开发环境使用手册

作者：hahaboy 版本日期：v1.1 | 2015-04-28

## 项目

目前主要分为以下项目：

lejs：PC站JS项目

ms：M站JS项目

diy：diy专题JS项目，包括主站和M站的专题

coop：移到端合作JS项目

koala：PC站CSS项目，使用模块化

letv：PC站老的CSS项目，未使用模块化

mcss：M站CSS项目，使用模块化

LMC：乐视组件库，不能独立打包上线

air：偏底层组件库，不能独立打包上线

comp：业务层组件库，不能独立打包上线

lib：公共库包，不能独立打包上线

vjs：播放器项目，不能独立打包上线

html：页面构建项目，使用模块化，不需要上线

player：废弃项目

old\_conf：盘古时期的JS配置

## 项目配置

项目配置文件：\_\_config.js

测试机配置文件：\_\_tester\_config.js

**1、配置CSS项目**

由于之前CSS的开发工具mokjs放在webpage下，现在全部统一使用website下的mokjs，所以需要简单调整一下配置。

如果你的webpage与website在同一个目录下，不需要进行任何修改。如果不在同一个目录，那么请修改 \_\_config.js 里的第6行指向你的webpage目录。



可以使用相对路径，也可以使用绝对路径。

**2、配置JS项目**

主要指配置指向分支或主干。例如主干开发lejs项目时，配置如下：



使用分支pad\_play开发lejs时，路径配置如下：



其他配置参数请勿修改。

## 本地开发调试

步骤：

**1、绑定host**

JS项目绑定：127.0.0.1 js.letvcdn.com

CSS项目绑定：127.0.0.1 css.letvcdn.com

**2、启动开发服务器**

Windows系统双击mokjs目录下的mok.cmd

Mac及Linux系统进入mokjs目录，执行命令node mok

**3、进入项目编写代码即可**

请注意查看你是在主干还是分支下开发。

如果需要移动端连接PC调试：可绑定代理，IP为开发机IP，端口为2001。可同时调试JS和CSS项目。如果单独调试JS或CSS，请修改 \_\_config.js ，注释掉JS或CSS配置：



开发过程中，修改上面的**代理配置proxy\_conf需要重启**服务器，修改其他配置**不需要重启**开发服务器。

## 部署测试机

修改mokjs目录下的 build-项目名.cmd 文件，将里面的 t=29 的29修改为你的测试机id，测试机id与测试机的对应关系请查看 \_\_tester\_config.js 文件。

例如你要测试的项目是mcss，你的测试机id是126，那么修改build-mcss.cmd里的 t=29 为 t=126 ，然后双击build-mcss.cmd即可将项目部署到测试机126。

部署到测试机后，请绑测试机IP自我验证通过后再交由测试部门测试。

除了直接运行cmd文件部署测试外，还可以从命令行部署测试机：从命令行进入mokjs目录，执行命令 node ./letv/build 项目名 t=测试机id 。例如：node ./letv/build coop t=126 ，会把coop项目部署到126上。

## 项目上线

CSS项目上线，直接双击release-项目名.cmd 文件即可获取上线地址。

JS项目上线，需要到发布机上上线，目前的发布机有：

10.154.252.73 不需要登录堡垒机

10.182.192.74 需要登录堡垒机，发布速度快

10.182.192.75 需要登录堡垒机，发布速度快

发布命令是：dist 项目名

例如执行 dist ms 命令即可上线ms项目，获取上线文件的地址。

## JS项目开发的改变

1、废除LTK.require、LTK.use函数，这两个函数都用require代替即可。

2、require一个模块的时候，就会自动引入模块所在的文件，不需要再通过\_\_req函数引入文件。

3、不要再使用 define(‘-module’… 定义一个立即运行的模块，使用require引用模块的时候就会执行模块了，要使用require取代\_\_req.

4、除特殊情况，不要再使用 define(‘module’… 定义模块，模块名要自动使用文件路径。使用define(‘module’… 定义的模块，在引用时必须用\_\_req引入文件！

5、推荐引用模块时使用“/”代替“.”，例如使用 require('bz/www\_play/events') 而不使用 require('bz.www\_play.events') 。前者是主流模块化规范例如node都使用的方式。

6、如果A模块和B模块在同一个目录，A引用B时，可使用相对路径引用 require('./B') ；如果B模块在上一级目录，则引用为 require('../B') 。

7、使用\_\_req函数跨项目引用文件一定要加上项目名，例如：\_\_req('vjs::index.js');