



React在大型后台管理项目中的工程实践

黄健 @今日头条 

自我介绍

黄健 前端工程师



@taikongfeizhu



@Abcat



huangjian@bytedance.com

2015 - 至今	今日头条	效率工程和客户增长 负责业务系统的前端开发和工程化构建
2014 - 2015	搜狐新闻	客户端研发部 负责前端基础平台和Hybird混合型架构开发
2013 - 2014	高德地图	移动应用开发部 负责LBS功能组件开发和活动运营研发
2010 - 2013	东华软件	重点客户事业部 负责企业系统集成平台和通用解决方案研发



TOPIC

01

业务情况概述

02

前端工程化设计

03

Redux项目结构改进

04

异步数据流优化

05

持续构建

Part1 业务情况概述

React技术栈产品线



业务复杂

由于业务系统本身是为满足用户的工作需要，系统往往基于复杂的数据流和业务逻辑



协同开发

大型业务系统的开发流程通常是将一个大功能点会拆解成多个小的业务模块，再由多人协作，同时开发



多端并行

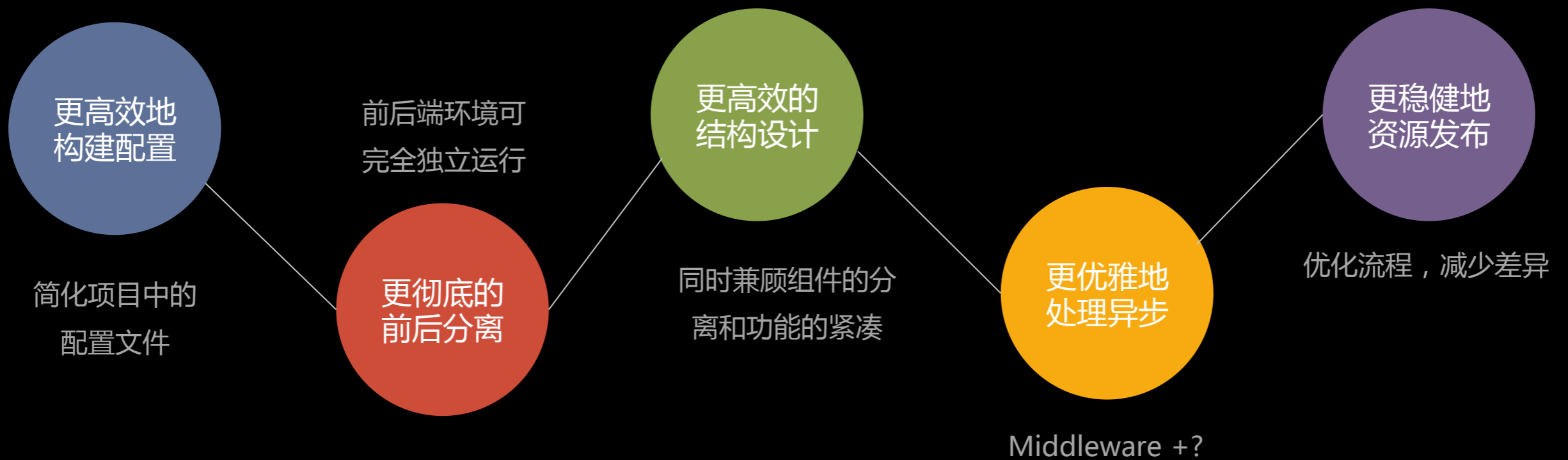
伴随着移动设备的普及和移动办公的效率需求，多数时候需要将大型业务中的高频功能同时拆解成PC模块和手机模块，多端同时进行开发



持续迭代

大型业务系统的由于用户的特殊性，对系统的持续迭代和功能响应都有着极高要求

诉求



Part2 前端工程化设计

工具链



webpack: 工程中的资源依赖管理和分析编译工具



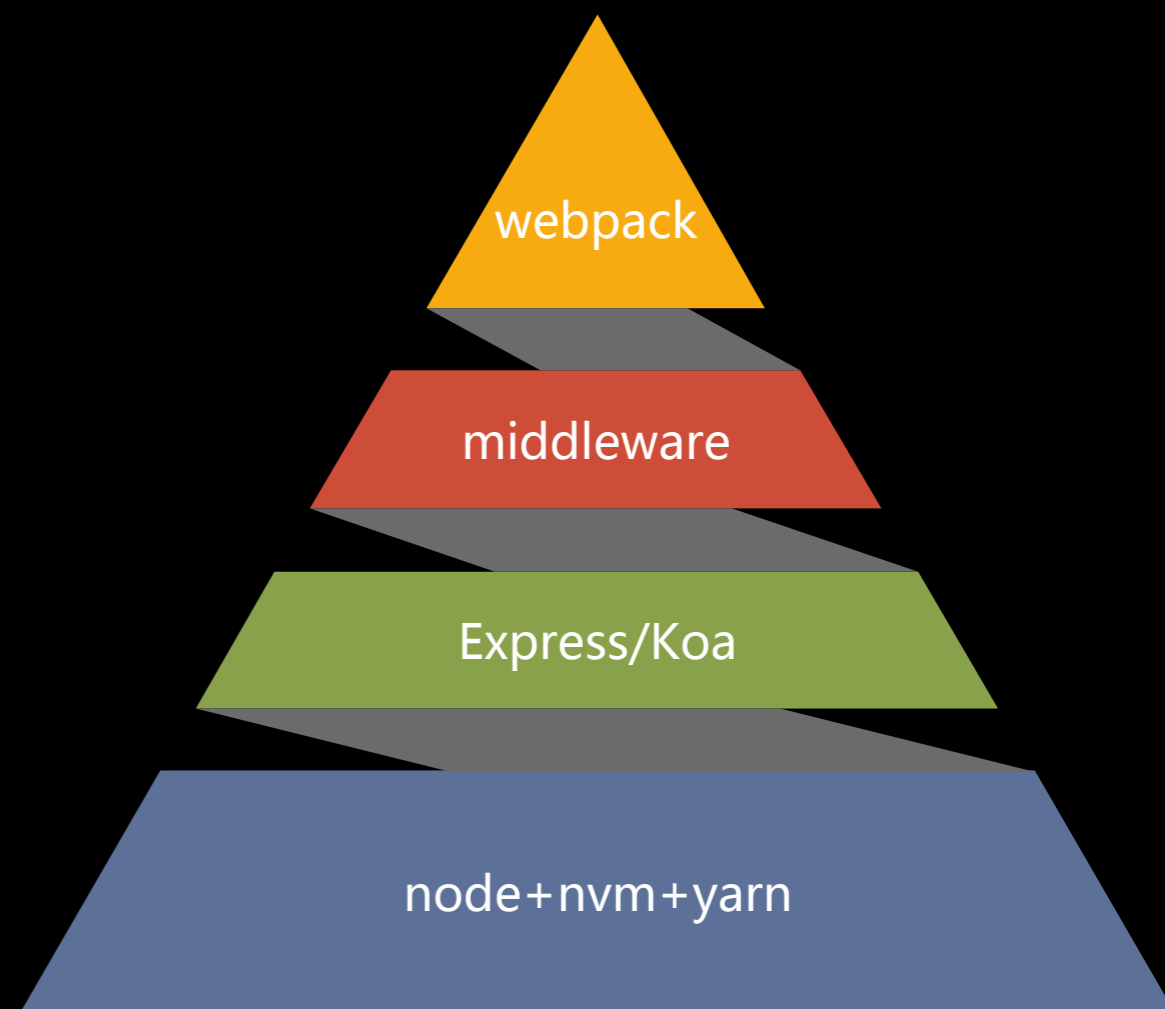
Middleware: 服务框架的中间件，能非常方便的为我们的前端环境提供各种增强服务功能



