今天学习的内容

1:复习昨天重点

2:视频项目

<video src=”” autoplay loop controls muted poster=””

preload=””></video>

#video 弱 [不能对视频加密/视频丢失]

#video 弱 直播不行

#强 微信 小程序 公众号 [pc/pad/ios/android]

2.1:同层播放 (苹果手机)

当点击视频播放,IOS手机默认行为,自动将视频弹出在一个独立，

窗口播放.(二个窗口)

***解决问题:同一层窗口播放视频***

在<video>标签中添加二个属性

webkit-playsinline="true" (IOS)让频视在小窗口内播放，

能不同跳出全屏播放

playsinline="true" (IOS)微信浏览器支持小窗口内播放

x5-video-player-type="h5" (Android)同层播放

示例:

<video webkit-playsinline="true" playsinline="true"

x5-video-player-type="h5" ></video>

***解决问题*: 全屏设置如果不设置如下属性，横屏会错位**

x5-video-player-fullscreen="true"

全屏设置对过监听resize事件实现 (Android )

x5-video-orientation="portraint">

2.2:video 专用样式

object-fit:选项

1. fill默认 失真
2. contain 包含(留白)
3. conver 覆盖(调整视频比例)

2.3:video 事件

(1)canplaythought 可以播放视频，视频全部加载完成

#iphone 点击播放

#android 可以加完成触发事件

(2)ended 播放结束

作业1:

项目: fly(h5视频项目(故事)) 微信公众号、小程序

玩:将fly保存你电脑->无线上网

你手机->无线上网

电脑->火狐浏览器-><http://192.160.10.1/fly/index.html>

火狐生成二维码

手机->微信扫描

3:今天的目标

3.1:Bootstrap 定制—(LESS 重点)

3.4:less样式语言

http://lesscss.cn

Less是一门CSS预处理语言—它扩展了CSS语言，增加变量，函数，等新特性.使CSS更易于维护和扩展.

3.5:less两种使用方法

(1)在客户端使用—学习(效率差)

编写一个x.less

在HTML中引入x.less文件，同时再引入LESS编译程序:

less.js

客户端请求html,下载x.less以及less.js 在客户端运行js文件

less编译为css

会减慢客户端样式的呈现速度

(2)在服务器端使用-Less(必需掌握)

编写x.less

在开发电脑安装less编译环境(Nodejs/lessc),执行x.less编译x.css

再编写html文件，引入编译x.css

客户端发出请求html/下载css

3.6:LESS语法

(1)Less支持所有CSS语法

(2)Less支持多行/单行注释

/\*\*/ 多行注释

// 单行注释—推荐使用(注释内容不会编译css文件)

(3) Less 支持(变量)

定义:@变量名:值;

使用:color:@变量名;

