19-2 React Lanes 模型的应用

Lane

位掩码，用于标记一个任务的优先级。每个 lane 上只有一个比特位为 1。

Update 的 lane

每个 update 会对应一个 lane。

所以在 Update 上有个属性：

react/packages/react-reconciler/src/ReactFiberClassUpdateQueue.js

```JavaScript
export type Update<State> = {
  lane: Lane,
  ...
};
```

react/packages/react-reconciler/src/ReactFiberHooks.js

```JavaScript
export type Update<S, A> = {
  lane: Lane,
  revertLane: Lane,
```

```
  action: A,
  hasEagerState: boolean,
  eagerState: S | null,
  next: Update<S, A>,
};
```

事件优先级

react/packages/react-reconciler/src/ReactEventPriorities.js

JavaScript

```
export opaque type EventPriority = Lane;

export const DiscreteEventPriority: EventPriority = SyncLane;
export const ContinuousEventPriority: EventPriority = InputContinuousL
export const DefaultEventPriority: EventPriority = DefaultLane; // 页面
export const IdleEventPriority: EventPriority = IdleLane;

let currentUpdatePriority: EventPriority = NoLane;
```

调度更新 scheduleUpdateOnFiber

Lanes

几个 Lane 位运算出来的位掩码，称为 Lanes。每个 lanes 上至少有一个比特位为 1。

UpdateQueue 的 lanes

react/packages/react-reconciler/src/ReactFiberClassUpdateQueue.js

JavaScript

```
export type SharedQueue<State> = {
  pending: Update<State> | null, // 单向循环链表
  lanes: Lanes,
  // 如果类组件是Activity(以前叫OffScreen)的后代组件，需要延迟执行的其setState
  // Activity目前还是unstable，了解即可~
  hiddenCallbacks: Array<() => mixed> | null,
};

export type UpdateQueue<State> = {
  baseState: State,
  // 单链表 firstBaseUpdate->...->lastBaseUpdate
  firstBaseUpdate: Update<State> | null,
  lastBaseUpdate: Update<State> | null,
  shared: SharedQueue<State>,
  callbacks: Array<() => mixed> | null,
};
```

Hooks UpdateQueue 的 lanes

react/packages/react-reconciler/src/ReactFiberHooks.js

JavaScript

```
export type UpdateQueue<S, A> = {
  pending: Update<S, A> | null,
  lanes: Lanes,
  dispatch: (A => mixed) | null,
  lastRenderedReducer: ((S, A) => S) | null,
  lastRenderedState: S | null,
};
```

Fiber 的 lanes

react/packages/react-reconciler/src/ReactInternalTypes.js

JavaScript

```
export type Fiber = {
  ...
```

```
  lanes: Lanes,
  childLanes: Lanes, // 后代fiber的lanes
  // 依赖，比如context
  dependencies: Dependencies | null,
  ...
};

export type ContextDependency<T> = {
  context: ReactContext<T>,
  next: ContextDependency<mixed> | null,
  memoizedValue: T,
  ...
};

export type Dependencies = {
  lanes: Lanes,
  firstContext: ContextDependency<mixed> | null,
  ...
};
```

因为一个 fiber 上可以有多个 update，而每个 update 可以有不同的优先级。

FiberRoot 的 lanes

react/packages/react-reconciler/src/ReactInternalTypes.js

JavaScript

```
  // updated lanes
  pendingLanes: Lanes,
  suspendedLanes: Lanes,
  pingedLanes: Lanes,
  expiredLanes: Lanes,
  errorRecoveryDisabledLanes: Lanes,
  shellSuspendCounter: number,

  finishedLanes: Lanes,

  // 记录当前更新与其他更新之间的关联性，即它们之间存在依赖或相关性。
  // 当一个更新被触发时，React 会根据其依赖关系计算出 entangledLanes，这些 ent
```

```
  entangledLanes: Lanes,
  entanglements: LaneMap<Lanes>,
```