《京程一灯》精英班第六讲试卷　　姓名：

请按要求作答，并工整书写试卷。

- 1.请简述一下React和Vue的特性。(5分)

  答：

（大家也可以根据自己的理解进行补充）本题考点分为如下：

1.React 后台是巨头 facebook，功能强大，社区也越来越火，资料文档、技术讨论等方面比 vue 要好些，jsx 可能成为 ECMA 的新规范，上手难度比 vue 大不少。

2.React 是单向数据流，状态机实现，Vue是双向绑定。

3.React 的开发趋势是将所有东西都放在 JavaScript 中,Vue 的 单文件组件 在把 CSS 封装到组件模块的同时仍然允许你使用你喜欢的预处理器。

4.React 团队雄心勃勃，计划让 React 成为通用平台的 UI 开发工具包括React-AR、React-VR，而 Vue 专注于为 Web 提供实用的解决方案(阿里实现了Weex)。

5. React，为了最优化的渲染需要处处实现 shouldComponentUpdate 和使用不可变数据结构。

- 2.请详细描述React和Vue SetState的原理(8分)

  答：

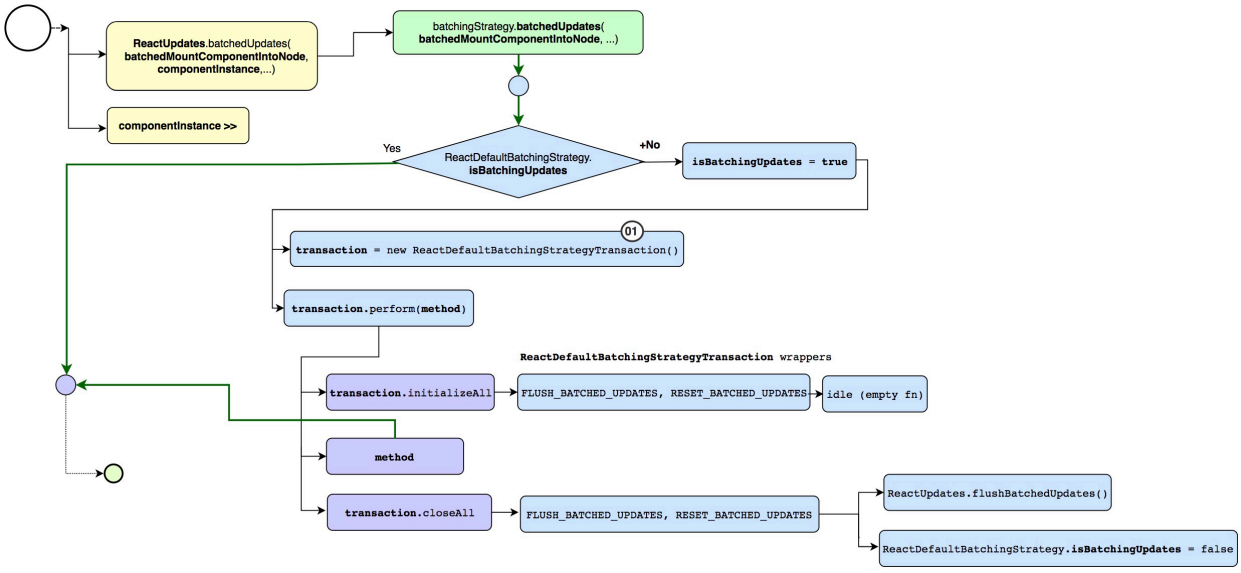
高频考题：

1.React调用的方法中连续setState走的是批量更新，此外走的是连续更新。就是说如果方法是通过React调用的比如生命周期函数，React的事件处理等，那么会进行批量更新，自己调用的方法比如setTimeout，xhr等则是连续更新。批量更新走的是事物更新batchedUpdates。也就是如下（图）

React在组件更新方面做了很多优化，这其中就包括了批量更新。在componentDidMount中执行了N个setState，如果执行N次更新是件很傻的事情。React利用其独特的事务实现，做了这些优化。再使用this.state时一定要注意组件的生命周期，很多时候在获取state的时候，组件更新还未完成，this.state还未改变，这是很造成bug