Express/Koa: 基础web服务框架

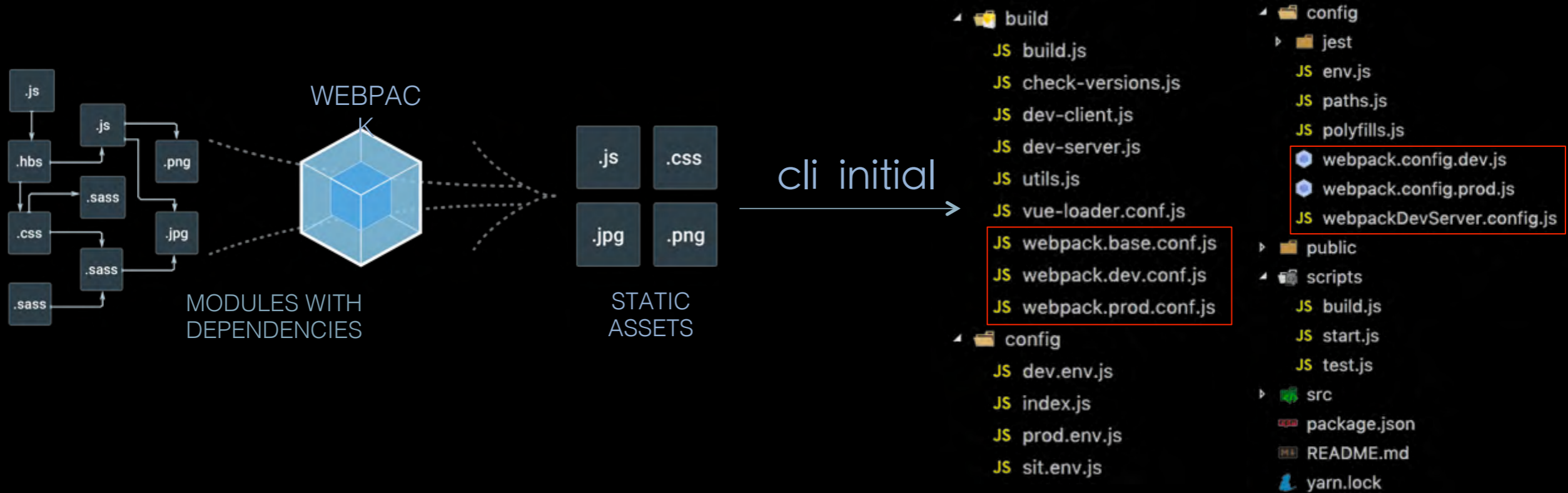


node: 底层依赖开发环境，yarn/npm作为资源管理工具，nvm用于管理多个node版本



仅此而已？

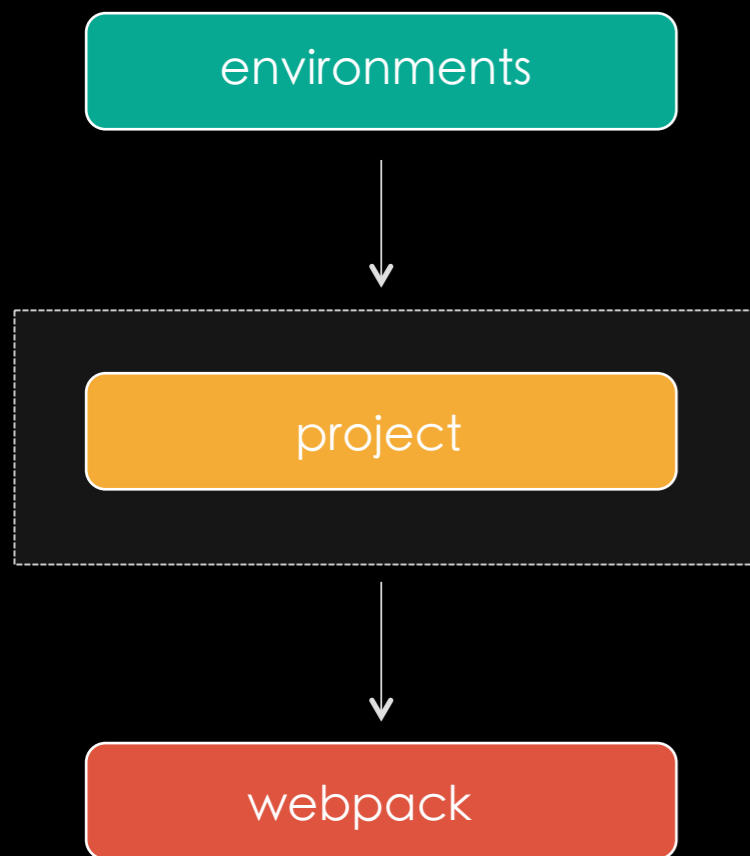
webpack结构



重复配置

```
webpack.config.dev.js x  webpack.config.prod.js x
55 // changing JS code would still trigger a refresh.
56 ],
57 output: {
58 // Next line is not used in dev but WebpackDevServer crashes without it:
59 path: paths.appBuild,
60 // Add /* filename */ comments to generated require()s in the output.
61 pathinfo: true,
62 // This does not produce a real file. It's just the virtual path that is
63 // served by WebpackDevServer in development. This is the JS bundle
64 // containing code from all our entry points, and the Webpack runtime.
65 filename: 'static/js/bundle.js',
66 // There are also additional JS chunk files if you use code splitting.
67 chunkFilename: 'static/js/[name].chunk.js',
68 // This is the URL that app is served from. We use "/" in development.
69 publicPath: publicPath,
70 // Point sourcemap entries to original disk location
71 devtoolModuleFilenameTemplate: info =>
72   path.resolve(info.absoluteResourcePath),
73 },
74 resolve: {
75 // This allows you to set a fallback for where Webpack should look for
76 // modules.
77 // We read 'NODE_PATH' environment variable in 'paths.js' and pass paths
78 // here.
79 // We placed these paths second because we want 'node_modules' to "win"
80 // if there are any conflicts. This matches Node resolution mechanism.
81 // https://github.com/facebookincubator/create-react-app/issues/253
82 modules: ['node_modules', paths.appNodeModules].concat(paths.nodePaths),
83 // These are the reasonable defaults supported by the Node ecosystem.
84 // We also include JSX as a common component filename extension to support
85 // some tools, although we do not recommend using it, see:
86 // https://github.com/facebookincubator/create-react-app/issues/290
87 extensions: ['.js', '.json', '.jsx'],
88 alias: {
89 devtool: 'source-map',
90 // In production, we only want to load the polyfills and the app code.
91 entry: [require.resolve('./polyfills'), paths.appIndexJs],
92 output: {
93 // The build folder.
94 path: paths.appBuild,
95 // Generated JS file names (with nested folders).
96 // There will be one main bundle, and one file per asynchronous chunk.
97 // We don't currently advertise code splitting but Webpack supports it.
98 filename: 'static/js/[name].[chunkhash:8].js',
99 chunkFilename: 'static/js/[name].[chunkhash:8].chunk.js',
100 // We inferred the "public path" (such as / or /my-project) from
101 // homepage |
102 publicPath: publicPath,
103 // Point sourcemap entries to original disk location
104 devtoolModuleFilenameTemplate: info =>
105   path.relative(paths.appSrc, info.absoluteResourcePath),
106 },
107 resolve: {
108 // This allows you to set a fallback for where Webpack should look for
109 // modules.
110 // We read 'NODE_PATH' environment variable in 'paths.js' and pass
111 // paths here.
112 // We placed these paths second because we want 'node_modules' to "win"
113 // if there are any conflicts. This matches Node resolution mechanism.
114 // https://github.com/facebookincubator/create-react-app/issues/253
115 modules: ['node_modules', paths.appNodeModules].concat(paths.nodePaths),
116 // These are the reasonable defaults supported by the Node ecosystem.
117 // We also include JSX as a common component filename extension to
118 // support
119 // some tools, although we do not recommend using it, see:
120 // https://github.com/facebookincubator/create-react-app/issues/290
121 extensions: ['.js', '.json', '.jsx'],
122 alias: {
```

抽象配置



“抽象优于重复”



`environments.config.js`

存放在特定环境下的需要变化的配置参数和设置
通常包含有：publicpath、devtools、warnings、uglify、hash等



`project.config.js`

存放于项目有关信息，如：server, output, loader, externals, plugin等配置；通过overrides实现对environments中的配置信息重载



`webpack.config.js`

主要是读取project.config.js中的配置，再按标准的webpack字段填入project中的配置信息，原则上是该文件中的信息只与构建工具有关，而与具体的项目工程配置无关，可以做到跨项目间复用

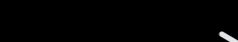
配置重构

- config
 - environments.config.js
 - project.config.js
 - webpack.config.js

object (env)



object.assign (override)



project.conf

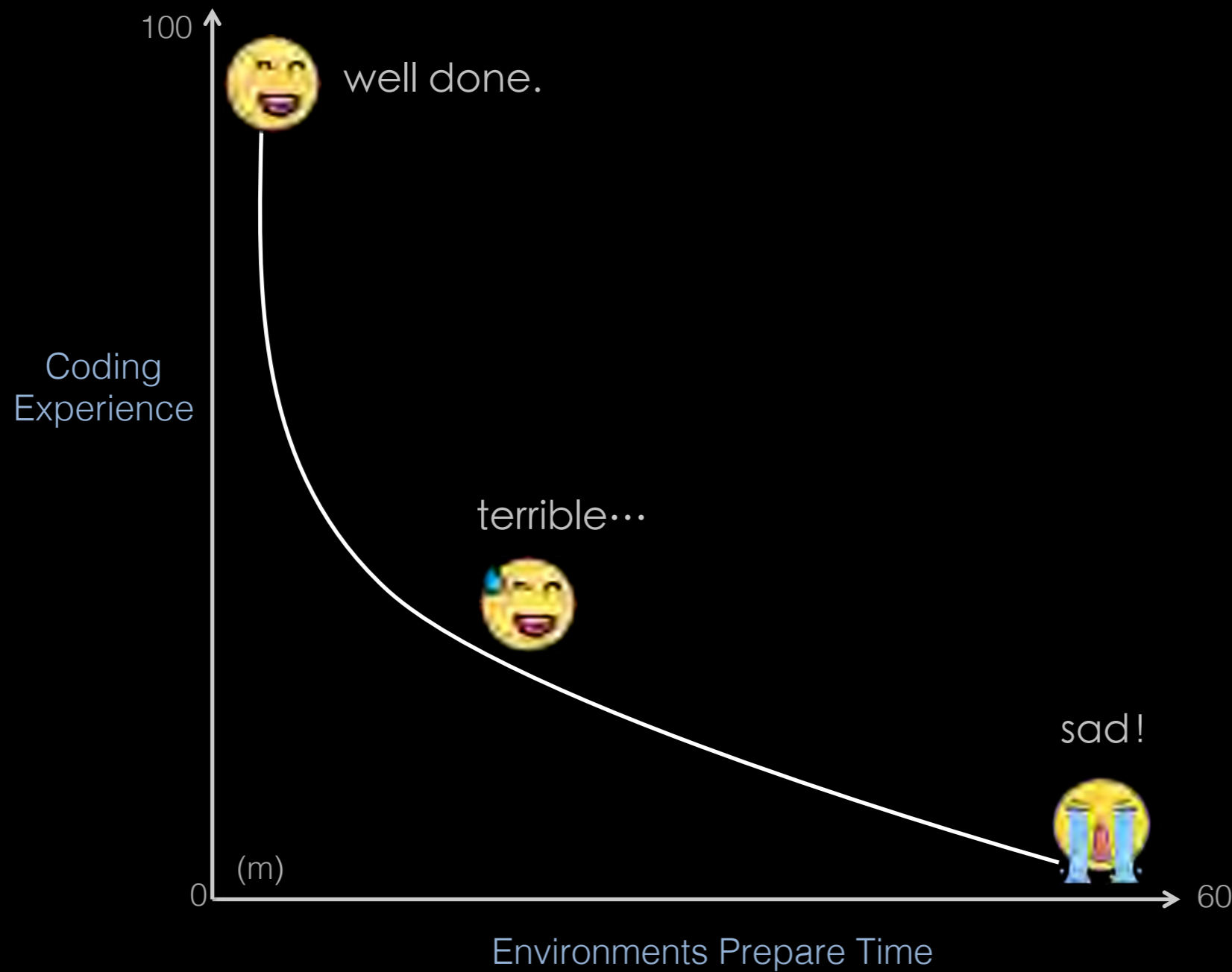
```
JS environments.config.js
1 module.exports = {
2   // =====
3   // Overrides when NODE_ENV === 'development'
4   // =====
5   // NOTE: In development, we use an explicit public path when the
6   // are served webpack by to fix this issue:
7   // http://stackoverflow.com/questions/34133808/webpack-ots-pars
8   development : (config) => ({
9     compiler_public_path : `http://${config.server_host}:${config.
10  }),
11
12  // =====
13  // Overrides when NODE_ENV === 'production'
14  // mersea_public_path: /static/mobile/
15  // express_public_path: /resource/os_mobile/
16  // cdn_public_path: //s3a.pstatp.com/cg_growth/resource/os_mobi
17  // =====
18  production : (config) => ({
19    compiler_public_path   : '//s3a.pstatp.com/cg_growth/resourc
20    compiler_base_route   : '/apps/',
21    compiler_fail_on_warning : false,
22    compiler_hash_type    : 'chunkhash',
23    compiler_devtool      : false,
24    compiler_stats        : {
25      chunks      : true,
26      chunkModules : true,
27      colors      : true
28    }
29  })
30 }
```

```
project.config.js
1 const path = require('path')
2 // =====
3 // Default Configuration
4 // =====
5 const config = {
6   env : process.env.NODE_ENV || 'development',
7
8   // =====
9   // Compiler Configuration
10  // =====
11  compiler_devtool      : 'source-map',
12  compiler_hash_type   : 'hash',
13  compiler_fail_on_warning : false,
14  compiler_quiet       : false,
15  compiler_public_path : '/',
16 }
17
18 // =====
19 // Environment Configuration
20 // =====
21 debug(`Looking for environment overrides for NODE_ENV "${config.env}".`
22 const environments = require('./environments.config')
23 const overrides = environments[config.env]
24 if (overrides) {
25   debug('Found overrides, applying to default configuration.')
26   Object.assign(config, overrides(config))
27 } else {
28   debug('No environment overrides found, defaults will be used.')
29 }
30 module.exports = config
```

```
webpack.config.js
1 const argv = require('yargs').argv
2 const os = require('os')
3 const webpack = require('webpack')
4 const project = require('./project.config')
5 const HtmlWebpackPlugin = require('html-webpack-plugin')
6 const ExtractTextPlugin = require('extract-text-webpack-plugin')
7 const UglifyJsParallelPlugin = require('webpack-uglify-parallel')
8
9 const __DEV__ = project.globals.__DEV__
10 const __PROD__ = project.globals.__PROD__
11 const __TEST__ = project.globals.__TEST__
12 const FILENAME = project.globals.filename
13
14 const webpackConfig = {
15   name : 'client',
16   target : 'web',
17   devtool : project.compiler_devtool,
18   resolve : {
19     modules : [project.paths.client(), 'node_modules'],
20     extensions : ['.web.js', '.js', '.jsx', '.json']
21   },
22   module : {}
23 }
24
25 // =====
26 // Entry Points
27 // =====
28 const APP_ENTRY = project.paths.client('main.js')
29
30 webpackConfig.entry = {
31   app : __DEV__
32     ? [APP_ENTRY].concat('webpack-hot-middleware/client?path=${proj
33     : [APP_ENTRY],
34   vendor : project.compiler_vendors
35 }
```

