知识点分析
几个笔记汇总
几处细节
高亮处的一个细节(添加了插件prism.js)
异步
笔记详解
打字原理
间歇调用
setInterval函数用法
超时调用结合递归来实现
加式样
完善会动的简历
下面看有回调的(有异步)

知识点分析

只要在html里面只要有回车,空格和tab,不管有多少个，都会压缩成一个空格,可以用 `pre` 标签(这个标签会保留回车,空格和tab);

几个笔记汇总

https://zhuanlan.zhihu.com/p/45394416
https://segmentfault.com/a/1190000018779991 (这个讲解的不错)
https://juejin.im/post/5d59ccff5188255b2131eb29
https://www.jianshu.com/p/ef71fed1371d

几处细节

高亮处的一个细节(添加了插件 `prism.js`)

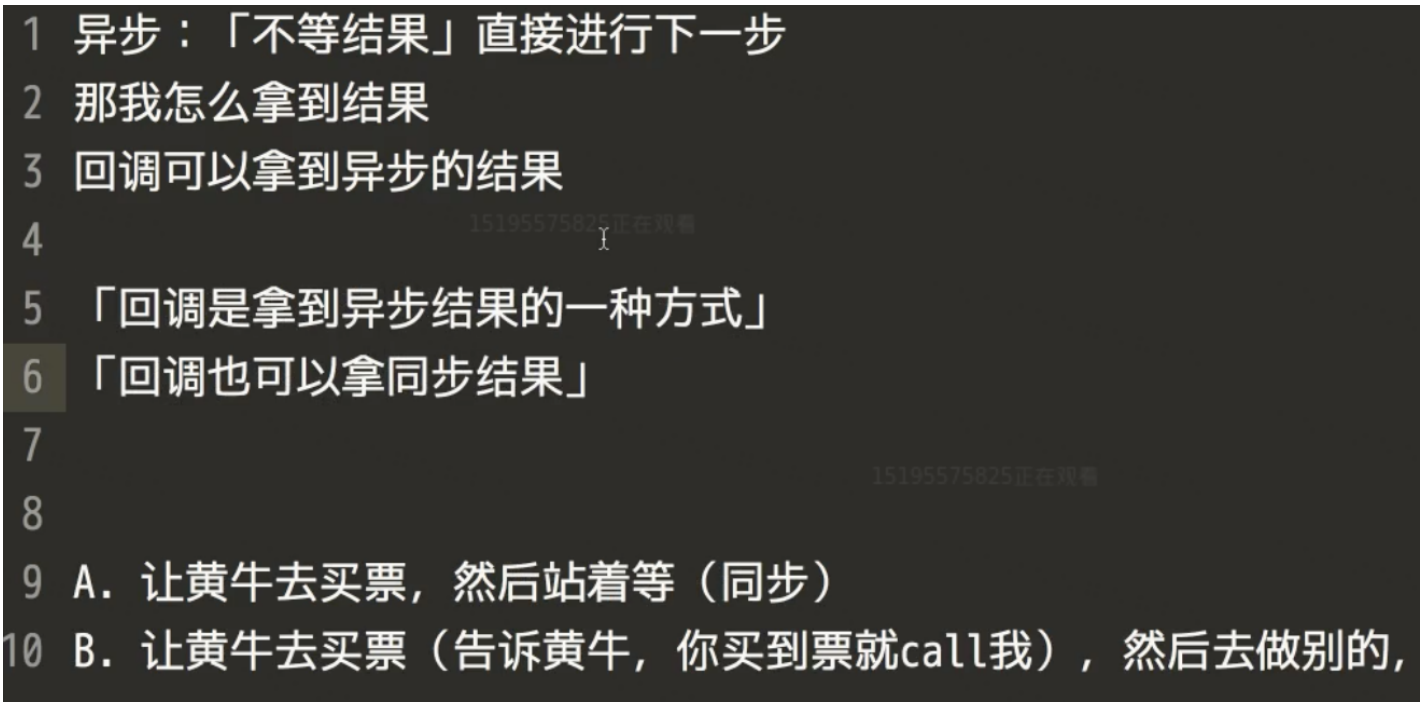
使用了一个库把没有span的东西变成了有span的东西


异步

异步:不等结果,直接执行下一步；

打个比方:
A .让黄牛去买票，然后我站着等,一直等到黄牛把票给我;(同步)

B.让黄牛去买票(告诉黄牛,你买到票就call我),然后我就去做别的。(异步)



笔记详解

打字原理

https://segmentfault.com/a/1190000018779991 (这个讲解的不错)

