项目脚手架的搭建

一：npm init初始化目的是输入参数，生成模块化描述文件package.json

1. 进入MINGW64 ~/mmall/doc/mmall-fe (mmall\_v1.0) （分支根目录）
2. 输入npm init回车 出现以package name:(mmall-fe)字段结尾的一段内容，
3. 一次回车出现version: (1.0.0)，二次回车出现description:。在description:后输入mmall前端代码

（三次回车）entry point:(index.js)

（四次回车）test commond:

（五次回车）git repository: (git@gitee.com:yqfhappymmall/mmall-fe.git)

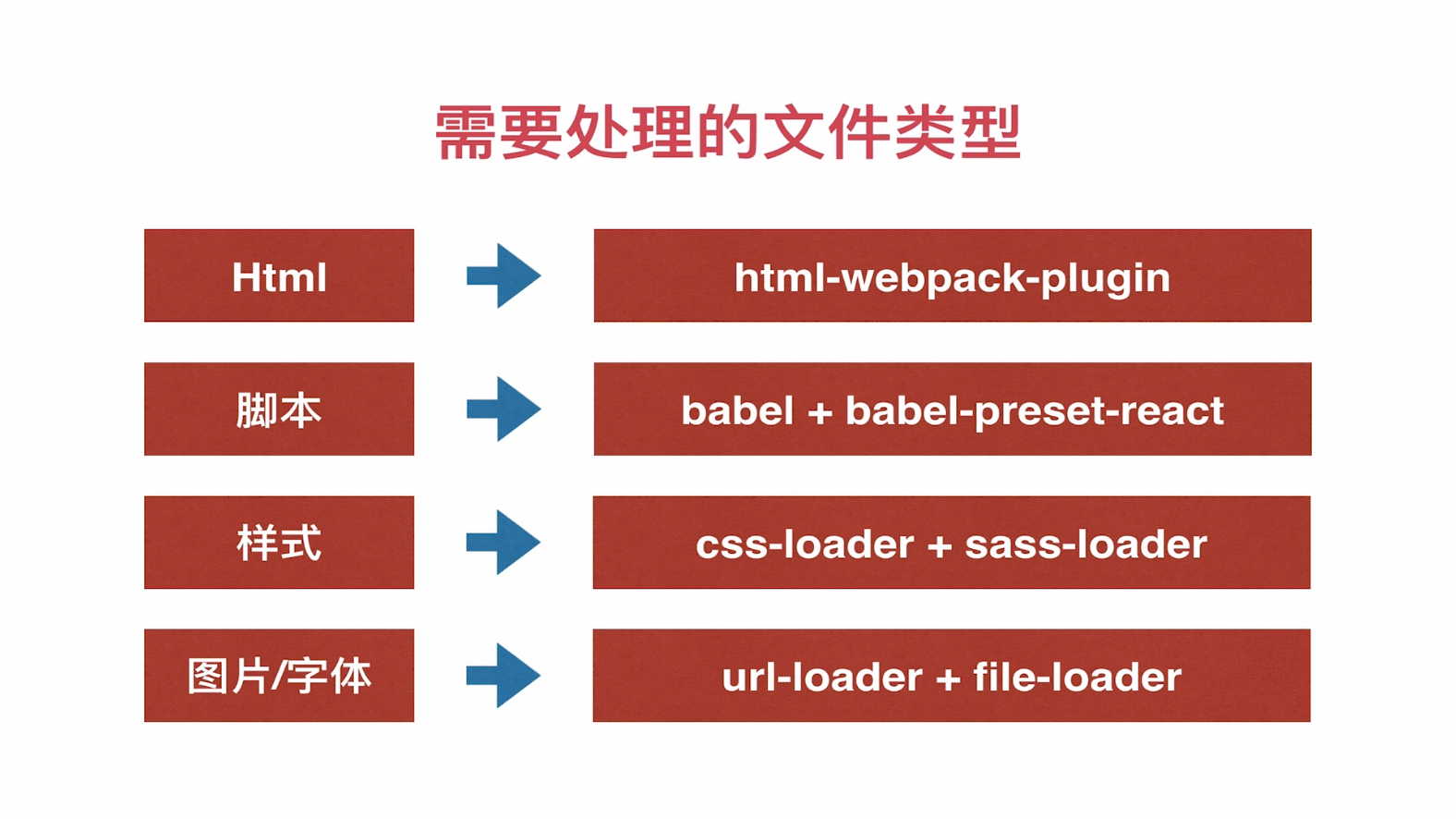
（六次回车）keywords:

（七次回车）author: Rosen

（八次回车）license: (ISC)

1. 再次回车在Is this ok? (yes)的后面输入yes 后再次回车
2. 先后可以输入ls,cat package.json进行查看内容

二：webpack的安装





Webpack常用模块

html-webpack-plugin打包生成单独的html文件,把对应的脚本和样式插入合适的位置，防止在更新的时候有浏览器缓存

extract-text-webpack-plugin将样式打包成单独文件

CommonsChunkPlugin提出通用模块

webpack-dev-server为webpack项目提供web服务，监听文件的改动自动刷新页面，路由的拦截和代理

babel:对ES6的语法进行处理，使其变为ES5的语法

css-loader和sass-loader对样式的处理

url-loader和file-loader是对图片和字体的处理

都在在这个目录下MINGW64 ~/mmall/doc/mmall-fe (mmall\_v1.0)