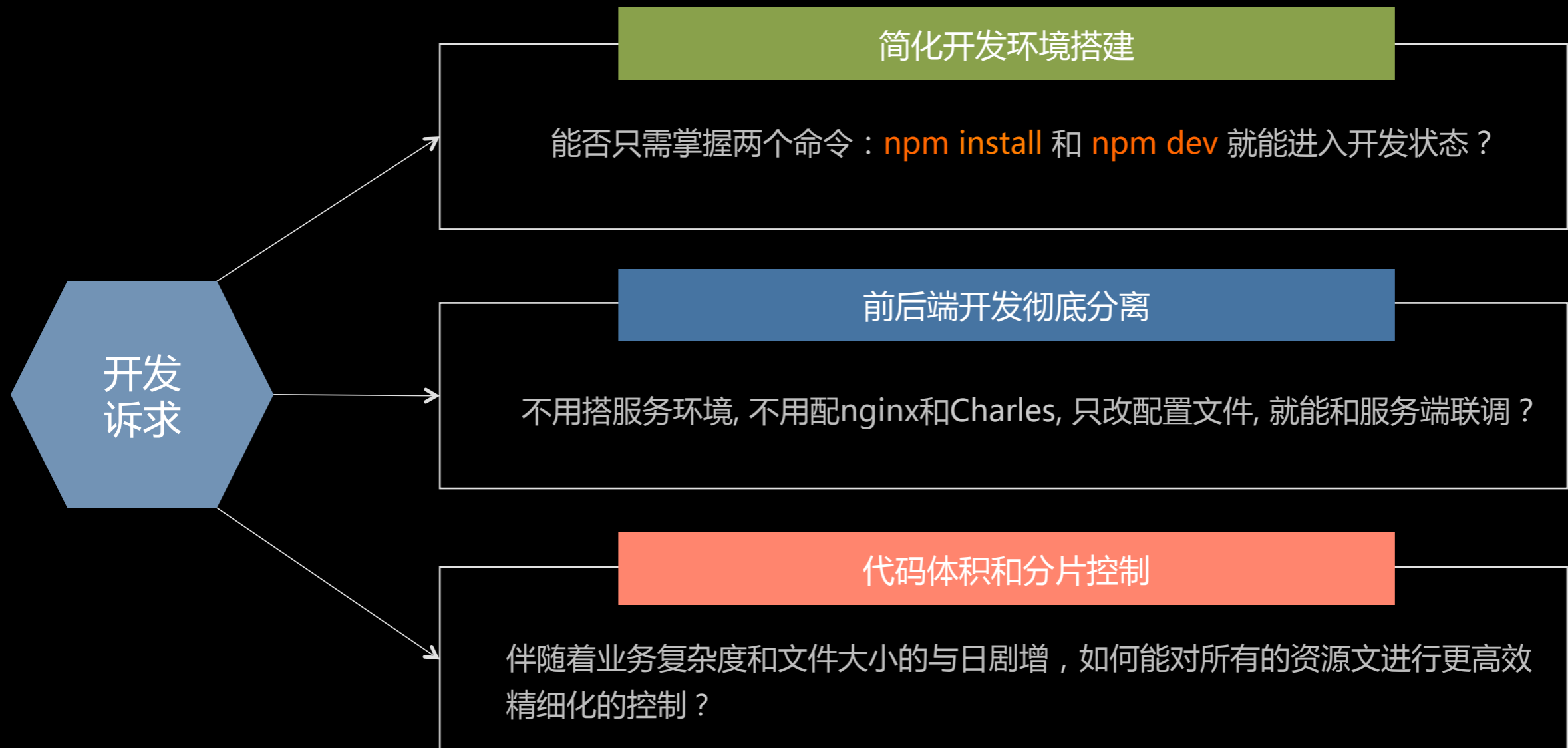
前后端分离

开发体验

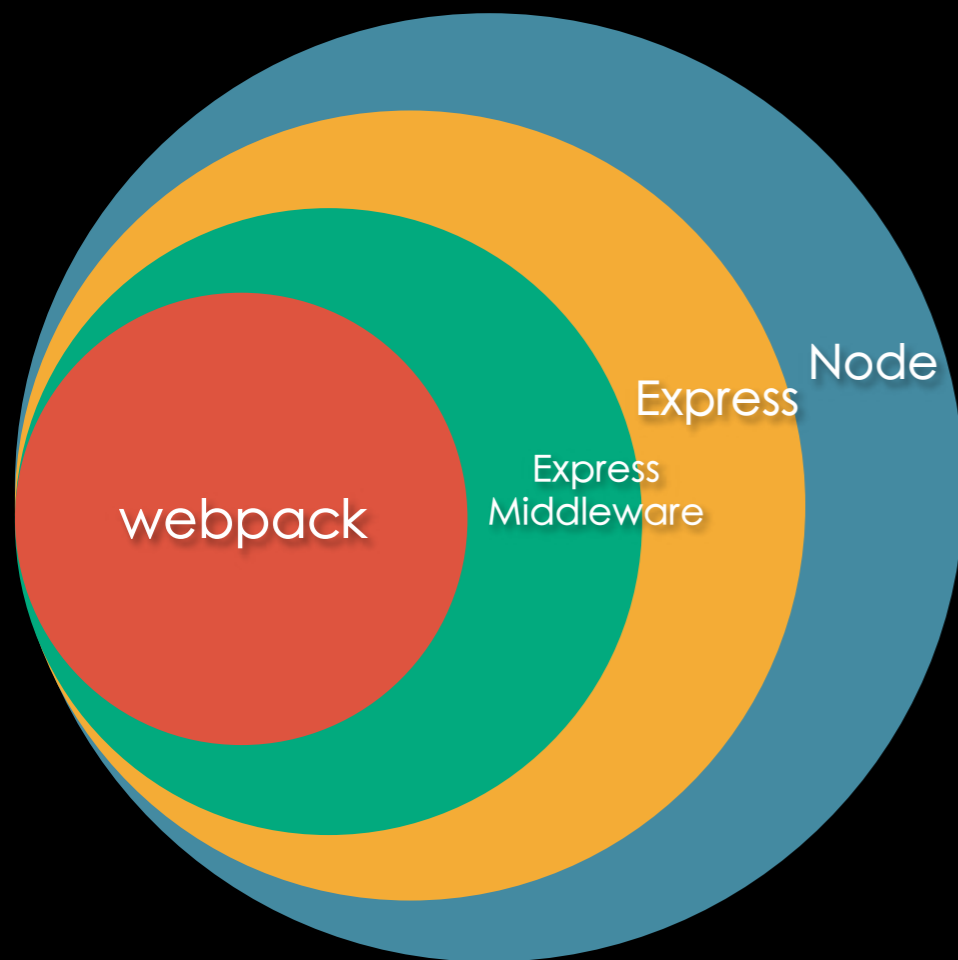


- PROJECT
 - bin
 - common
 - config
 - dev
 - apps
 - images
 - less
 - index.html
 - index.js
 - node_modules
 - output
 - public
 - server
 - .babelrc
 - .editorconfig
 - .eslintignore
 - .eslintrc
 - .gitignore
 - app.json
 - package.json
 - postcss.config.js
 - README.md
 - scm_build_resource.sh
 - scm_build.sh
 - scm_resource.py
 - settings.py
 - yarn.lock

开发诉求



环境搭建



依赖安装

```
npm install webpack-dev-middleware --dev
```

```
npm install webpack-hot-middleware --dev
```

```
npm install http-proxy-middleware --dev
```

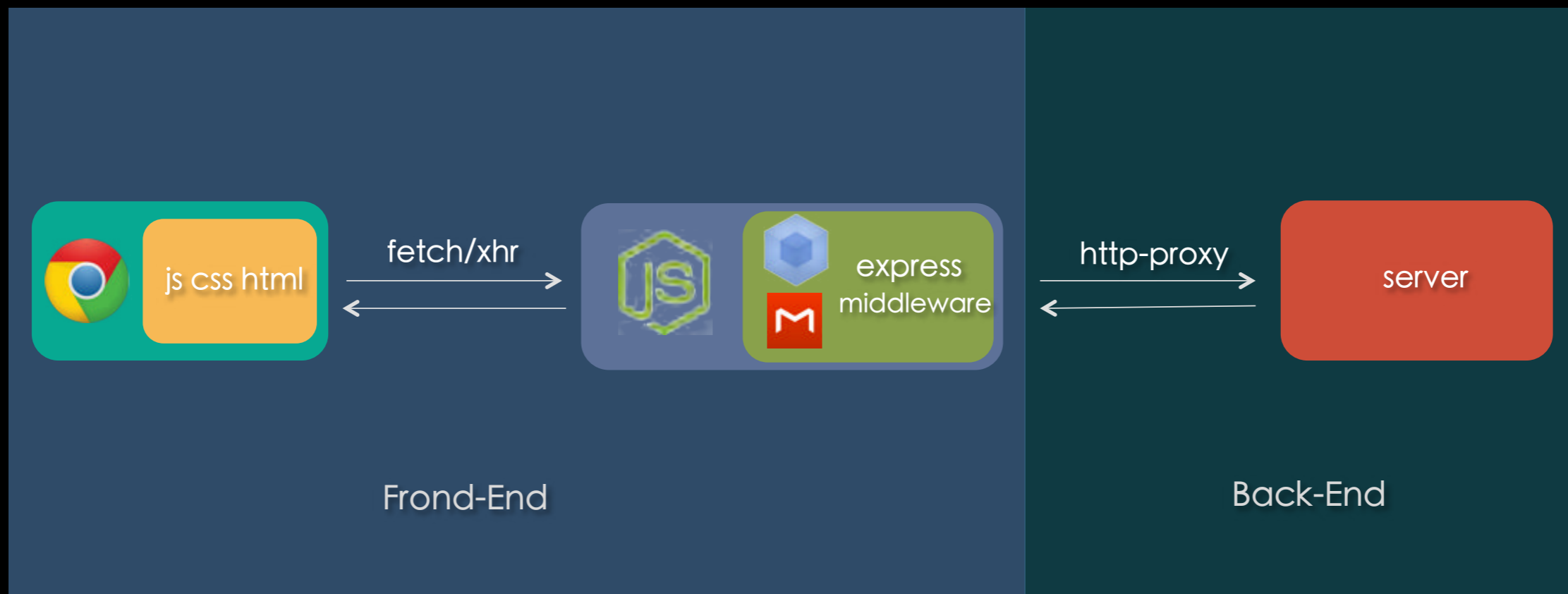
本地server (webpack-dev-server)