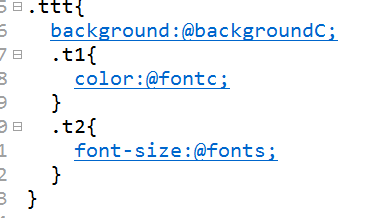
变量可以取值为任何合法样式值



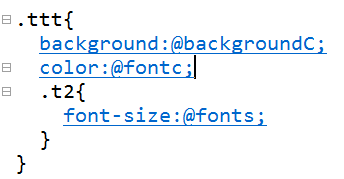
上图中有混合使用

但是有个问题，他和我想象的想过不一样（感觉自己好傻逼，尴尬）

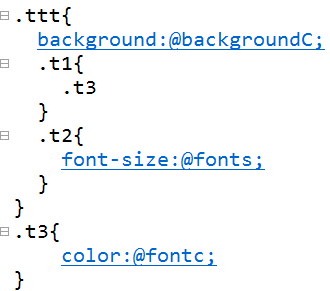
我想要的效果



但其实效果是这样的：



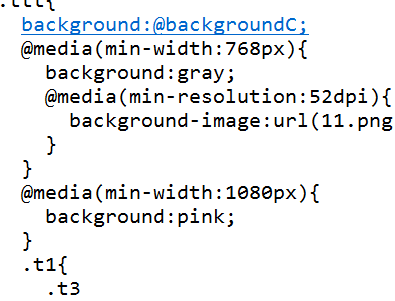
解决：实现我想要的效果



嵌套：

关于媒体查询的嵌套

Css3内容：resolution是查询分辨率

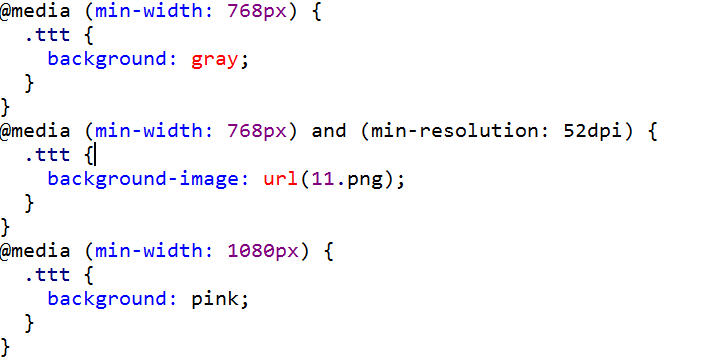


解读：当窗口大小为min-width:768px 元素背景色为gray

当 min-width:768px 并且 min-resolution:52dpi 背景图片为11.png

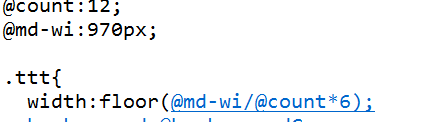
Min-width:1080px 元素背景色为pink

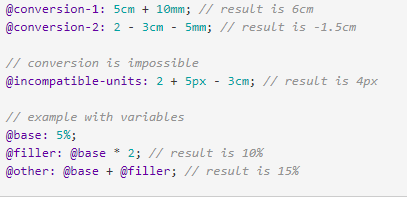
都不满足时为@backgroundC

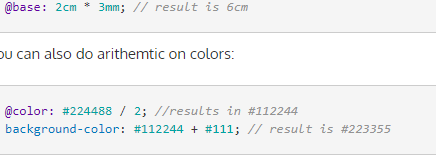


(4)Less 支持变量和常量算术运算

+ - \* / %



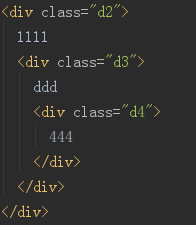
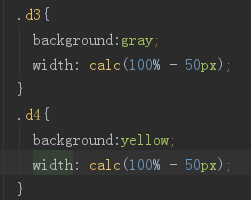


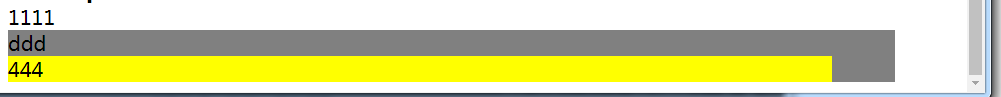


Calc **用于动态计算长度值**



宽度为父元素长度减去50px





Escaping 插入

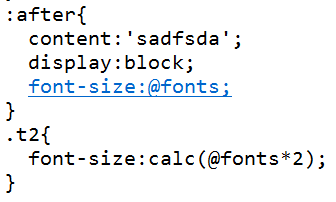
语法： ~”anything” , ~’ anything ’



