9-7 commit 阶段

performConcurrentWorkOnRoot

TypeScript

```ts
export function performConcurrentWorkOnRoot(root: FiberRoot) {
  // ! 1. render, 构建fiber树VDOM (beginWork|completeWork)
  renderRootSync(root);

  console.log(
    "%c [  ]-31",
    "font-size:13px; background:pink; color:#bf2c9f;",
    root
  );

  const finishedWork: Fiber = root.current.alternate as Fiber;
  root.finishedWork = finishedWork;

  // ! 2. commit, VDOM->DOM
  commitRoot(root);
}
```

Commit 阶段

```TypeScript
function commitRoot(root: FiberRoot) {
  const prevExecutionContext = executionContext;
  executionContext |= CommitContext;

  const finishedWork = root.finishedWork as Fiber;

  // mutation阶段
  commitMutationEffects(root, finishedWork);

  executionContext = prevExecutionContext;
  return null;
}
```

mutation 阶段

更新 DOM 树

```TypeScript
import { Placement } from "./ReactFiberFlags";
import { Fiber, FiberRoot } from "./ReactInternalTypes";
import { HostComponent, HostRoot } from "./ReactWorkTags";

export function commitMutationEffects(root: FiberRoot, finishedWork: F
  recursivelyTraverseMutationEffects(root, finishedWork);
  commitReconciliationEffects(finishedWork);
}

function recursivelyTraverseMutationEffects(
  root: FiberRoot,
  parentFiber: Fiber
) {
  let child = parentFiber.child;

  while (child !== null) {
```

```
    commitMutationEffects(root, child);
    child = child.sibling;
  }
}

// fiber.flags
// 新增插入
function commitReconciliationEffects(finishedWork: Fiber) {
  const flags = finishedWork.flags;

  if (flags & Placement) {
    commitPlacement(finishedWork);
    finishedWork.flags &= ~Placement;
  }
}

// 在dom上，把子节点插入到父节点里
function commitPlacement(finishedWork: Fiber) {
  const parentFiber = getHostParentFiber(finishedWork);

  // 插入父dom
  if (finishedWork.stateNode && finishedWork.tag === HostComponent) {
    // 获取父dom节点
    let parent = parentFiber.stateNode;

    if (parent.containerInfo) {
      parent = parent.containerInfo;
    }

    // dom节点
    parent.appendChild(finishedWork.stateNode);
  }
}

// 返回 fiber 的父dom节点对应的fiber
function getHostParentFiber(fiber: Fiber): Fiber {
  let parent = fiber.return;

  while (parent !== null) {
    if (isHostParent(parent)) {
      return parent;
```

```
    }
    parent = parent.return;
  }
  throw new Error(
    "Expected to find a host parent. This error is likely caused by a
      "in React. Please file an issue."
  );
}

// 检查 fiber 是否可以是父 dom 节点
function isHostParent(fiber: Fiber): boolean {
  return fiber.tag === HostComponent || fiber.tag === HostRoot;
}
```

main.tsx

React TSX

```
import { ReactDOM } from "../which-react";
import "./index.css";

const jsx = (
  <div className="box border">
    <h1 className="border">omg</h1>
  </div>
);

ReactDOM.createRoot(document.getElementById("root") as HTMLElement).re

// div.root 对应的是根fiber，Fiber, tag = HostRoot=3

// 原生标签Fiber, tag = HostComponent=5

// Host
// 1. HostRoot
// 2. HostComponent
// 3. HostText // 不能子节点
```