mobike
摩拜单车

个人介绍

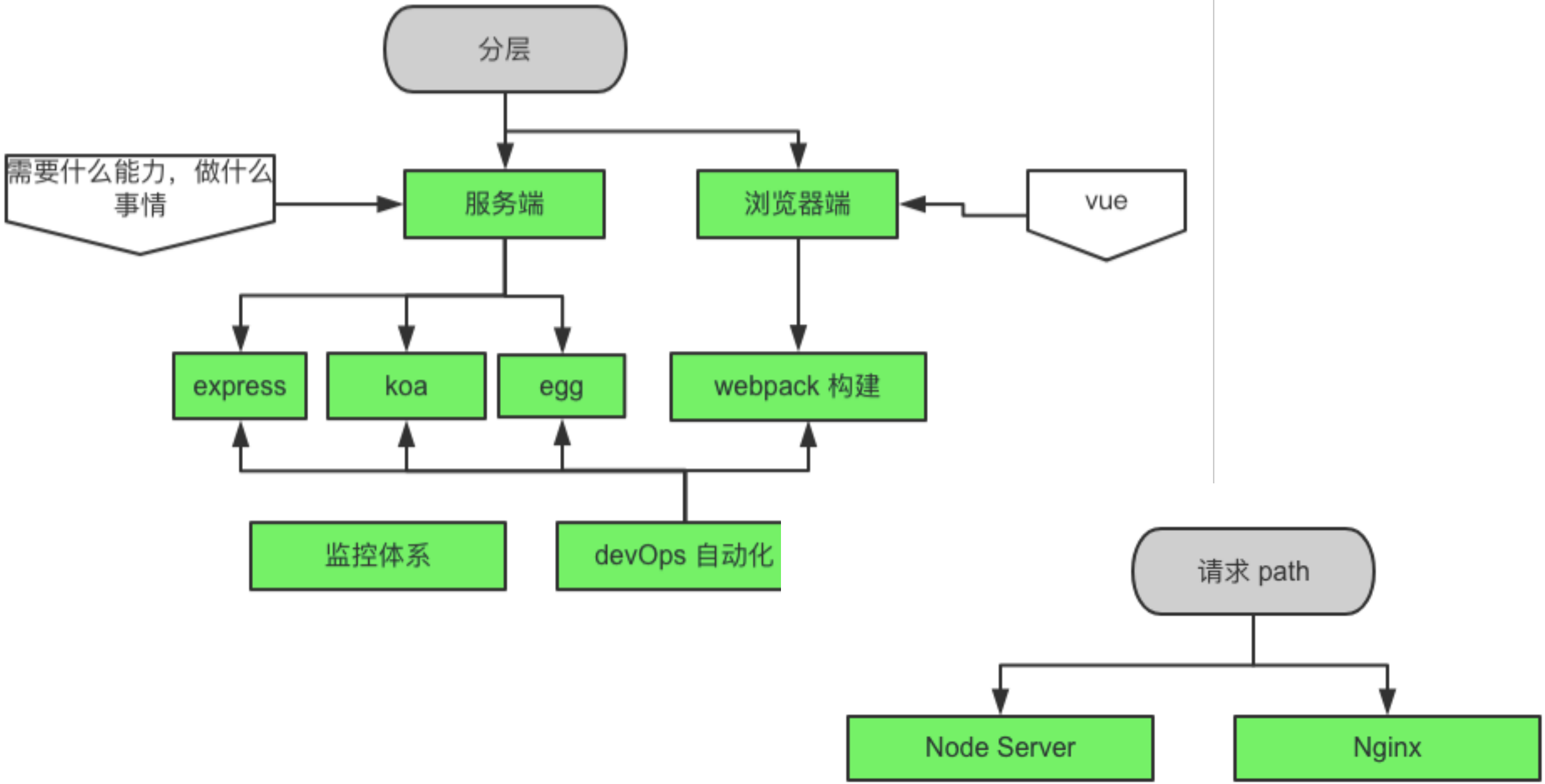
小春 - 摩拜技术总监 客户端&前端负责人

- 做过技术大会大前端出品人
- 参加过 SDCC、vueconf 等技术会议
- 写过《vue.js权威指南》、翻译过 Node.js 书籍等
- 个人新书收尾中、最近参与翻译了《SVG动画》出版在即
- 涉猎和兴趣较广、运营着多个技术号

mobike
摩拜单车

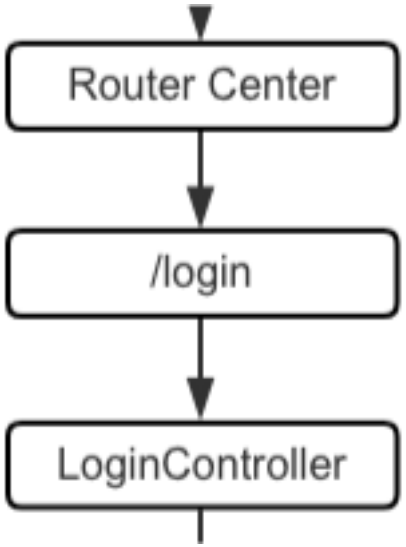
分享大纲

- Response 处理的历史演变和案例架构解析
- Node server 取之有道
- Dev server 的设计思路
- Nuxt 的设计思路
- Vue ssr 核心依赖的演变
- Egg 的设计思路
- 监控服务
- Node devOps


分层
需要什么能力，做什么事情
服务端
浏览器端
vue
express
koa
egg
webpack 构建
监控体系
devOps 自动化
请求 path
Node Server
Nginx

mobike
摩拜单车

第一步：请求



除了再上层的域名解析、nginx等

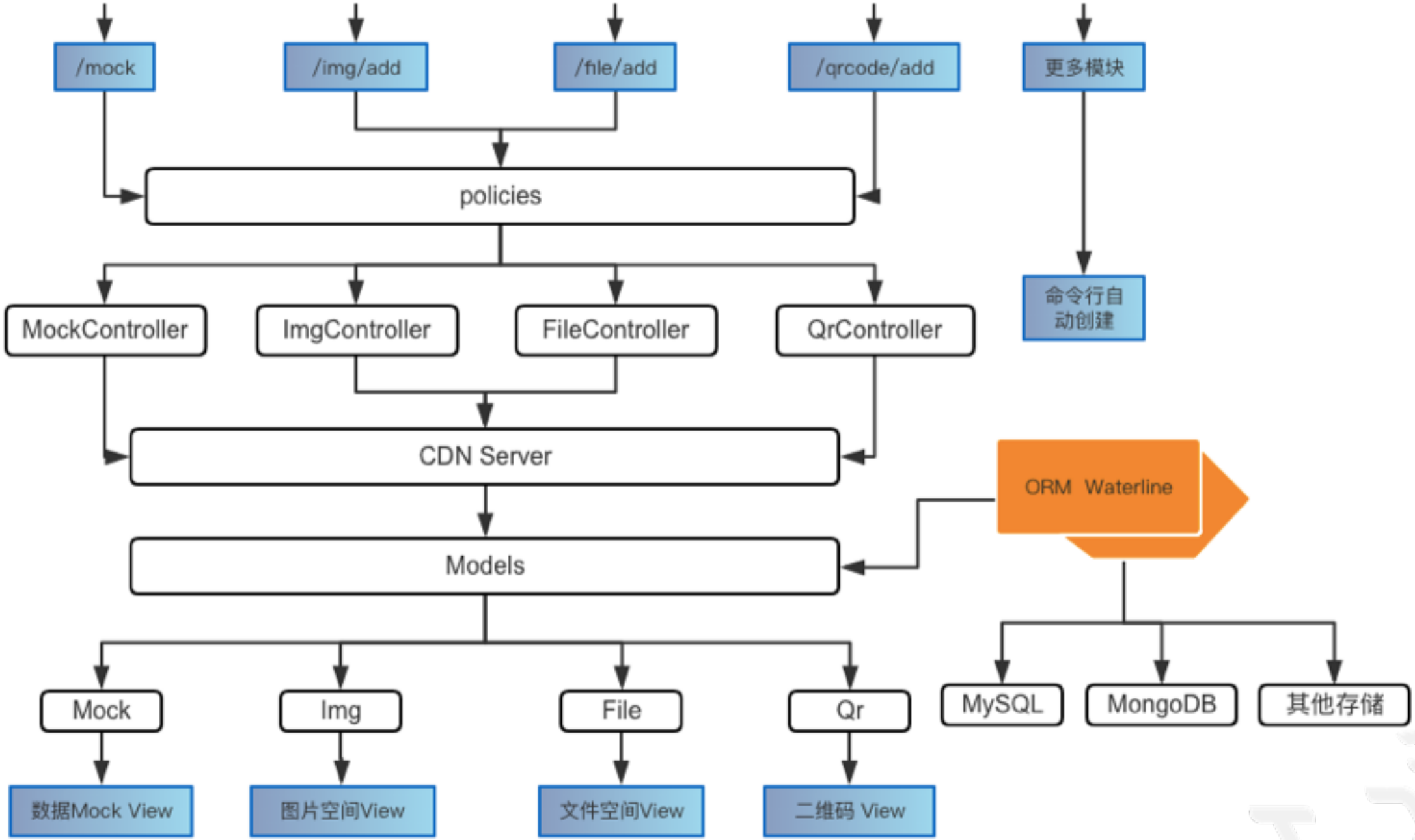
一般我们有路由中心、对应路由的控制器

关键问题：我们返回用户什么？

1、静态系统 --- index.html 文件

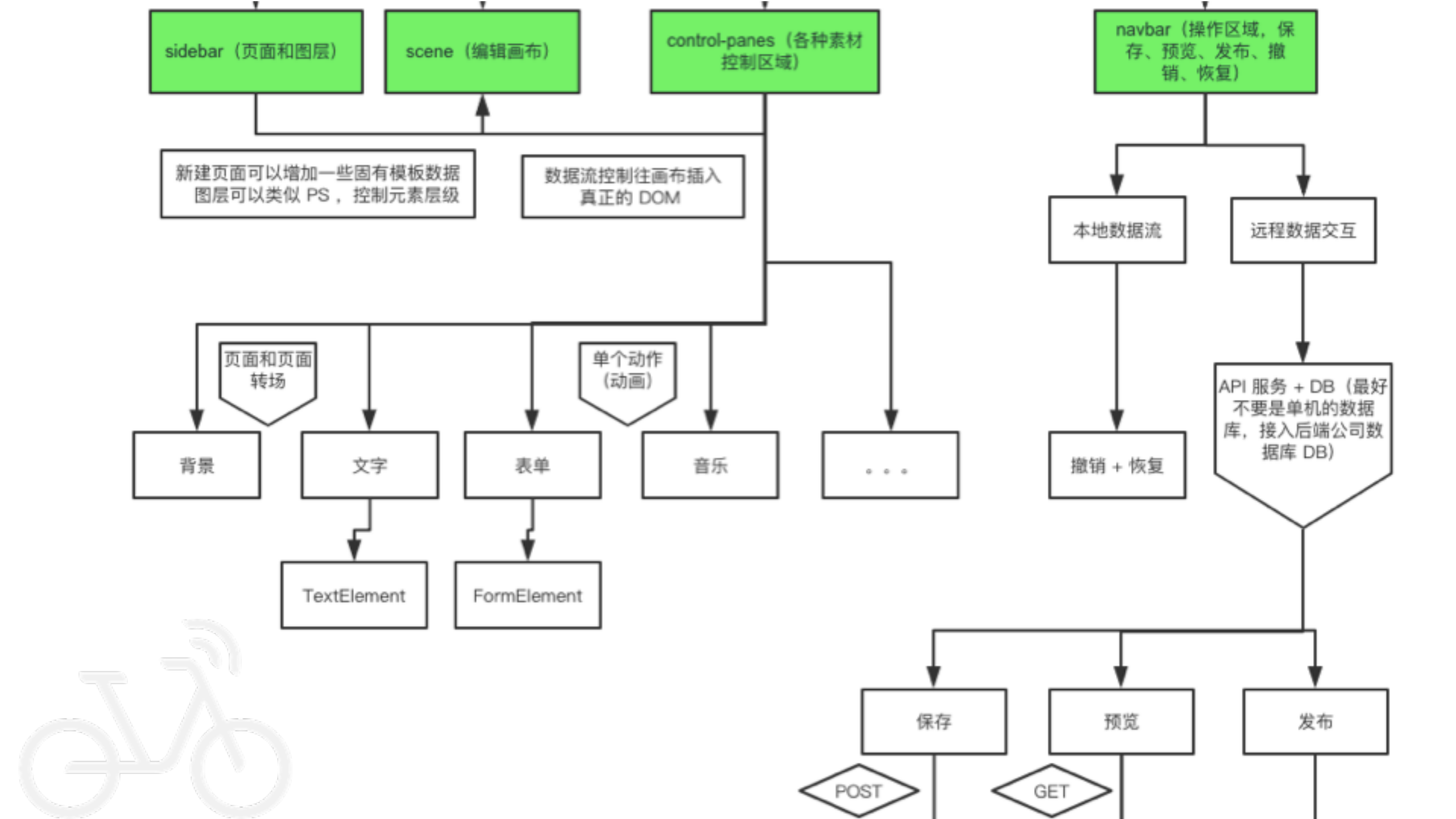
2、动态服务 --- ？？？

Time to content 内容到达时间


/mock
/img/add
/file/add
/qrcode/add
更多模块
policies
MockController
ImgController
FileController
QrController
命令行自动创建
CDN Server
ORM Waterline
Models
Mock
Img
File
Qr
MySQL
MongoDB
其他存储
数据Mock View
图片空间View
文件空间View
二维码 View

