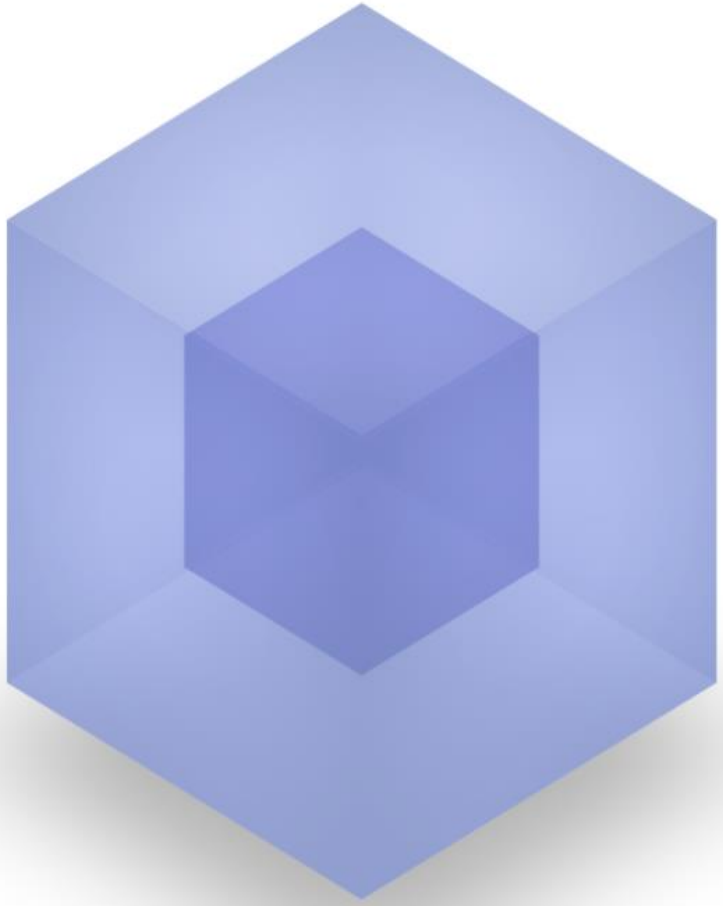


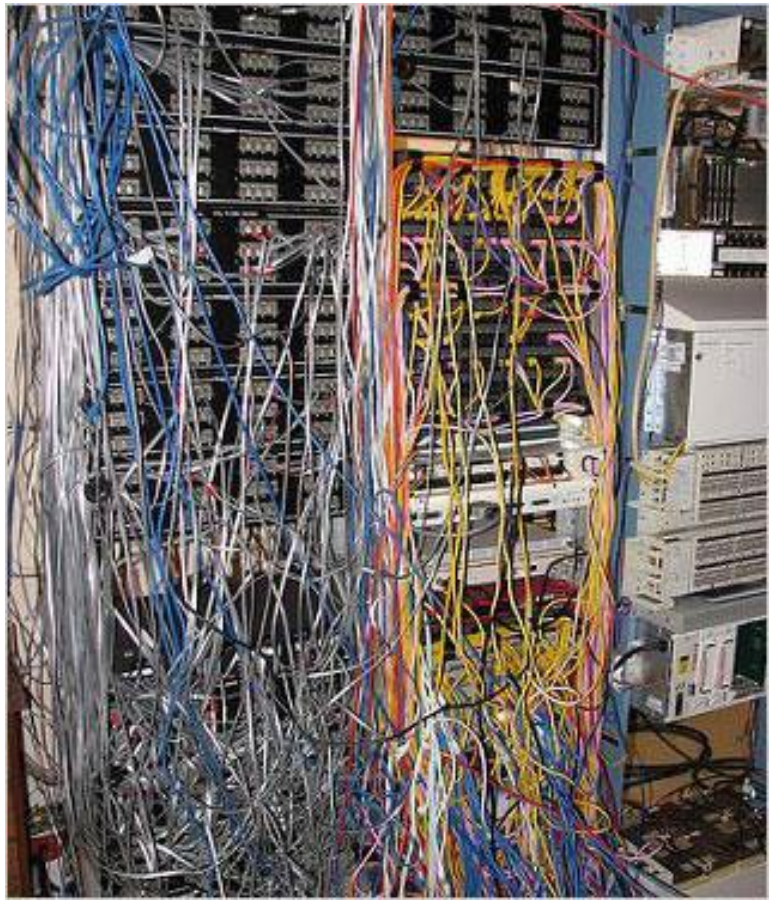
webpack

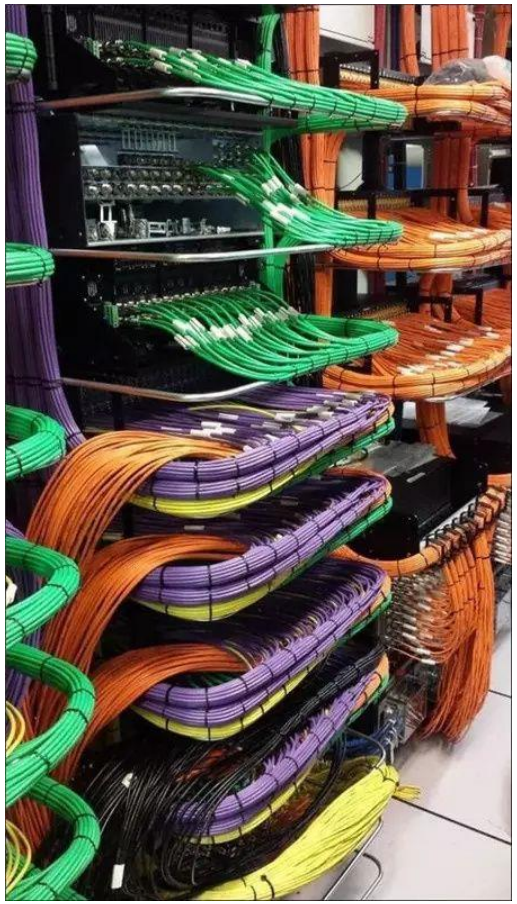
MODULE BUNDLER

以前

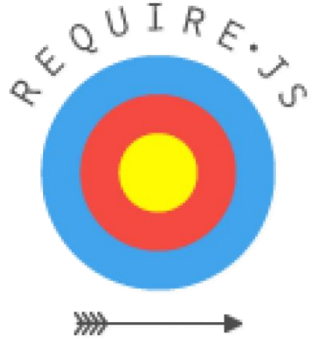
- <script src=“a.js"></script>
- <script src=“b.js"></script>
- <script src=“c.js"></script>
- <script src=“d.js"></script>
- <script src=“e.js"></script>
- <script src=“f.js"></script>

规范和模块化





AMD/CMD




```
require(['jquery’,  'underscore’, 'backbone'], function ($,  _, Backbone){

})
```

