Integrating MicrotaskContext to EnteredContext

tzik@

TL;DR:

A proposal to update HandleScopeImplementer to merge its microtask_context_ into entered_contexts_, so that we can nest RunMicrotasks for different MicrotaskQueues for Cooperative Scheduling and Layout Worklet.

Motivation

HandleScopeImplementer manages entered contexts and the microtask context separately, and we have two variants of getters of the last entered context: the microtask context is involved to one, GetEnteredContext(), and is not involved to another, GetEnteredOrMicrotaskContext(). However, GetEnteredOrMicrotaskContext() is more relevant than GetEnteredContext() for all callers, and the separation makes the implementations complex. Also, the current implementation doesn't work well for nested microtask queue as it's not managed as a stack.

This proposal merges the microtask context into entered contexts, and replaces GetEnteredContext() with GetEnteredOrMicrotaskContext() for simpler management and nestable microtask contexts.

Nested RunMicrotasks?

Cooperative Scheduling

This is an attempt to yield the main thread occupied by a third-party long running JS execution. While V8 is running an author JS, Blink stops running the task at certain safe points (e.g. Blink callbacks), and runs high priority tasks for better responsiveness. As the nested tasks will be a third party task, that is, the task don't touch the paused task synchronously, the execution is not visible for the page authors.

However, the current implementation of the microtask queue is problematic for this scenario. As it does not support nesting, we can not run microtasks in the inner task if the outer task is running a microtask.

Worklets

Worklets is a similar concept to Worker, but they may share a thread unlike Workers. Each of them has its own context and the event loop with a microtask queue, and may run synchronously while other script is running. E.g. Layout Worklet may run in the synchronous re-layout, and in this case, a microtask in the worklet may run while the outer script is running a microtask.

Overview

As a rough sketch, this proposes updating “Current” to “Proposed”, and to maintain the API compatibility, we probably need to go to “Transitional” first and stay there for a while.

Current

HandleScopeImplementer holds a stack of entered contexts and single microtask context separately. The microtask context is updated by RunMicrotasks builtin.

```
class Isolate {
 public:
  Local<Context> GetEnteredContext();
  Local<Context> GetEnteredOrMicrotaskContext();
};

class HandleScopeImplementer {
 public:
  void EnterContext(Handle<Context> context);
  void LeaveContext();
  bool LastEnteredContextWas(Handle<Context> context);
  Handle<Context> LastEnteredContext();

  void EnterMicrotaskContext(Handle<Context> context);
  void LeaveMicrotaskContext();
  Handle<Context> MicrotaskContext();
  bool MicrotaskContextIsLastEnteredContext() const;

 private:
  DetachableVector<Context*> entered_contexts_;
  Context* microtask_context_;
  size_t entered_contexts_count_;
  size_t entered_context_count_during_microtasks_;
};
```

Proposed

No separation between the entered context and microtask context. |entered_contexts_| contains both of them. As a preparation, we need to update DetachableVector to be accessible from builtins.

```
// No separation between the entered context and microtask context.
class Isolate {
 public:
  Local<Context> GetEnteredOrMicrotaskContext();
};

class HandleScopeImplementer {
 public:
  void EnterContext(Handle<Context> context);
  void LeaveContext();
  bool LastEnteredContextWas(Handle<Context> context);
  Handle<Context> LastEnteredOrMicrotaskContext();

 private:
  DetachableVector<Context*> entered_contexts_;
};
```

Transitional

To maintain the API compatibility, we need to keep the entered contexts and microtask contexts distinguishable. As there’s no known caller of GetEnteredContext outside of Blink and V8, the deprecation should be easy.

```
class Isolate {
 public:
  V8_DEPRECATED("Use GetEnteredOrMicrotaskContext",
                Local<Context> GetEnteredContext());
  Local<Context> GetEnteredOrMicrotaskContext();
};

class HandleScopeImplementer {
 public:
  void EnterContext(Handle<Context> context, bool is_microtask_context);
  void LeaveContext();
  bool LastEnteredContextWas(Handle<Context> context);

  Handle<Context> LastEnteredContext();
  Handle<Context> LastEnteredOrMicrotaskContext();

 private:
  DetachableVector<Context*> entered_contexts_;
  DetachableVector<bool> is_microtask_context_;
};
```

Steps

1. Update DetachableVector to use its own buffer, instead of std::vector<>, so that builtins can access the buffer directly. https://crrev.com/c/1297783
2. Replace all usage of GetEnteredContext with GetEnteredOrMicrotaskContext. https://crrev.com/c/1297654
3. Update EnterMicrotaskContext / LeaveMicrotaskContext for builtins to push/pop the given microtask context to entered_contexts_. https://crrev.com/c/1322290