mobike
摩拜单车

应用场景一：h5 编辑器


sidebar（页面和图层）
scene（编辑画布）
control-panes（各种素材控制区域）
navbar（操作区域，保存、预览、发布、撤销、恢复）
新建页面可以增加一些固有模板数据
图层可以类似 PS，控制元素层级
数据流控制往画布插入真正的 DOM
本地数据流
远程数据交互
页面和页面转场
单个动作（动画）
背景
文字
表单
音乐
. . .
撤销 + 恢复
API 服务 + DB（最好不要是单机的数据库，接入后端公司数据库 DB）
TextElement
FormElement
保存
预览
发布
POST
GET

mobike
摩拜单车

服务端 - 选择

```
var express = require('express')
var app = express()

app.get('/', function (req, res) {
  res.send('Hello World')
})

app.listen(3000)
```

```
const Koa = require('koa');
const app = new Koa();

// response
app.use(ctx => {
  ctx.body = 'Hello Koa';
});

app.listen(3000);
```

```
var http = require('http');

var hostname = '127.0.0.1';
var port = 3000;

var server = http.createServer(function(req, res) {
    res.statusCode = 200;
    res.setHeader('Content-Type', 'text/html');
    res.write('<head><meta charset="utf-8"/></head>');
    res.write('<b>GMTC Node.js 专场</b>');
    res.end('<h1>from mobike</h1>');
});

server.listen(port, hostname, function() {
    console.log('Server running at http://%s:%s', hostname, port);
});
```

mobike
摩拜单车

服务端 - 能力 - 本地静态资源访问

静态资源我们会放置到打包之后的目录 dist 或者 public 目录

1、express
express.static ➔ serve-static

```
80  exports.static = require('serve-static');
```

2、koa ➔ koa-static-cache

3、egg ➔ egg-static

直接访问指定根目录下的文件 dist/manifest.ff831299e65cb6ad7876.js
*****/ manifest.ff831299e65cb6ad7876.js

mobike
摩拜单车

Vue.js 静态类型项目本地 dev 服务启动

```
const compiler = webpack(webpackConfig)

const devMiddleware = require('webpack-dev-middleware')(compiler, {
  publicPath: webpackConfig.output.publicPath,
  quiet: true
})
app.use(devMiddleware)
```

webpack-dev-middleware

```
const hotMiddleware = require('webpack-hot-middleware')(compiler, {
  log: false,
  heartbeat: 2000
})
app.use(hotMiddleware)
```

webpack-hot-middleware

```
const staticPath = path.posix.join(config.dev.assetsPublicPath,
  config.dev.assetsSubDirectory)
app.use(staticPath, express.static('./static'))
```

mobike
摩拜单车

Vue.js 静态类型项目本地 dev 服务启动 webpack-dev-middleware

An express-style development middleware for use with webpack bundles and allows for serving of the files emitted from webpack.

benefits

1、 No files are written to disk, rather it handles files in memory
2、 If files changed in watch mode, the middleware delays requests until compiling has completed.

内存 -在请求来时，根据 URL path 拿到 output 文件路径，然后从 memory-fs 中读文件，写到响应里面。

mobike
摩拜单车

Vue.js 静态类型项目本地 dev 服务启动

webpack-dev-middleware

```
// constructor for the middleware
module.exports = function(compiler, options) {
    var context = {

    };
    var shared = Shared(context);

    // The middleware function
    function webpackDevMiddleware(req, res, next) {
    }

    webpackDevMiddleware.getFilenameFromUrl = getFilenameFromUrl.bind(this, c
    webpackDevMiddleware.waitUntilValid = shared.waitUntilValid;
    webpackDevMiddleware.invalidate = shared.invalidate;
    webpackDevMiddleware.close = shared.close;
    webpackDevMiddleware.fileSystem = context.fs;
    return webpackDevMiddleware;
};
```

middleware

memory-fs

```
var fs;
var isMemoryFs = !compiler.compilers && compiler.outputFileSystem inst
if(isMemoryFs) {
    fs = compiler.outputFileSystem;
} else {
    fs = compiler.outputFileSystem = new MemoryFileSystem();
}
context.fs = fs;
```

mobike
摩拜单车

Vue.js 静态类型项目本地 dev 服务启动 webpack-dev-middleware

```
module.exports = function Shared(context) {
    var share = {
    };
    share.setOptions(context.options);
    share.setFs(context.compiler);

    context.compiler.plugin("done", share.compilerDone);
    context.compiler.plugin("invalid", share.compilerInvalid);
    context.compiler.plugin("watch-run", share.compilerInvalid);
    context.compiler.plugin("run", share.compilerInvalid);

    share.startWatch();
    return share;
};
```

v2.0.0 重构过代码
最新 v3.1.3

```
// start watching
if(!options.lazy) {
    console.log('compiler.watch')
    var watching = compiler.watch(options.watchOptions, share.handleCompilerCal
    context.watching = watching;
} else {
    context.state = true;
}
```

startWatch

mobike 摩拜单车

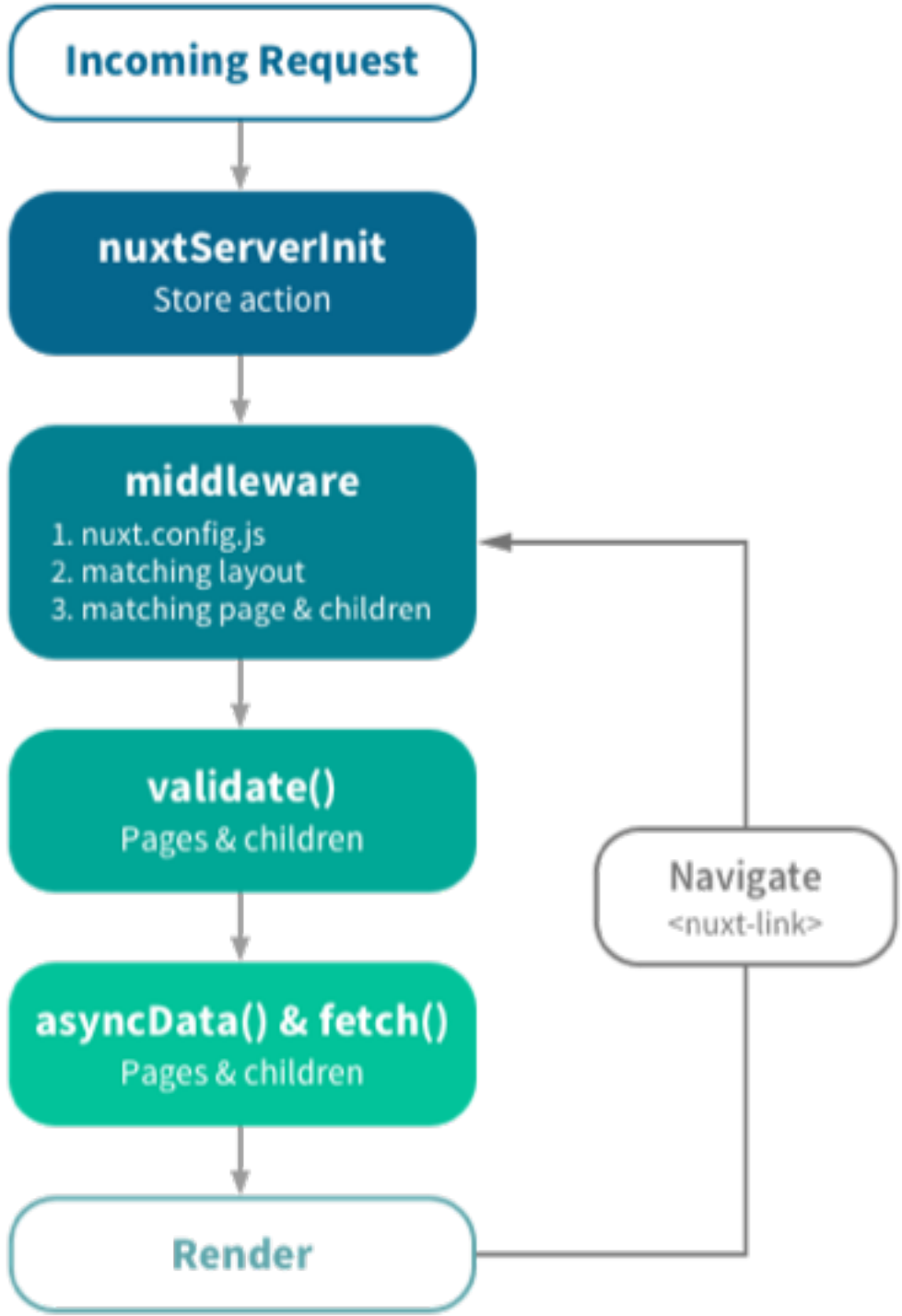
Nuxt 设计特性

Vue.js Meta Framework to create complex, fast & universal web application quickly.

1、Version choose：1.0.0-rc9 ➔ 1.4.0

2、Dependences：

- serve-static
- express
- vue-server-renderer
- vue
- vue-router
- vuex


Incoming Request
nuxtServerInit
Store action
middleware
1. nuxt.config.js
2. matching layout
3. matching page & children
validate()
Pages & children
asyncData() & fetch()
Pages & children
Render
Navigate
<nuxt-link>