1. 依赖前置，预先执行
2. 支持异步

Sea.js

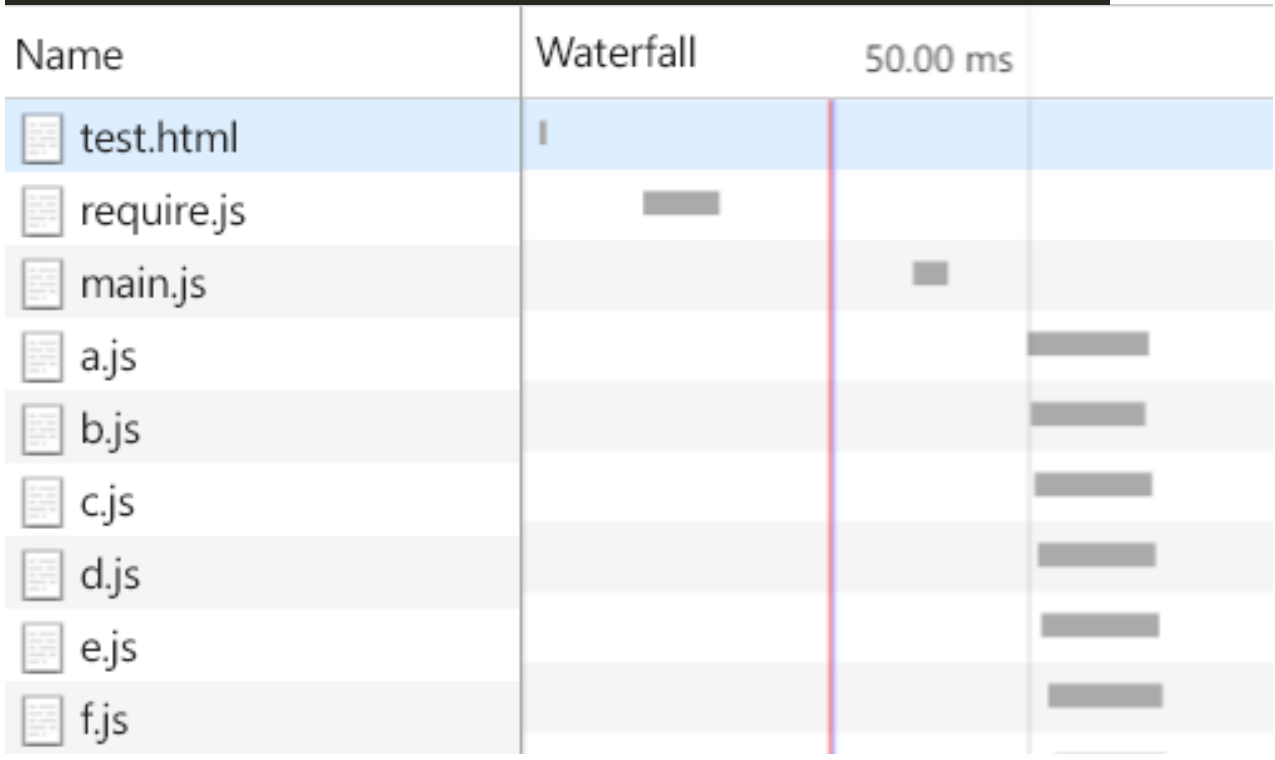
```
define(function(require, exports, module) {
    var $ = require(‘jquery’)
    var _ = require(‘underscore’)
    var Backbone = require(‘backbone’)
})
```

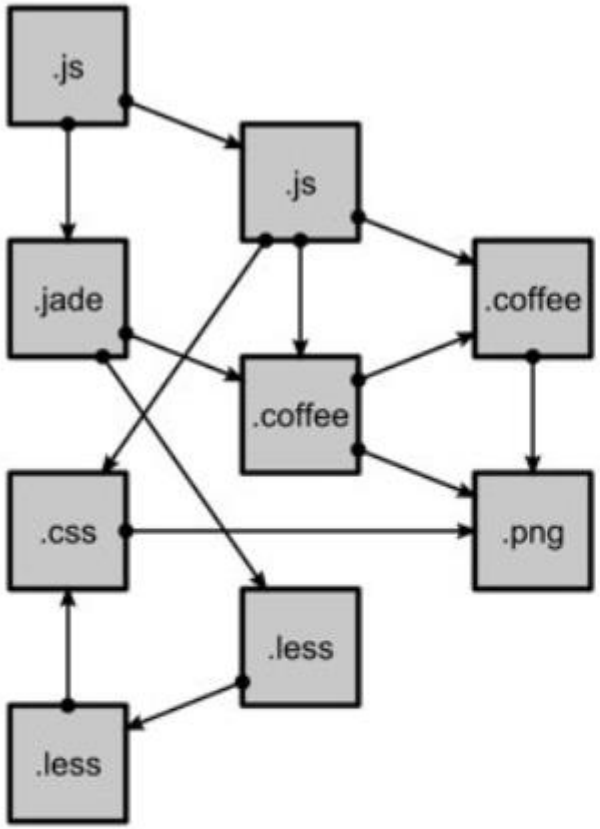
1. 依赖就近，延迟执行
2. 节省资源

存在问题

1. 合并请求和按需加载的冲突
2. 需要加载require.js文件
3. Others?