利用express搭建本地server环境，同时借用webpack-dev-middleware和webpack-hot-middleware为打包生成的资源文件提供Web(内存)服务，可将webpack和express集成到一个node进程中。

node-http-proxy

借用http-proxy-middleware实现跨域通信，本地的请求通过代理配置和url拦截，都可以转到本地服务器中进行请求转发

前后端通信



资源整合

```
5 main.js
// Apply Webpack HMR Middleware
// -----
if (project.env === 'development') {
  const compiler = webpack(webpackConfig)

  debug('Enabling webpack dev and HMR middleware')
  app.use(require('webpack-dev-middleware')(compiler, {
    publicPath : webpackConfig.output.publicPath,
    contentBase : project.paths.client(),
    hot        : true,
    quiet      : project.compiler_quiet,
    noInfo     : project.compiler_quiet,
    lazy       : false,
    stats      : project.compiler_stats
  })))
  app.use(require('webpack-hot-middleware')(compiler, {
    path: '/__webpack_hmr'
  })))
}
```

middleware

```
JS proxy.js
1 const project = require('../config/project.config')
2 const express = require('express')
3 const app = express()
4
5 // make http proxy middleware setting
6 const createProxySetting = function (url) {
7   return {
8     target: url,
9     changeOrigin: true
10  }
11 }
12
13 if (project.env === 'development') {
14   const proxyMiddleware = require('http-proxy-middleware')
15   const proxyConfig = [
16     {
17       url: '/back_end/page/*',
18       target: 'http://127.0.0.1:3000/mock'
19     },
20     {
21       url: '/back_end/*',
22       target: 'http://10.0.0.1:8351'
23     }
24   ]
25   // proxy middleware
26   proxyConfig.forEach(function (item) {
27     app.use(item.url, proxyMiddleware(createProxySetting(item.target)))
28   })
29 }
```

proxy

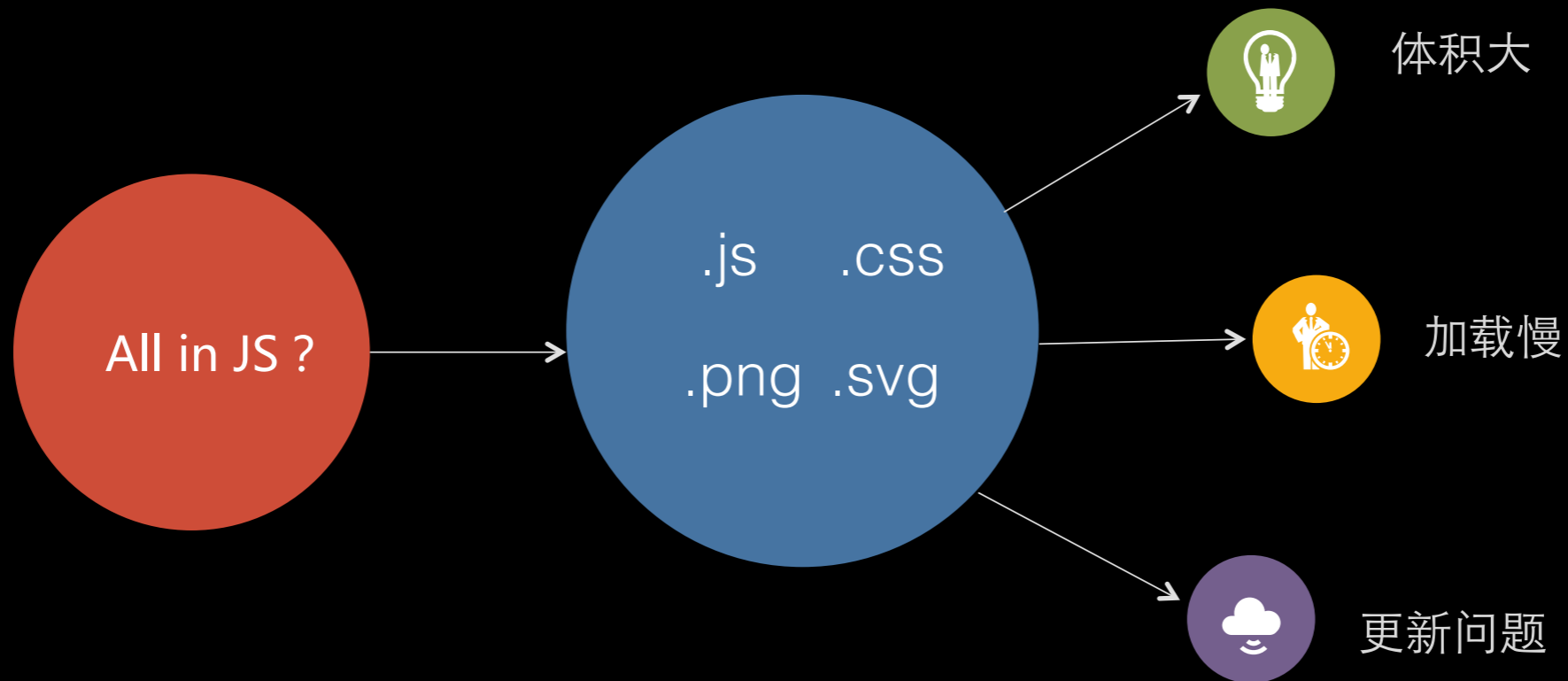
```
JS mock.js
1 // const mock = require('../routes/mock')
2 // app.use('/mock', mock)
3 const express = require('express')
4 const router = express.Router()
5 const opporList = require('../mock/opporList.json')
6 const Mock = require('mockjs')
7
8 router.get('/back_end/page/opporList', function (req, res) {
9   res.json(opporList)
10 })
11
12 router.get('/back_end/page/employee', function (req, res) {
13   var data = Mock.mock({
14     // 属性 list 的值是一个数组, 其中含有 1 到 10 个元素
15     'list|1-10': [{
16       // 属性 id 是一个自增数, 起始值为 1, 每次增 1
17       'id|+1': 1,
18       'name': 'name'
19     }]
20   })
21   res.json(data)
22 }
```

mock

```
{ "dev": "node ./bin/dev-server" }
```

代码分片

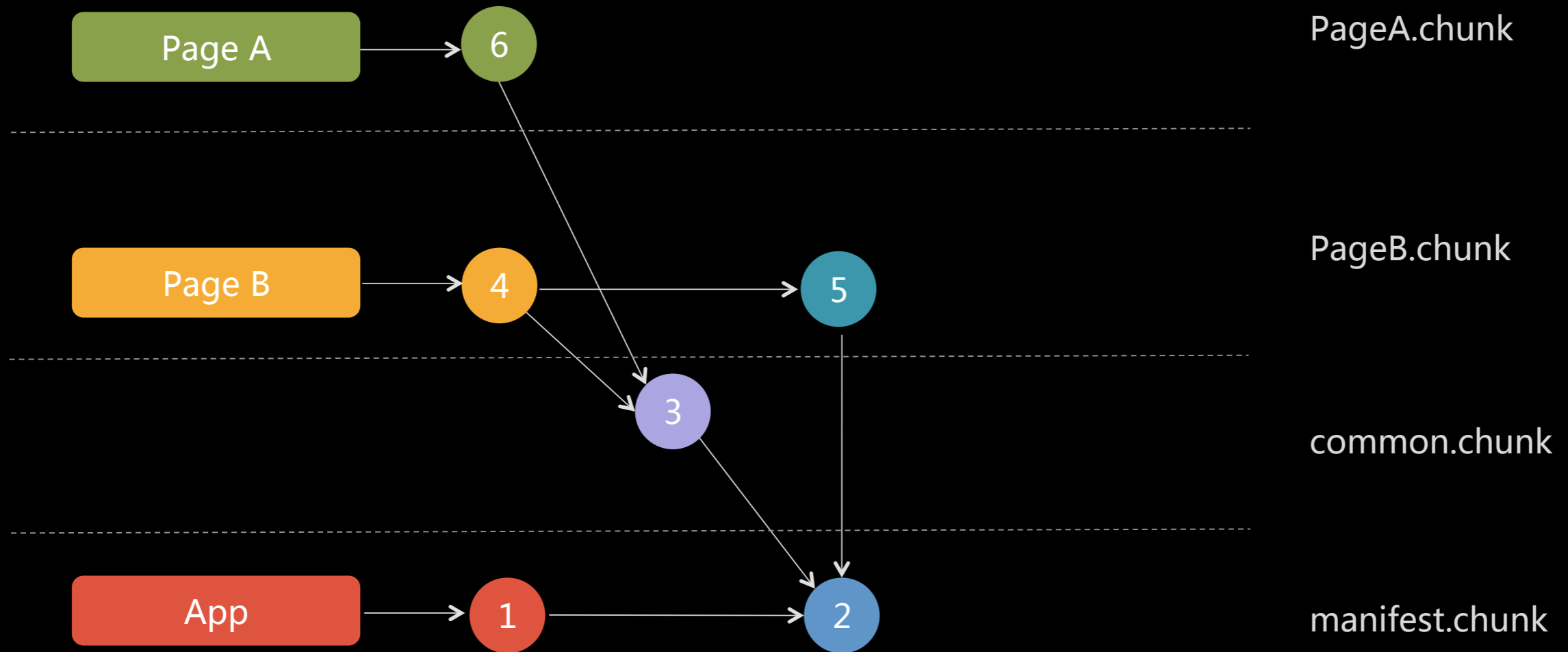
JSX的副作用



Asset	Size	Chunks		Chunk Names
js/app.001564477b55231a152a.js	5.19 MB	0	[emitted] [big]	app
js/vendor.001564477b55231a152a.js	1.33 MB	1	[emitted] [big]	vendor
js/manifest.001564477b55231a152a.js	30.4 kB	2	[emitted]	manifest
js/app.001564477b55231a152a.js.map	6.02 MB	0	[emitted]	app
js/vendor.001564477b55231a152a.js.map	1.57 MB	1	[emitted]	vendor
js/manifest.001564477b55231a152a.js.map	32.2 kB	2	[emitted]	manifest
favicon.ico	1.05 kB		[emitted]	
index.html	2.48 kB		[emitted]	