mobike 摩拜单车

Nuxt 设计特性之 dev

Backpack

命令：backpack dev

目录：

backpack-core
  bin
    dev

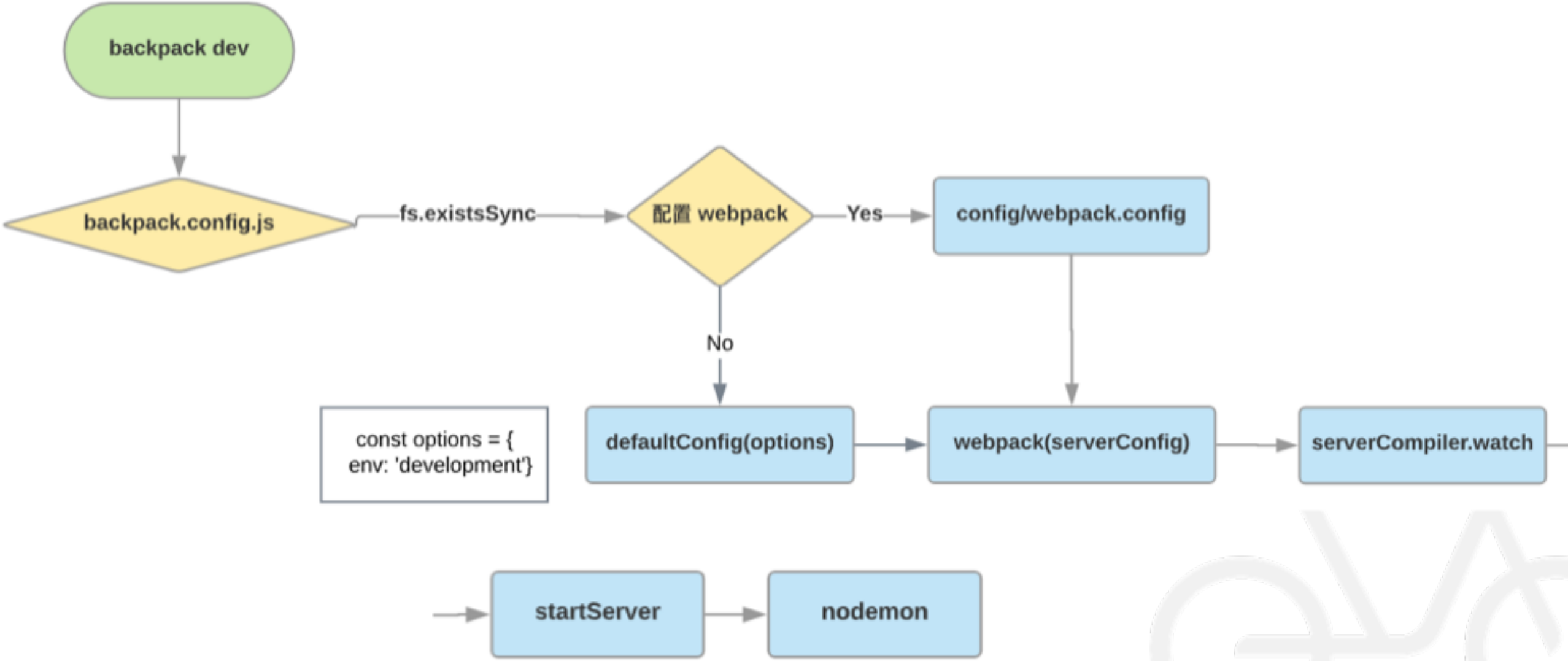
服务：

1、compiler.watch
2、 nodemon

mobike
摩拜单车

Nuxt 设计特性之 dev

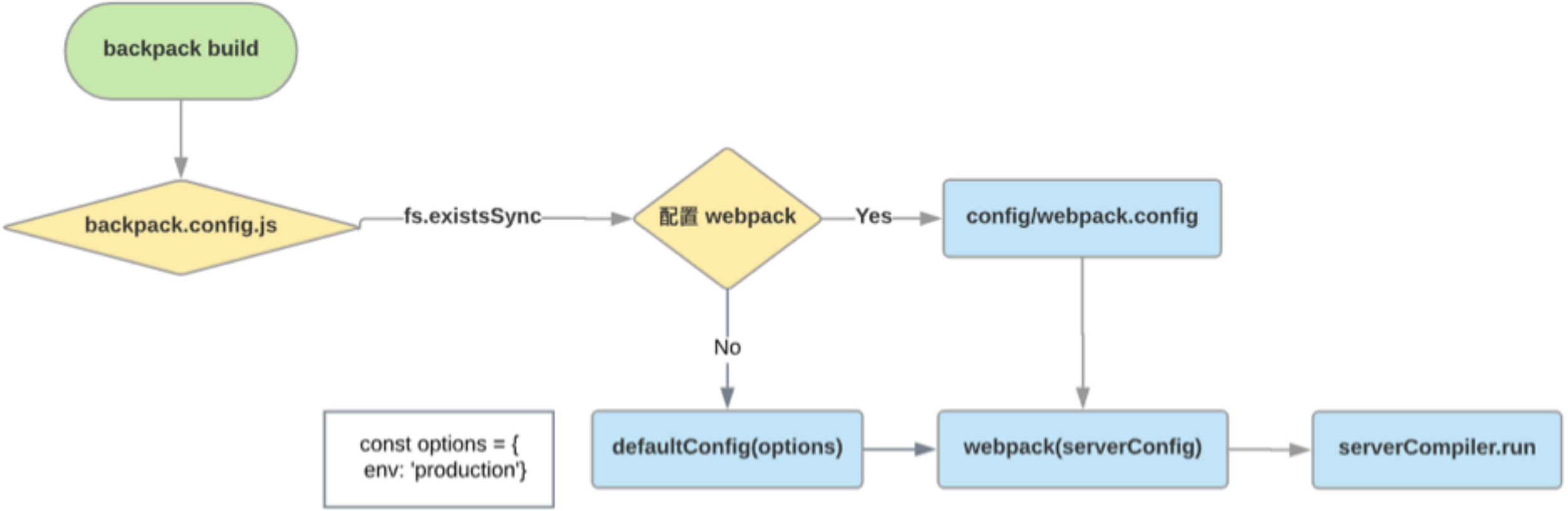
Backpack



mobike
摩拜单车

Nuxt 设计特性之 build

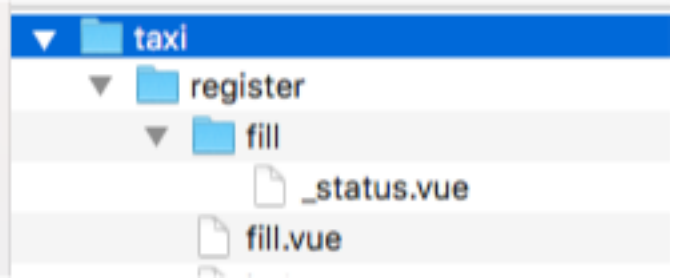
Backpack



mobike 摩拜单车

Nuxt 设计特性之 router

关键字： vue-router、glob 扫描 pages 目录、lodash 模板




```
{
  name: 'taxi-register-fill',
  path: '/taxi/register/fill',
  component: '/Users/zhangyaochun/latestmobike/taxi-driver/pages/taxi/register/fill.vue',
  chunkName: 'pages/taxi/register/fill',
  children: [ [Object] ]
}
```

```
import Vue from 'vue'
import Router from 'vue-router'

Vue.use(Router)

const _5c7048a8 = () => import('../pages/taxi/register/fill.vue'
    /* webpackChunkName: "pages/taxi/register/fill" */).then(
    m => m.default || m)
const _09dd71aa = () => import('../pages/taxi/register/fill/_status.vue'
    /* webpackChunkName: "pages/taxi/register/fill/_status" */).then(
    m => m.default || m)
```

```
// -- Routes --
debug('Generating routes...')
// If user defined a custom method to create routes
if (this._nuxtPages) {
  // Use nuxt.js createRoutes bases on pages/
  const files = {}
  ;(await glob(`${this.options.dir.pages}/**/*.{vue,js}`, {
    cwd: this.options.srcDir,
    ignore: this.options.ignore
  })).forEach(f => {
    const key = f.replace(/\.(js|vue)$/, '')
    if (/\.vue$/.test(f) || !files[key]) {
      files[key] = f
    }
  })
  templateVars.router.routes = createRoutes(
    Object.values(files),
    this.options.srcDir,
    this.options.dir.pages
  )
} else {
  templateVars.router.routes = this.options.build.createRoutes(
    this.options.srcDir
  )
}
```

```
// -- Templates --
let templatesFiles = [
  'App.js',
  'client.js',
  'index.js',
  'middleware.js',
  'router.js',
  'server.js',
  'utils.js',
  'empty.js',
  'components/nuxt-error.vue',
  'components/nuxt-loading.vue',
  'components/nuxt-child.js',
  'components/nuxt-link.js',
  'components/nuxt.js',
  'components/no-ssr.js',
  'views/app.template.html',
  'views/error.html'
]
```

```
  const templateVars = {
    options: this.options,
    extensions: this.options.extensions
      .map(ext => ext.replace(/^\./, ''))
      .join('|'),
    messages: this.options.messages,
    uniqBy: _.uniqBy,
    isDev: this.options.dev,
    debug: this.options.debug,
    mode: this.options.mode,
    router: this.options.router,
    env: this.options.env,
    head: this.options.head,
    middleware: existsSync(join(this.optic
    store: this.options.store,
```

1、循环 templatesFiles

```
templatesFiles.map(async ({ src, dst, options, custom }) => {
```

2、读取模板内容

```
const { remove, readFile, writeFile,
    mkdirp, existsSync } = require('fs-extra')
const fileContent = await readFile(src, 'utf8')
```

3、模板函数解析生成内容

```
let content
try {
  const _ = require('lodash')
  const template = _.template(fileContent, {
    imports: {
    }
  })
  content = template(
    Object.assign({}, templateVars, {
    })
  )
} catch (err) {
  /* istanbul ignore next */
  throw new Error(`Could not compile template ${src}: ${err.message}`)
}
```

4、创建目录

```
// Ensure parent dir exits
await mkdirp(dirname(path))
```

5、写入文件

```
// Write file
await writeFile(path, content, 'utf8')
```

mobike
摩拜单车

Nuxt 设计特性之 router

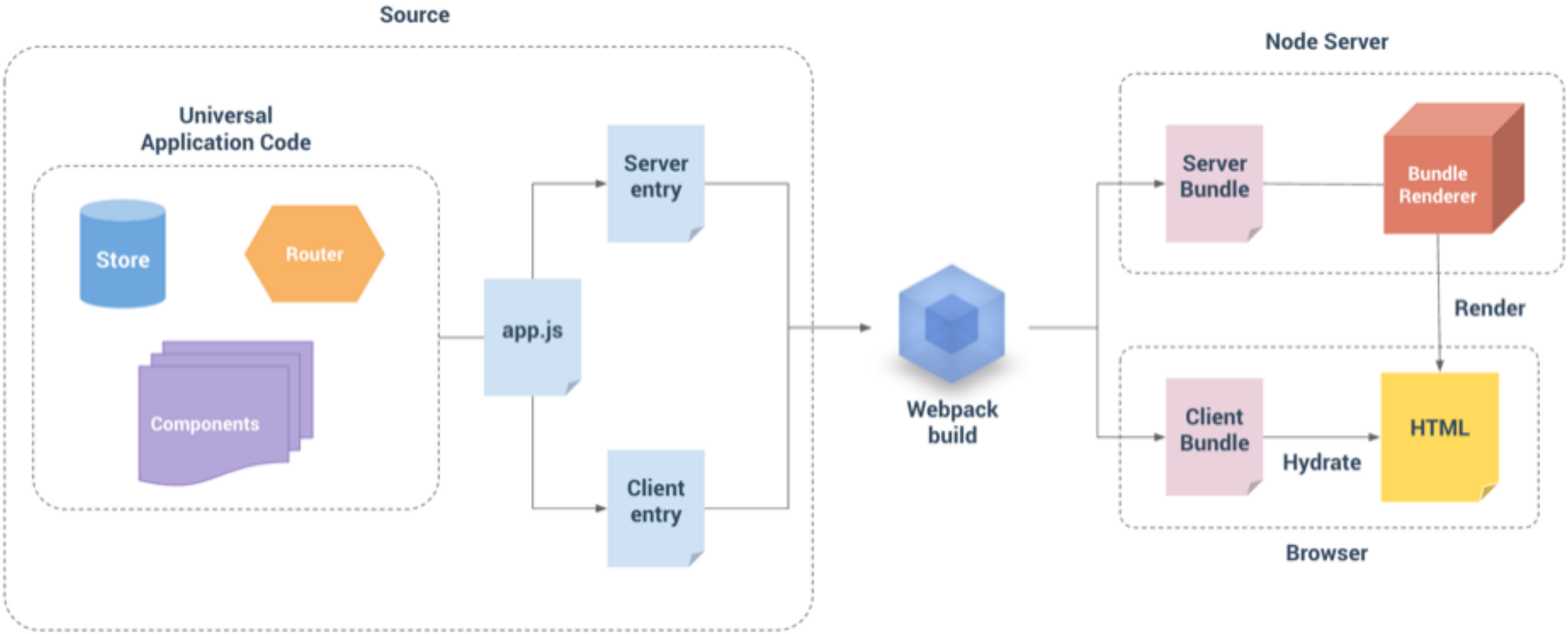
lib/app/router.js – 模板文件

```
import Vue from 'vue'
import Router from 'vue-router'

Vue.use(Router)

<%
function recursiveRoutes(routes, tab, components) {
  let res = ''
  routes.forEach((route, i) => {-
  })
  return res
}
const _components = []
const _routes = recursiveRoutes(router.routes, '\t\t
uniqBy(_components, '_name').forEach((route) => {
  %>const <%= route._name %> = () =>
  import('<%= relativeToBuild(route.component) %>'
<% }) %>
```

```
export function createRouter () {
  return new Router({
    mode: 'history',
    base: '/',
    routes: [
        {
            path: "/taxi/register",
            component: _469a49fd,
            name: "taxi-register"
        },
        {
            path: "/taxi/register/fill",
            component: _5c7048a8,
            name: "taxi-register-fill",
            children: [
                {
                    path: ":status?",
                    component: _09dd71aa,
                    name: "taxi-register-fill-status"
                }
            ]
        }
    ],


    fallback: false
  })
}
```

