcher(SingleThreadTaskRunner);
      void EnqueueEvent(Event* event);
      void CancelAllEvents();
      void Close();
private:
      HeapHashSet<Member<Event>> queued_events_;
}
```

As for CloseAllEvents, only IndexedDB would call this. This is different from Close since a new event can be enqueued after CloseAllEvents.

Steps

Remove PausableTimer usages from EventQueue

Now EventQueue uses a PausableTimer to aggregate events and the events are dispatched when the timer fires. This timer is for performance, and we are not sure if we can remove the timer usage without performance issue. Let's try to remove PausableTimer usage and see what will happen.

CL:

- https://chromium-review.googlesource.com/c/chromium/src/+/1075950
- https://chromium-review.googlesource.com/c/chromium/src/+/1076110

Unify the implementation of three EventQueue

There are some slight differences among three EventQueues (e.g. one event can be canceled or not), but after PausableTimer are gone, it should be easy to unify the implementation.

CL:

- https://chromium-review.googlesource.com/c/chromium/src/+/1082061

- https://chromium-review.googlesource.com/c/chromium/src/+/1082134
- https://chromium-review.googlesource.com/c/chromium/src/+/1082190
- https://chromium-review.googlesource.com/c/chromium/src/+/1088356
- https://chromium-review.googlesource.com/c/chromium/src/+/1088341
- https://chromium-review.googlesource.com/c/chromium/src/+/1088449
- https://chromium-review.googlesource.com/c/chromium/src/+/1090597
- https://chromium-review.googlesource.com/c/chromium/src/+/1105803
- https://chromium-review.googlesource.com/c/chromium/src/+/1098742

Move the owner of EventDispatcher from ExecutionContext to EventDispatcher users

- https://chromium-review.googlesource.com/c/chromium/src/+/1088534

Pass a TaskRunner instead of ExecutionContext

This enables that more detailed task types can be specified. Now DOMWindowEventQueue always uses TaskType::kUnthrottled since DOMWindowEventQueue can't assume what is the user.

Remove ExecutionContext::GetEventQueue()

- https://chromium-review.googlesource.com/c/chromium/src/+/1090697

Others

- https://chromium-review.googlesource.com/c/chromium/src/+/1090591