Rotes

# Yingyu

That will be really helpful if you can.

Parse 解析

Callback 回调函数

Filed 字段

Prevent cross-site tracking 阻止夸网站跟踪

Fraudulent website warning。欺骗性网站警告

Tweak 轻微调整

**主键约束primary key constraint**

**外健约束foreign key constraint**

Selector 选择符

Fade 淡入

淡入： fadeIn()

淡出: fadeout()

Opacity 不透明

Literal 字面量

Scripting language 脚本语言

Var

Keyword 关键字

Identifier 标识符

throw 抛出

throw exception 抛出异常

Built-in 内置

Event monitoring 事件监听

Bubble 冒泡

Capture 捕获

Lambda

Recruit。 招聘

Sweep the floor扫地，

Mop the floor拖地

Cockroach 蟑螂

Landlord 房东

全麦

蓝眼泪

中秋节

清明节

春节

劳动节

国庆节

fourth quarter 第四季度

It not really I can do it, it just I can’t do anymore to resolt the issue.

EMAIL:

Shoot me an email when you're working from home if you need anything.

# Question

Django的数据是怎么从models 通过view传给html的？怎么在html里使用？

【答】：

Ajax通过访问url传递数据，url会调用view里的html或者API函数。数据通过URL来传递，api对request提取数据，然后进行处理，存到数据库里，在通过response返回数据给ajax里的success。

用户，注册登录

ORM操作网站：

<https://www.cnblogs.com/sss4/p/7070942.html>

Django 的data是怎么工作的，api是怎么设置的? 尝试自己写一个看data怎么传递。

TemplateView

Tasksheet.html : why can use user.get\_short\_name?

SQLite doc: https://sqlitebrowser.org/

jQuery documentation

Ajax doc

Pandas DataFrame顺一遍 tqdm

1111：离圣诞还有43天

CSS Datatables

Ajax

把英语提高

Drink game 店铺和价格:

<https://www.google.com.au/shopping/product/15100208352303622024?lsf=seller:8035229,store:8945171676452000992,s:h&prds=oid:7928708653242191769&hl=en&ei=-SPcX5OELcHJpgeZnZJQ>

上班穿的衣服，小西服外套，白衬衫

给外公买烟,外公生日礼物

M:

2020

12月3号考试，考3们同一天

1207 开始work form home ， 周二周五来办公室

Amelia: 会计

Jacky:

去central coast 过圣诞，父母在central coast woywoy

Alice:

20201124离职去旅游。

Level:

Chris 0

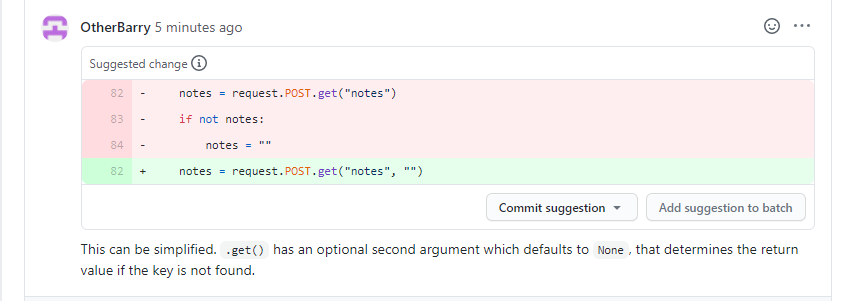
Nancy 1

Account 2

Stev 3

Other 4

# Code Improve



Python ： 判断数据类型 // isinstance(**“string”**, str)

JS: for in 和 for of 的区别 // for in的到的是下标。For of 的到的是值

Ajax：传递数据通过data，data里的值会通过url传到服务器，url相连接的后端函数的到request后，可以通过request.GET.get(“dataName”)取出数据

# Pandas

Tings should know:

Homebrew

Problem fix : download homebrew -> install sql -> install

1028 HTML, JavaScript, go through documentation

CSS

A list of job hold and other: see issue #44

Pd.DataFrame.strftime()

Df.eval()

Df.drop()

Df[‘’] = df[[“,”]].apply(lambda x=‘’.join(x), axis=1)

# HTML

No data being entered by the user, use <a>

<a class="btn btn-primary btn-lg" href="sample\_log/" target="\_blank">Sample Log Report</a>

+ <a class="btn btn-primary btn-lg" href="call\_log/" target="\_blank">Call Log Report</a>

Have data being entered by the user, use button submit.

<button type="submit" formaction="call\_log/" class="btn btn-primary">

- Call Log Report

- </button>

# Django

Unittest. : python standard library

Text

Description automatically generated

Graphical user interface, text, application

Description automatically generated

## 词汇

Tweak 轻微调整

好用的foreignkey的参数详解网站：

<http://mixoo.cn/2017/06/22/django-foreignkey-170622/>

## Instance 操作

$ python manage.py runserver

建表：

$ python manage.py migrate

Making model changes:

Change your models (in models.py)

$ python manage.py makemigrations # create migrations for those changes

$ python manage.py migrate # apply those changes to the database

$ python manage.py sqlmigrate clio 0001

进入shell：

$ python manage.py shell

## 菜鸟-view

Django 通过View 来对models的数据进行处理，再传递给template.