首先要分析打字的原理实现，假设我们定义一个字符串str,它等于一长串注释加CSS代码，并且我们看到，当css代码写完一个分号的时候，它写的样式就会生效。我们知道要想让一段CSS代码在页面生效，只需要将其放在一对 `<style>` 标签对中即可。比如:

代码:

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width">
  <title>JS Bin</title>
</head>
<body>
  红色字体
  <style>
    body{
      color:#f00;
    }
  </style>
</body>
</html>

```

上面的代码具体示例

间歇调用

当看到打字效果的时候，我们不难想到，这是要使用间歇调用(定时函数:setInterval())或超时调用(延迟函数:setTimeout())加递归去模拟实现间歇调用。一个包含一长串代码的字符串，它是一个个截取出来，然后分别写入页面中，在这里，我们需要用到字符串的截取方法，如slice(),substr(),substring()等，选择用哪个截取看个人，不过需要注意它们之间的区别。好了，让我们来实现一个简单的这样打字的效果，如下:

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width">
  <title>JS Bin</title>
</head>
<body>
    <div id="result"></div>
    <script>
        var r = document.getElementById('result');
        var c = 0;
        var code = 'body{background-color:#f00;color:#fff};'
        var timer = setInterval(function(){
          c++;
          r.innerHTML = code.substr(0,c);
          if(c >= code.length){
            clearTimeout(timer);
          }
        },50)
    </script>
</body>
</html>

```

示例代码在这里

让我们来分析一下以上代码的原理,首先放一个用于包含代码显示的标签，然后定义一个包含代码的字符串，接着定义一个初始值为0的变量，为什么要定义这样一个变量呢？我们从实际效果中看到，它是一个字一个字的写入到页面中的。初始值是没有一个字符的，所以，我们就从第0个开始写入，c一个字一个字的加，然后不停的截取字符串，最后渲染到标签的内容当中去，当c的值大于等于了字符串的长度之后，我们需要清除定时器。定时函数看着有些不太好，让我们用超时调用结合递归来实现。

通过调试的方法去理解:

调试代码

单步每过第24行右边就出现一个字符，

如上图,c的长度与code的长度一样长的时候就会清理定时器;
也可以在chrome里面调试(Mac OS: Cmd+Shft+R 为页面刷新)

setInterval函数用法

https://www.jianshu.com/p/6e0fbcf3adc5
setInterval(update,1000);
一秒后执行update函数;

超时调用结合递归来实现

定时函数看着有些不太好，让我们用超时调用结合递归来实现。

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width">
  <title>JS Bin</title>
</head>
<body>
  <div id="result"></div>
  <script>
debugger
     var r = document.getElementById('result');
     var c = 0;
     var code = 'body{background-color:#f00;color:#fff};';
     var timer;
     function write(){
debugger
        c++;
        r.innerHTML = code.substr(0,c);
        if(c >= code.length && timer){
            clearTimeout(timer)
        }else{
           setTimeout(write,50);
       }
    }
    write();
 </script>
```

```
</body>
</html>

```

调试代码 可能需要在关键地方加入debugger保证断点断下来
setTimeout(要执行的代码, 等待的毫秒数)
setTimeout(JavaScript 函数, 等待的毫秒数)
setTimeout
setInterval方法

注意:
setInterval() 方法可按照指定的周期（以毫秒计）来调用函数或计算表达式。
setInterval() 方法会不停地调用函数，直到 clearInterval() 被调用或窗口被关闭。由 setInterval() 返回的 ID 值可用作 clearInterval() 方法的参数。
提示： 1000 毫秒= 1 秒。

好了，到此为止，算是实现了第一步，让我们继续，接下来，我们要让代码保持空白和缩进，这可以使用 `<pre>` 标签来实现，但其实我们还可以使用css代码的 `white-space` 属性来让一个普通的 `div` 标签保持这样的效果，为什么要这样做呢，因为我们还要实现一个功能，就是编辑它里面的代码，可以让它生效。
更改一下代码,如下:

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width">
  <title>JS Bin</title>
  <style>
    #result{
      white-space:pre-wrap;
      oveflow:auto;
    }
  </style>
</head>
<body>
  <div id="result"></div>
  <script>
     var r = document.getElementById('result');
     var c = 0;
     var code = `
        body{
            background-color:#f00;
            color:#fff;
        }
```

```
    `
    var timer;
    function write(){
       c++;
       r.innerHTML = code.substr(0,c);
       if(c >= code.length && timer){
           clearTimeout(timer)
       }else{
          setTimeout(write,50);
      }
   }
   write();
 </script>
</body>
</html>
```

调试代码

操作同上,command+Shift+R刷新页面;加入适当的debugger;