1. npm install webpack -g回车进行全局安装 需要权限时加sudo 例如：sudo npm install webpack -g 例如：出现这个+ webpack@3.10.0表示安装成功
2. 在项目里进行安装npm install webpack@1.15.0 --save-dev 输入webpack -v可以查看webpack的版本 输入ls出现node\_modules/ package.json package-lock.json 可以输cd node\_modules/进去输入ls进行查看
3. 将mmall-fe文件就可以拖进sublime中，接下来就可以进行简单的配置 手动建文件夹如下 案例：



1. 在MINGW64 ~/mmall/doc/mmall-fe (mmall\_v1.0)输入 webpack ./src/page/index/index.js ./dist/app.js回车，那么这样这两个js文 件就被打包进入dist文件夹下（单入口的配置并不适用）

* Webpack.config.js的高级常用配置

1. 在mmall-fe的根目录下手动右键建一个webpack.config.js文件，填写 webpack.config.js一系列内容
2. 当一切结束后，在MINGW64 ~/mmall/doc/mmall-fe (mmall\_v1.0) $中输入webpack进行打包

* Jquery的引入是放在footer.html中引入的。如果想本地引入就输入npm install jquery --save

1. <script type=”text/javascript” src=”https://cdn.bootcss.com/jquery/1.11.3/jquery.min.js”></script>

* webpack提取公共模块 利用插件 CommonsChunkPlugin插件
* 项目目录下doc/mmall-fe (mmall\_v1.0) $这块需要安装的软件

1. npm install css-loader style-loader --save-dev （处理样式的）
2. npm install extract-text-webpack-plugin@1.0.1 --save-dev (用于css单独打包处理，阻止CSS被打包进入JS中混淆)
3. npm install html-webpack-plugin --save-dev （对html模块化的处理，将html页面和JS建立联系）
4. npm install html-loader --save-dev 来处理这一块的<%= require('html-loader!./layout/html-head.html') %>
5. npm install url-loader --save-dev 用来处理图片及字体好方便CSS文件引用
6. npm install file-loader@0.11 --save-dev 注意第5个命令依赖第6个插件
7. npm install webpack-dev-server@1.16.5 --save-dev （这里的版本号暂时先不要改成其他版本）
8. npm install webpack-dev-server@1.16.5 -g 进行全局安装服务器，用于边开发边浏览

* 执行命令

1. npm install hogan.js --save 用来渲染html模板的
2. npm install font-awesome --save 下载文字图标
3. MINGW64 ~/mmall/doc/mmall-fe (mmall\_v1.0) $ webpack-dev-server 测试启动服务器 得到重要的服务器信息 <http://localhost:8080/webpack-dev-server/>
4. 配置端口8088之后的启动方式是webpack-dev-server --inline --port 8088 回车之后获得localhost:8088

* 在package.json中配置短命令

"scripts": { //缩短启动服务服务器的命令和配置打包的命令

"dev": "WEBPACK\_ENV=dev webpack-dev-server --inline --port 8088", 配置启动服务器的命令

"dev\_win": "set WEBPACK\_ENV=dev && webpack-dev-server --inline --port 8088",

"dist": "WEBPACK\_ENV=online webpack -p", 配置打包的命令

"dist\_win": "set WEBPACK\_ENV=online && webpack -p"