mobike
摩拜单车

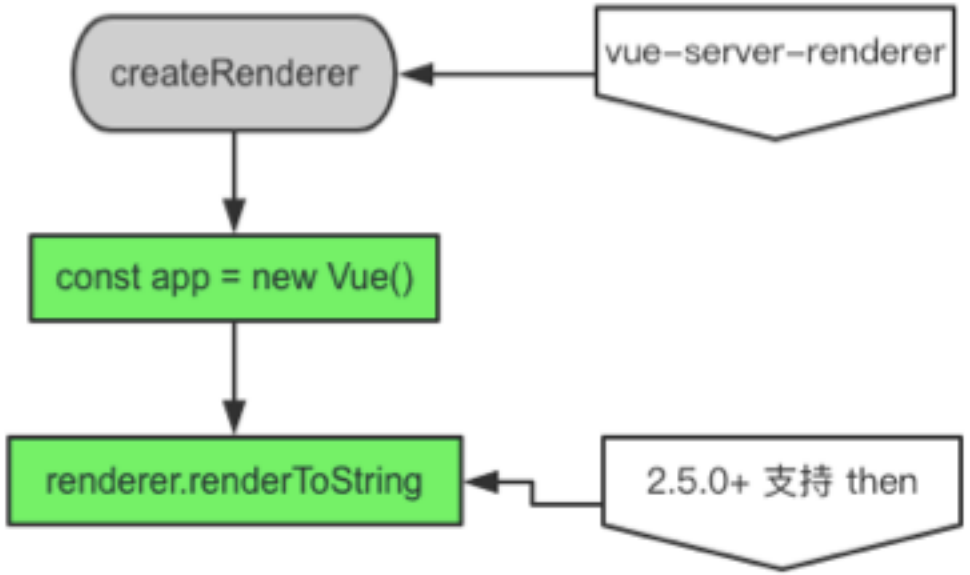
核心 vue-server-renderer

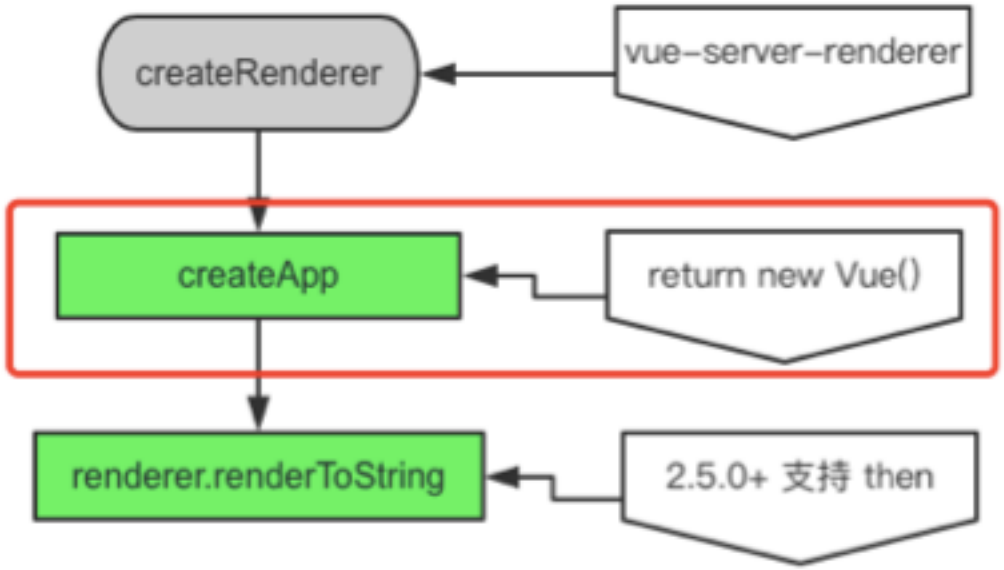


mobike
摩拜单车

核心 vue-server-renderer

升级的 3 个阶段





每个请求创建一个新的根 Vue 实例

runInNewContext

mobike 摩拜单车

核心 vue-server-renderer

client-plugin.js

生成： vue-ssr-client-manifest.json

```
compilation.assets['vue-ssr-client-manifest-zyc.json'] = {
  source: function () { return json; },
  size: function () { return json.length; }
};
```

```
{
  "publicPath": "https://*****",
  "all": [
  ],
  "initial": [
  ],
  "async": [],
  "modules": {
  }
}
```

注意：这些 key 是有用的，后面 build 生成各种类型的资源片段的源

mobike
摩拜单车

如何生成：一个指定内容的文件

```
var VueSSRClientPlugin = function VueSSRClientPlugin (options) {
  if ( options === void 0 ) options = {};

  this.options = Object.assign({
    filename: 'vue-ssr-client-manifest.json'
  }, options);
};

VueSSRClientPlugin.prototype.apply = function apply (compiler) {
  var this$1 = this;
  compiler.plugin('emit', function (compilation, cb) {
    var manifest = {
      publicPath: stats.publicPath,
      all: allFiles,
      initial: initialFiles,
      async: asyncFiles,
      modules: { /* [identifier: string]: Array<index: number> */ }
    };
    var json = JSON.stringify(manifest, null, 2);
    compilation.assets[this$1.options.filename] = {
      source: function () { return json; },
      size: function () { return json.length; }
    };
    cb();
  });
}
```

mobike
摩拜单车

核心 vue-server-renderer

server-plugin.js

生成 vue-ssr-server-bundle.json

mobike
摩拜单车

核心 vue-server-renderer

```
var VueSSRServerPlugin = function VueSSRServerPlugin (options) {
  if ( options === void 0 ) options = {};

  this.options = Object.assign({
    filename: 'vue-ssr-server-bundle.json'
  }, options);
};

VueSSRServerPlugin.prototype.apply = function apply (compiler) {
  var this$1 = this;
  compiler.plugin('emit', function (compilation, cb) {
    var bundle = {
      entry: entry,
      files: {},
      maps: {}
    };
    var json = JSON.stringify(bundle, null, 2);
    compilation.assets[this$1.options.filename] = {
      source: function () { return json; },
      size: function () { return json.length; }
    };
    cb();
  });
}
```

mobike
摩拜单车

easy-mock

```
1  const Koa = require('koa')
2
3  const app = module.exports = new Koa()
4
5  const port = config.get('port')
6  const host = config.get('host')
7
8  app.use(require('./middlewares/view').render(app))
9  app.listen(port, host)
```

Koa 服务 + 中间件（包含 ssr 生成 html）

1、调用 createBundleRenderer

```
const { createBundleRenderer } = require('vue-server-renderer')
```

2、读模板文件

```
const templatePath = resolve('../views/index.html')
const template = fs.readFileSync(templatePath, 'utf-8')
```

3、读两个 json

```
const bundle = require('../dist/vue-ssr-server-bundle.json')
const clientManifest = require('../dist/vue-ssr-client-manifest.json')
```

4、创建 renderer

```
let renderer = createBundleRenderer(bundle, Object.assign({
    template, clientManifest }, {
    cache: LRU({
      max: 1000,
      maxAge: 1000 * 60 * 15
    }),
    // this is only needed when vue-server-renderer is npm-linked
    basedir: resolve('../dist'),
    // recommended for performance
    runInNewContext: false
  }))
```

5、生成 html

```
renderer.renderToString
```

```
var TemplateRenderer = function TemplateRenderer (options) {
  this.options = options;
  this.inject = options.inject !== false;
  // if no template option is provided, the renderer is created
  // as a utility object for rendering assets like preload links and scripts
  this.parsedTemplate = options.template
    ? parseTemplate(options.template)
    : null;

  // extra functionality with client manifest
  if (options.clientManifest) {
    var clientManifest = this.clientManifest = options.clientManifest;
    this.publicPath = clientManifest.publicPath.replace(/\/$/, '');
    // preload/prefetch directives
    this.preloadFiles = (clientManifest.initial || []).map(normalizeFile);
    this.prefetchFiles = (clientManifest.async || []).map(normalizeFile);
    // initial async chunk mapping
    this.mapFiles = createMapper(clientManifest);
  }
};
```

```
<script src="https://[redacted]/mock-server/files/dist/
manifest.7ba9c2e….js" defer></script>
<script src="https://[redacted]/mock-server/files/dist/
vendor.c8f86fa….js" defer></script>
<script src="https://[redacted]/mock-server/files/dist/app.f7c9712….js
defer></script>
```

```
TemplateRenderer.prototype.renderScripts = function renderScripts (context) {
    var this$1 = this;

  if (this.clientManifest) {
    var initial = this.preloadFiles;
    var async = this.getUsedAsyncFiles(context);
    var needed = [initial[0]].concat(async || [], initial.slice(1));
    return needed.filter(function (ref) {
        var file = ref.file;

        return isJS(file);
      }).map(function (ref) {
        var file = ref.file;

      return ('<script src=\"" + (this$1.publicPath) + "/" + file + "\" defer></script>")
    }).join('')
  } else {
    return ''
  }
};
```

```
<link rel="preload" href="https://[redacted]/mock-server/files/dist/manifest.7ba9c2e….js" as="script">
<link rel="preload" href="https://[redacted]/mock-server/files/dist/vendor.c8f86fa….js" as="script">
<link rel="preload" href="https://[redacted]/mock-server/files/dist/app.f7c9712….js" as="script">
<link rel="preload" href="https://[redacted]/mock-server/files/dist/common.f7c9712….css" as="style">
```

```
TemplateRenderer.prototype.renderPreloadLinks = function renderPreloadLinks (
    var this$1 = this;

  var files = this.getPreloadFiles(context);
  var shouldPreload = this.options.shouldPreload;
  if (files.length) {
    return files.map(function (ref) {

      var extra = '';
      // by default, we only preload scripts or css
      if (!shouldPreload && asType !== 'script' && asType !== 'style') {
        return ''
      }
      // user wants to explicitly control what to preload
      if (shouldPreload && !shouldPreload(fileWithoutQuery, asType)) {
        return ''
      }
      if (asType === 'font') {
        extra = " type=\"font/" + extension + "\" crossorigin";
      }
      return ("<link rel=\"preload\" href=\"" + (this$1.publicPath) + "/" + file + "\"" + (as
    }).join('')
  } else {
    return ''
  }
};
```

```
<link rel="stylesheet" href="https://[redacted]/mock-server/files/dist/
common.f7c9712….css"> == $0
```

```
TemplateRenderer.prototype.renderStyles = function renderStyles (context) {
    var this$1 = this;

  var cssFiles = this.clientManifest
    ? this.clientManifest.all.filter(isCSS)
    : [];
  return (
    // render links for css files
    (cssFiles.length
      ? cssFiles.map(function (file) { return ("<link
        rel=\"stylesheet\" href=\"" + (this$1.publicPath) + "/" + file + "\">"); }).join('')
      : '') +
    // context.styles is a getter exposed by vue-style-loader which contains
    // the inline component styles collected during SSR
    (context.styles || '')
  )
};
```

```
TemplateRenderer.prototype.renderPrefetchLinks = function renderPrefetchLinks
    var this$1 = this;

  var shouldPrefetch = this.options.shouldPrefetch;
  if (this.prefetchFiles) {
    var usedAsyncFiles = this.getUsedAsyncFiles(context);
    var alreadyRendered = function (file) {
      return usedAsyncFiles && usedAsyncFiles.some(function (f) { return f.file === file;
    };
    return this.prefetchFiles.map(function (ref) {…

      if (shouldPrefetch && !shouldPrefetch(fileWithoutQuery, asType)) {
        return ''
      }
      if (alreadyRendered(file)) {
        return ''
      }
      return ("<link rel=\"prefetch\" href=\"" + (this$1.publicPath) + "/" + file + "\">")
    }).join('')
  } else {
    return ''
  }
};
```