DetailView:

model=Atticle*#指定了数据表。他的功能相当于取出了Article中的所有数据*

template\_name="blog/index.html"*#指定页面*

context\_object\_name="artcle\_list"*#listview默认使用object\_list作为上下文变量。可使用context\_object\_name重命名。*

get\_queryset(self)*#默认取出该表所有数据。想要过滤自定义只能在get\_queryset()中*

get\_context\_data(self,\*\*kwargs)*#这个方法用来添加额外的内容到上下文变量中。*

HttpRequest

常用request属性：GET，POST，body, path, method

Render()

Redirect() : 重定向，跳转新页面

修饰器

## 菜鸟-form

* Search\_form.html 是一个表单
* HTML 表单是网站交互的经典方式，server通过解析请求，获得client传来的数据，并根据URL来提供特定的服务。
* GET数据处理过程 ---用了2个url
* POST数据处理过程 {% csrf\_token %} ---用一个url来处理
* Request 对象 --- 属性：path method GET POST REQUEST COOKIES META user session raw\_post\_data, 其中user可以判断用户是否登陆.
* QueryDict对象 --- 在HTTPRequest对象中，get和post属性是django.http.QueryDict类的实例 --- 类（字典），单键对多值。

## 菜鸟-Routing

根据用户请求的URL连接来判断对应的处理程序。在urls.py配置

Include：路由分发

*路由别名，reverse（“路由别名”）*

HTML：{% url “路由别名” %}

## 菜鸟-admin

## 菜鸟-ORM

ORM操作网站：

<https://www.cnblogs.com/sss4/p/7070942.html>

## 菜鸟-template

## 菜鸟-

# 数据库---MySQL

## bilibili--- 简介

要使用mysql数据库，需要先下载mysql服务器到本地，官网可下。

**学习网站**：

B站视频 讲特别清楚： <https://www.bilibili.com/video/BV1Vt411z7wy?p=4>

菜鸟： <https://www.runoob.com/mysql/mysql-data-types.html>

## bilibili--- 数据库基本操作

**如何登陆数据库:**

Xie:~ X$ mysql -uroot -pXiechengzhu

**如何查询数据库服务器中的所有的数据库:**

Xie:~ X$ show databases;

**如何选中一个数据库进行操作**:

Xie:~ X$ use sushi

Xie:~ X$ select \* from admin;

**退出**:

Xie:~ X$ quit;

**创建数据库：**

Xie:~ X$ create database [name];

**查看创建好的数据表格：**

Xie:~ X$ describe [tablename];

## bilibili--- 表格操作

建表：

CREATE TABLE [name](

Name VARCHAR(20),

…

);

## bilibili--- 数据类型与类型选择

【菜鸟mysql数据类型里有详细的】

**如何选择数据类型：**

**日期 选择按照格式**

**数值和字符串按照大小**

**数值类型**：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **类型** | **大小** | **范围（有符号）** | **范围（无符号）** | **用途** |
| TINYINT | 1 byte | (-128，127) | (0，255) | 小整数值 |
| SMALLINT | 2 bytes | (-32 768，32 767) | (0，65 535) | 大整数值 |
| MEDIUMINT | 3 bytes | (-8 388 608，8 388 607) | (0，16 777 215) | 大整数值 |
| INT或INTEGER | 4 bytes | (-2 147 483 648，2 147 483 647) | (0，4 294 967 295) | 大整数值 |
| BIGINT | 8 bytes | (-9,223,372,036,854,775,808，9 223 372 036 854 775 807) | (0，18 446 744 073 709 551 615) | 极大整数值 |
| FLOAT | 4 bytes | (-3.402 823 466 E+38，-1.175 494 351 E-38)，0，(1.175 494 351 E-38，3.402 823 466 351 E+38) | 0，(1.175 494 351 E-38，3.402 823 466 E+38) | 单精度 浮点数值 |
| DOUBLE | 8 bytes | (-1.797 693 134 862 315 7 E+308，-2.225 073 858 507 201 4 E-308)，0，(2.225 073 858 507 201 4 E-308，1.797 693 134 862 315 7 E+308) | 0，(2.225 073 858 507 201 4 E-308，1.797 693 134 862 315 7 E+308) | 双精度 浮点数值 |
| DECIMAL | 对DECIMAL(M,D) ，如果M>D，为M+2否则为D+2 | 依赖于M和D的值 | 依赖于M和D的值 | 小数值 |