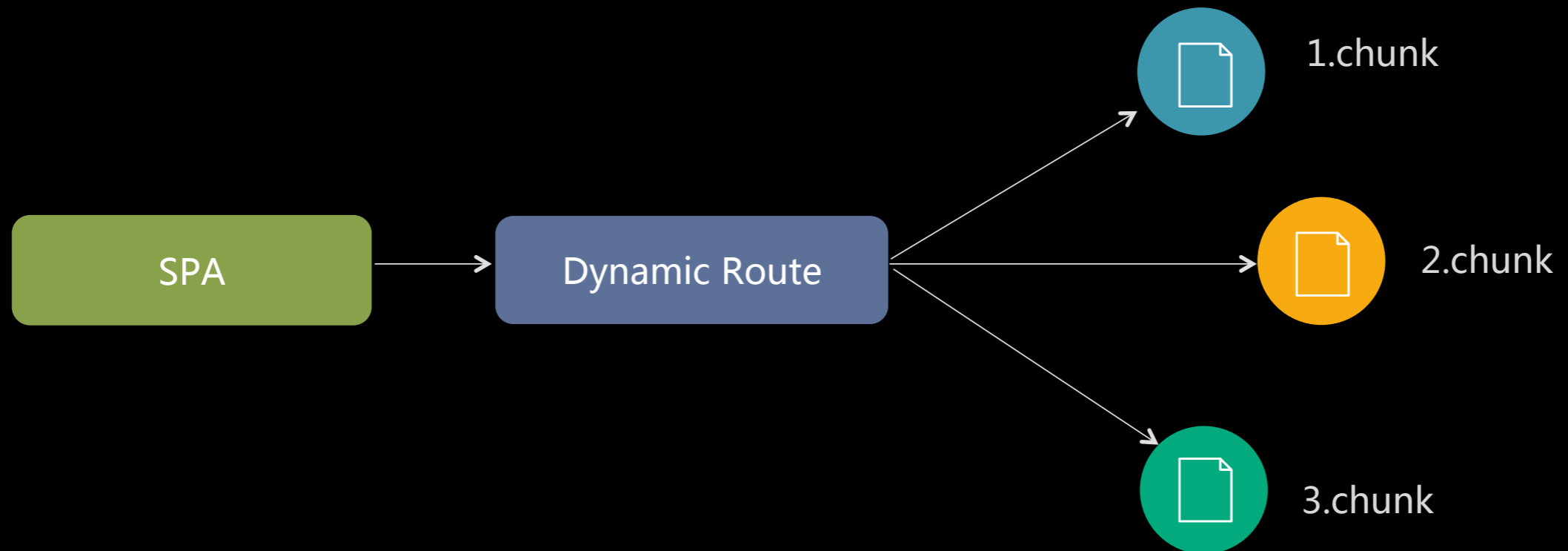
all in js \neq all in one js

模块提取



`webpack.optimize.CommonChunkPlugin`

动态加载

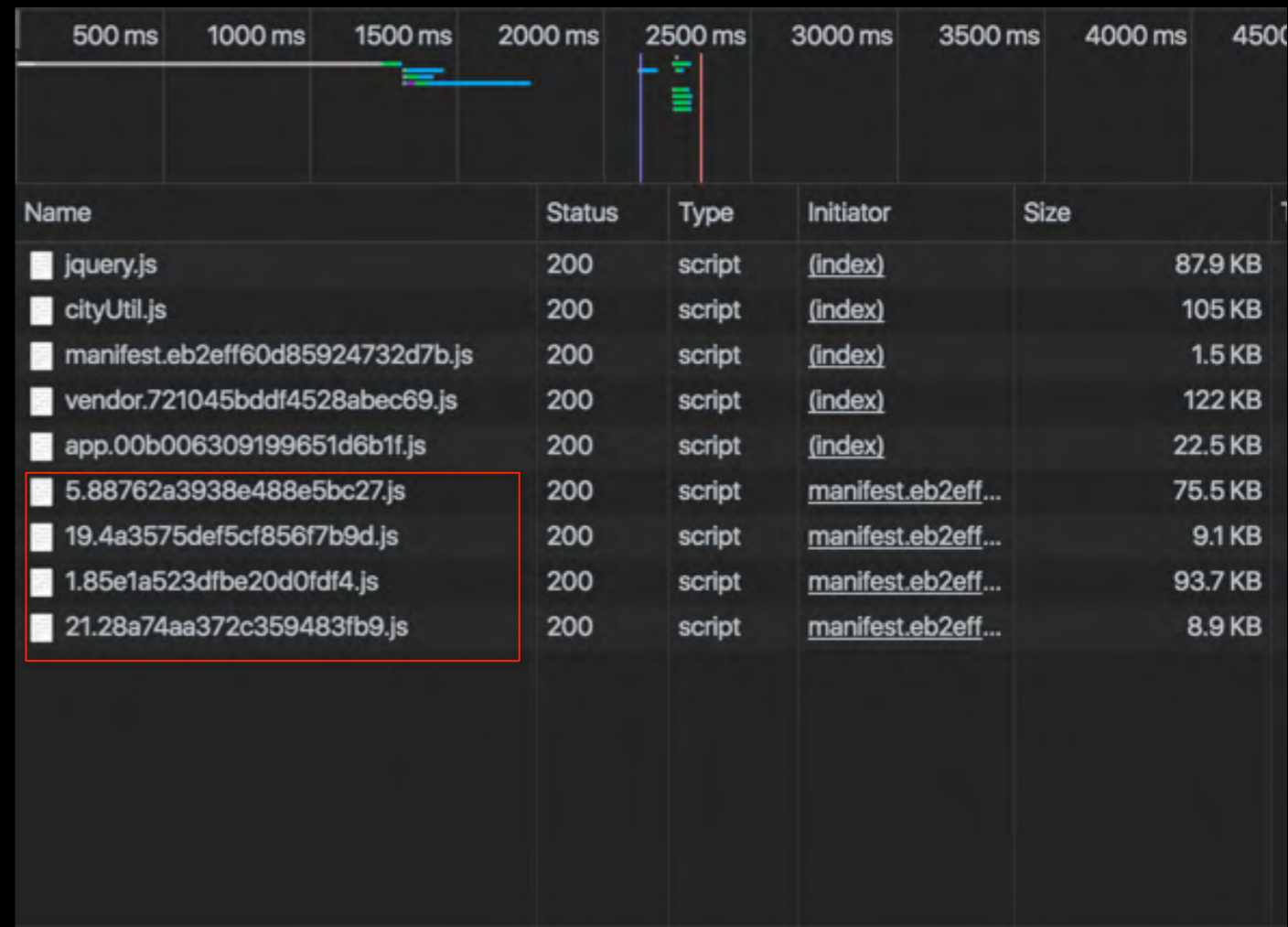


```
require.ensure() -> Promise.all([import(...)].then([modules])=> {})  
// webpack 1      webpack2
```


代码分片

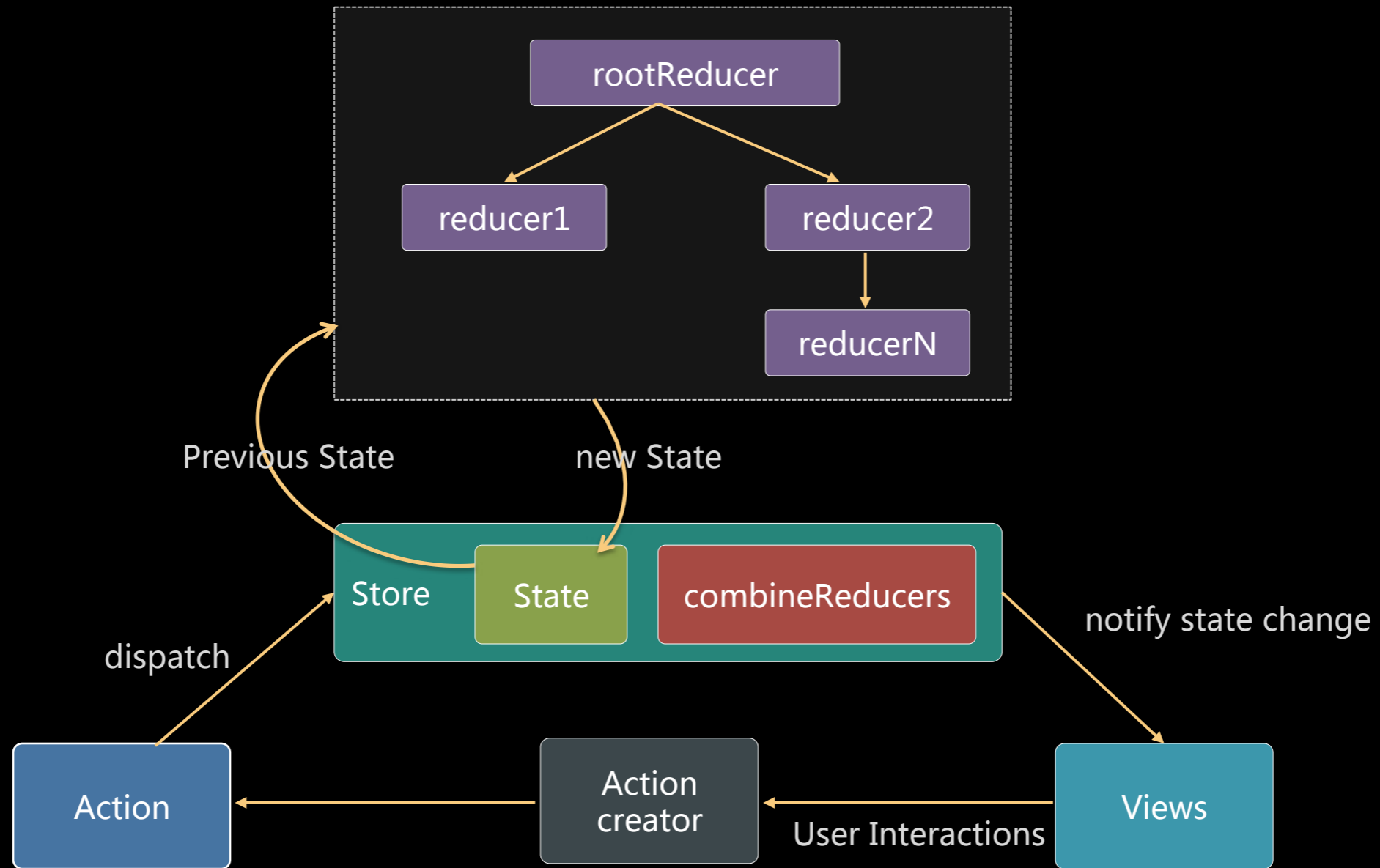
```
DynamicRoute.js
import { injectReducer } from 'store/reducers'

export default (store) => ({
  path:('/:dept/detailPage',
  /* Async getComponent is only invoked when route matches */
  getComponent (nextState, cb) {
    /* Webpack - use 'System.import' to create a split point
    and embed an async module loader (jsonp) when bundling */
    Promise.all([
      import('./containers/CheckLeaveContainer'),
      import('./modules')
    ]).then(([Container, modules]) => {
      const reducer = modules.default
      /* Add the reducer to the store on key 'counter' */
      injectReducer(store, { key: 'checkleave', reducer })
      /* Return getComponent */
      cb(null, Container.default)
    })
  }
})
```



Part3 Redux项目结构改进

Redux



views => action => reducer => store => react-redux => virtual dom => views

Rails风格

- ▶ css
- ▶ img
- ▶ js
 - ▶ actions ←----- here
 - ▶ api
 - ▶ components
 - JS ApprovalDetail.jsx ←----- here
 - JS ApprovalItems.jsx
 - JS ApprovalList.jsx
 - JS Entry.jsx
 - JS Header.jsx
 - JS Main.jsx
 - JS NotFound.jsx
 - JS ResourceItems.jsx
 - JS ViewBrief.jsx
 - JS ViewCustomerDetails.jsx
 - JS ViewTeamDetails.jsx
 - JS waterMark.jsx
 - ▶ constants ←----- here
 - ▶ containers
 - JS ApprovalDetail.js
 - JS ApprovalList.js
 - JS Entry.js
 - JS index.js ←----- here
 - JS Main.js
 - JS SalesDaily.jsx
 - ▶ helper
 - ▶ middleware
 - ▶ model
 - ▶ reducers
 - ▶ store

Just one module change !