},

之后在Git Bash中启动服务器就可以用自己的命令npm run dev\_win （在Mac下使用命令npm run dev）

* 依次输入ls ,git st,

1.git add .把文件先添加进来

2.git ca “initial” 提交到本地并加上注释

3.git ps 把文件都提交到远程上去

4.（第一次提交远程的时候）输入提示的git push --set-upstream origin mmall\_v1.0进行分支提交

5.再次执行git st可以查看已经没有改变了，可以看到这个nothing to commit, working tree clean

6.git merge origin master 因为第一次提交的时候master分支是空的，多人开发的时候进行合并

7.git tag tag-dev-initial （为了方便以后我们找分支方便，需要加个tag）

8.git push origin tag-dev-initial 把tag打到远程上。

十二.服务器的启动

1.npm run dev\_win （window系统）

2.npm run dev （Mac系统）

3.后期如果有webpack.config.js改动的话 先在Git Bash中执行webpack，在执行npm run dev\_win

十三．代码的提交

1.先进入我们的项目根目录（有分支的那一层）

2.输入git pull 养成良好的拉代码习惯

3.输入git st 回车 可以看到我们这次改动的文件

4.输入git add . 回车 把我们改动的文件追踪上

5.输入git ca “common部分” 回车 （这里是提交到本地）

6.输入git ps 提交到远程

7.输入git tag tag-dev-common 打一个common的tag标签

8.输入git push origin tag-dev-common 就算提交完成了

十四．注意每次重启服务之前一定先干掉之前的服务器

十五．代理服务器



注意：当使用fiddler代理时，当无参数时使用EXACT,当有参数时使用REGEX

1.webpack安装：

首先要进行全局安装npm install webpack -g，然后安装项目依赖 npm install webpack@1.15.0 --save-dev

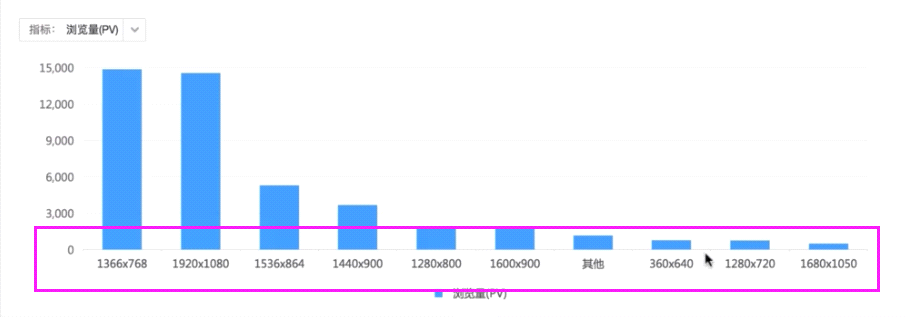
1. 查看webpack的版本号用webpack -v进行查看

Webpack.config.js知识点：

1. entry:js的入口文件
2. Externals:外部依赖的申明
3. Output：目标文件的内容
4. Resolve：配置别名
5. Module:各种文件，各种loader
6. Plugins:插件

知识点：

项目中的适配参考如下



1. npm是随同node.js一起安装的包管理工具
2. 检查npm的版本 使用npm -v进行查看。
3. 安装依赖包npm install xxx@v.v.v
4. 卸载依赖包npm uninstall xxx@v.v.v
5. 全局 需加参数-g
6. webpack是模块化管理工具
7. cat package.json 查看package.json的内容
8. 回去上一个文件操作顺序，cd . cd ..
9. 执行webpack打包的命令 直接在$后输入webpack后回车即可
10. mv ../test-img.jpg ./src/image/ 把一张图片放到新建的文件夹下
11. vim .gitignore 更改.gitignore的配置
12. git add .把文件先添加进来
13. <http://www.datatables.club/example/basic_init/alt_pagination.html> （分页功能）
14. 解决行内块元素之间的空白间隙问题：给共同父级加font-size:0px;
15. webpack用异步的形式来打包，常用于调试代码。webpack-p用做线上发布时的打包，会把所有文件做最小化压缩。webpack --watch用于接听文件的改变，而自动编译。一般用于开发过程
16. -save-dev表示你开发时依赖的包裹,--save表示发布时依赖的包裹

注意以下是三个路径特殊的引用：

css中引用图片的路径特殊采用一下方法

background: url(../../image/icon/loading.gif) no-repeat;

require('node\_modules/font-awesome/css/font-awesome.min.css');

Index.html中的图片引用格式：

<img class="floor-img" src="<%= require('../image/floor/floor1-1.jpg') %>" alt="双开门冰箱" />

index.string中图片的引用格式：

<img class="banner-img" src="../../image/banner/banner2.jpg" />

（原因：html文件是ejs模板，需要require图片路径才行，string里的走的html-loader，直接引用，loader就会帮着处理了

）

上线过程视频课

12-2线上服务器环境搭建

首先我们要有 一台服务器；服务器上安装我们需要的环境

1. 首先要登录我们的服务器，打开git Bash根目录（注意：不是项目根目录），在这里mmall@macbook ~ $ 后输入ssh +机器地址，例如输入：ssh rosen@112.126.73.177 回车进入这台机器
2. 输入vim .bash\_profile回车
3. 对bash\_profile文件进行配置，更多配置参考视频