2. Vue 的 DOM 更新是异步执行的。理解这一点非常重要。当侦测到数据变化时， Vue 会打开一个队列，然后把在同一个事件循环 (event loop) 当中观察到数据变化的 watcher 推送进这个队列。假如一个 watcher 在一个事件循环中被触发了多次，它只会被推送到队列中一次。然后，在进入下一次的事件循环时， Vue 会清空队列并进行必要的 DOM 更新。在内部，Vue 会使用 MutationObserver 来实现队列的异步处理，如果不支持则会回退到 setTimeout(fn, 0)。

3.Vue是如何利用事件循环的。需要明白两个东西，Microtask与MutationObserver（这两个东西加分！！）

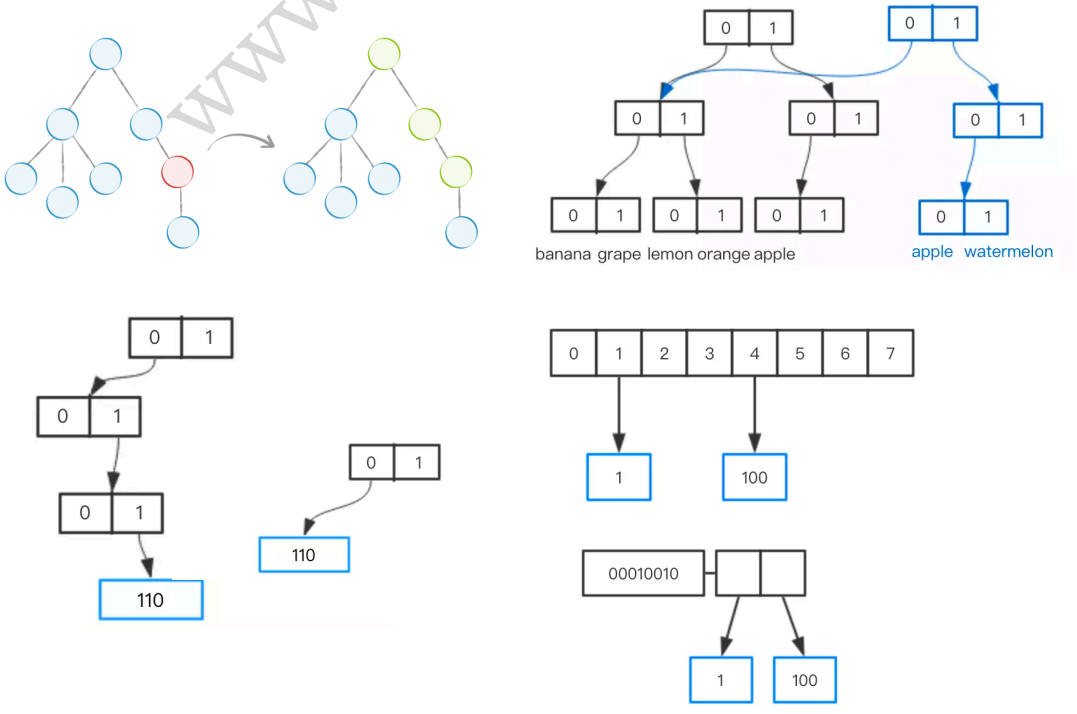


- 3.React生命周期的shouldComponentUpdate非常重要，请描述它代表什么并如何利用它进行性能调优。你还知道其他的生命周期么？在最新的版本中生命周期的改动？(12分)