serialize-javascript

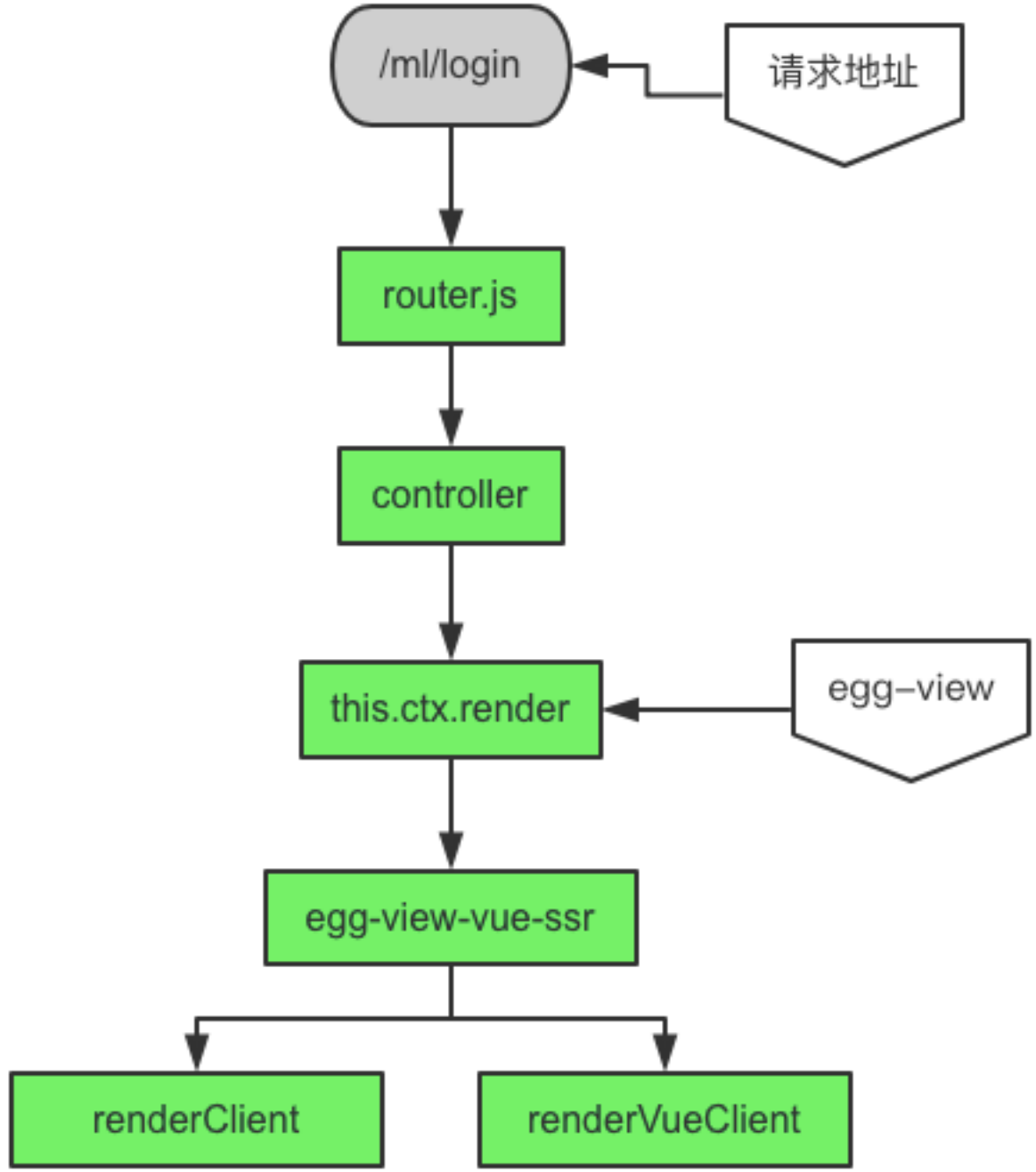
Serialize JavaScript to a _superset_ of JSON that includes regular expressions, dates and functions.

```
TemplateRenderer.prototype.renderState = function renderState (context, options) {
  var ref = options || {};
    var contextKey = ref.contextKey; if ( contextKey === void 0 ) contextKey = 'state';
    var windowKey = ref.windowKey; if ( windowKey === void 0 )
    windowKey = '__INITIAL_STATE__';
  var state = serialize(context[contextKey], { isJSON: true });
  var autoRemove = process.env.NODE_ENV === 'production'
    ? ';(function(){var s;(s=document.currentScript||document.scripts[document.scripts.le
    : '';
  return context[contextKey]
    ? ("<script>window." + windowKey + "=" + state + autoRemove + "</script>")
    : ''
};
```

模板插入设计

```
function parseTemplate (
  template,
  contentPlaceholder
) {
  if ( contentPlaceholder === void 0 ) contentPlaceholder = '<!--vue-ssr-outlet-->';

  if (typeof template === 'object') {
    return template
  }

  var i = template.indexOf('</head>');
  var j = template.indexOf(contentPlaceholder);

  if (j < 0) {
    throw new Error("Content placeholder not found in template.")
  }

  if (i < 0) {
    i = template.indexOf('<body>');
    if (i < 0) {
      i = j;
    }
  }

  return {
    head: compile$1(template.slice(0, i), compileOptions),
    neck: compile$1(template.slice(i, j), compileOptions),
    tail: compile$1(template.slice(j + contentPlaceholder.length), compileOptions)
  }
}
```

mobike
摩拜单车


/ml/login
请求地址
router.js
controller
this.ctx.render
egg–view
egg-view-vue-ssr
renderClient
renderVueClient

mobike
摩拜单车

基于 egg 的 ssr 设计

```
'use strict'
const Controller = require('egg').Controller

class ErrorController extends Controller {
  async e404 () {
    await this.ctx.render('error/e404.js')
  }

  async loginError () {
    await this.ctx.render('error/login-error.js')
  }
}

module.exports = ErrorController
```

egg-view-vue-ssr 为该项目的模板渲染插件，依赖于 vue-server-renderer 及 server-side-render-resource 。
该插件作为 egg-view 的一个模板引擎，会提供相应的 render renderClient 方法，支持模板渲染，静态资源注入。
启用插件配置位于 ${app_root}/config/plugin.js，插件配置位于 ${app_root}/config/config.default.js 。

2. egg-view-vue-ssr 插件主要功能

【egg-view-vue-ssr/config/config.default.js】下图 mapping 配置表示，render 方法传入的 .js 文件使用名称为 vue 的模板引擎来处理：

```
module.exports = app => {
  const config = {};

  config.view = {
    mapping: {
      '.js': 'vue',
    },
  };
```

【egg-view-vue-ssr/app/extend/application.js】挂载引擎，引擎逻辑位于 'lib/engine'，见下图：

```
const Engine = require('../../lib/engine');
const VUE_ENGINE = Symbol('Application#vue');

module.exports = {

  get vue() {
    if (!this[VUE_ENGINE]) {
      this[VUE_ENGINE] = new Engine(this);
    }
    return this[VUE_ENGINE];
  },
};
```

【egg-view-vue-ssr/app/extend/context.js】提供了renderClient 及 renderVueClient 方法，见下图：

```
module.exports = {
  renderClient(name, locals, options) {
    return this.renderVueClient(name, locals, options);
  },
  renderVueClient(name, locals, options = {}) {
    locals = this.app.vue.normalizeLocals(locals);
    return this.app.vue.renderClient(name, locals, options).then(html => {
      this.body = html;
    });
  },
};
```

【egg-view-vue-ssr/app.js】在插件入口文件，注册了一个 egg-view 的模板引擎 vue ， 具体逻辑处于 'lib/view' 下，见下图：

```
'use strict';
module.exports = app => {
  app.view.use('vue', require('./lib/view'));
};
```

mobike 摩拜单车

缓存设计

每次请求过来，从 jsBundle 转换成 html 的输出需要 cache

```
view: {
  root: path.join(appInfo.baseDir, 'app/view'),
  cache: true,
  defaultExtension: '.html',
  defaultViewEngine: '',
  mapping: {},
},
```

Egg-view

默认开启。
根据 root 配置的目录依次查找
如果匹配则会缓存文件路径，下次渲染相同路径时不会重新查找。

【render 方法】

环节一：this.app.vue.render(name, context, options) 首先调用了 vue 模板引擎提供的 render 方法返回渲染后的 html 部分

【egg-view-vue-ssr/lib/engine.js】

```
render(name, context, options) {
  context = context || /* istanbul ignore next */ {};
  options = options || /* istanbul ignore next */ {};

  return new Promise((resolve, reject) => {
    this.createBundleRenderer(name, options.renderOptions).renderToString(context, (err, html) => {
      if (err) {
        reject(err);
      } else {
        resolve(html);
      }
    });
  });
}
```

该方法中调用了 createBundleRenderer 方法，首先检测是否有 bundle 缓存，如果有，则获取缓存后直接返回；如果没有，则使用 vueServerRenderer 方法新创建，并设置缓存，然后返回。见下图：

```
if (this.config.cache === true) {
  this.bundleCache = LRU({
    max: 1000,
    maxAge: 1000 * 3600 * 24 * 7,
  });
} else if (typeof this.config.cache === 'object') {
  if (this.config.cache.set && this.config.cache.get) {
    this.bundleCache = this.config.cache;
  } else {
    this.bundleCache = LRU(this.config.cache);
  }
}
```

```
createBundleRenderer(name, renderOptions) {
  if (this.bundleCache) {
    const bundleRenderer = this.bundleCache.get(name);
    if (bundleRenderer) {
      return bundleRenderer;
    }
  }
  const bundleRenderer = this.vueServerRenderer.createBundleRenderer(name, Object.assign({}, this.renderOptions, renderOptions)
);
  if (this.bundleCache) {
    this.bundleCache.set(name, bundleRenderer);
  }
  return bundleRenderer;
}
```

mobike
摩拜单车

egg-cluster / egg-scripts

egg-cluster 来启动 Master 进程，不再需要使用 pm2

egg-scripts 来支持线上环境的运行和停止

mobike
摩拜单车

Inline source 的设计

文件内联的设计方式

Webpack 的方式：

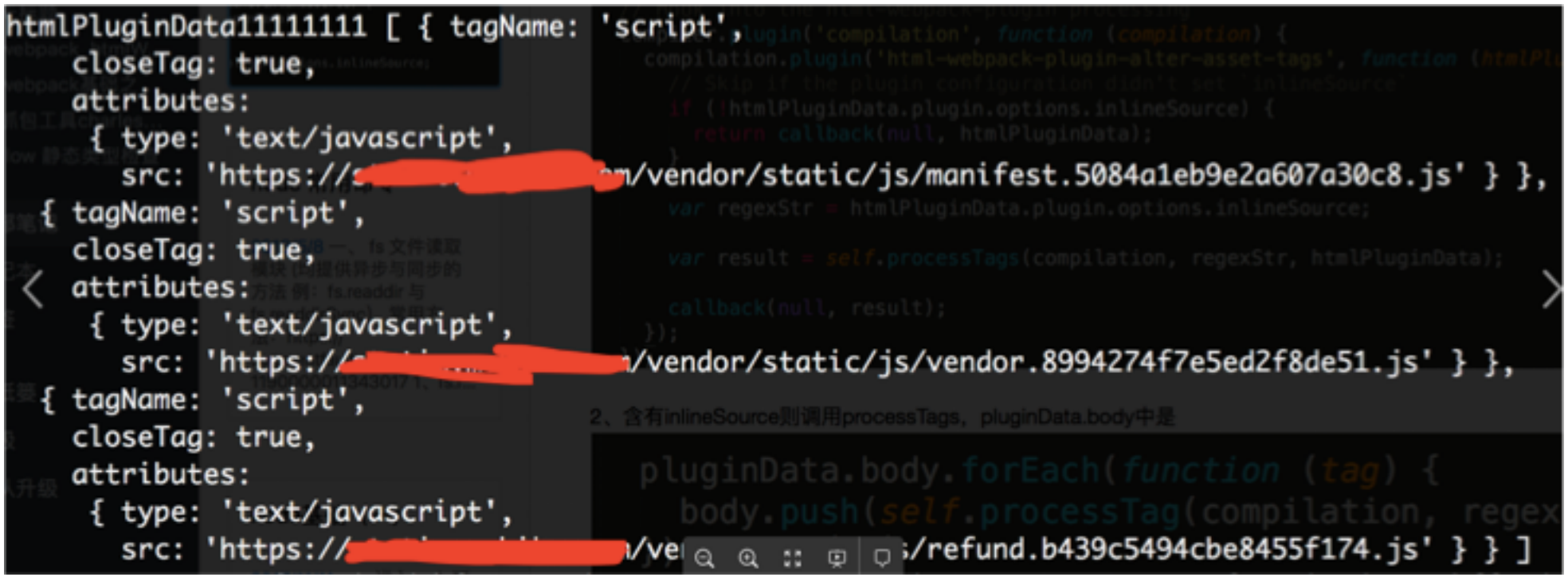
1、有 html 插件
html-webpack-plugin
html-webpack-inline-source-plugin