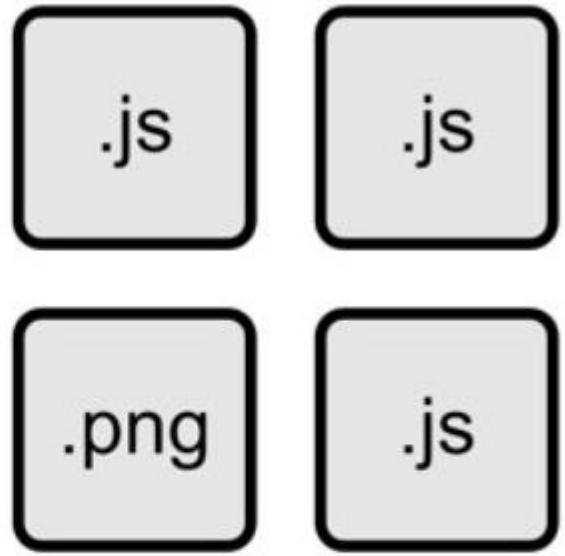
```
require(['a', 'b', 'c', 'd', 'e', 'f'],
function(a, b, c, d, e, f) {

})
```





modules
with dependencies

webpack
MODULE BUNDLER



static
assets

Hello world

编写代码

```
// app.js
import bar from './bar';

bar();


// bar.js
export default function bar() {
  //
}
```

使用 webpack 打包

```
// webpack.config.js
module.exports = {
  entry: './app.js',
  output: {
    filename: 'bundle.js'
  }
};
```

Html

```
<script src="bundle.js"></script>
```

entry and output

```
// @webpack.config.js
module.exports = {
  entry: {},
  output: {},
  module: {
    rules: [
    ]
  },
  plugins: [],
  externals: [],
  resolve: {},
  devtool: '',
}
```

```
entry:  String | [String] | Object | (() => {})
entry: {
  app: './src/main.js'
}


output: {
  path: path.resolve(__dirname, 'dist'),
  filename: 'bundle.js'/ '[name].[hash].[id].[chunkhash].js',
  chunkFilename: '[id].[chunkhash].js',
  publicPath: "/"
},
```

loader - 幕后功臣

```
// @webpack.config.js
module.exports = {
  entry: {},
  output: {},
  module: {
    rules: [
    ]
  },
  plugins: [],
  externals: [],
  resolve: {},
  devtool: '',
}
```

条件 + 结果 + 嵌套

```
{
  test: /\.js$/,
  loader: 'babel-loader',
  include: [resolve('src'), resolve('test')]
},
{
  test: /\.(png|jpe?g|gif|svg)(\?.*)?$/,
  loader: 'url-loader',
  query: {
    limit: 10000,
    name: utils.assetsPath('img/[name].[hash:7].[ext]')
  }
},
```