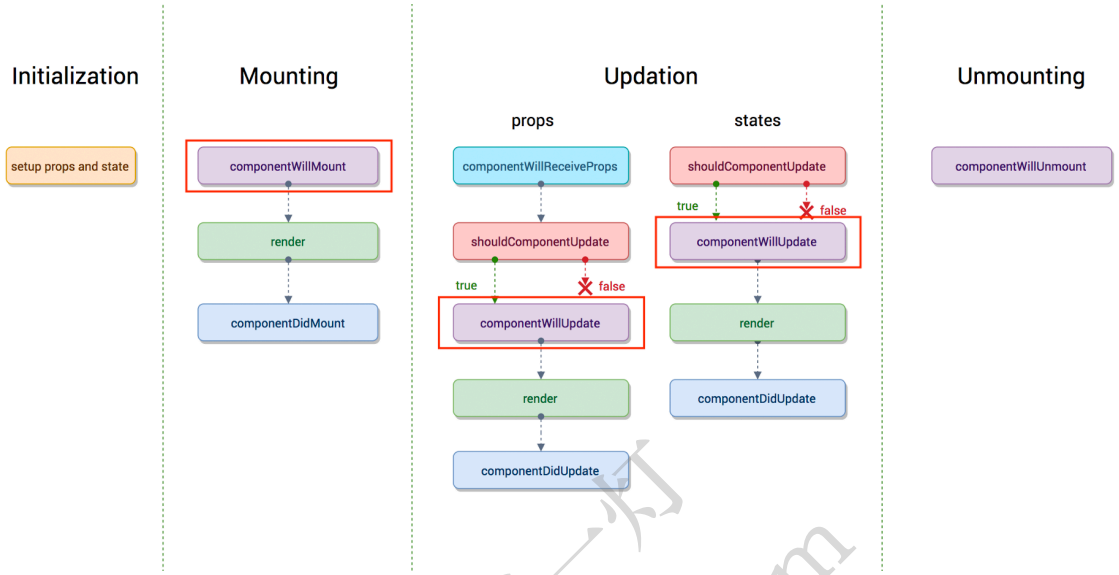
答：1.ShouldCompleteUpdate就是指明什么时候component（组件）需要进行更新。简单类型的比较如下。shouldComponentUpdate: function(nextProps, nextState) {

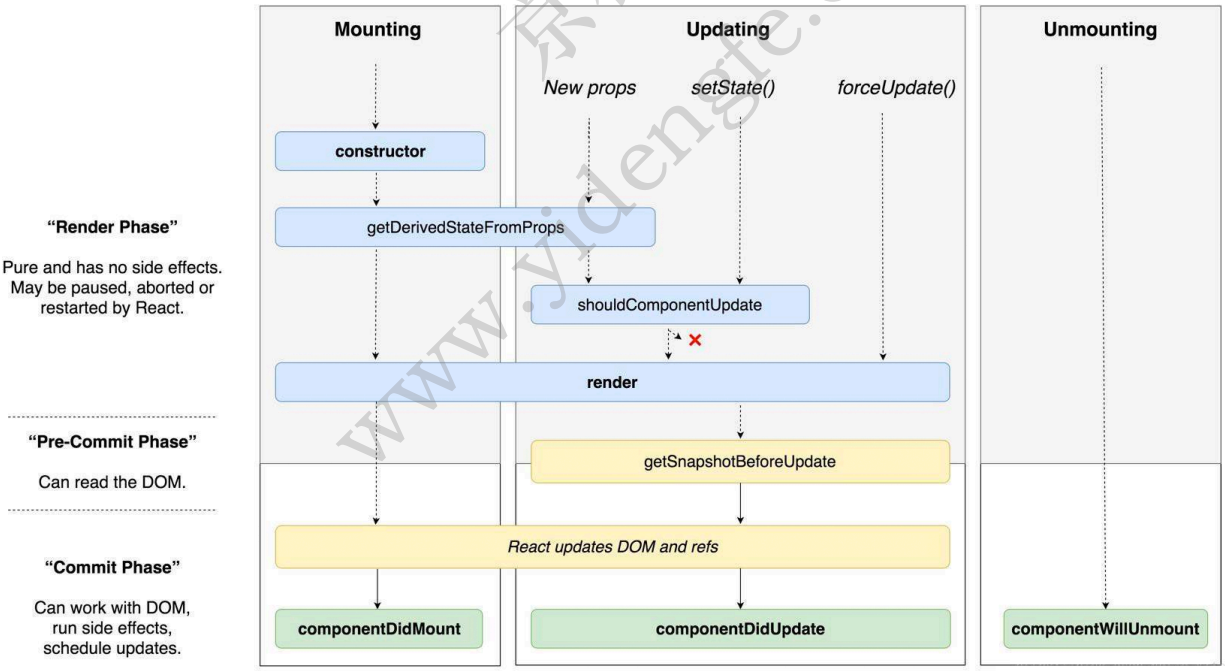
return nextProps.label !== this.props.label;

