



基于Vue 2.0的Element 通用组件库开发

曾海平

饿了么 大前端 机动架构组
高级前端开发工程师



一、设计目的

二、项目开发与实现

1. 组件设计模式和规范
2. 项目打包构建工程设计
3. 主题定制和覆盖

日渐增多的后台产品需求



企业后台项目的传统开发模式



产品经理+设计师



前端+后端+测试

企业资源尽量投入在核心业务上

痛点

- 开发与设计资源有限，没办法支持所有业务线
- 公司内部诸多后台产品使用体验不一致
- 各个项目中基础设施重复地造轮子



想马儿跑又想马儿不吃草



开发通用的 基础组件库



在节省研发成本的优势下
还保持了各个后台产品的高质量与体验

期望和目标

使用成本低廉

低耦合与可拓展性强

API 友好，文档清晰

主题可定制



Element

开发之路



×

Text text text

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad. Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad.

Button Text

Label ▼

Label Text

Label Text

Label Text

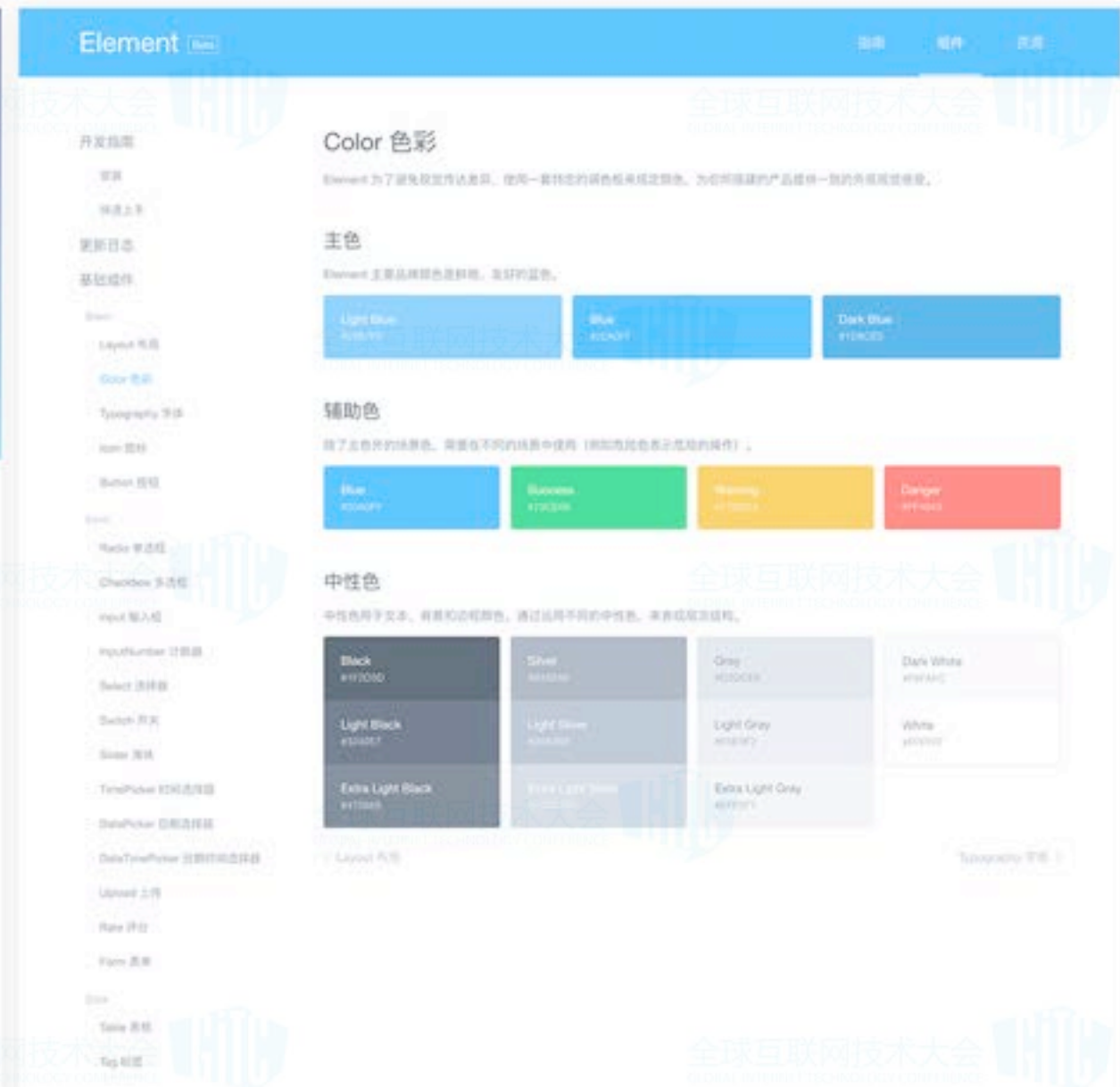
Label Text

Label Text

Tab 1

Tab 2

Tab 3

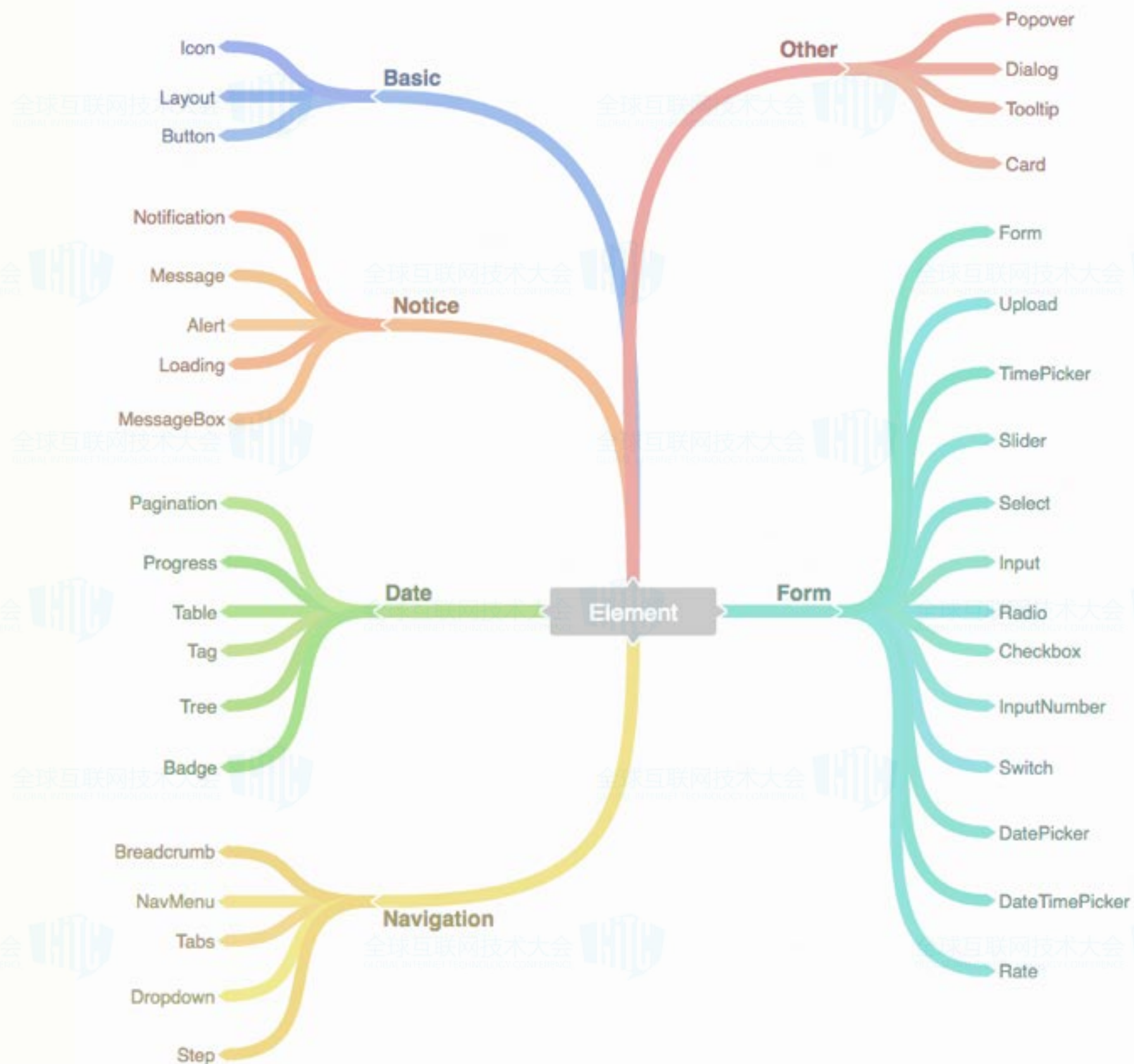




我们需要哪些组件？



企业级后台的组件生态





技术实现

技术选型

编程规范

打包工程

主题定制



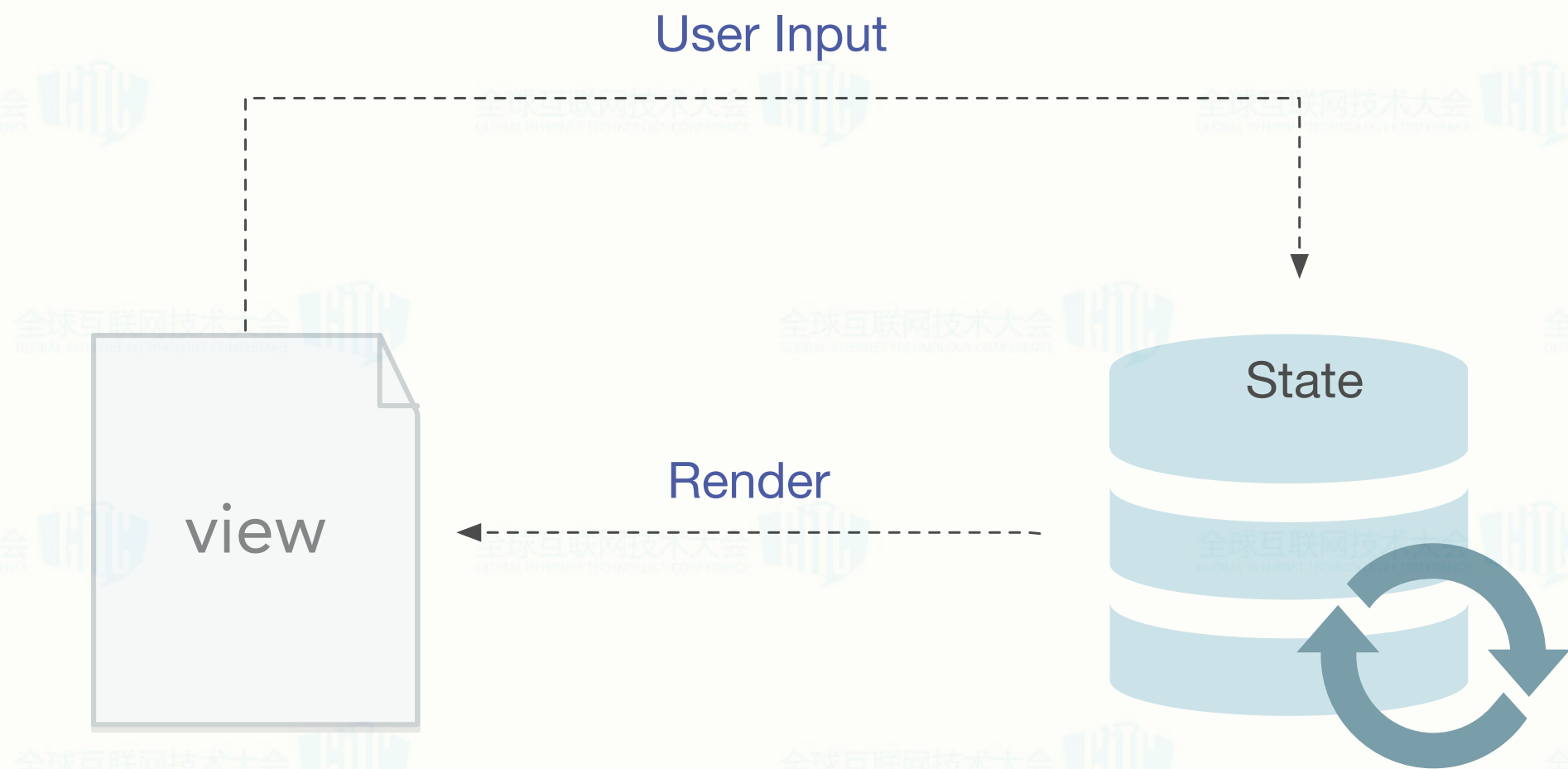
技术选型



新时代的前端，拥抱MVVM框架




通过双向数据绑定把 View 层和 Model 层连接起来





Vue.js

符合厂内前端的技术盏



elemefe

Shanghai
China

[Tweet your ranking](#)

[Refresh your profile](#)