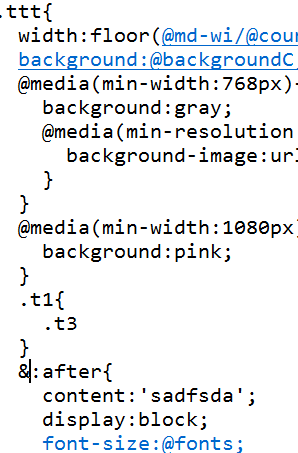
可以直接将字符串

放到 

相当于 

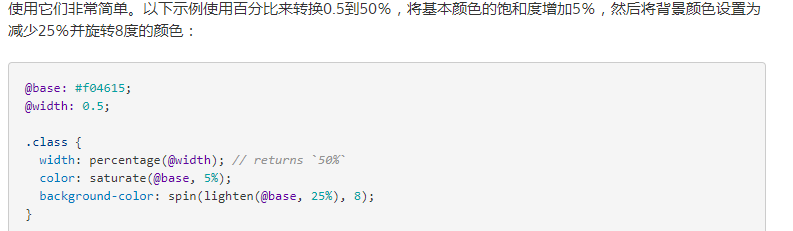
伪类是嵌套在.t2中

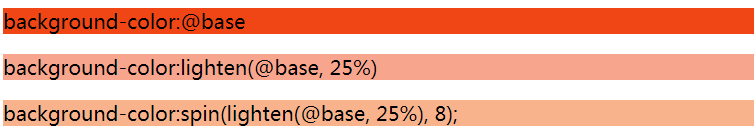
伪类是嵌套在.ttt中



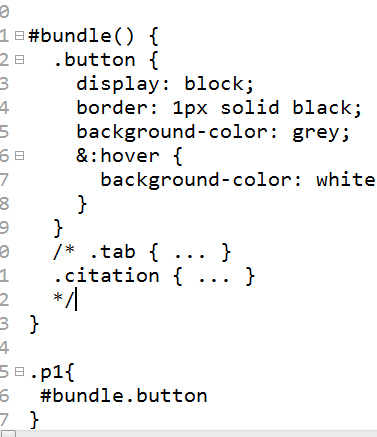
函数（function）







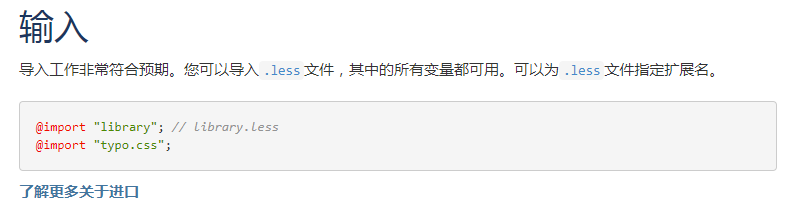
命名空间，相当于对象



相当于.p1用了#bundle中.button的样式



@import 一般不会用



(5)less支持一个选择器中”混入”另一个选择器的样式

选择器1{...}

选择器2{.. 选择器1}

(6)less在样式混入时可以指定参数

选择器1(@参数1,@参数2..){}

选择器2{选择器1(值1,值2...)}

(7)less支持样式嵌套(\*\*\*)

选择器1{

选择器2{

}

}

上述代码会被编译为:

选择器1{}

选择器 2{}

3.7:通过修改Bootstrap的less源文件实现定制

达到三个目标

(1)瘦身:删除不需要样式

(2)粗粒度定制

只需要修改variables.less中定的变量值即可

(3)细粒度定制 针对某个组件细节修改

修改组件对应less文件. dropdown.less

3.8:如何实现定制

(1)下载并且安装nodejs – (js解析器)

在命令行 node –v 可以看到版本信息

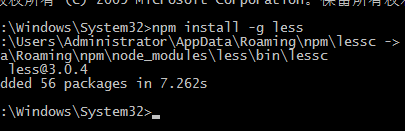
(2)下载less编译程序

e:\node\_modules\less\bin\lessc

示例:

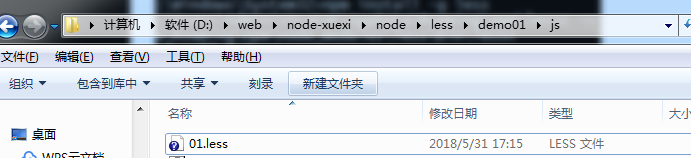
e:\node\_modules\less\bin\lessc x.less x.css

1 npm install -g less 全局安装less

.

1. 在less所在文件下 lessc 01.less 01.css





3.9 定制Bootstrap

(1) 查找software/bootstrap/less

(2) e:\node\_modules\less\bin\lessc less/bootstrap.less bootstrap.css

4.0:Nodejs

面试题:用户在浏览器中输入 [www.taobao.com](http://www.taobao.com)直到看到页面之间发生了什么?