}复杂的类型 使用 Immutable.js,有关于 Immutable原理这块还是希望大家能够好好的复习整理一下。Immutable首先使用Vector trie存储数字分区方式（然后实现了一个hash函数）存储节点，可以把一个值转换成相应数字，接下来Hash maps array trie 压缩tree。Immutable一个基本的结构 HashArrayMapNode，拥有的子节点数量 ＞16 ，拥有的数组长度为 32、BitmapIndexedNode，拥有的子节点数量 ≤16 ，拥有的数组长度与子节点数量一致，经由 bitmap 压缩
、ValueNode，叶子节点，存储 key 和 value、HashCollisionNode的节点去处理发生冲突的键值。



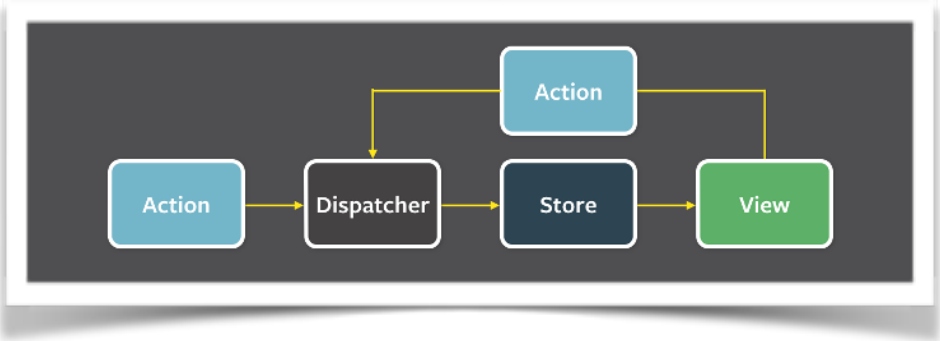
2.具体的React16.8相关的具体API变动大家看直播课哈
3.放上前后生命周期对比图如下：





- 4.如下是Flux架构图，请描述每一部分所代表的含义。(5分)

- 答：



🍄必知必会要不Redux就彻底泡汤了：

1.用户访问 View
2.View 发出用户的 Action
3.Dispatcher 收到 Action，要求 Store 进行相应的更新
4.Store 更新后，发出一个"change"事件
5.View 收到"change"事件后，更新页面