Help Github Awards stay alive !

We're looking for a sponsor :

vue ranking

Shanghai	1 / 32 🏆
China	1 / 208 🏆
Worldwide	1 / 1 134 🏆
Repos :	9 📁
Stars :	7879 ★

javascript ranking

Shanghai	10 / 3 017 🏆
China	75 / 18 139 🏆
Worldwide	1 882 / 689 668 🏆
Repos :	27 📁
Stars :	1522 ★

*.vue 向 web 组件看齐的单文件组件



支持声明式渲染

自带组件系统

支持样式隔离



编程规范

JavaScript 的世界进步的太快

面向未来开发是项目保持生命力的必要条件

面向标准的编程方式

ECMAScript 6 + CSS4

BABEL (ES2015)

```
import Clickoutside from 'element-ui/src/utils/clickoutside';
import Emitter from 'element-ui/src/mixins/emitter';

export default {
  render(h) {
    let { hide, splitButton, type, size } = this;

    const handleClick = _ => {
      this.$emit('click');
    };

    const triggerElm = !splitButton
      ? this.$slots.default
      : (<el-button-group>
        <el-button type={type} size={size} nativeOn-click={handleClick}
          {this.$slots.default}
        </el-button>
        <el-button ref="trigger" type={type} size={size} class="el-dro