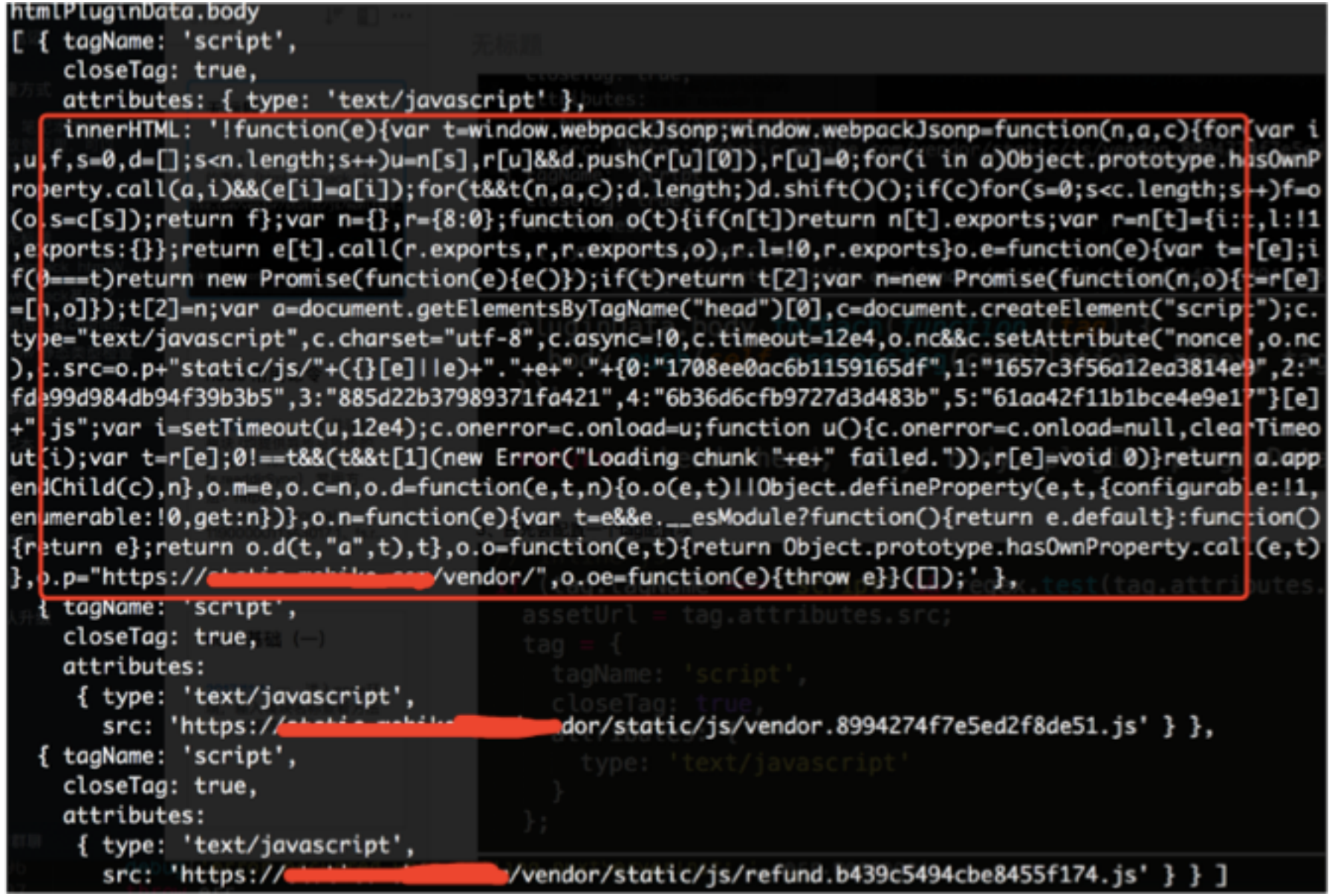
2、无 html 插件

mobike
摩拜单车

Inline source 的设计

```
HtmlWebpackInlineSourcePlugin.prototype.apply = function (compiler) {
  var self = this;

  // Hook into the html-webpack-plugin processing
  compiler.plugin('compilation', function (compilation) {
    compilation.plugin('html-webpack-plugin-alter-asset-tags', function (htmlPluginData, callback) {
      // Skip if the plugin configuration didn't set `inlineSource`
      if (!htmlPluginData.plugin.options.inlineSource) {
        return callback(null, htmlPluginData);
      }

      var regexStr = htmlPluginData.plugin.options.inlineSource;

      var result = self.processTags(compilation, regexStr, htmlPluginData);

      callback(null, result);
    });
  });
```


htmlPluginData11111111 [{ tagName: 'script',
closeTag: true,
attributes:
{ type: 'text/javascript',
src: 'https://[redacted]/vendor/static/js/manifest.5084a1eb9e2a607a30c8.js' } },
{ tagName: 'script',
closeTag: true,
attributes:
{ type: 'text/javascript',
src: 'https://[redacted]/vendor/static/js/vendor.8994274f7e5ed2f8de51.js' } },
{ tagName: 'script',
closeTag: true,
attributes:
{ type: 'text/javascript',
src: 'https://[redacted]/refund.b439c5494cbe8455f174.js' } }]
compilation.plugin('html-webpack-plugin-alter-asset-tags', function
if (!htmlPluginData.plugin.options.inlineSource) {
return callback(null, htmlPluginData);
var regexStr = htmlPluginData.plugin.options.inlineSource;
var result = self.processTags(compilation, regexStr, htmlPluginData);
callback(null, result);
2、含有inlineSource则调用processTags，pluginData.body中是
pluginData.body.forEach(function (tag) {
body.push(self.processTag(compilation, regex


htmlPluginData.body
[{ tagName: 'script',
closeTag: true,
attributes: { type: 'text/javascript' },
innerHTML: '!function(e){var t=window.webpackJsonp;window.webpackJsonp=function(n,a,c){for(var i
,u,f,s=0,d=[];s<n.length;s++)u=n[s],r[u]&&d.push(r[u][0]),r[u]=0;for(i in a)Object.prototype.hasOwnP
roperty.call(a,i)&&(e[i]=a[i]);for(t&&t(n,a,c);d.length;)d.shift()();if(c)for(s=0;s<c.length;s++)f=o
(o.s=c[s]);return f};var n={},r={8:0};function o(t){if(n[t])return n[t].exports;var r=n[t]={i:t,l:!1
,exports:{}};return e[t].call(r.exports,r,r.exports,o),r.l=!0,r.exports}o.e=function(e){var t=r[e];i
f(0===t)return new Promise(function(e){e()});if(t)return t[2];var n=new Promise(function(n,o){t=r[e]
=[n,o]});t[2]=n;var a=document.getElementsByTagName("head")[0],c=document.createElement("script");c.
type="text/javascript",c.charset="utf-8",c.async=!0,c.timeout=12e4,o.nc&&c.setAttribute("nonce",o.nc
),c.src=o.p+"static/js/"+({}[e]||e)+"."+e+"."+{0:"1708ee0ac6b1159165df",1:"1657c3f56a12ea3814e9",2:"
fde99d984db94f39b3b5",3:"885d22b37989371fa421",4:"6b36d6cfb9727d3d483b",5:"61aa42f11b1bce4e9e17"}[e]
+".js";var i=setTimeout(u,12e4);c.onerror=c.onload=u;function u(){c.onerror=c.onload=null,clearTimeo
ut(i);var t=r[e];0!==t&&(t&&t[1](new Error("Loading chunk "+e+" failed.")),r[e]=void 0)}return a.app
endChild(c),n},o.m=e,o.c=n,o.d=function(e,t,n){o.o(e,t)||Object.defineProperty(e,t,{configurable:!1,
enumerable:!0,get:n})},o.n=function(e){var t=e&&e.__esModule?function(){return e.default}:function()
{return e};return o.d(t,"a",t),t},o.o=function(e,t){return Object.prototype.hasOwnProperty.call(e,t)
},o.p="https://[redacted]/vendor/",o.oe=function(e){throw e}}([]);' },
{ tagName: 'script',
closeTag: true,
attributes:
{ type: 'text/javascript',
src: 'https://[redacted]dor/static/js/vendor.8994274f7e5ed2f8de51.js' } },
{ tagName: 'script',
closeTag: true,
attributes:
{ type: 'text/javascript',
src: 'https://[redacted]/vendor/static/js/refund.b439c5494cbe8455f174.js' } }]

```
// inline js
if (tag.tagName === 'script' && regex.test(tag.attributes.src)) {
  assetUrl = tag.attributes.src;
  tag = {
    tagName: 'script',
    closeTag: true,
    attributes: {
      type: 'text/javascript'
    }
  };
```

```
if (assetUrl) {
  // Strip query string (e.g. cache busting hash) from asset URL
  assetUrl = assetUrl.replace(/\?.*$/, '');
  // Strip public URL prefix from asset URL to get Webpack asset name
  var publicUrlPrefix = compilation.outputOptions.publicPath || '';
  // 去掉publicPath以后的路径
  var assetName = path.posix.relative(publicUrlPrefix, assetUrl);
  // 从编译环境中取到对应的源码
  var asset = compilation.assets[assetName];
  var updatedSource = this.resolveSourceMaps(compilation, assetName, asset);
  tag.innerHTML = updatedSource;
}

return tag;
```