- 5.请描述Vuex的几大模块和关键作用（提示：State、Getters、Mutations、Actions、Modules） （20分）

- 答：

🍄必会（如果可以。。我说如果。。你要研究一下它的源码）：

1.State 单一状态树，用一个对象就包含了全部的应用层级状态，对象展开符号.....mapState({})，

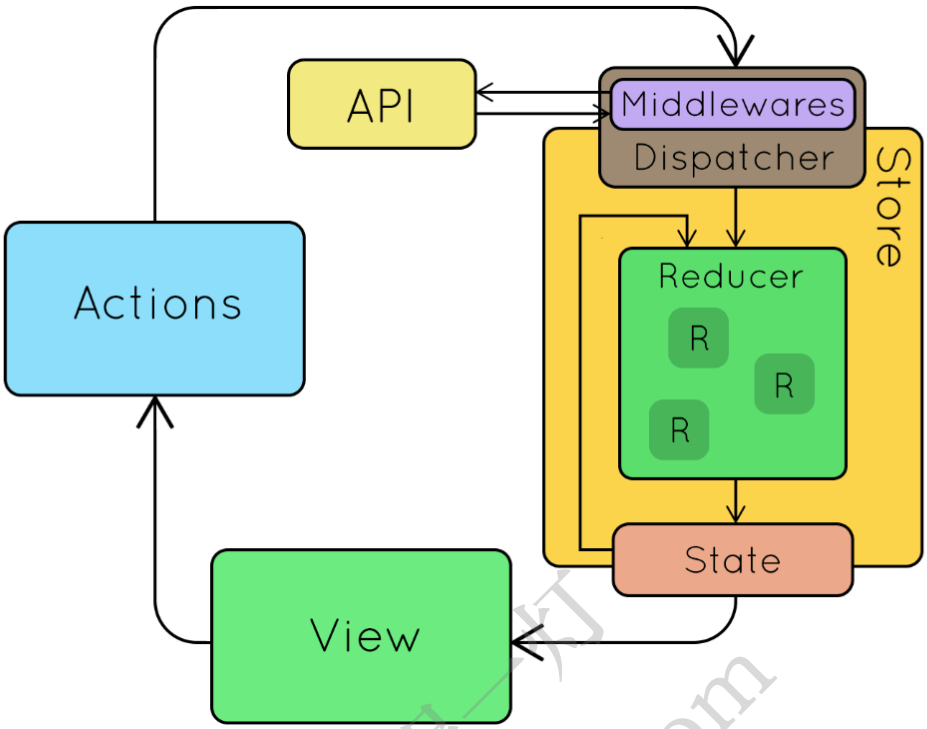
2.Getters 从 store 中的 state 中派生出一些状态，mapGetters 辅助函数仅仅是将 store 中的 getters 映射到局部计算属性 ，当一个组件需要获取多个状态时候，将这些状态都声明为计算属性会有些重复和冗余。

3.Mutations更改 Vuex 的 store 中的状态的唯一方法是提交 mutation（不能直接调用句柄，必须通过触发）mutation 必须是同步函数，通常使用常量替代 Mutation 事件类型。每个 mutation 都有一个字符串的 事件类型 (type) 和 一个 回调函数 (handler)

4.Actions提交的是 mutation，Action 可以包含任意异步操作。mapActions 辅助函数将组件的 methods 映射为 store.dispatch 调用

5.Modules 运行我们将 store 分割到模块（module）。每个模块拥有自己的 state、mutation、action、getters、甚至是嵌套子模块——从上至下进行类似的分割。 store 创建之后，你可以使用 store.registerModule 方法注册模块。

- 6.请描述Redux每一部分的作用。 （10分）



- 答：

  请看图说话，务必能够将整体流程详细描述（不懂？这个得看看上课咱们课程了）这里我们要求大家能够手写出Redux，群里有分步骤的代码拆解方案，希望大家能够从头敲一遍。哪里不懂再随时呼叫老袁。同时大家也可以关注下Mobx，他跟最新的React Hooks更新还是很密切的。如下是联想下和第一周的函数式编程的关系。

- * store -> container
- * currentState -> __value
- * action -> f
- * currentReducer -> map
- * middleware -> IO functor (解决异步操作的各种问题。)

- 7.请用你的知识解释一下GraphQL和Relay？(15分)

- 答：

这是跟RESTful同样地位的新趋势，这里可以在面试过程中稍微多加谈论，必成加分项：

1.GraphQL 既是一种用于 API 的查询语言也是一个满足你数据查询的运行时。 GraphQL 对你的 API 中的数据提供了一套易于理解的完整描述，使得客户端能够准确地获得它需要的数据，而且没有任何冗余，也

让 API 更容易地随着时间推移而演进，还能用于构建强大的开发者工具。其中query，mutation是核心处理数据必备的两种类型，大家还是要记得不停的尝试遇到问题💊及时解决。

2.Relay是一个框架，它负责把React组件连接到GraphQL服务器，此连接是通过一个实现了动作、调度器和存储的容器实现的。开发人员不需要对动作、调度器和存储进行编程，而可以触发这些动作并通过Relay API来访问相应的结果。若要配置容器，开发人员必须提供GraphQL查询和突变片段(mutation fragments)向容器描述数据的图结构;此外，Relay还会负责照顾数据管理的所有细节。

（其实关于实现这些技术还有非常多的技术框架比如Redux的替代品mobx，GraphQL的Sangria及lokka）基于GraphQL的BFF架构也是可以给企业解耦的。