alias gommallnew=’ssh rosen@112.126.73.177’ 然后按esc :wq保存

1. 强制执行这个文件 source .bash\_profile
2. mmall@macbook ~ $后输入gommallnew
3. 进入这台机器后在[rosen@iZ2z2fic7kr4s79qs6azuZ ~] $后输入 ifconfig回车查看这台机器是什么地址。
4. 在[rosen@iZ2z2fic7kr4s79qs6azuZ ~] $后输入ifconfig | grep 112回车。可以看到inet addr：112.126.73.177 Bcast：112.126.75.255 Mask：255.255.252.0
5. 分别输入node -v, ruby -v,git -v检查一下这台机器是否安装了node.js,ruby,git这三款软件
6. 在[rosen@iZ2z2fic7kr4s79qs6azuZ ~] $先后输入cd回车ls回车进入到用户目录下
7. 输入wget <https://nodejs.org/download/release/v4.4.7/node-v4.4.7-linux-x64.tar.gz>回车下载安装输入ls可以进行查看
8. 进行解压在[rosen@iZ2z2fic7kr4s79qs6azuZ ~] $后输入tar -xzvf node-v4.4.7-linux-x64.tar.gz回车，输入ls回车可以看到node-v4.4.7-linux-x64 node-v4.4.7-linux-x64.tar.gz
9. 输入cd node-v4.4.7-linux-x64，再输入ls，输入cd bin回车。再输入ls回车可以看到node npm，输入./node -v可以查看node的版本
10. 把node配置成全局都可以使用的，输入cd /usr/local回车再次输入ls回车
11. 输入mv ~/node-v4.4.7-linux-x64 ./如果出现错误就输入sudu mv ~/node-v4.4.7-linux-x64 ./回车
12. 输入管理员密码再次输入ls
13. 输入pwd查看路径，输入vim /etc/pro按下TAB键，输入vim /etc/profile回车如果不成功则需要加sudu权限。例如输入sudo vim /etc/profile回车，如果需要就输入系统密码
14. 通过输入/path找到类似于export CATALINA\_HOME=/developer/apache-tomcat-7.0.73，然后输入i进行编辑，输入export NODE\_HOME=/usr/local/node-v4.4.7-linux-x64，然后在export PATH=$PATH用:进行区分补上:$NODE\_HOME/bin然后退出保存
15. 然后输入source /etc/profile回车再次执行node -v就可以看到v4.4.7
16. 然后在[rosen@iZ2z2fic7kr4s79qs6azuZ local]$后输入npm install webpack -g --registry=https://registry.npm.taobao.org回车稍等几分钟，执行webpack -v可以看到webpack -v安装成功。
17. ruby的安装，输入sudo yum install ruby回车，输入y稍等一会，输入ruby -v检查ruby是否安装成功
18. sass的安装，输入sudo su -c “gem install sass”回车，输入sudo yum install rubygems回车，稍等输入y，稍等一会，输入gem -v检查gem是否安装成功，再次执行sudo su -c “gem install sass”
19. 检查后端git是否安装成功，如果没有安装，则需要自己安装

该学到12-3：发布流程及发布脚本的开发

* 首先输入gommallnew登录上机器， 如mmall@macbook ~ $ gommallnew，如需输入密码，则输入密码。

1. 输入ls可以看到node-v4.4.7-linux-x64.tar.gz，依次输入cd /回车，再输入ls，回车，再次输入ls -al，再输入cd developer/，再次输入ls，ls -al可以看到git -repository，这里需要提高自己的权限，在[rosen@iZ2ze2fic7kr4s79qs6azuZ developer]$后输入sudo usermod -g root rosen回车，再依次输入ls, ls -al

在这一层目录中，更改developter权限。[rosen@iZ2ze2fic7kr4s79qs6azuZ /]$ sudo chmod -R 775 developer/回车，再依次输入cd dev回车，输入cd ../developer/，输入ls，再次输入cd git-repository/回车再次输入ls可以看到mmall\_learning

1. 找到git仓库，找到admin-fe点击，点击“克隆，下载”，复制ssh中的地址，在[rosen@iZ2ze2fic7kr4s79qs6azuZ git-repository]$后输入git clone +刚刚复制的地址（如：git clone git@git.oschina.net:happymmall/admin-fe.git）回车，中途需要输入yes

但是并没有进行ssk的配置，现在就进行配置ssk：

3.在[rosen@iZ2ze2fic7kr4s79qs6azuZ git-repository]$后输入cat ~/.ssh/id按了几下tab发现出不来。原因是Rosen这个账户还没有生成公钥。

给rosen账户生成公钥。回到根目录[rosen@iZ2ze2fic7kr4s79qs6azuZ ~]$后输入ssh-keygen -t rsa -C ‘gaorunsheng1@126.com’一直回车，直到出现[rosen@iZ2ze2fic7kr4s79qs6azuZ ~]$，在后面输入cat ~/.ssh/id\_rsa.pub回车。将ssh-rsa AAAAB3NzaC1yc2EAAAABIw\*\*\*\*\*\*\*gaorunsheng1@126.com复制下来。找到码云的管理——项目公钥的管理——添加部署公钥——添加标题，将复制的ssh-rsa AAAAB3NzaC1yc2EAAAABIw\*\*\*\*\*\*\*gaorunsheng1@126.com粘贴进“在这里粘贴公钥”——添加。

4.[rosen@iZ2ze2fic7kr4s79qs6azuZ ~]$后输入cd /developter/git-repository/回车，之后输入ls，可以见到mmall\_learning

5.在[rosen@iZ2ze2fic7kr4s79qs6azuZ git-repository]$后输入git clone +（项目的ssh地址）回车直到项目被下载下来。输入ls，可以见到admin-fe mmall\_learning两个文件夹。

6.在[rosen@iZ2ze2fic7kr4s79qs6azuZ git-repository]$后输入cd admin-fe/回车，输入ls，可以看到favicon.ico package.json README.md src webpack.config.js

7.先回去把用户端的代码也下载下来，在[rosen@iZ2ze2fic7kr4s79qs6azuZ admin-fe]$中输入cd ..，回到[rosen@iZ2ze2fic7kr4s79qs6azuZ git-repository]$，在码云中找到mmall-fe，点击“克隆下载”，把ssh中的地址复制下来，在[rosen@iZ2ze2fic7kr4s79qs6azuZ git-repository]$后输入git clone +ssh复制的地址。发现报错，因为用户端的key还没有配置，所以需要先进性配置。