**日期和时间类型：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **类型** | **大小 ( bytes)** | **范围** | **格式** | **用途** |
| DATE | 3 | 1000-01-01/9999-12-31 | YYYY-MM-DD | 日期值 |
| TIME | 3 | '-838:59:59'/'838:59:59' | HH:MM:SS | 时间值或持续时间 |
| YEAR | 1 | 1901/2155 | YYYY | 年份值 |
| DATETIME | 8 | 1000-01-01 00:00:00/9999-12-31 23:59:59 | YYYY-MM-DD HH:MM:SS | 混合日期和时间值 |
| TIMESTAMP | 4 | 1970-01-01 00:00:00/2038  结束时间是第 **2147483647** 秒，北京时间 **2038-1-19 11:14:07**，格林尼治时间 2038年1月19日 凌晨 03:14:07 | YYYYMMDD HHMMSS | 混合日期和时间值，时间戳 |

**字符串类型：**

|  |  |  |
| --- | --- | --- |
| **类型** | **大小** | **用途** |
| CHAR | 0-255 bytes | 定长字符串 |
| VARCHAR | 0-65535 bytes | 变长字符串 |
| TINYBLOB | 0-255 bytes | 不超过 255 个字符的二进制字符串 |
| TINYTEXT | 0-255 bytes | 短文本字符串 |
| BLOB | 0-65 535 bytes | 二进制形式的长文本数据 |
| TEXT | 0-65 535 bytes | 长文本数据 |
| MEDIUMBLOB | 0-16 777 215 bytes | 二进制形式的中等长度文本数据 |
| MEDIUMTEXT | 0-16 777 215 bytes | 中等长度文本数据 |
| LONGBLOB | 0-4 294 967 295 bytes | 二进制形式的极大文本数据 |
| LONGTEXT | 0-4 294 967 295 bytes | 极大文本数据 |

## bilibili--- 约束constraint-主键约束|外健约束

**主键约束primary key constraint：**

它能够唯一确定一张表中的一条记录，就是通过主键约束可以保证字段不重复且不为空

它联合主键：只要联合的主键加起来不重复就可以

如果创建表的时候忘记创建主键约束：

可以通过alter table user1 add primary key(name); 添加或删除

删除，alter table user1 drop primary key;

还可以使用modify 修改字段

**外健约束foreign key constraint：**

* 主表classes 中没有的数据，在副表中，是不可以使用的。
* 主表中的记录被副表引用，是不可以被删除的。

主表-班级 副表-学生：

**Text

Description automatically generated**

**自增约束：**

它可以和主键约束一起用

Eg：

Create table user(

id int primary key auto\_increment,

name varchar(20)

)

**唯一约束：**

**非空约束：**

**默认约束：**

## bilibili--- 范式

第一范式：

* 数据表中的所有字段都是不可分割的原子值

如果一个字段里的信息值是可以再分割的，就不满足第一范式

范式设计的越详细，对于实际操作可能更好，但不一定是最好的方式

第二范式：

* 必须满足第一范式的前提下
* 除主键外的每一列都必须完全依赖于主键
* 如果出现不完全依赖，只可能发生在联合主键的情况下

不满足的时候通常会拆表

第三范式：

必须满足第二范式的前提下

除开主键列的其他列之间，不能有传递依赖关系

## bilibili---

## bilibili---

# jQuery

## 菜鸟--- 教程

jQuery是一个JavaScript库

简化了JS编程

实例：

$(document).ready(function(){

$("p").click(function(){

$(this).hide();

});

});

【jQuery方式】

function myFunction()

{

$("#h01").html("Hello jQuery");

}

$(document).ready(myFunction);

## 菜鸟--- 简介

【jQuery库功能】

HTML元素选取

HTML元素操作

CSS操作

HTML事件函数

JS特效和动画

HTML DOM 遍历和修改

AJAX

Utilities

## 菜鸟--- 安装

jQuery是一个库，需要下载，可以下载到本地，或者通过CDN（内容分发网络）引用

例如：通过google CDN引用它

<head>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js">

</script>

</head>

查看版本：

在Console窗口 $.fn.jquery

## 菜鸟--- 语法

Selector 选择符

语法

**$(*selector*).*action*()**

美元符号定义jQuery

Selector “查询”“查找”HTML元素

jQuery的action()执行对元素的操作

$(this).hide()

$(“p”).hide()

$(“p.test”).hide()

$(“#test”).hide()

文档就绪：

$(document).ready(function(){

// 开始写 jQuery 代码...

});

为了防止文档在完全加载之前运行jQuery代码，比如获得未完全加载的图像的大小。还有一种简洁写法：

$(function(){

// 开始写 jQuery 代码...