- 8.请写出用Vue实现单页应用的全部技术细节，并阐述SSR哪里最为耗时？（5分）

- 答：

🍄可以分成几个具体的步骤：

1. 客户端Webpack使用 new VueSSRClientPlugin() 生成vue-ssr-client-manifest.json

2.服务端Webpack使用 new VueSSRServerPlugin() 生成vue-ssr-server-bundle.json

3.node使用createBundleRenderer进行Bigpipe输出,前端注意使用window

4.Vue组件内部要自定义异步请求事件，因为SSR进行输出的过程要提前fetch

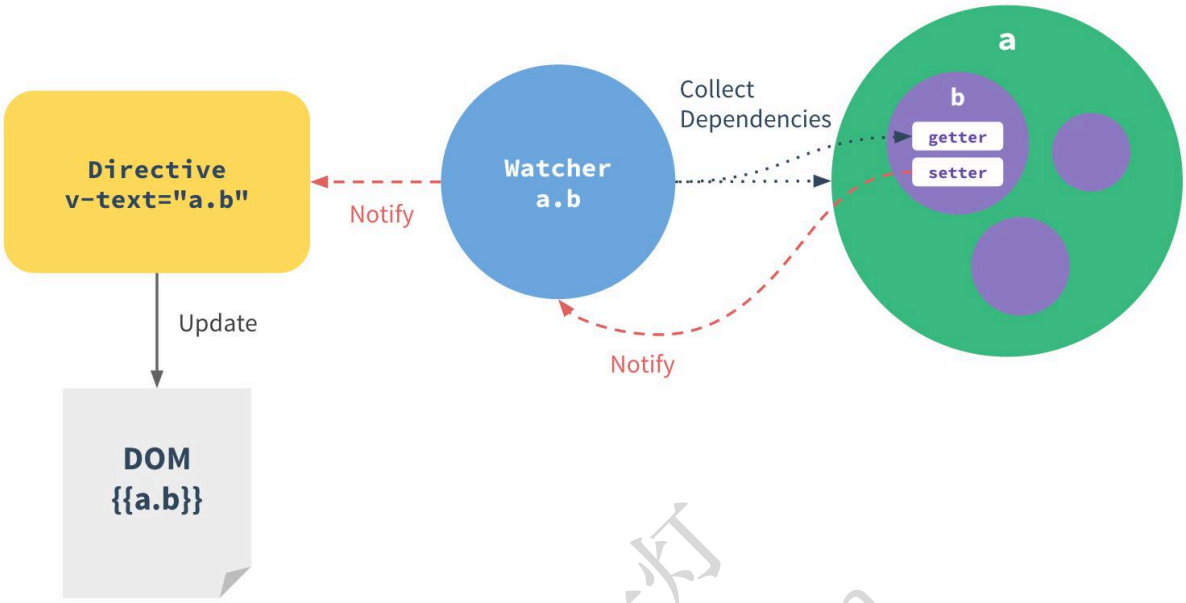
5.后台数据成功之后把值打到__INITIAL_STATE__ Vuex直接注入到state

6.最耗时在后端解析模板，使用lru-cache进行模板缓存。

- 9.请解释一下Vue源代码双向数据绑定的原理，对于最新Proxy版本你能描述么？(10分)

- 答：

🍄请看图说话，务必能够将整体流程详细描述（不懂？这块是课程的重头戏）,Vue3也即将发布，大家也要把基于Proxy的版本讲清楚，其实对于Proxy很有用，比如现在需要使用函数式编程上的执行代码并对执行堆栈进行重新的编排。function yideng(){} yideng.a = ...; yideng.b = ...; 这个时候我们就可以代理到yideng上然后每次执行a、b的时候先记录在编排（类似于fiber）