8.找到码云的管理，部署公钥管理，公钥管理，我们可以看到happymmallnew的公钥创建于：3分钟前，后面有个“启用”按钮，点击“启用”

9.在[rosen@iZ2ze2fic7kr4s79qs6azuZ git-repository]$中从新执行git clone +ssh地址，把代码拉下来了，输入ls可以看到admin-fe mmall-fe mmall\_learning三个文件夹。

10.以mmall-fe为例看下发布过程。在[rosen@iZ2ze2fic7kr4s79qs6azuZ git-repository]$中输入cd mmall-fe/回车，输入ls可以看到只有README.md这一个文件夹。说明master分支还是空的。需要进行分支的合并。

11.在码云中找到mmall-fe，点下方的Pull Requests，点击“新建Pull Request”按钮，把源分支合并到目标分支上，happymmall/mmall-fe mmall\_v1.0 到happymmall/mmall-fe master上，在创建Pull Request中输入标题和说明，填写好审查人员，测试人员，其他默认，点击“创建”但是会有问题，作为一个管理员：

在git根目录中输入cd mmall/doc/mmall-fe进入@macbook ~/mmall/doc/mmall-fe(master)$中输入ls回车可以见到README.md src，输入git merge mmall\_v1.0回车，但是会有问题。

1. 输入ls，输入vim README.md回车，进入之后将里面重复的删除，按下esc输入：wq保存，做一个提交先后输入git st，git add .，git ca “conflict”，git ps
2. 在服务器[rosen@iZ2ze2fic7kr4s79qs6azuZ mmall-fe]$中，执行git pull稍等一会就可以看到文件都被拉下来了，输入ls可以看到favicon.ico package.json README.MD src webpack.config.js文件。

13.需要进行一下初始化npm install --registory=https://registory.npm.taobao.org回车（这个等待时间比较长）

14.在等待期间我们再开一个窗口，把我们的admin做一个初始化，在mmall@macbook ~$输入gommallnew回车，如果需要输入密码，则输入密码回车。

15.在[rosen@iZ2ze2fic7kr4s79qs6azuZ ~]$中输入cd /developer/git-repository/回车，输入cd admin-fe/回车进入[rosen@iZ2ze2fic7kr4s79qs6azuZ admin-fe]$中输入ls,再拉一下代码git pull回车，再次输入npm install --registory=https://registory.npm.taobao.org回车

15.在[rosen@iZ2ze2fic7kr4s79qs6azuZ mmall-fe]$中输入ls就可以看到node\_modules，再输入npm run dist（window系统输入的是npm run dist\_win）看看能不能打包成功。

16.输入ls回车可以看到dist文件就出来了，最后一步就是把dist文件放进product中，输入cd /product/，回车输入ls，检查是否有其他文件夹，先建一个文件夹，输入mkdir frontend，出现了Permission denied说明frontend这个文件夹权限也不对。输入cd ..

16.在[rosen@iZ2ze2fic7kr4s79qs6azuZ /]$中输入ls回车，再次输入ls -al，可以发现product的权限不对，输入sudo chmod 775 -R product/回车输入密码。

17.输入cd product/回车再次输入ls，依次回车输入mkdir frontend 输入ls可见到frontend ftpfile文件。依次输入cd frontend/，ls，mkdir mmall-fe，mkdir admin-fe，ls，cp -R /developer/git-repository/mmall-fe/dist/ ./mmall-fe/

18.再依次输入ls，cd mmall-fe/，ls，cd dist/，ls可以看到css，favicon，js，resource，view到这里我们mmall-fe就完事了，接下来我们用脚本把他做下来。

19.接下来我们用脚本把他做下来。在[rosen@iZ2ze2fic7kr4s79qs6azuZ developer]$中输入vim fe-deploy.sh回车，输入i，把ssshell脚本里的内容粘贴过来，退出保存。

20.依次输入ls，./fe-deploy.sh，发现报错Permission denied，说明权限不够，输入ls -al检查权限。

21.更改权限，在[rosen@iZ2ze2fic7kr4s79qs6azuZ developer]$中输入sudo chmod 775 fe-deploy.sh，如果需要输入电脑密码，则输入电脑密码。