});

【jQuery入口函数】

$(document).ready(function(){

    // 执行代码

});

或者

$(function(){

    // 执行代码

});

【JS入口函数】

window.onload = function () {

// 执行代码

}

【jQuery入口函数 和 JS入口函数区别】

jQuery的入口函数是在HTML所有标签（DOM）都加载之后，就会去执行。

JS的window.onload事件是等到所有内容，包括外部图片之类的文件加载完后，才会执行。

Table

Description automatically generated

## 菜鸟--- 选择器selector

jQuery selector基于元素的id、类、类型、属性、属性值等“查找”或选择HTML元素。

【selector】 菜鸟- CSS选择器 有详细的

实例：

$(document).ready(function(){

$("button").click(function(){

$("p").hide();

});

});

用户点击按钮后，所有<p>元素都隐藏。

【各种selector语法】 :

$(“\*”) $(this) $(“p.intro”) $(“ul li:first”) $(“ul li:first-child”) $(“p:first”)

$(“[href]”) $(“a[target=’\_blank’]”) $(“a[target!=’\_blank’]”)

$(“:button”) $(“tr:even”) $(“tr:odd”)

【实操笔记】

$(‘#idname’) 中idname 如果含’.’需要进行特殊处理，因为是特殊符号不能识别

cb = "#" + "1.33".replace(/\./g, '\\.');  
$(cb).prop("checked", true);

## 菜鸟--- 事件event

Event 事件

Trigger 触发

Focus 焦点

【事件】： 详细见菜鸟

Click dblclick mouseenter mouseleave hover

Keypress keydown keyup

submit change focus blur

load resize scroll unload

|  |  |  |  |
| --- | --- | --- | --- |
| **鼠标事件** | **键盘事件** | **表单事件** | **文档/窗口事件** |
| [click](https://www.runoob.com/jquery/event-click.html) | [keypress](https://www.runoob.com/jquery/event-keypress.html) | [submit](https://www.runoob.com/jquery/event-submit.html) | [load](https://www.runoob.com/jquery/event-load.html) |
| [dblclick](https://www.runoob.com/jquery/event-dblclick.html) | [keydown](https://www.runoob.com/jquery/event-keydown.html) | [change](https://www.runoob.com/jquery/event-change.html) | [resize](https://www.runoob.com/jquery/event-resize.html) |
| [mouseenter](https://www.runoob.com/jquery/event-mouseenter.html) | [keyup](https://www.runoob.com/jquery/event-keyup.html) | [focus](https://www.runoob.com/jquery/event-focus.html) | [scroll](https://www.runoob.com/jquery/event-scroll.html) |
| [mouseleave](https://www.runoob.com/jquery/event-mouseleave.html) |  | [blur](https://www.runoob.com/jquery/event-blur.html) | [unload](https://www.runoob.com/jquery/event-unload.html) |
| [hover](https://www.runoob.com/jquery/event-hover.html) |  |  |  |

实例：

<script>

$(document).ready(function(){

$("input").focus(function(){

$(this).css("background-color","#cccccc");

});

$("input").blur(function(){

$(this).css("background-color","#ffffff");

});

});

</script>

## 菜鸟--- 效果

### 菜鸟--- hide show

Fade 淡入

【效果】：隐藏，显示，切换，滑动，淡入淡出，动画

语法：

$(*selector*).hide(*speed,callback*);  
$(*selector*).show(*speed,callback*);

可选的speed参数规定隐藏/显示的速度，可选的callback参数是完成后所执行的函数名称

Toggle() 切换显示/隐藏

$(*selector*).toggle(*speed,callback*);

### 菜鸟--- 淡入淡出 fade

淡入： fadeIn()

淡出: fadeout()

切换淡入淡出： fadeToggle()

渐变为给定的不透明度： fadeTo()

### 菜鸟--- 滑动

slideDown()

slideUp()

slideToggle()

语法：

$(*selector*).slideDown(*speed,callback*);

$(*selector*).slideUp(*speed,callback*);

$(*selector*).slideToggle(*speed,callback*);

### 菜鸟--- 动画

Opacity 不透明

animate()

语法：

$(*selector*).animate({*params*}*,speed,callback*);

必需的params参数定义形成动画的CSS属性。

\*\* default情况下，元素无法移动，对位置进行操作是，要首先把元素的CSS position属性设置为relative、fixed 或 absolute

\*\* 要使用animate()时，必须使用Camel标记法书写所有属性名

【属性的动画值可设置为show hide toggle】

【animate()的队列功能】---多个变化

### 菜鸟--- 停止动画

语法：

$(*selector*).stop(*stopAll,goToEnd*);

可选stopAll：规定是否应该清除动画队列，队列会向后执行。 default：false

可选goToEnd：规定是否立即完成当前动画。Default：false

实例：

$("#stop").click(function(){

$("#panel").stop();

});

### 菜鸟--- callback

### 菜鸟--- 链chaining

Chaining：允许我们在相同的元素上运行多条jQuery命令，一条接着另一条，这样浏览器就不必多次查找相同的元素。

实例：

$("#p1").css("color","red").slideUp(2000).slideDown(2000);