- 10.请了解Zone.js 和RX.js么。你能解释一下前端IOC的原理么？

- 答：

这两库是AngularJS的灵魂：

1.zone.js为JavaScript提供了执行上下文，可以在异步任务之间进行持久性传递，看代码

```
var log = function(phase){
    return function(){
        console.log("I am in zone.js " + phase + "!");
    };
};
zone.fork({
    onZoneCreated: log("onZoneCreated"),
    beforeTask: log("beforeTask"),
    afterTask: log("afterTask"),
}).run(function(){
    var methodLog = function(func){
        return function(){
```

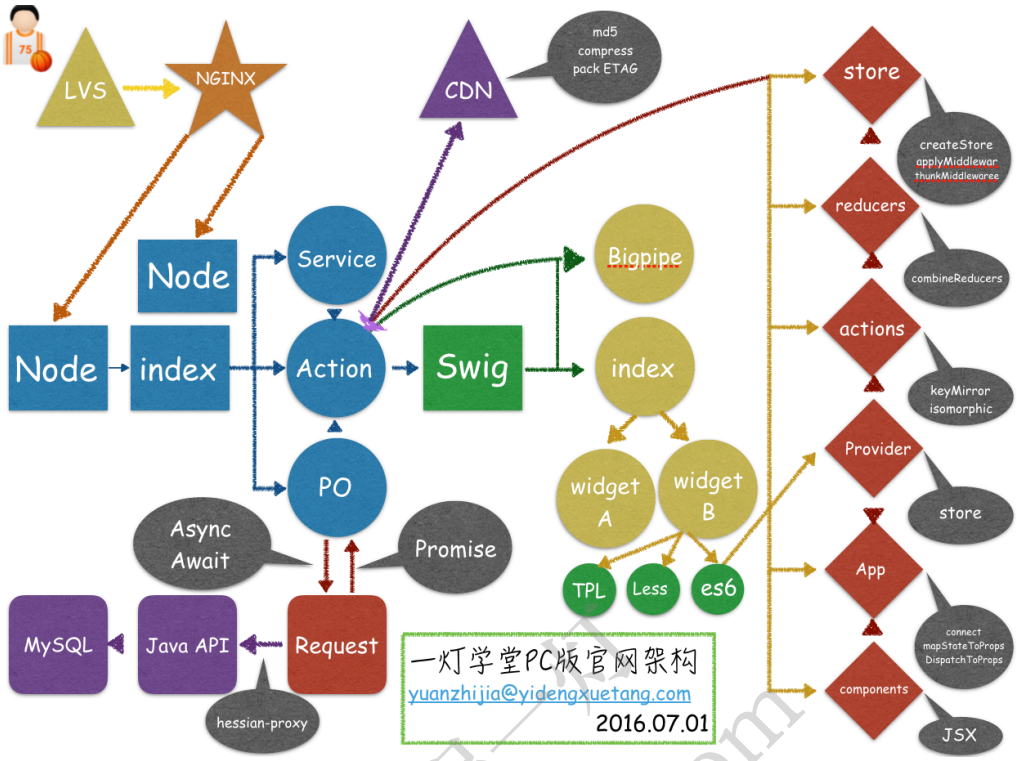
```
            console.log("I am from " + func + " function!");
        };
    },
    foo = methodLog("foo"),
    bar = methodLog("bar"),
    baz = function(){
        setTimeout(methodLog('baz in setTimeout'), 0);
    };

    foo();
    baz();
    bar();
});
```

执行这段示例代码的输出是：

```
I am in zone.js beforeTask!
I am from foo function!
I am from bar function!
I am in zone.js afterTask!
I am in zone.js onZoneCreated!
I am in zone.js beforeTask!
I am from baz in setTimeout function!
I am in zone.js afterTask!
```

2.RxJS 是使用 Observables 的响应式编程的库，它使编写异步或基于回调的代码更容易。这个项目是 Reactive-Extensions/RxJS(RxJS 4) 的重写，具有更好的性能、更好的模块性、更好的可调试调用堆栈，同时保持大部分向后兼容，只有一些破坏性的变更(breaking changes)是为了减少外层的 API 。
3.依赖注入（Dependency Injection）是用于实现控制反转（Inversion of Control）的最常见的方式之一，本质原理就是把需要的类注入到接口供你调用。