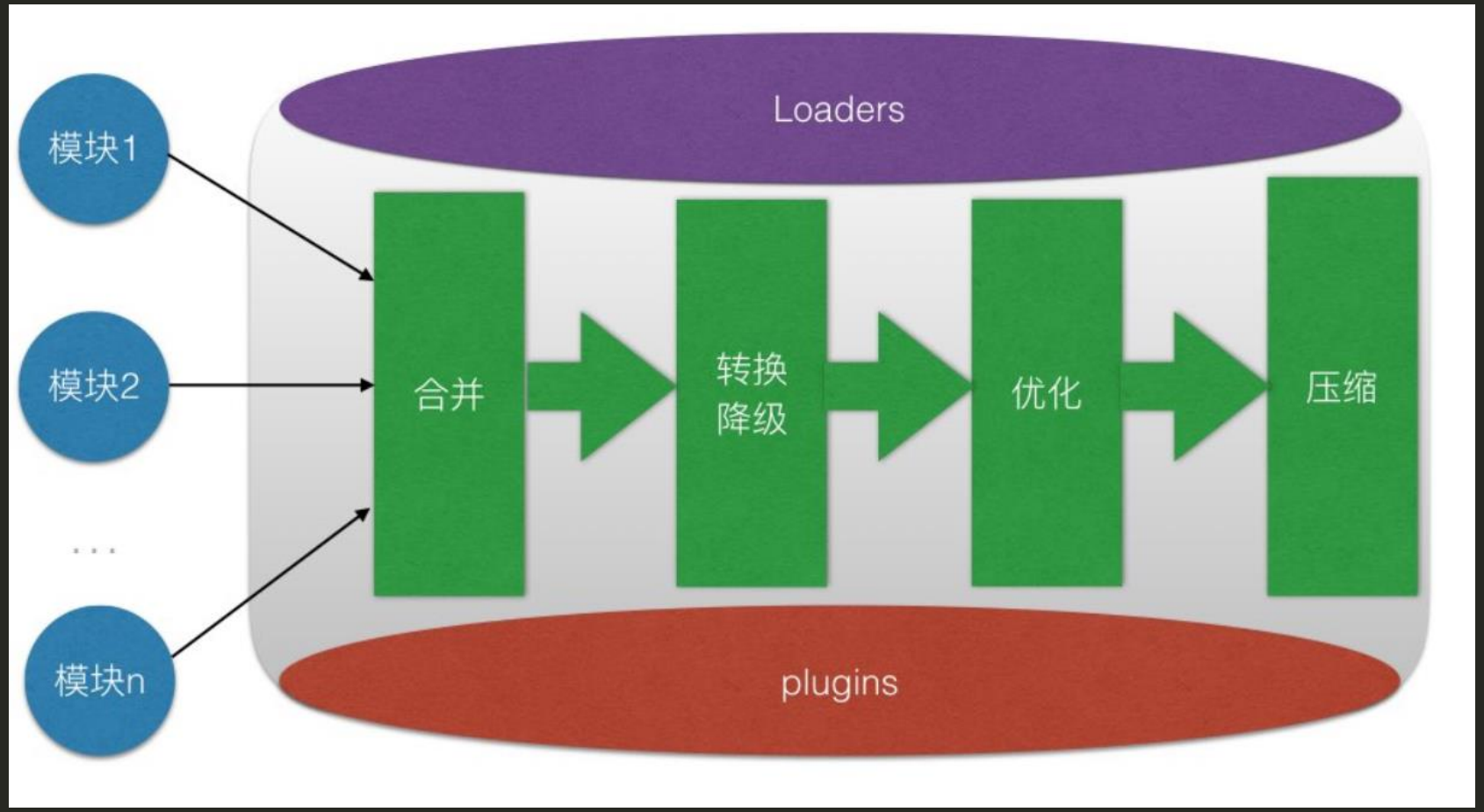
plugins - 庞大

```
// @webpack.config.js
module.exports = {
  entry: {},
  output: {},
  module: {
    rules: [
    ]
  },
  plugins: [],
  externals: [],
  resolve: {},
  devtool: '',
}
```

文件优化

```
new webpack.optimize.CommonsChunkPlugin({
  name: ['vendor', 'manifest'],
}),
new HtmlWebpackPlugin({
  filename: 'index.html',
  template: 'index.prod.html',
}),
new webpack.optimize.UglifyJsPlugin({
  compress: {
    warnings: false,
  }
}),
```