(1)操作系统访问网络DNS服务器，把域名转换IP地址

(2)浏览器发起HTTP请求

(3)淘宝服务器接收请求消息，查找指定资源[默认index.html]

可能访问数据库，构建并且返回HTTP响消息

(4)浏览器接收响应

(5)浏览器缓存接收响应消息，并解析和渲染内容

4.1:静态网页和动态网页

静态:网页内容任何人在任何时间访问都是不变的

HTML/CSS/JS/视频/音频

动态:网页内容不同人在不同时间访问可能是不同的

JSP/PHP/ASP.NET/Node.JS

JSP = HTML+JAVA 功能强大可靠，适合大型企业级项目

PHP = HTML+PHP 简单易用,适合互联网项目

ASP.NET = HTML+C# 易用,适合windows 平台

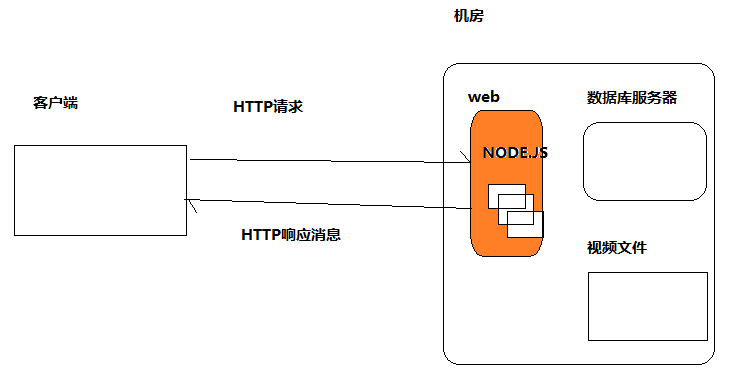
30w==2w+0.5w+20w+10w

Node.JS = HTML+”JS” 性能好,适合服务器端IO密集型项目,

不适合CPU密集型项目

CPU密集项目:滴滴打车,天气预报,大数据分析,大数据计算..

IO密集项目:查询,修改,删除,查询,查询,修改



4.2:Nodejs

Node.js不是JS,是一种软件开发平台，它的竞争对象JSP/PHP/ASP.NET,历史第一次有一种语言可以通吃前后端.

网站：阿里云镜像

https://npm.taobao.org/mirrors/node/

版本:0.12(16年初) 4.x(16年中) 6.x(16年底)

LTS:Long Term Support

Current:最新版本

4.3:Node运行的两种方式

(1)交互模块---简单了解(测试新功能)

REPL:输入一行代码执行一行

注意:交互模自带输出功能,不必一定要 console.log()

node 回车 进入交互模式

.exit 退出

(2)脚本模块—正式项目中使用

把要执行所有语句编写一个文本文件中(后缀任意，没有),一次性提交node解析器执行.

node 完整路径/x.js 回车

提交:只要安装成node.js,重启webstorm,ws可以自动发现node.exe解析，新的项目和文件编写一定UTF-8

Phpstorm运行nodejs ctrl + shift + f10

练习:创建04\_node.js ,打印九九乘法表，以脚本模式运行在ws中

故障:运行 nodejs 进程不能正常工作

原因:创建文件夹 node。js/03\_node.js

建议:所有目录名称,不要有中文,不要有空格

4.4:面试题:如何自学一门新语言

(1)了解背景 – 百度百科（事件驱动/非阻塞IO）

(2)搭建开发环境,编写HelloWorld

(3)数据类型—重点

|  |
| --- |
| 前端JS中数据类型   1. 基本/原生/值类型   string/number/boolean/null/undefined  (2)引用类型  ES对象类型:Error/Data/Math/Array/RegExp/Object/....  BOM:window;docuemnt,location,event,screen,....  DOM:Element,Attribute,.....  用户自定义对象类型:{} |
| 前端NODE JS中数据类型  (1) 基本/原生/值类型  string/number/boolean/null/undefined  (2)引用类型  ES对象类型:Error/Data/Math/Array/RegExp/Object/....  用户自定义对象类型:{}  Node.JS 第三方对象:十万多 https://www.npmjs.com/ |

(4)变量和常量

var age = 20;

const PI = 3.14;

(5)运算符

算术运算符

比较运算符

逻辑运算符

位算运算符

赋值运算符

特殊运算符 . instanceof typeof

(6)逻辑结构

循环结构:while do ..shile for(;;) for(..in..) for(..of..)