## 菜鸟--- HTML

## 菜鸟--- 遍历

## 菜鸟--- Ajax

## 菜鸟--- 其他

## 菜鸟--- 实例

## 菜鸟--- 参考手册

## 菜鸟--- 插件

# Javascript

## 菜鸟--- 用法

document.getElementById(“id\_name”)

document.getElementById().innerHTML= “Hello”

<script></script>

Alert

Document.write(“<p>Hello</p>”)

Onclick=”myFunction()”

外部的JavaScript：保存在外部文件。<script src="myScript.js"></script>

## 菜鸟---输出

window.alert() 弹出警告框

document.write() 将内容写道HTML文档中

innerHTML 写入到HTML元素

console.log() 写入到浏览器的控制台

## 菜鸟---语法

Literal 字面量

Scripting language 脚本语言

Var

Keyword 关键字

Identifier 标识符

注释 //

大小写敏感

使用Unicode字符集

## 菜鸟---语句statement

语句后添加；

空格：JS会忽略多余空格

字符串中可以使用 \ 进行换行

document.write("你好 \

世界!");

## 菜鸟---注释comment

多行注释 /\* \*/

## 菜鸟---变量variable

声明变量

声明可以横跨多行：

Var lastname="Doe",

age=30,  
job="carpenter";

声明的多个变量不可以同时赋同一个值：

## 菜鸟---数据类型type of data

值类型：string, umber, Boolean, Null, Undefined, Symbol

引用数据类型：Object, Array, Function

## 菜鸟---对象object

对象的属性和寻址：使用 { } var car = {type:"Fiat", model:500, color:"white"};

Addressing 寻址

New 用来声明变量类型：var x=      new Number;

对象中的方法可以是函数：

var person = {

firstName: "John",

lastName : "Doe",

id : 5566,

fullName : function()

{

return this.firstName + " " + this.lastName;

}

};

## 菜鸟---函数

函数是包裹在 { } 中的代码块， 前面使用到了关键字function

大小写敏感

函数变量和传参必须以相同顺序出现

Local variable 局部变量

Global variable 全局变量

变量的生存期

为声明变量可删除： delete var1；

## 菜鸟---作用域scope

全局变量有俩种：

在函数外定义

在函数内不使用var 关键字声明

## 菜鸟---事件event

HTML事件：html页面完成加载 / input 字段改变 / 按钮被点击

可以在html中使用single quote 或者double quote 来添加 JS代码：

<*some-HTML-element* *some-event*=**"*JavaScript 代码*"**>

Quote里面也可以是JS函数：

<button onclick="displayDate()">点这里</button>

<script>

function displayDate(){

document.getElementById("demo").innerHTML=Date();

}

</script>

<p id="demo"></p>

This.innerHTML= 改变自身元素的内容

常见HTML event ：

onchange 元素改变

onclick 点击元素

onmouseover 在元素上移动鼠标

onmouseout 在元素上移开鼠标

onkeydown 按下键盘按键

onload 浏览器已完成页面的加载

## 菜鸟---字符串string

引号怎么使用

字符串长度： str.length

特殊字符： \’ \” \\ \n \r \t \b \f

New 可以使用new 关键字将string定义为objact

=== ：绝对等号

字符串属性： constructor, length, prototype

字符串方法：charAt(), charCodeAt(), Concat(), fromCharCode(), indexOf(), lastIndexOf(), localeCompare(), match(),

replace(),search(), slice(), split(), substr(), substring(), toLocaleUpperCase(), toLocaleLowerCase(),

toLowerCase(), toString(0, toUpperCase(), trim(), valueOf()

## 菜鸟---运算符operator

算数运算符： + - \* / % ++ ---

赋值运算符： = += -== \*= /= %=

字符串相加

## 菜鸟---比较

比较运算符： == ==== != !==不觉对等于

逻辑运算符： && : and ||: or ！: not

## 菜鸟---条件语句if

if (time<20)

{

x="Good day";

}

## 菜鸟--- Switch语句

用于基于不同条件来执行不同的动作

Default 关键字来规定匹配不存在时

## 菜鸟--- for循环

for (var i=0;i<cars.length;i++)

{

document.write(cars[i] + "<br>");

}

【不同类型的循环】

For/in :

var person={fname:"Bill",lname:"Gates",age:56};

for (x in person) // x 为属性名

{

txt=txt + person[x];

}

【实操笔记】

Graphical user interface, text, application

Description automatically generated

## 菜鸟--- While循环

Do/while 循环 ： 是while的变体，循环会至少执行一次

## 菜鸟--- Break 和continue语句

Break ： 用于跳出循环，之后会继续执行循环后的代码

Continue： 用于跳过循环中的一个迭代。中断循环中的迭代

JS标签：

label:

statements

语法：

break labelname;

continue labelname;

## 菜鸟--- Typeof

检查数据类型

typeof "John”;