解耦&紧凑

```
src
├── components
├── containers
├── helpers
├── redux
│   ├── middleware
│   └── modules
│       ├── JS auth.js
│       ├── JS counter.js
│       ├── JS info.js
│       ├── JS reducer.js
│       ├── JS survey.js
│       └── JS widgets.js
│       ├── JS create.js
│       ├── theme
│       └── utils
│           ├── JS client.js
│           ├── JS config.js
│           └── JS routes.js
```

```
1 // const
2 const SAVE_SUCCESS = 'redux-example/widgets/SAVE_SUCCESS';
3 const SAVE_FAIL = 'redux-example/widgets/SAVE_FAIL';
4
5 const initialState = {
6   loaded: false,
7   editing: {},
8   saveError: {}
9 };
10
11 // reducer
12 export default function reducer(state = initialState, action = {
13   switch (action.type) {
14     case LOAD:
15       return {
16         ...state,
17         loading: true
18       };
19     case LOAD_SUCCESS:
20       return {
21         ...state,
22         loading: false,
23         loaded: true,
24         data: action.result,
25         error: null
26       };
27   }
28 }
29 // action
30 export function isLoading(globalState) {
31   return globalState.widgets && globalState.widgets.loaded;
32 }
```

<https://github.com/erikras/ducks-modular-redux>

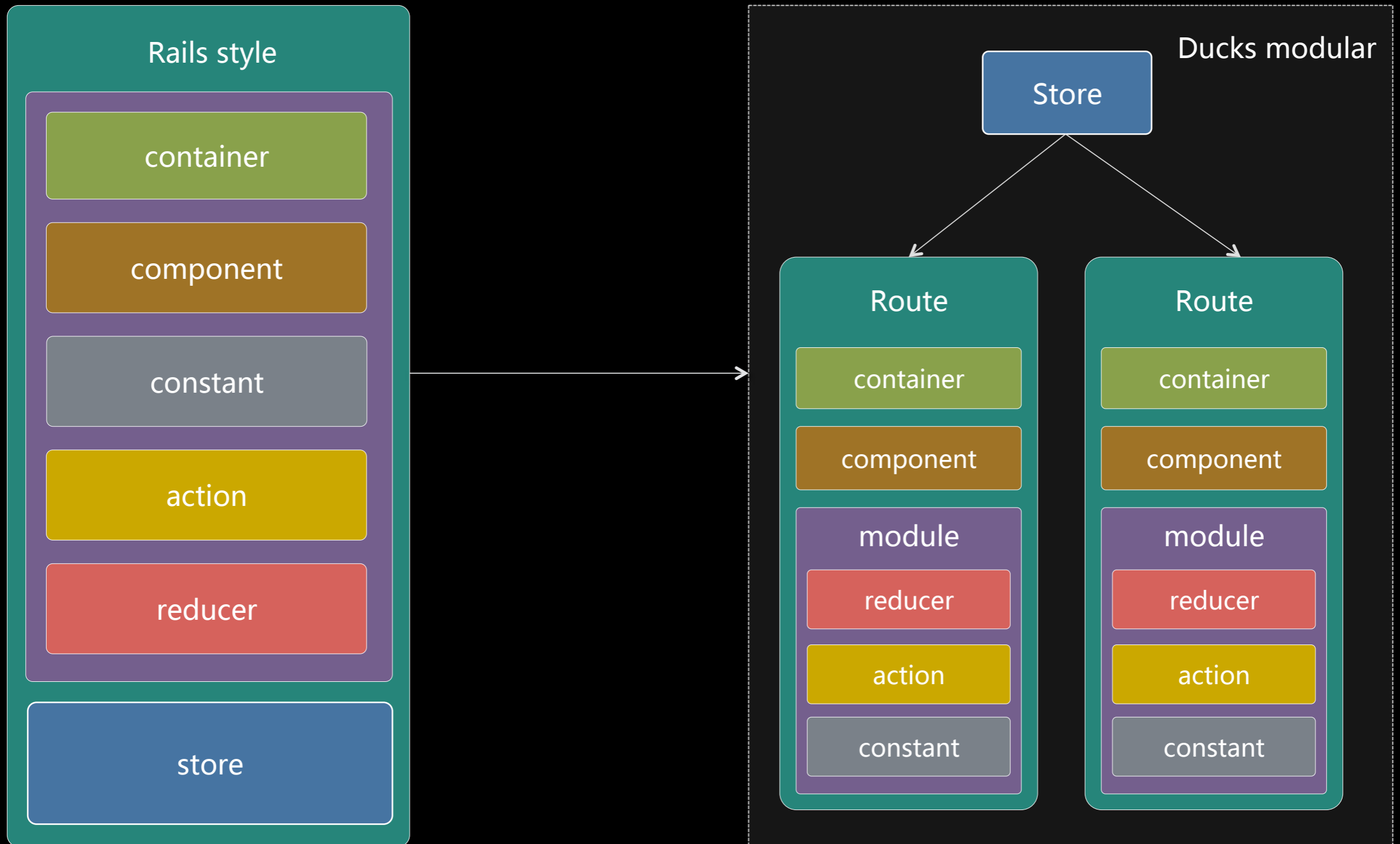
Ducks: Redux Reducer
Bundles



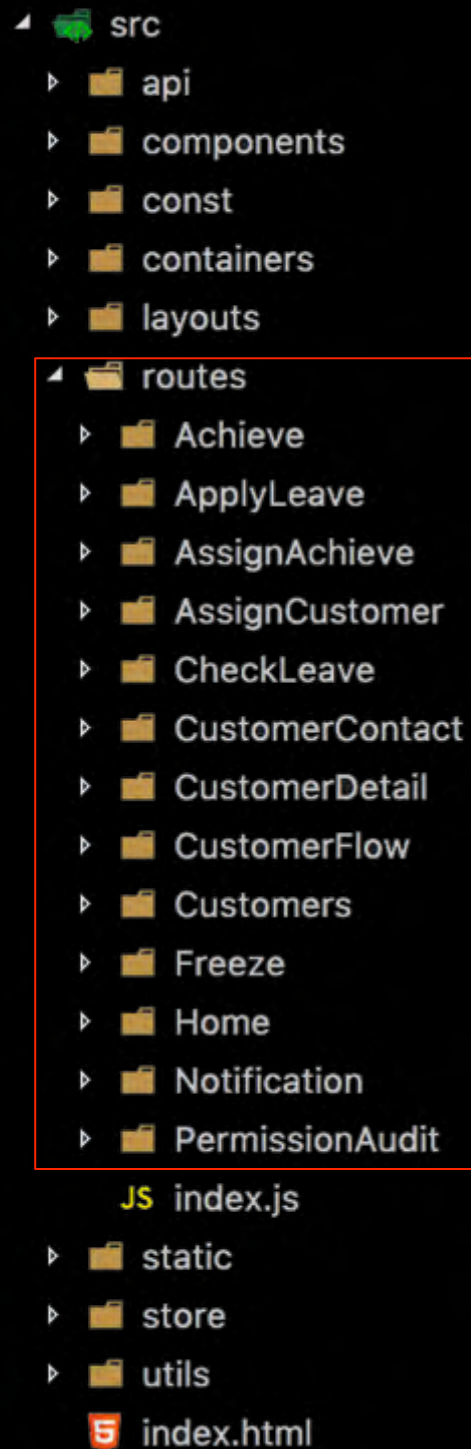
特点概述

- 1、业务代码可集中归类
- 2、少冲突，方便多人协作开发
- 3、易于业务功能的横向扩展
- 4、模块化，可插拔式管理

modules



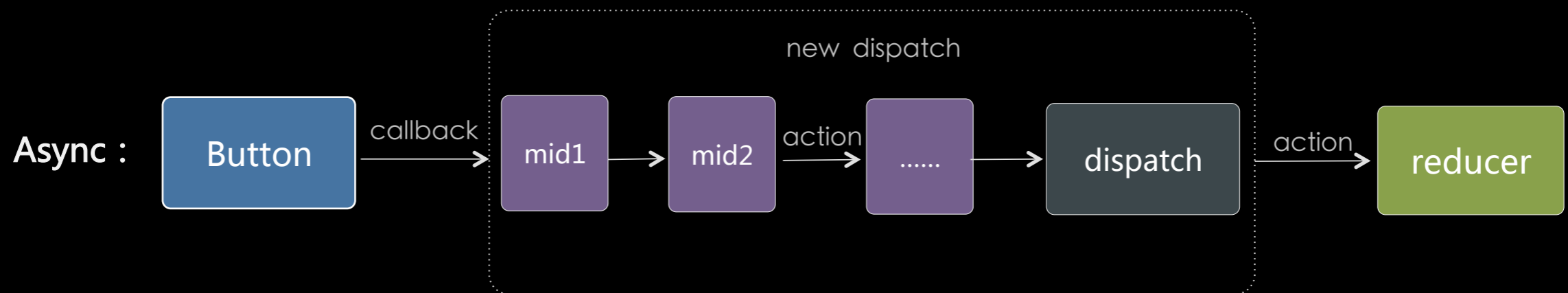
业务组合



```
JS index.js x
8   import CheckLeave from './CheckLeave'
9   import AssignAchieve from './AssignAchieve'
10  import AssignAchieveExt from './AssignAchieve/ext'
11  import AssignCustomer from './AssignCustomer'
12  import CustomerContact from './CustomerContact'
13  import CustomerDetail from './CustomerDetail'
14  import CustomerFlow from './CustomerFlow'
15  import Freeze from './Freeze'
16  import Achieve from './Achieve'
17  import Notification from './Notification'
18
19  /* Note: Instead of using JSX, we recommend using react-router
20     PlainRoute objects to build route definitions. */
21
22  export const createRoutes = (store) => ({
23    path      : '/',
24    component : CoreLayout,
25    indexRoute : Home,
26    childRoutes : [
27      PermissionAudit(store),
28      Customers(store),
29      OtherCustomers(store),
30      ApplyLeave(store),
31      CheckLeave(store),
32      AssignAchieve(store),
33      AssignAchieveExt(store),
34      CustomerContact(store),
35      AssignCustomer(store),
36      CustomerDetail(store),
37      CustomerFlow(store),
38      Freeze(store),
```

Part4 异步数据流优化

Data in Redux



Redux thunk

```
function createThunkMiddleware(extraArgument) {  
  return ({ dispatch, getState }) => next => action => {  
    if (typeof action === 'function') {  
      return action(dispatch, getState, extraArgument);  
    }  
    return next(action);  
  };  
}
```



```
function upload(data){  
  return dispatch => {  
    // 显示出加载效果  
    dispatch({ type: 'SHOW_WAITING_MODAL' });  
    // 开始上传  
    api.upload(data)  
      .then(res => {  
        // 成功, 隐藏加载效果, 并显示出预览图  
        dispatch({ type: 'PRELOAD_IMAGES', data: res.images });  
        dispatch({ type: 'HIDE_WAITING_MODAL' });  
      })  
      .catch(err => {  
        // 错误, 隐藏加载效果, 显示出错误信息, 2秒后消失  
        dispatch({ type: 'SHOW_ERROR', data: err });  
        dispatch({ type: 'HIDE_WAITING_MODAL' });  
        setTimeout(_ => dispatch({ type: 'HIDE_ERROR' }), 2000);  
      })  
  }  
}
```