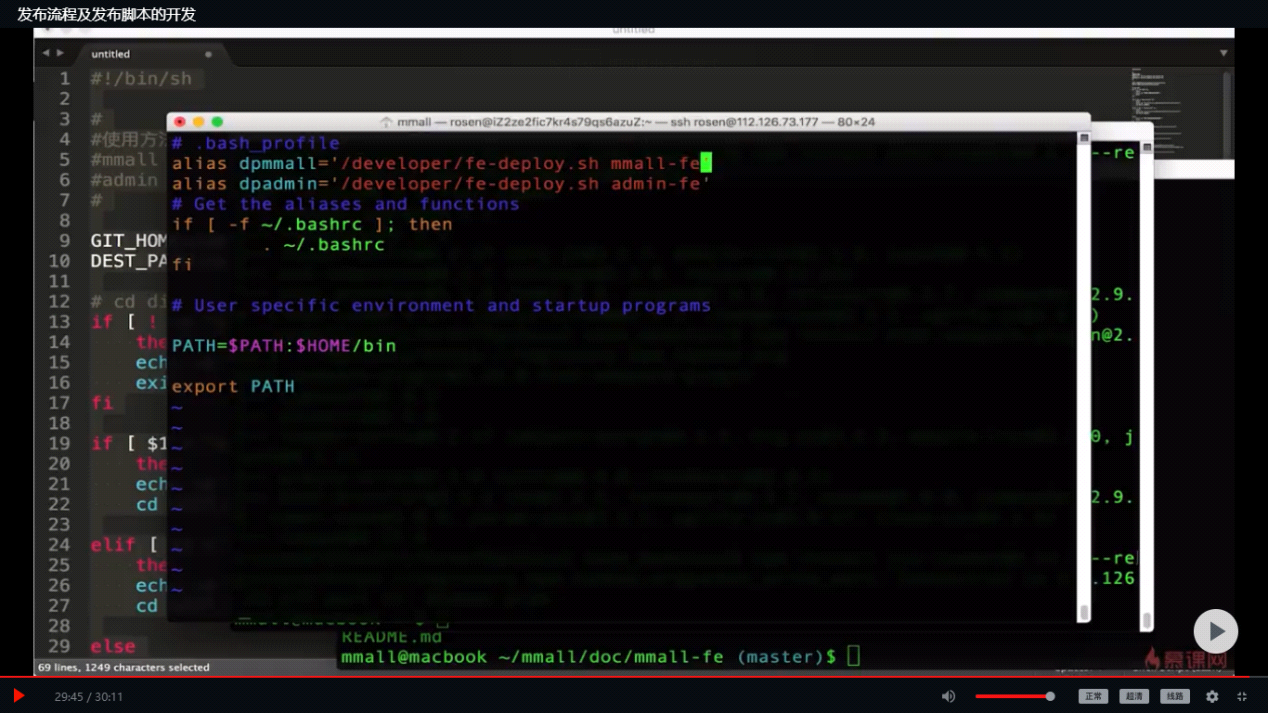
22.在[rosen@iZ2ze2fic7kr4s79qs6azuZ developer]$中，再次输入./fe-deploy.sh，再次输入./fe-deploy.sh mmall-fe，稍等一会，直到执行成功。可以依次输入ls，cd /product/frontend/mmall-fe/，ls，ls -al，date查看是否成功。

23.可以输入cd /developer/git-repository/回车，进入到[rosen@iZ2ze2fic7kr4s79qs6azuZ git-repository]$，输入ls，可以看到三个文件夹admin-fe mmall-fe mmall\_learning

24.依次输入cd admin-fe/，ls，npm run dist（或者npm run dist\_win）可以检查是否成功，输入ls，rm -rf dist/，ls，cd ..，ls，cd ..，ls，回到[rosen@iZ2ze2fic7kr4s79qs6azuZ developer]$中，执行./fe-deploy.sh admin-fe回车。

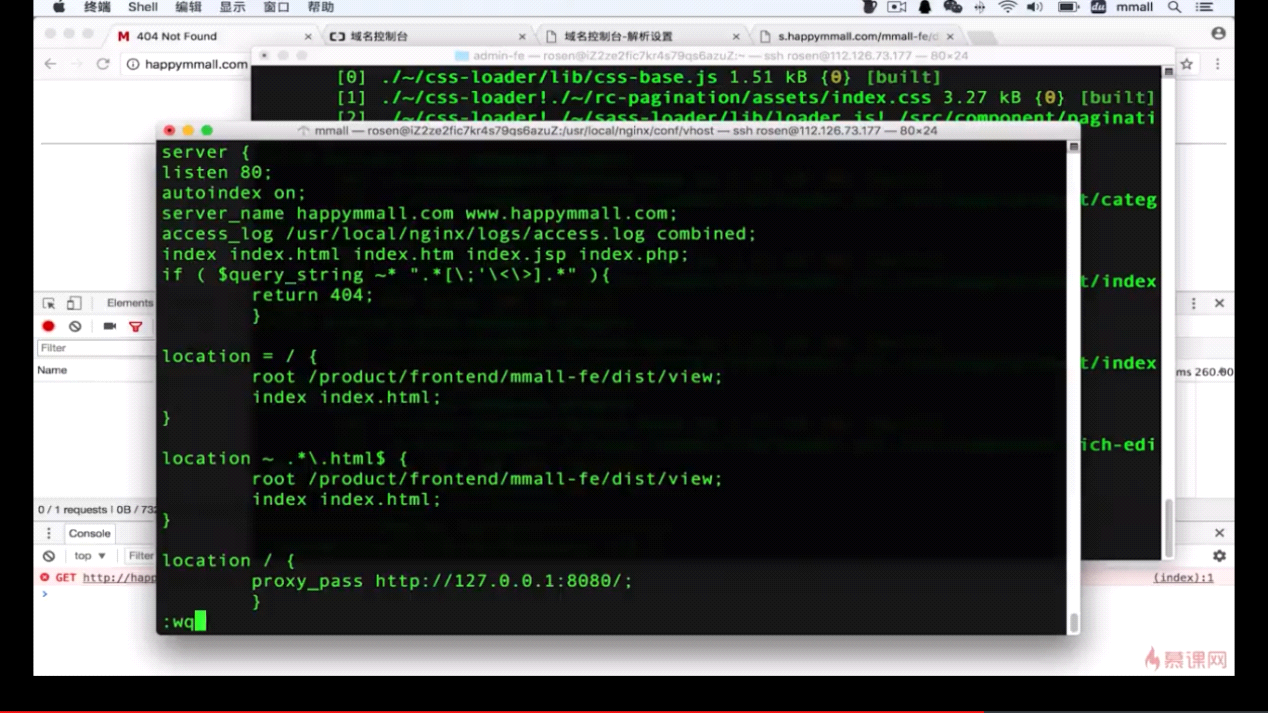
25.配置：在[rosen@iZ2ze2fic7kr4s79qs6azuZ developer]$中，依次输入ls，cd /product/frontend/admin-fe/，ls，cd直接回到根目录