练习:创建05.js 声明一个保存5个学生成绩的数组,

使用三种for循环依次打印出每个成绩

选择结构:if..else switch..case

练习:创建一个变量 var path = “/index”;

使用二种选择结构,判断path的值为哪一种

(/index;/search;/login) 调用不同执行函数

(7)通用小程序

练习:打印出100以内所有质数

(8)函数和对象

(9)常用的组件，第三方工具库,框架

(10)实际小项目

|  |
| --- |
| Modal: 模态框  Model: 模型  Module:模块 |

4.5:nodejs中特有概念—模块---(重点)

一个项目中多个”模块”,订单模块，用户模块，支付模块....

node.js按照功能不是，可以把函数，对象分别保存不同文件或目录下，这些目录和文件在node.js称为一个“模块”

04\_node.js 05.js

##注意:Node.JS每个一个模块都是一个独立构造函数，

Node.JS解析器自动为每一个.js文件添加如下代码.

(function(exports,require,module,\_\_filename,\_\_dirname){

exports:{} 用于声明向外部导出自己成员

require:fn 用于导入其它的模块，并且创建指定模块对象

module: 当前模块对象

\_\_filename

\_\_dirname

var i = 10;

var j = 11;

console.log(i+j);

});

|  |
| --- |
| Node.js 模块中exports和module.exports对象区别是什么?  二者都可以用于向外导出自己内部成员，  但:实际向外导出成员 module.exports  exports 对应一个引用 |

每个模块都可以使用自己require();引入另一个模块—底层本质是创建指定模块一个对象实例.

08\_data.js 声明二个变量一个函数 ->导出一个变量一个函数

09\_app.js 引入

今日作业2:

仿写node.js提供一个模块:url

创建模块MyUrl.,向外导出一个方法

resolve(url);该方法接收参数如http://www.jd.com:80/admin

返回对象:

{

protocol:"http",

server:"www.jd.com",

port:80,

path:"/admin"

}

今天学习的内容

1:复习昨天重点

2:作业

3:今天的目标

练习1:创建文件模块 circle.js 其中常量PI,

声明二个方法 getSize(r) 求面积

getPerimiter(r); 求周长

创建主模块 02\_app.js 调用其公开二个方法

练习2:创建文件模块 arrayutils.js

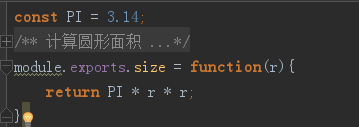
包含一个sum(arr); 返回数组值和,

包含一个avg(arr); 返回数组值平均值

创建主模块 03\_app.js 调用其公开二个方法

自定义一个node模块

1 在node\_modules中cirlce目录

2 创建一个 index.js 随便一点东西并导出

3 设置package.json 并设置

{   
 "name": "circle", // 可以随便写，没关系  
 "version": "0.0.1", // 版本号  
 "main": "./index.js" // 引入文件名称，如果目录中引入文件，则默认引入index.js  
 }

3.1:nodejs 模块-目录模块

自定义模块的两种形式

(1)**文件模块**

创建一个JS文件,如m3.js,导出其公开数据，

其它模块可以require("./m3");引入

(2)**目录模块**

方式一:创建一个目录,假设m4,其中创建index.js文件，导

出需要公开数据。其它模块引入

var m = require(“./m4”);

方式二:创建一个目录,假设m5,其中创建5.js文件，导出需要

公开数据数据,创建package.json文件 main指定启动

文件 5.js 其它模块引入

var m = require(“./m5”);

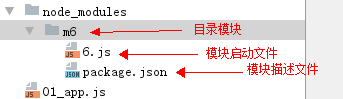
方式三:创建一个目录,必须名为 node\_modules

,其中再创建目录模块,假设 m6 其中创建 package.json

文件，其中声明 main属性指定默认执行启动js,如6.js，

其中导出需要公共数据，其它模块引入

require(“m6”);



|  |  |  |
| --- | --- | --- |
|  | 文件模块 | 目录模块(方式三) |
| 程序结构 | m1.js | node\_modules  m2  package.json  2.js |
| 模块名称 | m1 | m2 |
| 导出数据 | module.exports | module.exports |
| 导入模块 | var m = require(“./m1”); | var m = require(“m2”); |

