CLGG前端开发指南

目前，CLGG前端开发采用的主要技术和工具：jQuery, CSS, BackboneJS, NodeJS; 前两者不用多说，BackboneJS, 以及NodeJS是什么，为什么使用，以及性能等关切的问题会在本文中一一做出回答。

# 背景知识

## BackboneJS是什么

[Backbone](http://baike.baidu.com/view/342697.htm) 为复杂Javascript应用程序提供模型(models)、集合(collections)、视图(views)的结构。其中模型用于绑定键值数据和自定义事件；集合附有可枚举函数的丰富API； 视图可以声明事件处理函数，并通过RESTful JSON接口连接到应用程序。

参考：百度 或 http://backbonejs.org/

## BackboneJS优势

数据和界面分离。

将事件的绑定很好的剥离出来，便于管理和迭代。

程序的模块化更加清晰、明了。

## NodeJS是什么

参考<http://www.ibm.com/developerworks/cn/opensource/os-nodejs/>

## NodeJS性能

国外针对NodeJS的性能测试：

<http://blog.shinetech.com/2013/10/22/performance-comparison-between-node-js-and-java-ee/>

<http://www.nodejs-news.com/nodejs-tech/nodejs-wheat-benchmark/>

<http://www.ziggytech.net/technology/web-development/experiences-with-node-js-porting-a-restful-service-written-in-java/>

## NodeJS缺点

一位大神对NodeJS的离别感言：

<https://medium.com/code-adventures/farewell-node-js-4ba9e7f3e52b>

重复回调，

* you may get duplicate callbacks
* you may not get a callback at all (lost in limbo)
* you may get out-of-band errors
* emitters may get multiple “error” events
* missing “error” events sends everything to hell
* often unsure what requires “error” handlers
* “error” handlers are very verbose
* callbacks suck

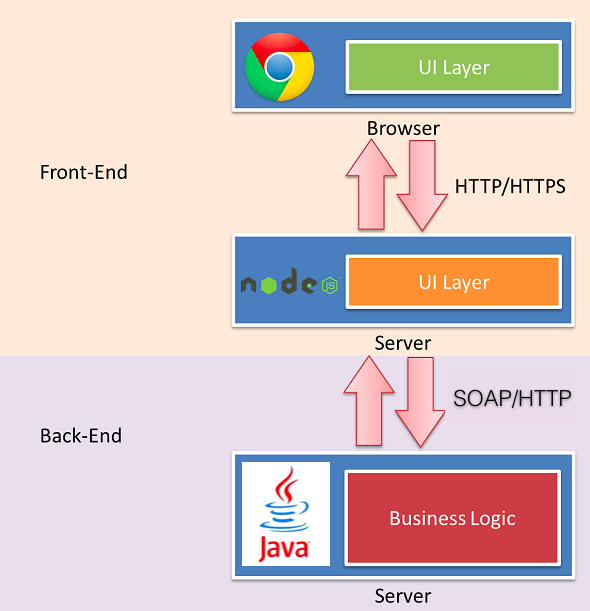
虽然有如此缺点，但用NodeJS搭建web站点依然是个不错的选择。

# 开发工具

WebStorm

# 技术选型

参考<https://github.com/lifesinger/lifesinger.github.com/issues/185>



采用基于NodeJS的Web应用框架: Express.模版引擎采用ejs 或Jade，两者可同时使用。

熟悉传统HTML编写的开发人员可选用ejs, jade是面向后台开发员的模板语言。

<http://expressjs.com/>

# 第三方JS库

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 版本 | 功能 | URL |
| jQuery | V1.7.2 | jQuery | <http://jquery.com> |
| Highcharts | v3.0.9 | 图表 | <http://www.highcharts.com/> |
| FileAPI | v2.0.2 | 文件操作：多文件、拖放式和分块的文件上传，图片裁剪、 调整大小和自动定位 | <https://github.com/mailru/FileAPI> |
| jQuery.validate | v1.10.0 | Jquery校验插件 | <http://bassistance.de/jquery-plugins/jquery-plugin-validation/> |
| easyui | latest | easyui | <http://www.jeasyui.com> |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# 规范

AMD规范：浏览器端, CommonJS规范：服务端

# Hello World

以用户登录为例(如下图，未考虑异常流程)说明：



## 目录结构

examples/

|-- node-modules 存放 NodeJS 项目的依赖包

|-- controllers 路由控制器目录

|-- fonts 验证码字体目录

|-- models 模型目录

|-- static 存放前端资源文件(js,css,image,html等)

| |-- assets 静态资源文件

| | |-- css 样式文件

| | |-- fonts web字体

| | |-- images 图片

| | |-- js 自定义JS

| | |-- lib 第三方JS库

| | `--vendor JS库(样式、JS、图片)

| |-- login 登录DEMO

| | |-- collections Backbone集合目录

| | |-- models Backbone模型目录

| | | `--credential.js

| | |-- routes Backbone路由目录

| | | `--login.js

| | |-- templates Backbone 模版目录

| | | |-- login.html

| | | `--verifycode.html

| | |-- views Backbone 视图目录

| | | |-- login.js

| | | `--verifycode.js

| | `--main.js 登录入口文件

| |-- config.js

| |-- init.js

| `--util.js

|-- views 视图/模版目录，存放模版/HTML文件

|-- util 工具包

|-- views 视图文件目录(模版,Server端)

| `--login.html 登录页

|-- uploads 上传文件目录

|-- config.js 配置文件

|-- init.js 初始化文件

|-- package.json 项目依赖包配置

`-- server.js 项目入口文件

## 源码

附件

## 依赖

|  |  |  |
| --- | --- | --- |
| 名称 | 版本 | 地址 |
| NodeJS | latest | http://nodejs.org/ |
| GraphicsMagic | latest | http://www.graphicsmagick.org/ |
|  |  |  |