模块处理



其他

```
// @webpack.config.js
module.exports = {
  entry: {},
  output: {},
  module: {
    rules: [
    ]
  },
  plugins: [],
  externals: {},
  resolve: {},
  devtool: '',
}
```

externals: 防止模块被打包

```
externals: {
  jquery: 'jQuery'
}
```

```
// @index.prod.html
<script src="https://code.jquery.com/jquery-3.1.0.js"
integrity="sha256-slogkvB1K3VOkzAI8QITxV3VzpOnkeNVsKvtkYLMjfk="
crossorigin="anonymous"></script>
```

resolve: (解析) 后缀，模块路径等

devtool: 开发专用~

其他： devServer, context 等等

NPM – 依赖管理

- 55w+ 个模块

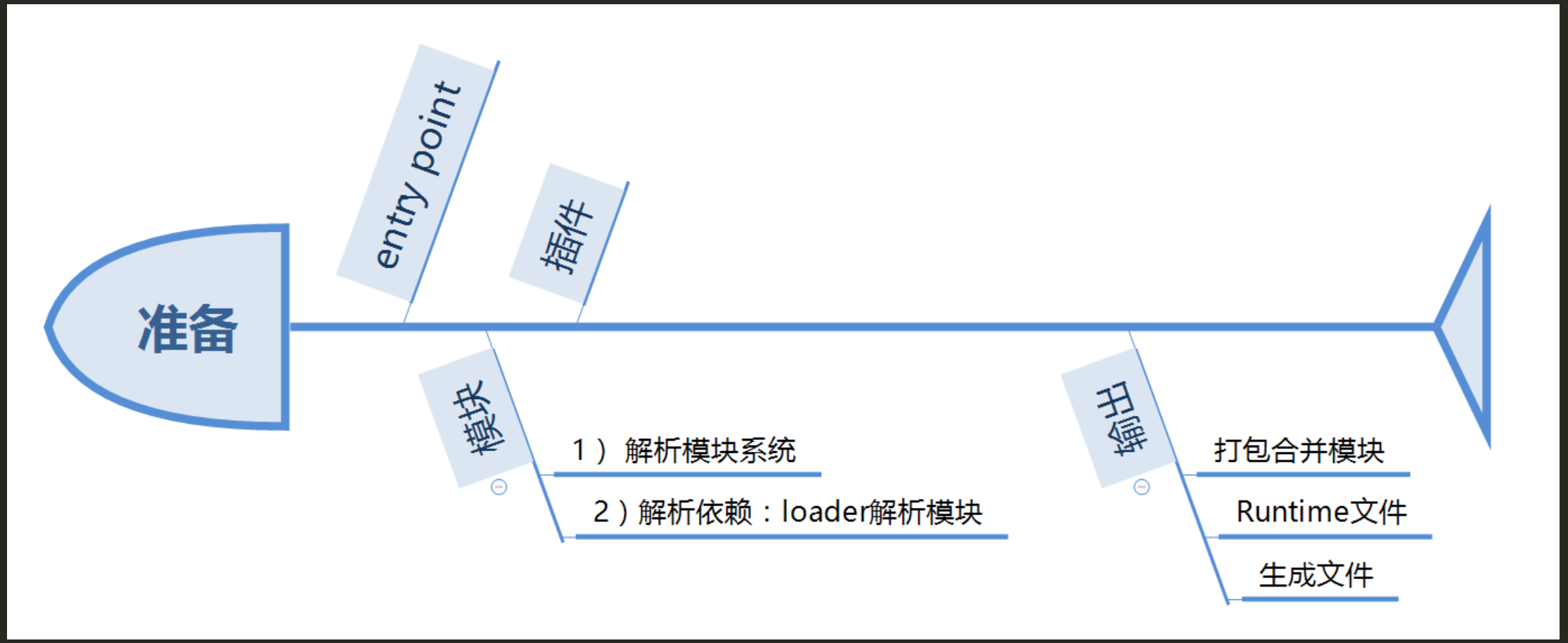
- Npm init配置package.json文件
- Npm install安装依赖

```
"scripts": {
  "dev": "node build/dev-server.js",
  "build": "node build/build.js"
},
"dependencies": {
  "vonic": "^2.0.0-beta.14",
  "vuex": "^2.3.0"
},
"devDependencies": {
  "autoprefixer": "^6.7.2",
  "babel-core": "^6.22.1",
  "babel-loader": "^6.2.10",
```

扩展部分

[2/2]