副作用

- 1、dispatch使用泛滥成灾
- 2、重复代码多, 处理流程抽象度低
- 3、多请求间的衔接处理无规范
- 4、取消, 超时, 队列等复杂情况未考虑。

Redux saga

```
import { take, put, call, delay } from 'redux-saga/effects'
/*
 * put (产生一个 action)
 * call (阻塞地调用一个函数)
 * fork (非阻塞地调用一个函数)
 * take (监听且只监听一次 action)
 * delay (延迟)
 * race (只处理最先完成的任务)
 */
function *uploadFlow(action) {
  // 显示出加载效果
  yield put({ type: 'SHOW_WAITING_MODAL' });
  // 简单的 try-catch
  try{
    const response = yield call(api.upload, action.data);
    yield put({ type: 'PRELOAD_IMAGES', data: response.images });
    yield put({ type: 'HIDE_WAITING_MODAL' });
  }catch(err){
    yield put({ type: 'SHOW_ERROR', data: err });
    yield put({ type: 'HIDE_WAITING_MODAL' });
    yield delay(2000);
    yield put({ type: 'HIDE_ERROR' });
  }
}

function* watchUpload() {
  yield* takeEvery('BEGIN_REQUEST', uploadFlow)
}
```

<https://github.com/redux-saga/redux-saga>

An alternative side effect model for Redux apps

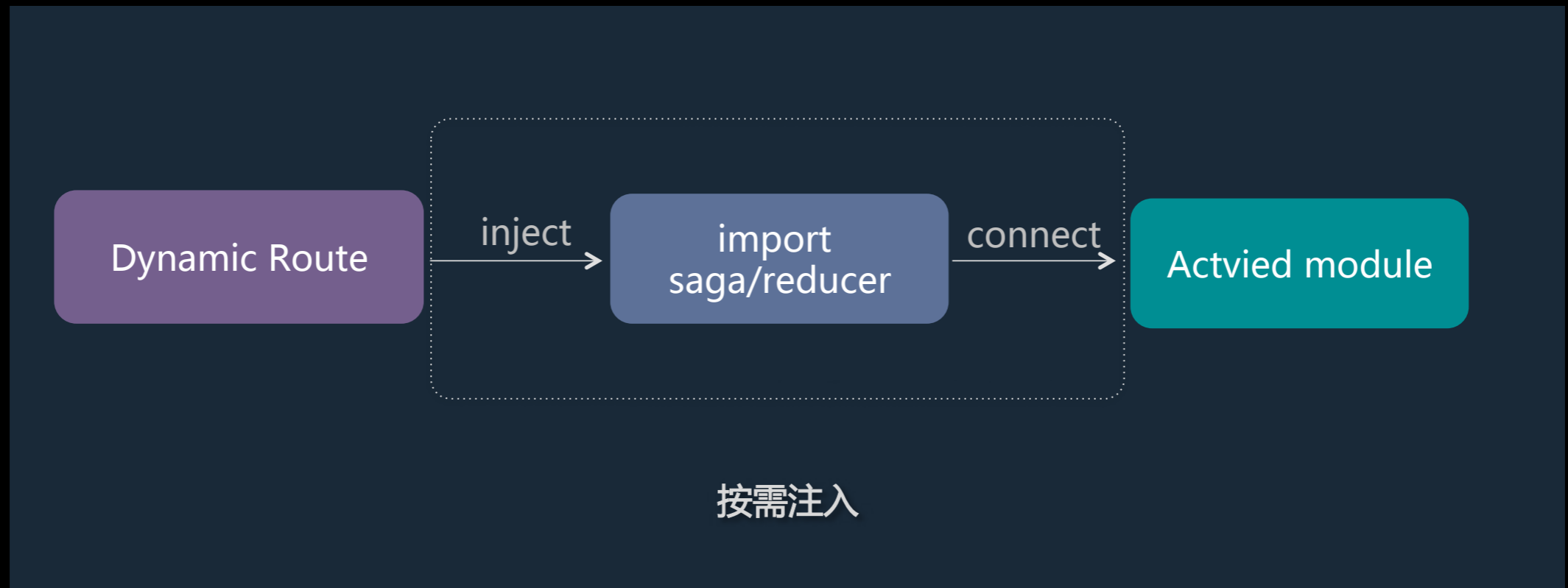
优点概述

- 1、集中处理 redux 副作用问题
- 2、异步处理实现为 generator
- 3、类redux-thunk中间件。
- 4、watch/worker(监听->执行) 的工作形式。

Saga请求事件队列

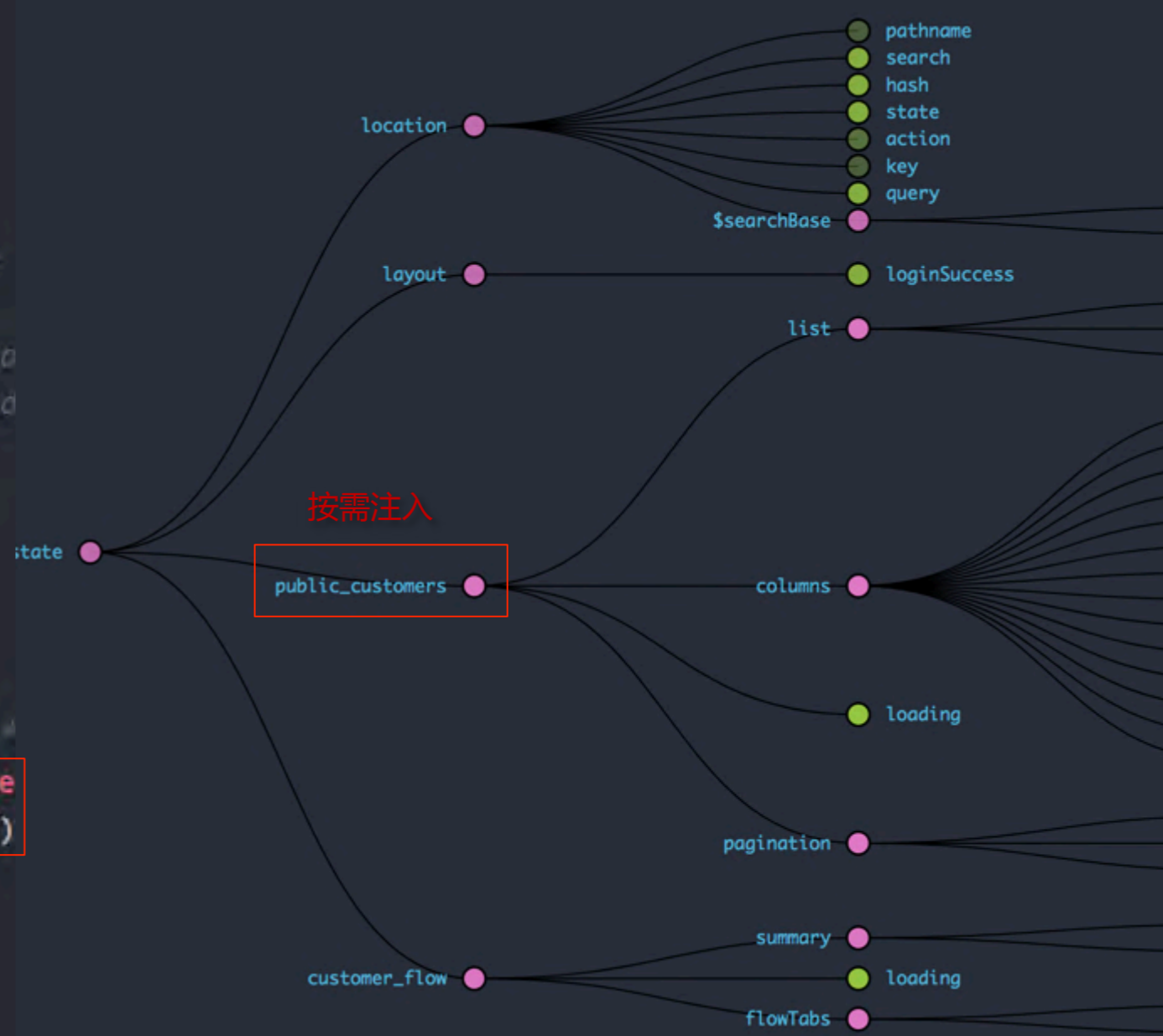
```
// -----  
// Sagas请求队列  
// -----  
export function* fetchOppportunityDeatil(type, body) {  
  while (true) {  
    // 1- 创建一个针对请求事件的 channel  
    const { payload } = yield take(REQUEST_OPPOR_PAGE_DATA)  
    // 2- 从 channel 中拿出多个事件。注意这里我们使用的是阻塞的函数调用  
    const [detail, contacts, trackTypes, isList, osList, remark, visit, industry] = yield [  
      call(fetchAPI, types.getOpporDetail, payload.id),  
      call(fetchAPI, types.getContacts, payload.id),  
      call(fetchAPI, types.getTrackType),  
      call(fetchAPI, types.getTrackList, payload.list + `&sale_type=${SALE_TYPE.IS}`),  
      call(fetchAPI, types.getTrackList, payload.list + `&sale_type=${SALE_TYPE.OS}`),  
      call(fetchAPI, types.getRemarkInfo, payload.first + `&type=${REMARK_TYPE.FOLLOW}`),  
      call(fetchAPI, types.getVisitList, payload.first + `&status=${VISIT_STATUS.UNVISIT}`),  
      call(fetchAPI, types.getIndustry),  
      call(fetchAPI, types.getPhoneRecord, payload.list)  
    ]  
    // 3- 请求完成后再对reducer进行修改  
    yield put(addInitData({ detail, contacts, trackTypes, isList, osList, remark, visit,  
      industry })))  
  }  
}
```

进一步优化



按需注入

```
import { injectReducer } from 'store/reducers'  
import { injectSagas } from 'store/sagas'  
  
export default (store) => ({  
  path: 'oppoAudio',  
  /* Async getComponent is only invoked when route matches */  
  getComponent(nextState, cb) {  
    /* Webpack - use 'System.import' to create a split point  
    and embed an async module loader (jsonp) when bundling */  
    Promise.all([  
      import('./containers/OppoAudioContainer'),  
      import('./modules/')  
    ]).then(([OppoAudioContainer, modules]) => {  
      const reducer = modules.default  
      const sagas = modules.sagas  
      /* Add the reducer to the store on key 'counter' */  
      injectReducer(store, { key: 'oppoAudioList', reducer })  
      injectSagas(store, { key: 'oppoAudioList', sagas })  
      /* Return getComponent */  
      cb(null, OppoAudioContainer.default)  
    })  
  }  
})
```



没有银弹

名称	特点	地址
redux-thunk	原始方案，使用简单	https://github.com/gaearon/redux-thunk
redux-saga	方案完善，颗粒级控制	https://github.com/redux-saga/redux-saga
redux-observable	观察者模式，自动响应	https://redux-observable.js.org/
redux-promise-middleware	保留thunk，promise优化	https://github.com/pburtchaell/redux-promise-middleware
redux-loop	类Elm模式	https://github.com/redux-loop/redux-loop
redux-action-tools	抽象合理，上手简单	https://github.com/kpaxqin/redux-action-tools

Part5 持续构建

风格约束

```
npm install husky --save-dev
```

```
package.json:
```

```
"scripts": {  
  "precommit": "npm lint",  
  "prepush": "npm lint",  
  "...": "..."  
}
```

```
git commit -m "Keep calm and commit" (git hook)
```

<https://github.com/typicode/husky>

Husky can prevent bad commit, push and more 🐶 woof!

