全栈实训课程

# 基础总结思路

班级：BW-1503

讲师：大象

日期：2017年7月7日

大象老师

QQ : 2890594972

E-mail : 2890594972@qq.com

|  |
| --- |
|  |

复习

|  |
| --- |
|  |

Git复习

|  |
| --- |
|  |

1、免费的、开源的、分布式的、版本控制器

|  |
| --- |
|  |

2、分布式与集中式

|  |
| --- |
|  |

Git 工作流程

工作区->暂存区->本地仓库->远程仓库

|  |
| --- |
|  |

Git bash指令

工作存->暂存区

指令：git add 文件名

git add 文件名 文件名 文件名

Git add --all

暂存区->本地仓库

指令：git commit -m “备注”

本地仓库->远程仓库

指令：git push origin master

PS: 第一次 git push -u origin master

拉取远程数据：远程克隆

指令：git clone 路径(https/ssh)

获取最新：git pull

|  |
| --- |
|  |

Webpack复习

|  |
| --- |
|  |

什么webpack

Webpack是一个工程化工具

PS: webpack是一个打包工具，也是一个多人协作工具

Webpack.config.js

1. 定义模块写法：Module.exports
2. Entry 本地真实的地址
   1. 字符串 唯一
   2. 数组 合并
   3. 对象 多个入口
3. Output 发布地址、上线地址
   1. 对象
   2. Path : path.resolve(\_\_dirname,”./dist/”)
   3. Filename: 文件名
   4. [name][hash

4、loader 加载器

Why: webpack只能解析js模块，对于.css,.png,.less,.scss...，就需要loader做转换

Css: style-loader,css-loader,url-loader

Less: less,less-loader

Html-webpack-plugin：自动生成一个Html文件

4.1

1. 获取webpack
   1. Var webpack = Require(“webpack”)
2. 获取压缩的属性
   1. webpack.optimize.UglifyJsPlugin
3. 插件中使用
   1. new UglifyJsPlugin()

|  |
| --- |
|  |

|  |
| --- |
|  |

敲写熟练：ES6写法

|  |
| --- |
| 1. //定义一个类 2. class Star{ 3. //定义属性 4. constructor(name){ 5. this.name = name; 6. } 7. //定义方法 8. getName(){ 9. return this.name; 10. } 11. //定义静态方法 12. static getGirlFriend(){ 13. return 'no'; 14. } 15. } |

ES5写法：

|  |
| --- |
| 1. function Star(name){ 2. this.name = name; 3. // this.getName = function(){ 4. // return this.name; 5. // } 6. } 7. Star.prototype.getName = function(){ 8. return this.name; 9. } 10. Star.hasGirlFriend = function(){ 11. return 'no'; 12. } 13. //prototype解决什么问题，(为什么要定义在prototype) 14. //共享,只有一个，换种说法：定义在对象中的方法，会有多个 15. var cl = new Star("成龙"); 16. console.log(cl.getName()); 17. console.log(Star.hasGirlFriend()); |

|  |
| --- |
|  |

创建文件

|  |
| --- |
| 1. .babelrc |

{ "presets": [ "es2015" ] }

安装三个包：

Babel-loader

Babel-core

babel-preset-es2015

下午：

复习

ES6:

|  |
| --- |
| Class Star{    Constructor(name){  This.name = name;  }  getName(){  Return this.name;  }  Static hasGirlFriend(){  }  } |
| Var chenglong = new Star(“成龙”); |

ES5

|  |
| --- |
| Function Star(name){  This.name = name;  }  Star.prototype.getName = function(){}  Star.hasGirlFriend = function(){}  Var chenglong = new Star(“成龙”); |
|  |

浏览器的转换：了解 -》1、引入两个插件 2、type=”module”

服务端转换：常用 -> babel-loader ,babel-core,babel-preset-es2015

|  |
| --- |
|  |

函数级作用域和块级作用域

|  |
| --- |
|  |

作用域：scope

变量作用的范围，分类：局部变量，全局变量(可变的范围)

函数级作用域: es5

块级作用域：es6

|  |
| --- |
| 1. If(true){ 2. Var width = 100; 3. } 4. Console.log(width); |

|  |
| --- |
| 1. Function demo(){ 2. Var height = 200; 3. } 4. Console.log(height); |
|  |

Node:js运行环境

|  |
| --- |
|  |

函数级作用域

|  |
| --- |
|  |

总结var和let比较

作用域

var 支持函数级作用域，不支持块级作用域

Let 支持函数级作用域，支持块级作用域

声明前置

Var 支持声明前置

Let 不支持声明前置

重复定义

Var 能

Let 不能

循环存值

Var 不存值

Let 能

常量const

不可改变的值

· 不能修改 不能

· 块级作用域 支持

· 不能重复定义 不能

· 不能声明前置 不支持

· 属性值只能是值类型 写代码

解构语法

对象：var {key1,key2} = obj

数组：var [p1,p2,p3,...arrData] = Array

arrData: 剩余子数组