Webpack


准备
entry point
插件
模块
1）解析模块系统
2）解析依赖：loader解析模块
输出
打包合并模块
Runtime文件
生成文件

模块系统

ES6模块

```
import '...'
import abc from '....'
import { abc, cba } from '...'
import * as abc from '...'
import { cba as abc } from '...'

export let a = 1
export default class Person extends Component {}
```

CMD: var a = require(‘a’)

AMD: require([‘a’], function(a) { })

manifest.js: __webpack_require__

runtime与manifest，CommonsChunkPlugin

缓存：

Vendor

```
output: {
  path: '/dist',
  filename: 'js/[name].[chunkhash].js',
  chunkFilename: 'js/[id].[chunkhash].js'
}
```

runtime: 连接模块代码， webpackJsonp，__webpack_require__

manifest :

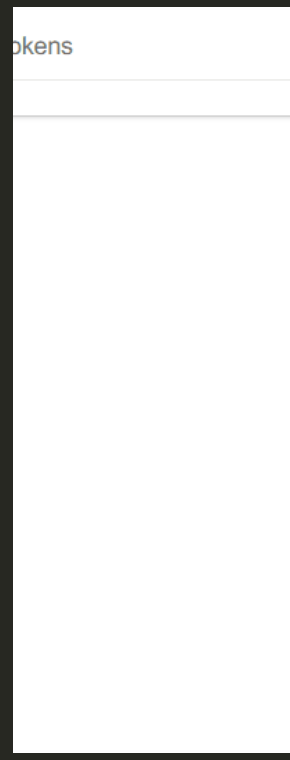
```
{
  "app.css": "cfld_static/css/app.78b2a68d400f1bfd4aded542b436d03a.css",
  "app.css.map": "cfld_static/css/app.78b2a68d400f1bfd4aded542b436d03a.css.map",
  "app.js": "cfld_static/js/app.4a79c23b98ccb38da3ad.js",
  "app.js.map": "cfld_static/js/app.4a79c23b98ccb38da3ad.js.map",
  "manifest.js": "cfld_static/js/manifest.7d4a07e3079f8d188760.js",
  "manifest.js.map": "cfld_static/js/manifest.7d4a07e3079f8d188760.js.map",
  "vendor.js": "cfld_static/js/vendor.d9b009f72a0461349f5c.js",
  "vendor.js.map": "cfld_static/js/vendor.d9b009f72a0461349f5c.js.map"
}
```

Asset	Size	Chunks		Chunk Names
cfld_static/js/app.4a79c23b98ccb38da3ad.js	113 kB	0	[emitted]	app
cfld_static/js/vendor.d9b009f72a0461349f5c.js	32.8 kB	1	[emitted]	vendor
cfld_static/js/manifest.7d4a07e3079f8d188760.js	6 kB	2	[emitted]	manifest
tic/css/app.78b2a68d400f1bfd4aded542b436d03a.css	25.2 kB	0	[emitted]	app

简单的webpack

1. 分析模块依赖关系esprima
2. 找到模块
3. 拼接output
   a. 依赖关系depTree
   b. chunks and modules
   c. 拼接
   d. 输出

```
var depTree = {
  modules: {},
  modulesById: {},
  chunks: {},
  nextModuleId: 0,
  nextChunkId: 0,
  chunkModules: {}
}
modules = {
  'F:\\example\\example.js': {
    id: 0,
    filename: 'F:\\example\\example.js',
    name: '',
    require: [],
    source: 'console.log(123456)',
    chunks: []
  }
}
```



细说 webpack 之流程篇

code-splitting

1. 代码分离类型
   a. Entry 配置
   b. CommonsChunkPlugin插件
   c. Require.ensure

2. 动态加载语法
() => import('./Foo.vue')

```
// react-router
const chooseProducts = (location, cb) => require.ensure([], require => {
    cb(null, require('../Component/chooseProducts').default)
},'chooseProducts')


// Vue-router
const orderDetail = resolve => require.ensure([], () =>
    resolve(require('../page/order/children/orderDetail')),
'orderDetail')
```

Q&A