          <i class="el-dropdown__icon el-icon-caret-bottom"></i>
        </el-button>
        </el-button-group>);

    return (
      <div class="el-dropdown" v-clickoutside={hide}>
        {triggerElm}
        {this.$slots.dropdown}
      </div>
    );
  }
};
```



Postcss

BEM风格

模块化

CSS 4 选择器

Sass 语法

```
@charset "UTF-8";
@import "../common/var.css";
@import "vue-popup/lib/popup.css";

@component-namespace el {

  @b dialog {
    position: absolute;
    left: 50%;
    transform: translateX(-50%);
    background: #fff;
    border-radius: var(--border-radius-small);
    box-shadow: var(--dialog-box-shadow);
    box-sizing: border-box;

    @modifier tiny {
      width: var(--dialog-tiny-width);
    }

    @modifier small {
      width: var(--dialog-small-width);
    }

    @modifier large {
      width: var(--dialog-large-width);
    }
  }
}
```




项目管理与打包构建



如何管理大型的多组件项目？



	主仓库	独立仓库	
代码冗余度	低	高	组件公共模块无法复用，相互依赖造成组件重复打包。 公共模块重复打包
可维护性	高	低	每个组件都需要单独维护和打包，同时还要维护组件库项目的各依赖的版本号。
文档	一份文档	两份文档	独立仓库下的组件需要各自去做组件展示，不利于主仓库文档网站的生成

组件代码与主仓库共同维护管理

.	
build	# 打包配置与构建脚本
bin	# 打包前的预处理脚本
config.js	# 公用打包配置
cooking.common.js	# element 整体打包配置文件
cooking.component.js	# 组件独立打包的配置文件
cooking.demo.js	# 本地开发的与文档网站的打包的配置文件
...	
examples	# 网站与文档
lib	# 输出目录
packages	# 组件包和样式包
alert	
autocomplete	
badge	
breadcrumb	
breadcrumb-item	
button	
theme-default	# 独立的样式包
...	
src	
index.js	# 入口文件
locale	# 国际化模块
mixins	# 组件公用mixins
utils	# 通用工具、指令、类
test	# 单元测试

```
examples
├── app.vue
├── assets
├── components
├── docs
│   ├── en-US
│   └── zh-CN
│       ├── alert.md
│       ├── badge.md
│       ├── breadcrumb.md
│       ├── button.md
│       ├── card.md
│       ├── checkbox.md
│       ├── color.md
│       ├── custom-theme.md
│       ├── date-picker.md
│       ├── datetime-picker.md
│       ├── dialog.md
│       ├── dropdown.md
│       ├── form.md
│       └── ...
├── entry.js
└── ...
```