26.在[rosen@iZ2ze2fic7kr4s79qs6azuZ ~]$中输入vim .bash，vim .bash\_profile对.bash\_profile进行配置如下

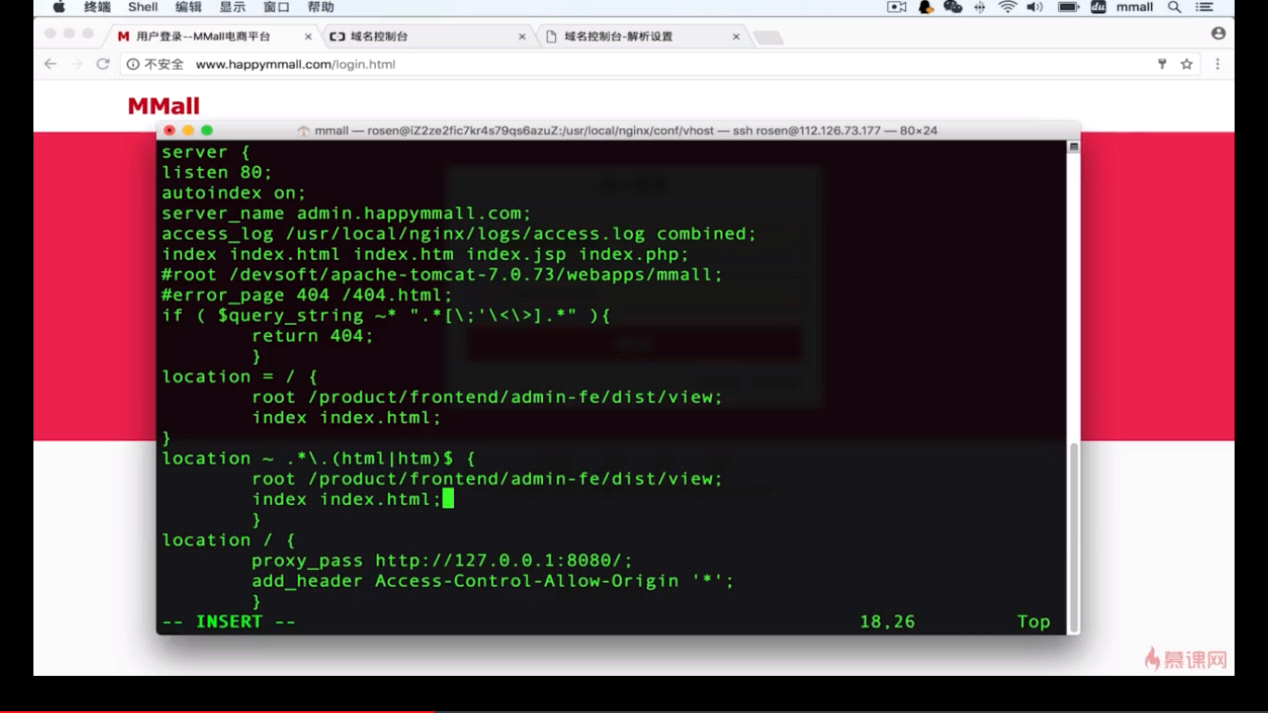
27.在[rosen@iZ2ze2fic7kr4s79qs6azuZ ~]$中依次输入source .bash\_profile，cd，dpmmall， dpadmin，

