概述

# 页面框架

php

包含内容日期树的创建：读取本地文件data.json创建

（频道日期树的创建通过读抓取的文件目录创建）

# 内容抓取

通过phantomjs和electorn抓取，方式都是通过linux服务器的定时任务定时执行：

# 服务器定时任务一览

\*/3 \* \* \* \* www-data cd /var/frontend/system/compare && phantomjs clip.js

**phantomjs抓取程序**

16,36,56 \* \* \* \* www-data cd /var/frontend/system/compare && sh killXvfb.sh

**杀掉残余electron抓取程序**

2-59/5 \* \* \* \* www-data cd /var/frontend/system/compare && xvfb-run -a electron channel\_fetch\_es6.js

**electron抓取程序**

0 0,9,12,21 \* \* \* www-data cd /var/frontend/system/compare/c && phantomjs editor.js

**已经去掉，因为现在目录中已经没有editor.js文件，应该是之前的抓取程序**

1 1 \* \* \* www-data cd /home/xqwei/work/compare && ./rmlog

**定时清理日志程序**

29 \* \* \* \* www-data /home/xqwei/bin/killzombie29

59 \* \* \* \* www-data /home/xqwei/bin/killzombi

**已经去掉，上边两个都是杀掉phantomjs残余进程，但是现在已经修改过phantomJs脚本，不会存在残余进程**

# 文件对比

只有内容抓取（使用phantomJs抓取时候）有对比，是抓取时候通过clip.js内部的实现的。

# 内容抓取页面

就是主页面:

首先页面通过jsonp调用<http://datacompare.qa.developer.163.com/api/date/get?begin=>获得一个日期list，然后判断请求url有没有date参数，如果没有的话，重定向至?date=xx（xx是最新日期），然后根据时间等参数通过后台获得html文件，写入到返回的html页面中。

# 频道页面

通过传递参数channel和date，来获取对应频道和时间的html内容。实际上只能获取当天的。后台会读取channle\_files文件夹里对应channel和date的文件夹，获得一个文件列表，传递到前台用来生成左侧的树。

当请求的参数没有date的时候，前台channe.js会执行一个脚本，发送一个包含begin,end,callback参数的url请求，用来获取一个包含所有时间的list，然后返回callback(list)到前台的形式，然后取出最新的时间，用这个参数再次发送上边的请求

# phantom.Js抓取程序的问题及重编写

源程序逻辑：

1. 先进行门户比对的逻辑。
2. 比对主程序结束之后，开始抓取程序，通过setTimeout异步实现

源程序的想法是通过setTimeout，让抓取程序每隔3秒开始抓取

1. 抓取到数据之后，进行style的设置：把每个dom元素的computedStyle取出来，放到style属性里，是css得一种替代。然后判断配置文件里有没有对应频道和网站的event属性，这个event属性实际上就是对这个dom的一些操作，如果有的话，执行event。
2. 最后通过一个setTimeout(100)的异步方式把抓取的html写入到文件。

问题及修改：

1. 把源程序中杂乱无章的逻辑归并到函数中进行命名和调用。
2. 一开始的每隔3秒开始抓取是没有意义的，因为phantomJs的open本来就是异步的，所以同时开始抓取没有问题，因此修改成统一setTimeout(0)开始5个网站的抓取程序
3. 关于配置中的event：配置文本中配置了大量的data-src属性替换url的src属性，因此把配置文件中的这一块逻辑统统拿掉，统一在evaluate逻辑中进行替换
4. evaluate是同步操作，没有必要setTimeout（100），因此把上面第四条改成同步。
5. 增加了日志记录函数wlog，统一调用，用来记录成功及错误日志，以及日志写入程序flush，在退出时候一口气写入日志文件。
6. 增加了phantomJs的错误回调函数和逻辑里的try catch，解决了程序因为抛出错误而崩溃的问题。
7. 在遍历配置文件的时候记录处理的总数，在每一个异步处理完成的时候进行--判断，如果归零说明结束，记录日志，exit
8. 更新所有失效的配置中的选择器
9. 解决了源程序不会在运行结束后退出成为僵尸进程的问题

一堆大坑：

1. 在刚运行的时候发现程序总是莫名其妙的报错，测了很久后发现是随机的错误，然后更新了phantomJS程序（1.9升级为2.2）后，在windows环境里没有任何问题。
2. 但是在linux环境下，这个2.2的phantomJs程序在对sohu的网站进行抓取的时候，总是随机（差不多40%几率）的出现下面这个问题：有时候open（最开始的phantomJS方法，open之后才能做其他操作）之后没有任何响应，没有报错，注册的onErrr回调函数里也没有响应，open的回调也不触发，就是停在那里。其他的网站都没有问题，就sohu有。找了好久原因没找到。。。最后解决的方法是，加一个2分钟的回调函数，通过一个flag判断sohu的open方法是不是死掉了，如果是的话，再执行一次sohu的抓取，通过retry变量控制最多4次，每两分钟调用一次sohu的抓取。通过这种方式，目前能做到100%的获取内容

# php程序重写成nodejs

新增： /www nodejs文件

/index.jade node模板

/channel\_list.jade

/channel.jade

/404.jade

/\_list.jade

/\_history.jade

/\_head.jade

/log.txt 成功日志信息

/errlog.txt 错误日志信息

修改： /clip.js

/phantomJs.exe