文档与网站统一管理



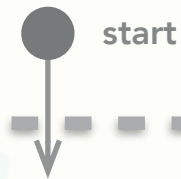
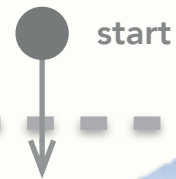
Cooking

基于**webpack**的二次封装

- 提供更简单的配置项
- 提供的脚手架功能能快速搭建基于 Vue 或 React 的项目

entry.js

文档网站入口文件



packaging in webpack

babel-loader

postcss-loader

markdown-it-loader

output

element-common.js
index.js

docs site

组件和主题的独立发布

Lerna



Lerna

A tool for managing JavaScript projects with multiple packages.

npm: v1.3.2 | travis: passing | build: passing | slack: 6/106

<https://github.com/lerna/lerna>

```
→ element git:(master) ✕ lerna publish
Lerna v2.0.0-beta.18
Independent Versioning Mode
Checking for updated packages...
? Select a new version for el-alert (currently 0.0.0) (Use arrow keys)
> Patch (0.0.1)
  Minor (0.1.0)
  Major (1.0.0)
  Custom
```



样式管理与主题定制

避免选择器嵌套，保持简洁的层级

采用 BEM 的方式管理类名

```
▼<label class="el-checkbox">
  ▼<span class="el-checkbox__input">
    ▶<span class="el-checkbox__inner is-checked">...</span>
      <input type="checkbox" class="el-checkbox__original" value>
    </span>
  ▼<span class="el-checkbox__label">
    "备选项"
    <!-->
  </span>
</label>
```

命名空间 组件名 元素 状态

```
.el-checkbox__input {
  white-space: nowrap;
  cursor: pointer;
  outline: none;
  display: inline-block;
  line-height: 1;
  position: relative;
  vertical-align: middle;
}
```


表现层的属性值全部用变量

```
:root {  
  
  /* Transition */  
  --fade-transition: opacity 300ms cubic-bezier(0.23, 1, 0.32, 1);  
  --fade-linear-transition: opacity 200ms linear;  
  --md-fade-transition: transform 300ms cubic-bezier(0.23, 1, 0.32, 1) 100ms,  
  --border-transition-base: border-color .2s cubic-bezier(.645,.045,.355,1);  
  --color-transition-base: color .2s cubic-bezier(.645,.045,.355,1);  
  
  /* Colors */  
  --color-primary: #20a0ff;  
  --color-success: #13ce66;  
  --color-warning: #f7ba2a;  
  --color-danger: #ff4949;  
  --color-info: #50BFFF;  
  --color-blue: #2e90fe;  
  --color-blue-light: #5da9ff;  
  --color-blue-lighter: rgba(var(--color-blue), 0.12);  
  --color-white: #fff;  
  --color-black: #000;  
  --color-grey: #C0CCDA;  
  
  /* Link */  
  --link-color: #475669;  
  --link-hover-color: var(--color-primary);  
  
  /* Border
```

基于 Postcss 的CSS处理器

Postcss-salad

(<https://github.com/ElemeFE/postcss-salad>)

足够简单的主题定制方式

<https://github.com/ElementUI/element-theme>

引入默认
的主题包

替换变量并编译

et --watch [--config variable file path] [--output path]

自动生成
变量文件

定制化的主题

工具链

构建



webpack
(cooking)

模块管理



Lerna

测试



Karma

文档



Markdown
(markdown-it)



babel
(ES2015)



PostCSS
(postcss-salad)



Mocha



Travis CI



Vue.js



FAQ