加式样

接下来，我们还要让样式生效，这很简单，将代码在style标签中写一次即可，请看:

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width">
  <title>JS Bin</title>
  <style>
    #result{
      white-space:pre-wrap;
      overflow:auto;
    }
  </style>
</head>
<body>
   <div id="result"></div>
   <style id="myStyle"></style>
   <script>
      var r = document.getElementById('result'),
          t = document.getElementById('myStyle');//多出来的
```

```
      var c = 0;
      var code = `
         body{
            background-color:#f00;
            color:#fff;
         }
      `;
     var timer;
     function write(){
       c++;
       r.innerHTML = code.substr(0,c);
       t.innerHTML = code.substr(0,c);//多出来的
       if(c >= code.length){
         clearTimeout(timer);
       }else{
         setTimeout(write,50);
       }
     }
     write();
   </script>

</body>
</html>

```

主要是多了这两行,与前面相比:

完善会动的简历

```
$cd jrg-system-senior-pratice-41
$ls
01依次展示数字              04封装                    07markdown
02增加css效果               05回调
03完善会动的简历            06自动滚动
$http-server -c -1
Starting up http-server, serving ./
Available on:
  http://127.0.0.1:8080
  http://192.168.0.102:8080
  http://192.168.117.1:8080
  http://172.16.134.1:8080
```

```
46
47 var n = 0
48 var id = setInterval(()=>{
49     n += 1
50     styleTag.innerHTML = result.substring(0,n)
51     code.innerHTML = result.substring(0,n)
52     // 高亮标签—方法1
53     //code.innerHTML = code.innerHTML.replace('html','<span style="color:red;">html</span>')
54     //高亮标签—方法2
55     code.innerHTML =
56         Prism.highlight(code.innerHTML,Prism.languages.css)
57     if(n>=result.length){
58         window.clearInterval(id)
59         fn2()
60         fn3(result)
61     }
62 },10)
63
64 function fn2(){
65     var paper = document.createElement('div')
66     paper.id = 'paper'
67     document.body.appendChild(paper)
68 }
69
70 function fn3(preResult){
71     var result = `
72 #paper{
73     width:100px;height:100px;
74     background:red;
75 }
76     `
77     var n = 0
78     var id = setInterval(()=>{
79         n += 1
80         code.innerHTML = preResult + result.substring(0,n)
81         code.innerHTML = Prism.highlight(code.innerHTML,Prism.languages.css)
82         styleTag.innerHTML = preResult + result.substring(0,n)
83         if(n>=result.length){
84             window.clearInterval(id)
85         }
86
87     })
88 }
```

源代码看这里

上面的图都消失了,看这里的源代码

main.js

```
1 /**
2  * Created by Administrator on 2018/2/4 0004.
3  */
4
5 var result = `/*
6 *面试官你好，我是xxx
```

```
*我将以动画的形式来介绍我自己
*只用文字介绍太单调了
*我就用代码来介绍吧
*首先准备一些样式
*/

*{
    transition:all 1s;
}
html{
    background:rgb(222,222,222);
    font-size:16px;
}
#code{
    border:1px solid red;
    padding:16px;
}

/*我需要一点代码高亮*/

.token.selector{
    color:#690;
}
.token.property{
    color:#905;
}
.token.function{
    color:#DD4A68;
}

/*加点3D效果*/
#code{
    transform:rotate(360deg);
}

/*不玩了，我来介绍一下我自己吧*/
/*我需要一张白纸*/

`

var n = 0
var id = setInterval(()=>{
    n += 1
    styleTag.innerHTML = result.substring(0,n)
    code.innerHTML = result.substring(0,n)
    // 高亮标签——方法1
```

```
    //code.innerHTML = code.innerHTML.replace('html','<span style="color:red;">html</s
    //高亮标签——方法2
    code.innerHTML =
        Prism.highlight(code.innerHTML,Prism.languages.css)
    if(n>=result.length){
        window.clearInterval(id)
        fn2()
        fn3(result)
    }
},10)

function fn2(){
    var paper = document.createElement('div')
    paper.id = 'paper'
    document.body.appendChild(paper)
}

function fn3(preResult){
    var result = `
#paper{
    width:100px;height:100px;
    background:red;
}
    `
    var n = 0
    var id = setInterval(()=>{
        n += 1
        code.innerHTML = preResult + result.substring(0,n)
        code.innerHTML = Prism.highlight(code.innerHTML,Prism.languages.css)
        styleTag.innerHTML = preResult + result.substring(0,n)
        if(n>=result.length){
            window.clearInterval(id)
        }

    })
}
```

下面看有回调的(有异步)

全部源代码