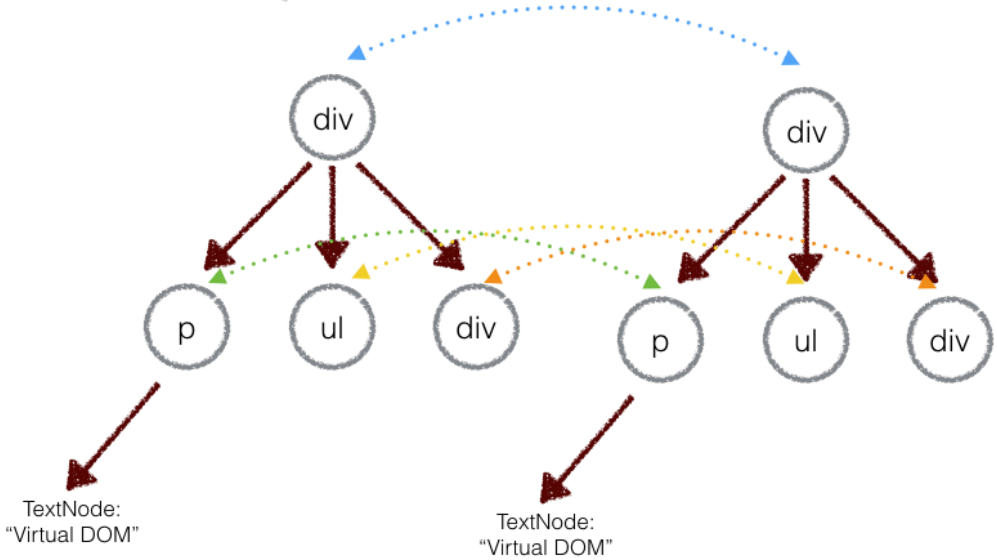
- 11.请写出如下的架构的全部技术思路。



- 答：

- 前端架构师启蒙课，咱们到底都学了些什么？？？来，请开始你的表演。这个图在面试的时候你可以当成你自己的 🔔

- 12.我给你空了这么一大页的纸，就是想让你知道接下来是个送分题。请先生或者女士解释什么叫Virtual DOM并默写Dom Diff的代码?



- 答：

🍄以下是老袁总结的一些考点：

对新旧两棵树进行一个深度优先的遍历，这样每个节点都会有一个唯一的标记，在深度优先遍历的时候，每遍历到一个节点就把该节点和新的的树进行对比。如果有差异的话就记录到一个对象里面。p是patches[1]，ul是patches[3]，类推。

简陋版diff https://github.com/livoras/simple-virtual-dom/blob/master/lib/diff.js

简陋版path https://github.com/livoras/simple-virtual-dom/blob/master/lib/patch.js

- 13.请解释React的新引擎—React Fiber是什么?

- 答：

这块我们在群内有非常详细React16.9源码分析的的文档（附录也我放了哈），具体的scheduler源码requestAnimationFrame+MessageChannel模拟requestIdleCallback的源码老袁是详细些的注释的哈，大家好好复习复习哈，这块面试肯定是有大用的。

- 14.Redux的原理?vue-router的原理还有高阶组件？

- 答：1.Redux基本完全是基于函数式编程的，范畴论将世界抽象为对象和对象之间的联系，Redux将所有事件抽象为action，第一周我们学过Container 中含有 __value 和 map 两个属性，而修改 __value 的方法只有 map，在操作完 __value 后将新值放回 Container 中。如何操作或修改 __value 由 f 给出。对照如下 【store -> container】【 currentState -> __value】【action -> f】【 currentReducer -> map】【middleware -> IO functor (解决异步操作的各种问题。)】

  2.Vue-router 完全基于是pushstate的，这块知识的话主要是怎么实现CSR+SSR混合搭配。加油👊

  比如vue-cli+koa 核心使用koa2-connect-history-api-fallback 真假路由混合。

前端路漫漫，这是最好的结束，也是全新的开始。京程一灯永远是您的后盾，无论何时需要帮助，我们永远都在，无论你在哪，如果需要任何帮助请随时联系我们，祝好~