## 菜鸟---类型转换

JS有6种不同的数据类型： string number boolean object function symbol

3中对象类型： Object Date Array

2个不含任何值的数据类型： null undefined

constructor 属性：返回所有JS变量的构造函数

用来判断是否为日期，是否为string ：myDate.constructor.toString().indexOf("Date")

String() : 可用于任何类型的数字，字母，变量，表达式 转换成string

== .toString()

Number 方法：

toFixed() : 把数字转换为字符串，结果的小数点后有指定位数的数字

toPrecision(): 把数字格式化为指定的长度

【将布尔值转化为字符串】

【将日期转化为字符串】

【将字符串转换为数字】

Operator + 可用于将变量转换为数字

var y = "5";      // y 是一个字符串

var x = + y;      // x 是一个数字

【将日期转化为数字】 Number()

【自动转换为字符串】

【不同数值转换为数字，字符串，布尔值的结果】

## 菜鸟---正则表达式regular expression

语法：

/正则表达式主体/修饰符(可选)

【使用字符串方法】

Search() : 检索字符串中指定的字符串，或检索与正则表达式相匹配的子字符串，并返回子串的起始位置

Replace() ： 在字符串中用一些字符替换另一些字符，或替换一个与正则表达式匹配的子串

【正则表达式修饰符】 regex modifier

Modifier 修饰符

正则表达式方法：test() exec()

/e/.test("The best things in life are free!") // 结果true

【判断输入字符串是否为数字、字母、下划线组成】

【判断输入字符是否全部为字母】

【判断输入字符串是否全部为数字】

## 菜鸟---错误

throw 抛出

throw exception 抛出异常

Try 语句测试代码块的错误

catch 语句处理错误

throw 语句创建自定义错误

finally 语句在try 和 catch 语句之后，无论是否有触发异常，该语句都会执行

语法grammar：

try {

... //异常的抛出

} catch(e) {

... //异常的捕获与处理

} finally {

... //结束处理

}

【throw】

## 菜鸟---调试debugging

Built-in 内置

浏览器启用调试工具一般是按下 F12 键，并在调试菜单中选择 "Console" 。

【个浏览器debugging的方法】

## 菜鸟---变量提升hoisting

JS中函数和变量的声明都将被提升到函数的最顶部，也就是说可以先使用再声明。

初始化不会提升。

可以

x = 5；

var x; // 声明 x

不可以，会报错：

var x = 5; // 初始化 x

elem = document.getElementById("demo"); // 查找元素

elem.innerHTML = x + " " + y;           // 显示 x 和 y

var y = 7; // 初始化 y

## 菜鸟---严格模式strict mode

【use srict】

<script>

"use strict";

myFunction();

function myFunction() {

y = 3.14; // 报错 (y 未定义)

}

</script>

## 菜鸟---使用误区

浮点数： JS 中所有数据都是以64位浮点数来存储 // 可以使用整数乘除法来解决

【return使用注意事项】

数组中不能使用名字来索引，使用名字会被重新定义为标准对象

【Null 和 Undefined区别】

## 菜鸟---表单form

fname 表单字段

Constraint validation约束验证

document.forms[“formname”][“fname”].value

【约束验证】 表单被提交时浏览器用来实现验证的一种算法

HTML约束验证基于：

【HTML输入属性】 disabled max min pattern required type

【CSS伪类选择器 : disabled :invalid :optional :required :valid

【DOM属性和方法】

## 菜鸟---表单验证validation

【email 验证】

## 菜鸟---验证API

checkValidity()如果input元素中的数据是合法的返回true，否则返回false

setCustomValidity()用于自定义错误提示信息的方法

object.validationMessage

【约束验证DOM属性】 ： validity validationMessage willValidate

【Validity属性】 有很多，看菜鸟

## 菜鸟---保留关键字reversed keywords

避免使用关键字作为名字

【window关键字】

【HTML事件句柄】

## 菜鸟--- This

表示对当前对象的一个引用。但是会随着执行环境的改变而改变：

在方法中，this便是该方法所属的对象。

单独使用时，this表示全局对象。

在函数中，this表示全局对象。

在函数中，严格模式下，this是未定义的。

在事件中，this表示接受事件的元素。

类似call() 和apply()方法允许切换函数执行的上下文环境，即this绑定的对象，可以将this引用到任何对象。

This.stayle.display=’none’ // 消失

【显示函数绑定】

Call() apply() --函数对象的方法

var person1 = {

fullName: function() {

return this.firstName + " " + this.lastName;

}

}

var person2 = {

firstName:"John",

lastName: "Doe",

}

person1.fullName.call(person2); // 返回 "John Doe"

