SCHEDULER

BHAVDEEP DHANJAL

github: alivenotions

twitter: bhavdeepdhanjal

mail: dhanjalbhavdeep@gmail.com

WHAT THIS TALK IS ABOUT?

An introduction to very internal topics

Making you all interested in reading source code

Sharing what I've been learning on my own

Scheduler

WHAT IS SCHEDULING?

Scheduling is the way work is assigned to somebody that will do the work.

This can be done to optimize various things.

DOES THE BROWSER NEED A SCHEDULER?

Why don't we make our code run fast?

Our dreams are bigger than our processors

Why don't we let things be just as they are?

Too many updates can cause wasted renders

Long updates can cause frames to drop

FIFO is not always that cool

HOW DOES IT FIT IN REACT?

Scheduling is the way a unit of work is assigned to somebody that will do the work.

What is a unit of work for React?

FIBER

Fiber is a data structure that represents work to be done on a component or work that was done

It allows the scheduler to control when to render something and when to yield for some other event

Let's go to ReactFiber.js

FIBER

It's a tree mapped to your components with some additional properties

Activities during reconciliation is called work

Unlike elements, fibers aren't recreated and are mutable

Allows the architecture to run in two phases: render and commit

FIBER: RENDER PHASE

Work in progress tree (think of it as a draft)

Caused by updates to components

Fiber tree is marked with effects

Can be async

Look at ReactFiberWorkLoop.js

performUnitOfWork

beginWork

completeUnitOfWork

completeWork

FIBER: COMMIT PHASE

Synchronously applies the fiber nodes marked with effects to its instances

Look at ReactFiberCommitWork.js

Scheduler assigns priorities to work and can choose what chunk of work should be executed when

If a higher priority task is fired whilst a lower priority one is running, the latter is suspended to be run later

PRIORITIES

Immediate(Synchronous tasks)

User Blocking(250ms timeout)

Normal(5s timeout)

Low (10s timeout)

Idle (no timeout)

Why don't they leverage APIs from the browser like requestIdleCallback?

It waits for an idle period before executing the associated work

HOW SCHEDULER WORKS (CURRENTLY)

1. Store all work callbacks in a list ordered by expiration time
2. Register a callback that is run after the next frame is drawn by the browser
3. Achieved by postMessage inside a requestAnimationFrame callback
4. Called right after the frame is rendered and it executes as many of the registered callbacks as possible

Control is either yielded based on isInputPending but the scheduler still maintains a max frame length after which it yields regardless

All tasks that go through the scheduler, go through this function(ReactFiberWorkLoop/performConcurrenWorkO

LIMITATION

- Scheduler doesn't have access to lower level APIs like GC or rendering APIs
- Multiple scheduling mechanisms can fight within each other

WORK IN BROWSERLAND

- When to yield to the browser?

  scheduler.isInputPending()
- When to regain control

  scheduler.yield()
- Coordination between cooperating actors

  A standardized shared notion of priority
- Standard set of scheduling APIs

  Post task api

https://github.com/WICG/main-thread-scheduling

THANK YOU

github: alivenotions

twitter: bhavdeepdhanjal

mail: dhanjalbhavdeep@gmail.com