mobike
摩拜单车

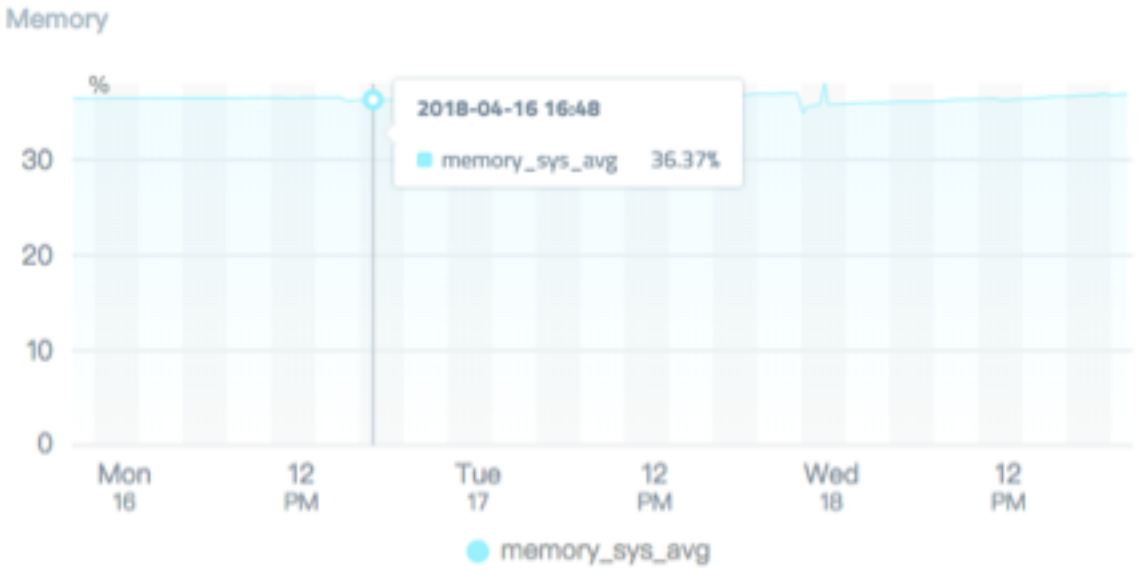
监控服务 egg-alinode

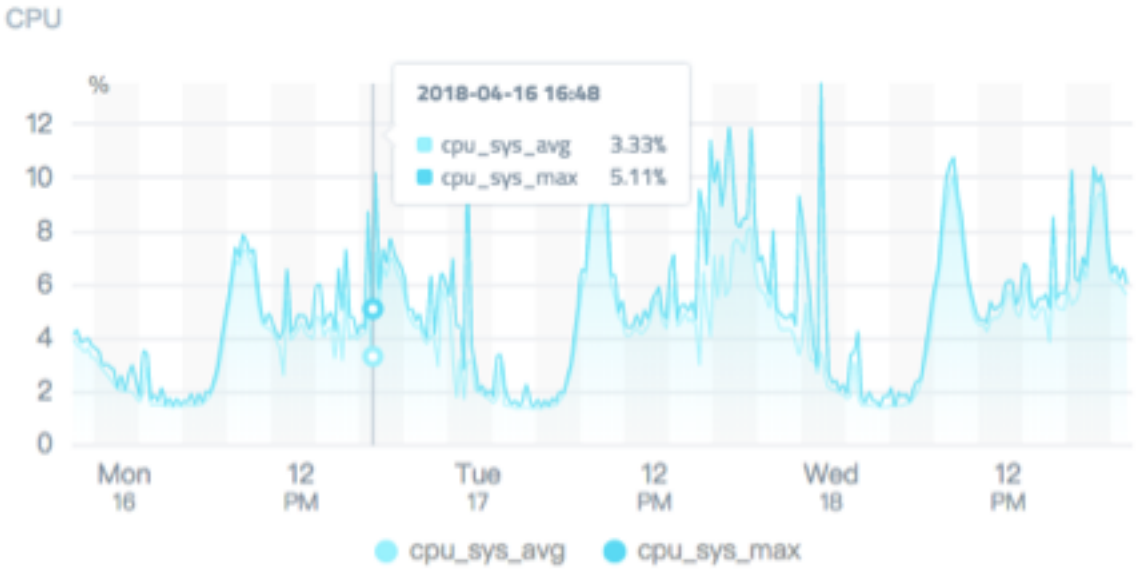
问题：

1、容器化部署方式：
配置 ip 来防止名字不变导致的重复

2、多个 appid 来区分环境

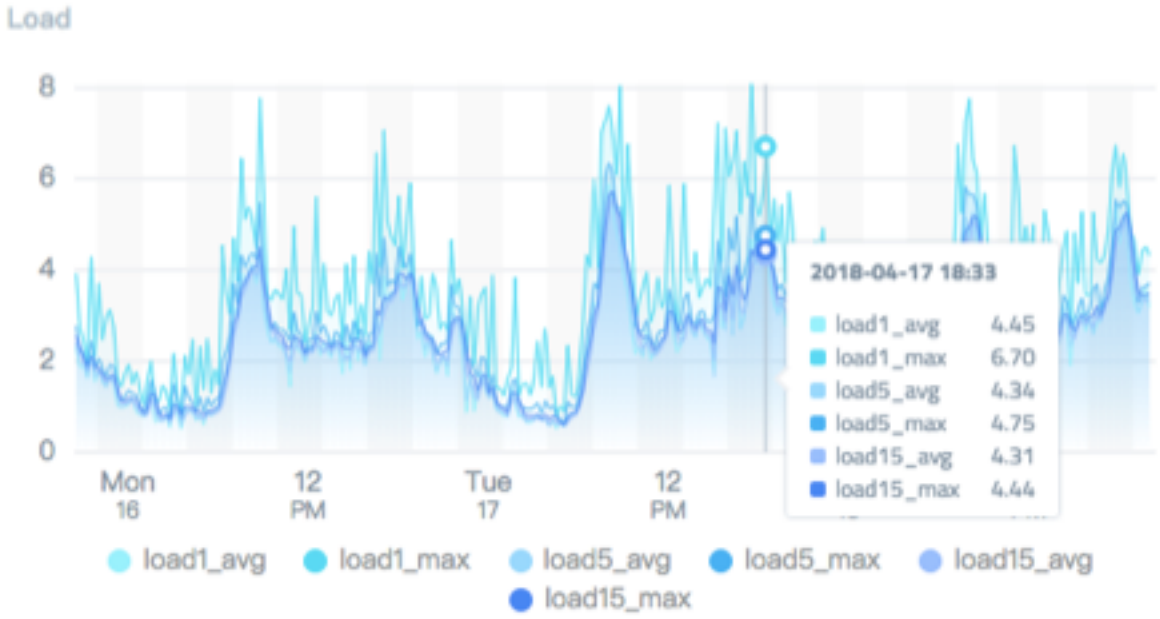
```
exports.alinode = {
  appid: '**',
  secret: '****',
  agentidMode: 'IP'
};
```

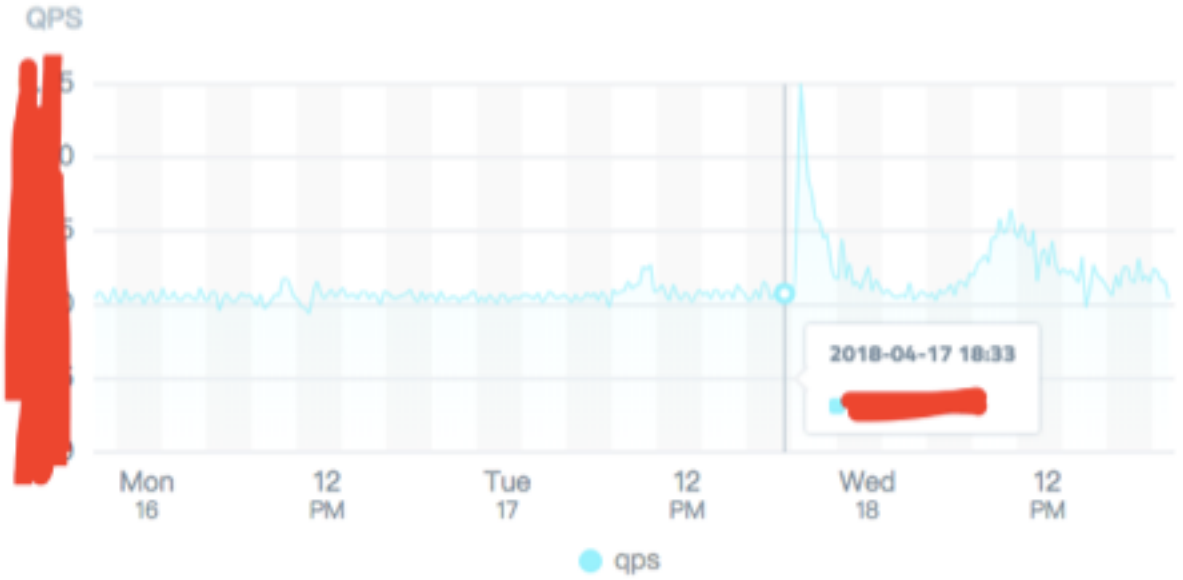




mobike
摩拜单车

监控服务 egg-alinode



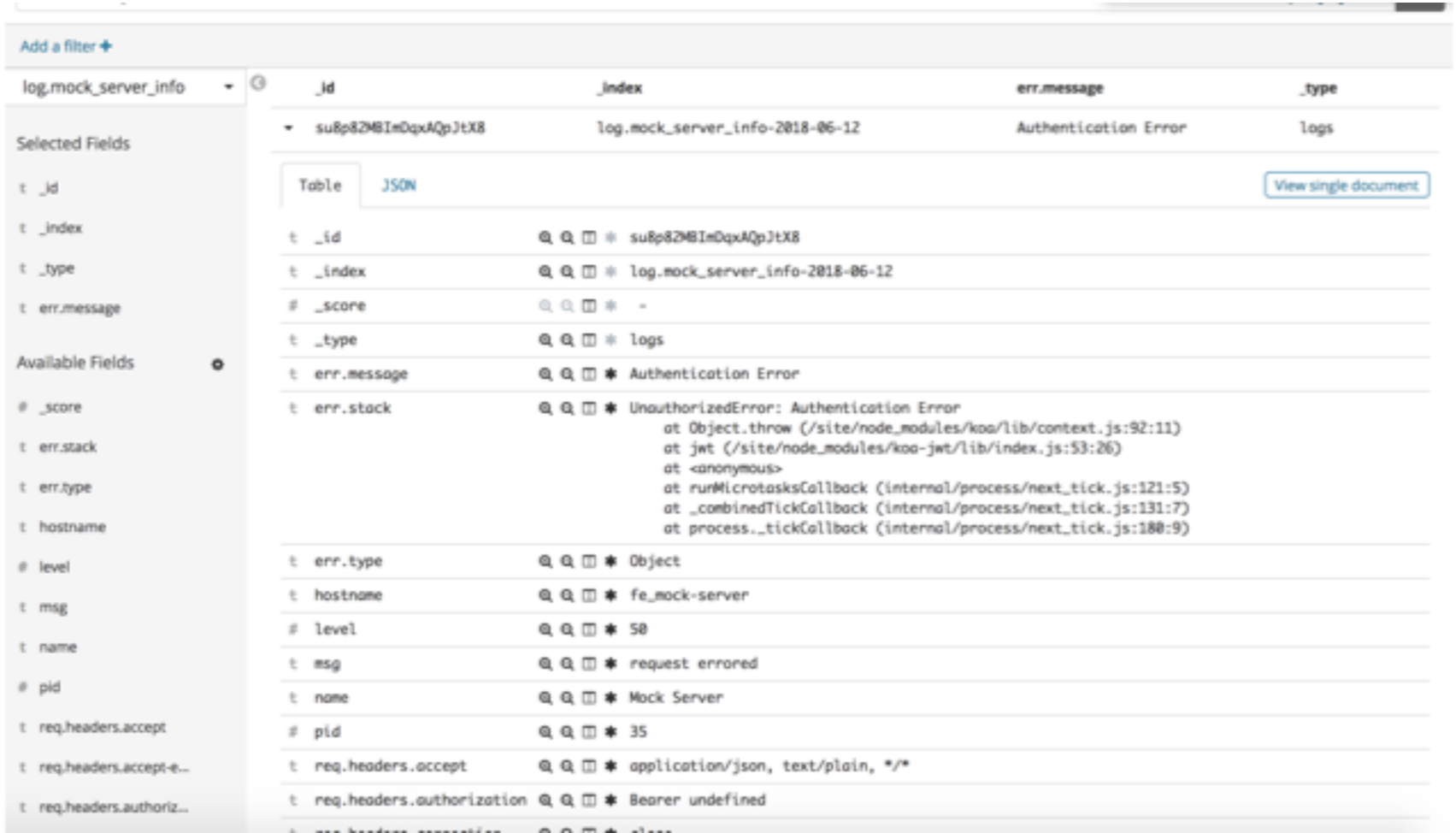


核心还是：agentx

mobike
摩拜单车

摩拜监控服务流程设计

Docker 化 + elk + dashboard



mobike
摩拜单车

摩拜devOps – 配置

.gitlab-ci.yml

```
npm:
  stage: build
  script:
  - [redacted]
  - docker build -f DockerfileDev -t $DOCKER_REPO/$CI_PROJECT_PATH:$CI_COMMIT_TAG .
  - docker tag  $DOCKER_REPO/$CI_PROJECT_PATH:$CI_COMMIT_TAG $DOCKER_REPO/$CI_PROJECT_PATH:latest
  - docker push $DOCKER_REPO/$CI_PROJECT_PATH:$CI_COMMIT_TAG
  - docker push $DOCKER_REPO/$CI_PROJECT_PATH:latest
  only:
  - /^dev_[redacted]/

npm_production:
  stage: build
  script:
  - docker build -t $DOCKER_REPO/$CI_PROJECT_PATH:$CI_COMMIT_TAG .
  - docker tag  $DOCKER_REPO/$CI_PROJECT_PATH:$CI_COMMIT_TAG $DOCKER_REPO/$CI_PROJECT_PATH:latest
  - docker push $DOCKER_REPO/$CI_PROJECT_PATH:$CI_COMMIT_TAG
  - docker push $DOCKER_REPO/$CI_PROJECT_PATH:latest
  only:
  - /^release_[redacted]/
```