## 菜鸟--- Let 和 const

Let声明的变量只在let命令所在的代码块内有效 // 用来保证局部变量和全局变量不冲突

Const声明一个只读的常量，一旦声明，常量的值就不能改变。

JS只有俩种作用域：全局变量 与 函数内的局部变量。

在JS中，全局作用域是针对JavaScript环境。

在HTML中，全局作用域是针对window对象。

【var和let区别】：

使用var关键字声明的变量在任何地方都可以修改

在相同的作用域或块级作用域中，不能使用let关键字来重置var/let关键字声明的变量

在不同的作用域或块级作用域中，可以使用let关键字来重置let关键字声明的变量

Let/ const不能提升变量

const用于声明一个或多个常量，声明时必须进行初始化，且之后不可修改。

【const和let相同点】：

都是块级作用域

都不能和所在作用域内的其他变量或函数拥有相同的名称

Const本质：定义的对象或者数组，是可变的。但不能对常量数组重新赋值

Const在不同作用域可以重新声明赋值

## 菜鸟--- Json ： JavaScript Object Notation

语法：

数据为键/值 对

数据由逗号分隔

大括号保存对象

方括号保存数组

JSON.parse() ：  JavaScript 内置函数 JSON.parse() 可以将字符串转换为 JavaScript 对象

JSON.stringify(): 用于将 JavaScript 值转换为 JSON 字符串。

## 菜鸟--- Void

Void是JS中非常重要的关键字，该操作符operator指定要计算一个表示式但是不返回值。

Grammar:

void func()

javascript:void func()

or

void(func())

javascript:void(func())

【href="#"与href="javascript:void(0)"的区别】：

#包含了一个位置信息

javascript:void(0)仅仅表示一个死链接

## 菜鸟---异步编程Asynchronous

同步：按照代码顺序执行

异步：不按照代码顺序执行。效率更高

回调函数callback

setTimeout(arg1,arg2) : arg1是函数，arg2是时间

XMLHttpRequest() 属性：onload onerror open() send()

## 菜鸟--- Promise

是一个ECMAScript 6 提供的类，一些旧的浏览器并不支持.

Promise的构造函数（constructor）只有一个参数，是一个函数，这个函数在构造之后会直接被异步运行，所以我们称之为起始函数。起始函数包含2个参数resolve和reject。

新建：

new Promise(function (resolve, reject) {

// 要做的事情...

});

new Promise(function (resolve, reject) {

console.log(1111);

resolve(2222);

}).then(function (value) {

console.log(value);

return 3333;

}).then(function (value) {

console.log(value);

throw "An error";

}).catch(function (err) {

console.log(err);

});

方法：

.then() 将参数中的函数添加到当前的正常执行序列

.catch() 设定promise的异常处理序列

.finally() 最后一定会执行的序列

resolve() 中可以放置一个参数用于向下一个 then 传递一个值，then 中的函数也可以返回一个值传递给 then。但是，如果 then 中返回的是一个 Promise 对象，那么下一个 then 将相当于对这个返回的 Promise 进行操作。

* resolve 和 reject 的作用域只有起始函数，不包括 then 以及其他序列；
* resolve 和 reject 并不能够使起始函数停止运行，别忘了 return

【一些then catch finally的顺序及使用的问题】

## 菜鸟---代码规范

命名规则CamelCase

每行代码字符小于80个

HTML CSS的横杠（-）字符

## 菜鸟--- JS 函数

### 函数定义

【自调用函数】 ：表达式后面紧跟 ()，会自动调用

【JS函数可以作为表达式使用】

var x = myFunction(4, 3) \* 2;

【函数是对象，有属性和方法】 还有call() apply()

arguments.length : 这个属性返回函数调用过程收到的参数个数

toString() ：这个方法将函数作为一个字符串返回。

function myFunction(a, b) {  
    return a \* b;  
}  
  
var txt = myFunction.toString();

【箭头函数】

(参数1, 参数2, …, 参数N) => { 函数声明 }

(参数1, 参数2, …, 参数N) => 表达式(单一)

// 相当于：(参数1, 参数2, …, 参数N) =>{ return 表达式; }

没有参数时：

() => {函数声明}

// ES5

var x = function(x, y) {

return x \* y;

}

// ES6

const x = (x, y) => x \* y;

箭头函数的this的值会默认绑定外层this的值。

箭头函数不能提升，使用前需要定义

### 函数参数

【显性参数】

functionName(parameter1, parameter2, parameter3) {

// 要执行的代码……

}

【隐形参数】：函数调用时传递给函数的真正的值

【为参数设置一个默认值】 y = y ||0；

ES6 支持函数带有默认参数

【argument对象】 JS函数的内置对象

### 函数调用

JS函数有4种调用方式，他们的不同在于this 的初始化。

【this指向】 ：一般指向函数执行时的当前对象。

【构造函数】： 函数调用前使用 new 关键字。

【函数方法调用函数】；

call() apply() 他们俩个方法的第一个参数必须是对象本身。

他俩区别是传参方式，apply传入一个参数数组

### 闭包

## HTML DOM

### DOM 简介

Dom简介:

Dom文档对象模型

【查找html元素】：

通过id

通过标签名

通过类名 var x=document.getElementsByClassName("intro");