12-4 nginx配置和域名解析

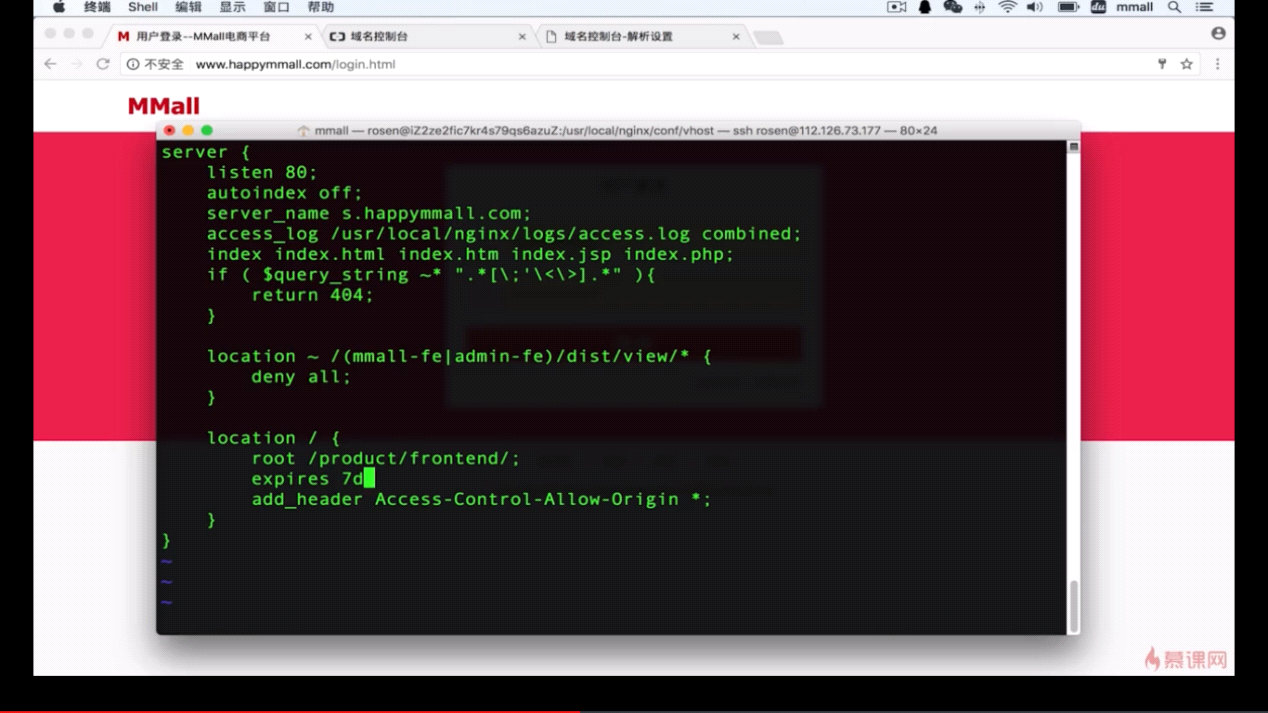
1.在mmall@macbook ~/doc $ 输入gommallnew，如需输入电脑密码，则输入电脑密码。进入服务器[rosen@iZ2ze2fic7kr4s79qs6azuZ ~]$依次输入cd /usr/local/nginx/回车，输入ls，cd conf/，ls，cd vhost/，ls可以看到admin.happymmall.com.conf img.happymmall.com.conf happymmall.com.conf s.happymmall.com.conf这四个文件夹。输入sudo vim happymmall.com.conf更改配置如下



保存

3.在[rosen@iZ2ze2fic7kr4s79qs6azuZ vhost]$中再输入sudo vim admin.happymmall.com.conf回车

（源码中add\_header Access-Control-Allow-Origin ‘\*’;前面需要加个#注释起来。如 #add\_header Access-Control-Allow-Origin ‘\*’;）保存。

4.在[rosen@iZ2ze2fic7kr4s79qs6azuZ vhost]$中输入sudo vim s.happymmall.com.conf回车

5.回到[rosen@iZ2ze2fic7kr4s79qs6azuZ nginx]$中依次输入cd sbin/，ls，sudo ./nginx -t,sudo ./nginx -s reload

6.测试一下：在Charses中，找到tools ——DNS Spoofing——勾选Enable DNS Spoofing——点击Add——四次填写Host Name和Address。（例如：HostName: happymmall.com，Address: 112.126.73.177。HostName: admin.happymmall.com，Address: 112.126.73.177。HostName: s.happymmall.com，Address: 112.126.73.177。HostName: [www.happymmall.com](http://www.happymmall.com/)，Address: 112.126.73.177。） 最后点击”OK”

7.在[rosen@iZ2ze2fic7kr4s79qs6azuZ sbin]$中依次输入ls，cd ..，ls，cd logs/，ls，tail -f access.log。多次回车刷新，

8.webpack.config.js做修改，在mmall@macbook -/mmall/doc/admin-fe (master)$中输入vim webpack.config.js

9.征得了后端的同意以后，在[rosen@iZ2ze2fic7kr4s79qs6azuZ ~]$中依次输入cd /developter/，ls，ls -al;，./deploy.sh，稍等一会，代码执行到可以见到”\*\*\*\*\*\*/ROOT.war’?”后面需要输入一个y等待执行成功。

十：域名的控制

1.

09：36

知识点：

1. 查看服务器IP。在回到[rosen@iZ2ze2fic7kr4s79qs6azuZ sbin]$中ifconfig |grep 12
2. [rosen@iZ2ze2fic7kr4s79qs6azuZ vhost]$ sudo /usr/local/nginx/sbin/nginx -s reload是什么意思？

要学习的东西：linux基本命令