mobike
摩拜单车

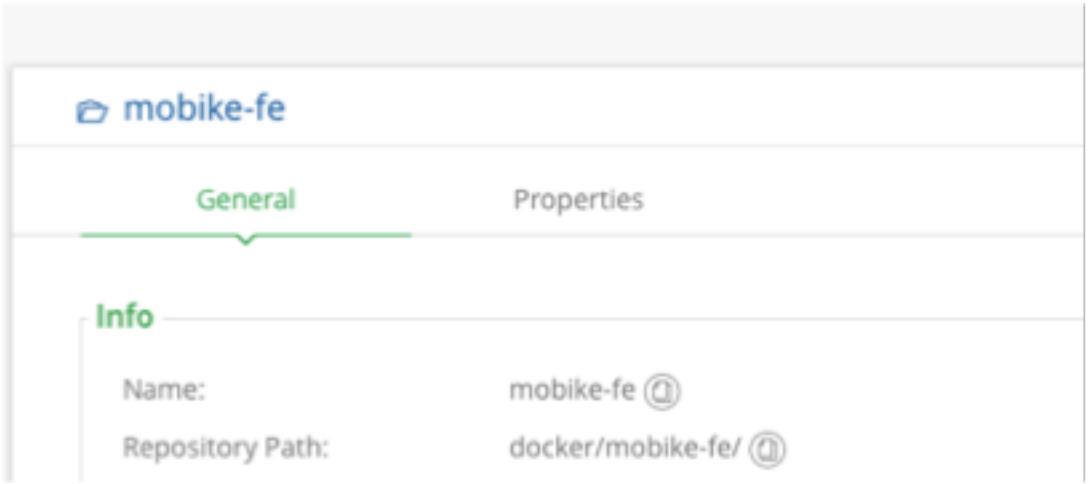
摩拜devOps – 私服

1、npm

npm init --scope=mobike

2、docker

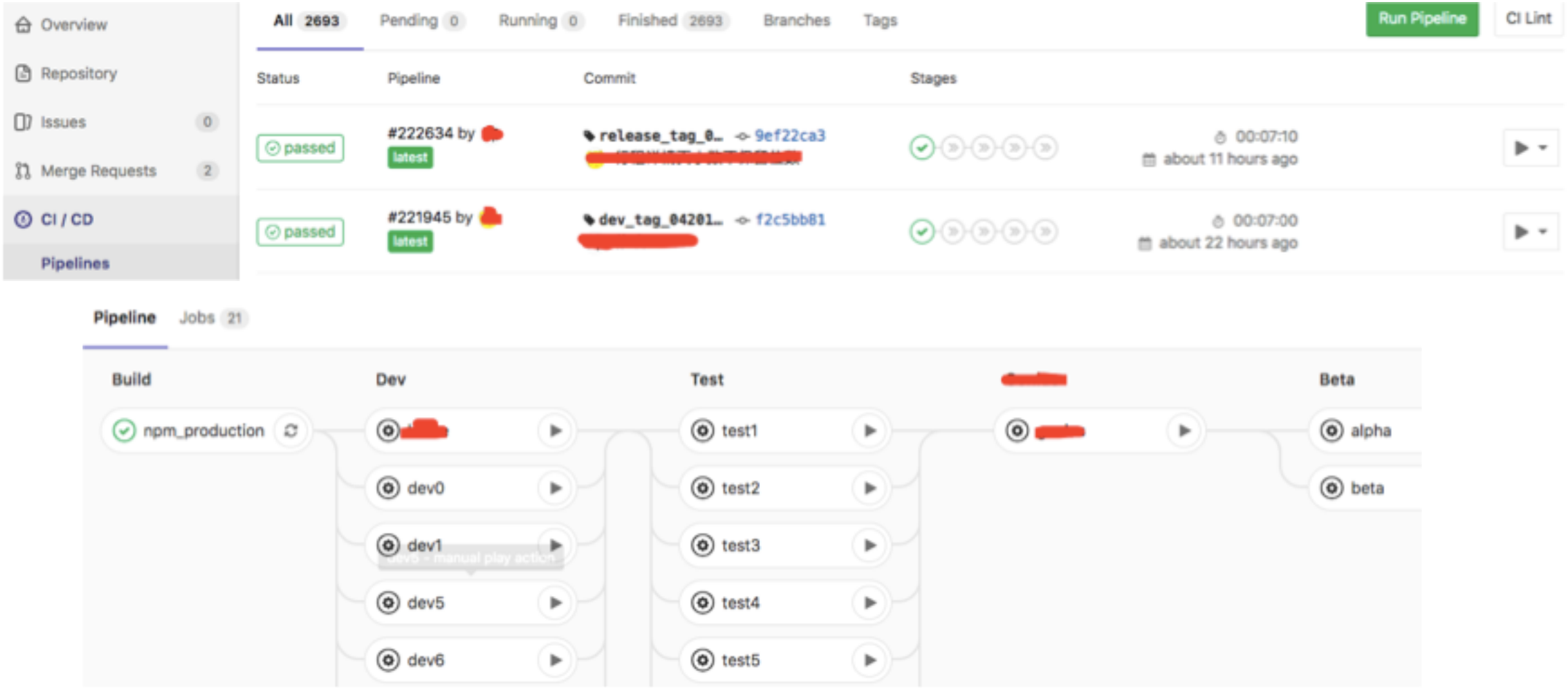
docker tag
docker push



3、node 多版本镜像

摩拜devOps – 发布流程自动化

从 make tag 开始



mobike 摩拜单车

摩拜devOps – 发布流程自动化

静态项目编译做了什么？

Running runner
Checking out 9ef22ca3 as tag名字
npm install
rm -fr dist
npm run deploy
docker build
docker tag
docker push　一般会有 2 次！！！

Node.js Dockerfile

```
1 FROM docker.***/infra/***-node:node8
2 RUN mkdir -p /***
3 ENV CONSUL_TAGS fe,***
4 COPY . /***
5 WORKDIR /****
6 EXPOSE ***
7 RUN npm install
8 RUN npm run build
9 CMD ["start"]
```

mobike
摩拜单车

摩拜devOps – 发布流程自动化

做了什么？

```
Step 1/9 : FROM docker.[redacted]/infra/[redacted]-node:node8
 ---> e401ee540821
Step 2/9 : RUN mkdir -p /app
 ---> Using cache
 ---> 6910fc07509f
Step 3/9 : ENV CONSUL_TAGS fe,[redacted]
 ---> Using cache
 ---> e52681b0f266
Step 4/9 : COPY . /app
 ---> a3dddd704ef2
Step 5/9 : WORKDIR /app
 ---> 4f0d532f8a55
Removing intermediate container 56214d3ef01f
Step 6/9 : EXPOSE 3000
 ---> Running in 73cb972c915d
 ---> 4216f1491e9d
Removing intermediate container 73cb972c915d
Step 7/9 : RUN npm install
 ---> Running in bc418d05ac56
```

```
docker push $DOCKER_REPO/$CI_PROJECT_PATH:$CI_COMMIT_TAG

docker push $DOCKER_REPO/$CI_PROJECT_PATH:latest
```

mobike
摩拜单车

配置存取 config

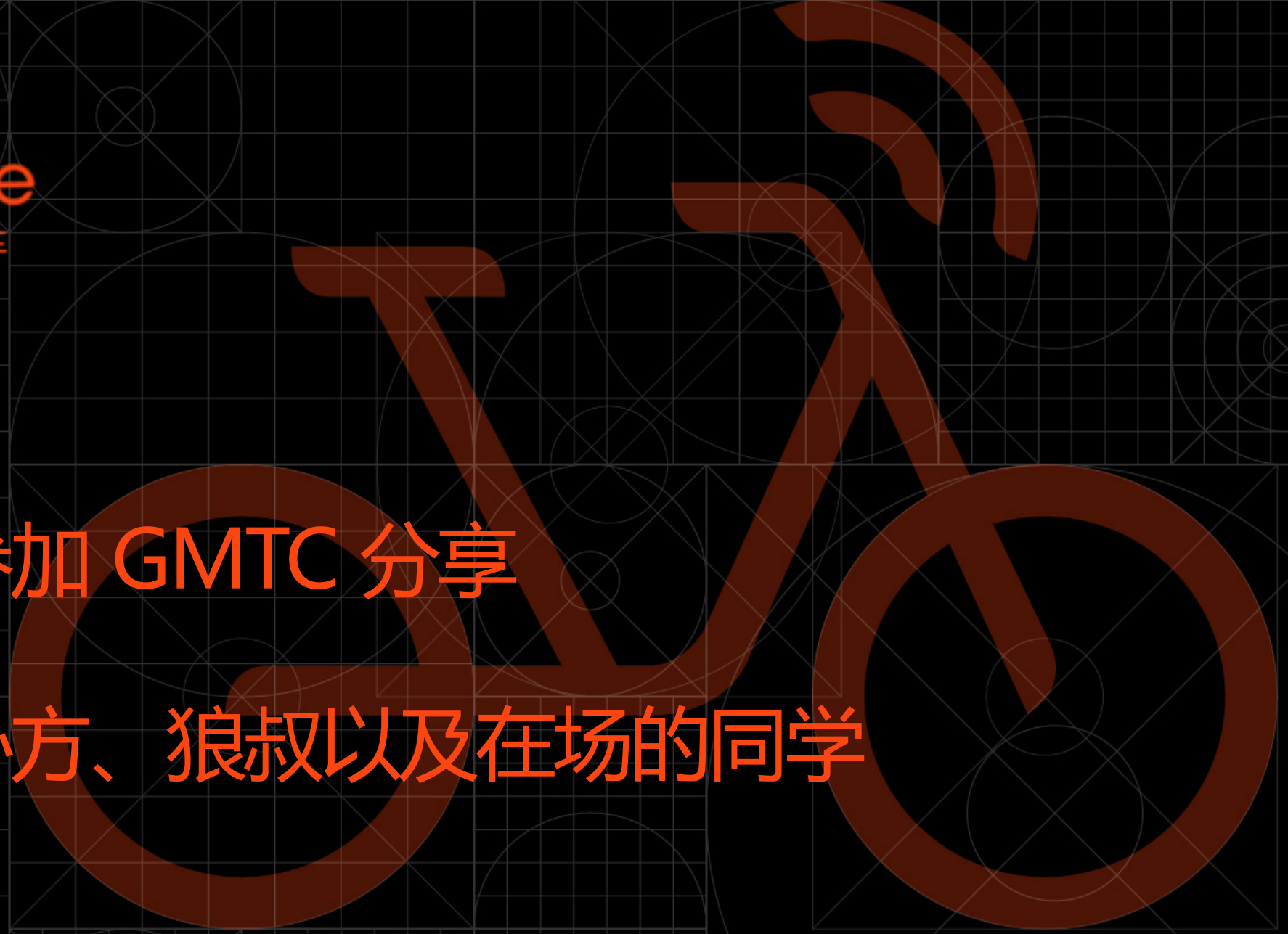
https://www.npmjs.com/package/config

Easy-mock 和 nuxt-config 多内置了它

Egg 中的 egg-core 也类似文件级别，但是是自己解析

```
12
13   /**
14    * Load config/config.js
15    *
16    * Will merge config.default.js 和 config.${env}.js
17    *
18    * @method EggLoader#loadConfig
19    * @since 1.0.0
20    */
21   loadConfig() {
22     this.timing.start('Load Config');
23     this.configMeta = {};
24
25     const target = {};
```

mobike
摩拜单车



很荣幸参加 GMTC 分享

感谢主办方、狼叔以及在场的同学

mobike
摩拜单车

关注我们

前端新视野：原创 + 日刊（每天 3 篇推荐过滤的文章）

mobike
摩拜单车

关注我们

安卓编程：日刊（每天 2 篇推荐过滤的文章）