“将可预见问题控制在提交前”

```
> husky - npm run -s precommit
```

```
/Users/huangjian/docker/react/webpack-develop-startkit/src/routes/Home/containers/HomeContainer.js
```

```
3:38 error Extra semicolon semi
```

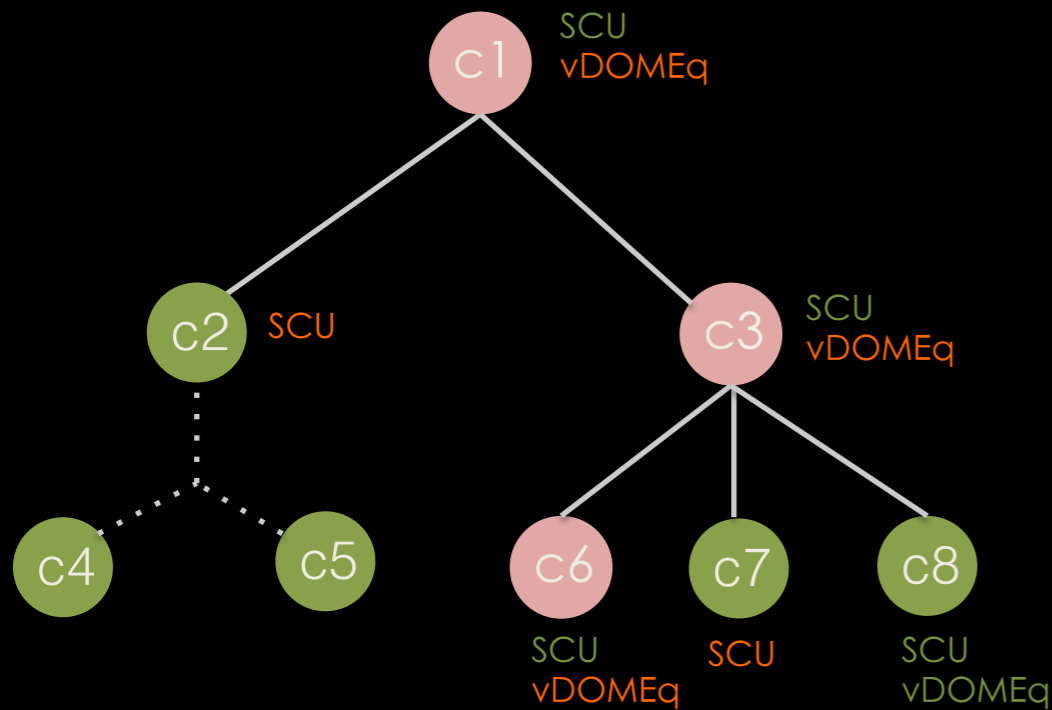
```
15:1 error Too many blank lines at the end of file. Max of 1 allowed no-multiple-empty-lines
```

```
* 2 problems (2 errors, 0 warnings)
```

```
> husky - pre-commit hook failed (add --no-verify to bypass)
```

```
> husky - to debug, use 'npm run precommit'
```

组件优化



- No Reconciliation needed SCU ShouldComponentUpdate?
- Reconciliation needed vDomEq Are virtual DOMS equivalent?
vDomeq

单组件优化

shouldComponentUpdate: PureComponent -> Immutable.js

多组件优化

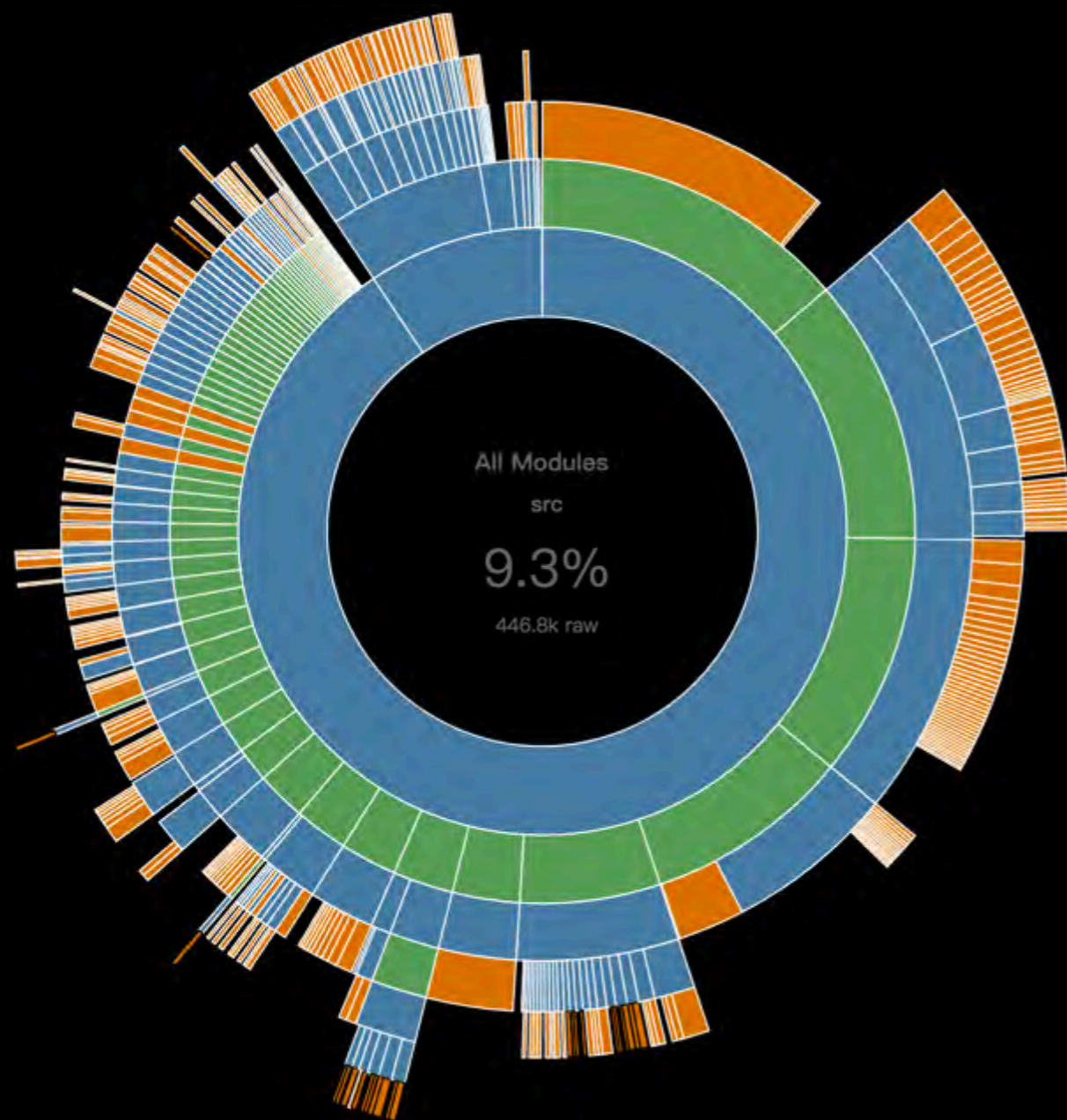
Reconciliation: key (use id instead index)

Redux优化

Store: Reselect、Normalizr

"react-addons-perf"

构建优化



Tree
shaking

DllPlugin

速度优化

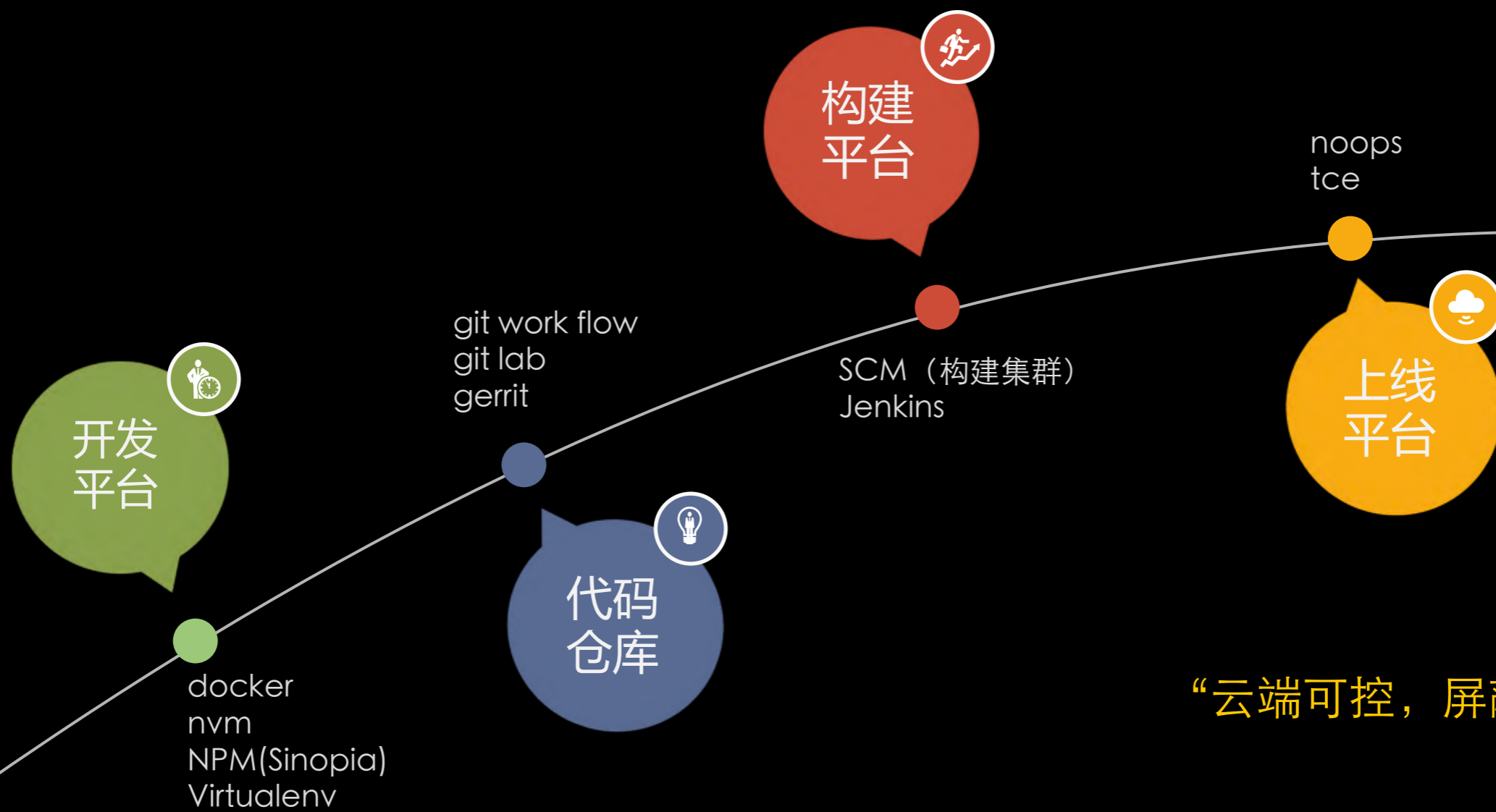
体积优化

uglify
parallel

happy
Pack

<https://techblog.toutiao.com>

工作流



“云端可控，屏蔽差异”

总结

总结



Q&A

THANKS!