练习1:创建二个目录模块,circle,rectangle都对外公开两个方法size(); perimeter();返回指定图形面积和周长

最后在最外层主模块中引入上述二个模块四个方法

circle.size(r); circle.perimeter(r);

rectangle.size(w,h);

rectangle.perimeter(w,h);

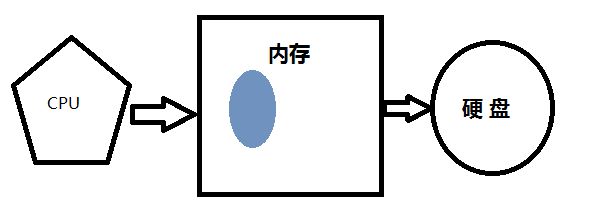
3.2:nodejs 原生模块-->(重点)

Node.js 解析引擎自带

3.3: Node.js 原生模块—Buffer(缓冲区)

Buffer:缓冲区,本质上是一块内存区域，用于暂存以后要用到数据

(数字,字符串,二进制图片/音频/视频),该区域称为“缓存”



#分配置一个指定大小缓冲区 1024字节

建议：缓冲区大小不超过 512KB 512\*1024

var buff1 = Buffer.alloc(1024);

#使用一个数组创建缓冲区

var buff2 = Buffer.from([1,2,3]);

#使用一个字符串创建缓冲区

var buff4 = Buffer.from(“abc”);

#将一个缓冲区内容转换字符串{如果缓冲区数据是字符串}

var str = buff4.toString();

3.4: Node.js 原生模块—QueryString

querystring模块用于处理HTTP请求URL中查询字符串

var obj = qs.parse(str); 把查询字符串解析js对象

var str = qs.stringify(obj) 把js对象转换为查询字符串

3.5: Node.js 原生模块—URL

url模块用于解析一个HTTP请求地址，获取其中各个部分

var obj = url.parse(str); 把一个URL地址为js对象

var obj = url.parse(str,true); 功能同上，并且把其中查询字符串转换

js obj

3.6: Node.js 原生模块—fs—(重点)

文件:在node.js 我们系统文件或者目录统称为文件

fs 模块提供对文件和目录进行，增删改查，读定文件内容

1. 同步读写文件(阻塞)

#同步读取文件内容 var data = fs.readFileSync(fileName);

#同步向文件写内容 fs.writeFileSync(fileName,data);

注意:(1)如果当前写入文件并不存在，自动创建一个

a.txt

(2)如果当前写入目录并不存在，报错

d:/abc/a.txt

(3)如果当前文件己经存在内容，清空

#同步向文件追加内容

fs.appendFileSync(fileName,data);

练习:使用上述方法，实现文件复制操作

./public/data.log-->backup\_data.log

1. 异步读写文件(非阻塞)

#异步读取数据 fs.readFile(fileName,function(err,data){ });

函数:文件读取完成后调用

err:读取文件不存在，权限不足,

自动创建错误对象

#异步写数据 fs.writeFile(fileName,data,function(err){})

函数:文件写结束调用

#异步追加数据 fs.appendFile(fileName,data,function(err){})

函数:文件追加结束

练习:使用非阻塞方法,实现public/data.txt文件,复制backup\_data.txt

3.7: Node.js 原生模块--http

HTTP 模块可用于编写基于HTTP协议客户端程序(即浏览器),也可以编写一个基于HTTP协议服务器(APACHE)

APACHE[1000并发量] nginx[28000] IIS[微软]

用http模块编写一个web服务器

(1)创建服务器对象 var server = http.createServer();

(2)绑定监听端口 1-65535 server.listen(3000);

(3)接收客户端请求， server.on(“request”,(req,res)=>{..})

(4)并且响应客户消息 res.end(str...);

req:请求对象:保存客户端请求消息

req.url 请求url地址

res:响应对象:服务器发送数据客户端,修改响应格式

作业1:使用node.js创建服务器，根据客户端的请求地址不同

输出不同的html页面内容

http://127.0.0.1:3000/login.html

http://127.0.0.1:3000/admin.html

若请求资源不存在，则返回 404.html