### DOM HTML

Dom html：

改变元素属性：document.getElementById(*id*).*attribute=新属性值*

改变元素的样式：document.getElementById("p2").style.fontSize="larger";

### DOM CSS

【html允许通过触发事件来执行代码】：

元素被点击

页面加载完成

输入框被修改

等等

### DOM 事件

Dom事件：

【事件的例子】

【onclick】：onclick=*JavaScript*

【onload 和 onunload】

【onchange】常结合对输入字段的验证来使用

【onmousever 和 onmouseout】移动鼠标触发

【onmousedown 和 onmouseup】

【onfocus】

### DOM EventListener

Event monitoring 事件监听

Bubble 冒泡

Capture 捕获

addEventListener()

removeEventListener()

语法：

*element*.addEventListener(*event, function, useCapture*);

第一个参数是事件的类型 (如 "click" 或 "mousedown").

第二个参数是事件触发后调用的函数。

第三个参数是个布尔值用于描述事件是冒泡还是捕获。该参数是可选的。

可以向同一个元素中添加多个事件句柄

### DOM 元素

【向文档中添加和移除元素（节点）】 具体看菜鸟

### HTMLCollection对象

### NodeList对象

## JavaScript高级教程

### JavaScript 对象

### JavaScript prototype

### JavaScript Number 对象

### JavaScript String

### JavaScript Date（日期）

### JavaScript Array（数组）

### JavaScript Boolean（布尔）

### JavaScript Math（算数）

### JavaScript RegExp（对象）

## JS浏览器BOM

## JS 库

### JavaScript 测试jQuery

需在网页中引用它：

<script src="https://cdn.staticfile.org/jquery/1.8.3/jquery.min.js"> </script>

【jQuery方式】

function myFunction()

{

$("#h01").html("Hello jQuery");

}

$(document).ready(myFunction);

### JavaScript 测试Prototype

### Prototype

## JS 实例

## JS 参考手册

# Ajax

Ajax可以用来：

注册时，输入用户名自动检测用户是否已经存在

登陆时，提示用户名密码错误

删除数据行时，将行ID发送到后台，后台在数据库中删除，数据库删除成功后，在页面DOM中将数据行也删除。

Ajax和jQuery一起封装优点：

Developer不用考虑参数顺序及get post的大小写问题，更加方便

## Bilibili-遇见狂神说

<https://www.bilibili.com/video/BV1Kt411u7BV?from=search&seid=16457670512570027463>

jQuery所有参数：

Text

Description automatically generated

## Bilibili-

<https://www.bilibili.com/video/BV17W41137jn?p=142>

创建异步对象

设置请求方式和请求地址

发送请求

监听状态的变化

处理返回的结果

Ajax-jQuery封装：

Text

Description automatically generated

Text

Description automatically generated

## Bilibili-

## Bilibili-

## 菜鸟-教程

AJAX = asynchronous JS and XML. 异步的JS和XML

是一种使用现有标准的新方法

最大的优点在于不重新加载整个页面的情况下，可以与服务器交换数据并更新部分网页内容

不需要任何浏览器插件

应用：

运用JS操作DOM来执行动态效果

## 菜鸟-简介

Diagram

Description automatically generated

## AJAX是基于现有的Internet标准

AJAX是基于现有的Internet标准，并且联合使用它们：

* XMLHttpRequest 对象 (异步的与服务器交换数据)
* JavaScript/DOM (信息显示/交互)
* CSS (给数据定义样式)
* XML (作为转换数据的格式)

lamp  AJAX应用程序与浏览器和平台无关的

## 菜鸟-实例

## 菜鸟-创建对象

XMLHttpRequest 是AJAX的基础

创建XMLHttpRequest语法：

*variable*=new XMLHttpRequest();

【要检查浏览器版本是否支持】 详见菜鸟

## 菜鸟-请求

XMLHttpRequest对象用于和服务器交换数据

Open()

Send()

xmlhttp.open("GET","ajax\_info.txt",true);  
xmlhttp.send();

A picture containing graphical user interface, text

Description automatically generated

例子：

xmlhttp.open("GET","/try/ajax/demo\_get.php",true);

xmlhttp.send();

如果需要通过GET发送信息，向url添加

【POST 像HTML form 一样post数据】 详见菜鸟

使用setRequestHeader()

xmlhttp.open("POST","/try/ajax/demo\_post2.php",true); xmlhttp.setRequestHeader("Content-type","application/x-www-form-urlencoded"); xmlhttp.send("fname=Henry&lname=Ford");

A picture containing graphical user interface

Description automatically generated

Xmlhttp.onreadystatechange=function()

## 菜鸟-响应

## 菜鸟-readyState

## 菜鸟-ASP/PHP

## 菜鸟-数据库

## 菜鸟-XML

## 菜鸟-实例

# Select2

https://blog.csdn.net/fukaiit/article/details/81750755