开发笔记

致谢

- Space Galaxy Wallpaper NewTab freeaddon.com)
  一个chrome插件，借用了其源代码
- 枫林晚
  一个hexo主题的部分源码，宇宙星辰背景
- 插图下载网站 多给了几个 因为大多 一天免费下两张
  - pnghost 很强大 英文搜索 需要注册 下载免费 有次数限制
  - pikpng
  - pngdirs
  - cleanpng 免费 无次数限制 好网站 东西多 英文搜索
  - 图精灵
  - 3png
  - 阿里巴巴矢量图标库 丰富，但其实不咋样，好处是免费无广告等等
- 网站js特效来源
  - 素材8
  - bootstrap 模板库
- 抠图网站
  - 佐糖 微信扫码关注微信公众号 免费 人像A抠图还不错

背景音乐

- 想要恋爱的日子
- 告白の夜
- 1945-骆集益
- kiss the rain
- river flows in you
- 卡农
- 梦中的婚礼（钢琴版）
- 给你们, 菊次郎的夏天
- 贝加尔湖畔
- 风中的蒲公英

  注：自动播放失败，考虑到可能确实需要将背景音乐是否播放交给用户

日志

开始日期：2022-8-4

第一阶段：爱一年你四季

一年之中，爱你四季，春夏秋冬，矢志不渝

参考Space Galaxy Wallpaper NewTab(freeaddon.com)

- **飘花特效** 作为 **春天** 使用，**十里春风**，**怦然心动** 相遇 第一次遇见，心中也乐开花
- **下雨特效** 作为 **夏天** 使用，**盛夏相交**，**携风浴雨**　相识 渐渐了解（可能是追求），果实慢慢长大
- **落叶特效**作为 **秋天** 使用，**一叶知秋**，**此心君知**　相知 有了一定的了解，果实成熟
- **下雪特效** 作为 **冬天** 使用，**风花雪月**，**情投意合**　相恋 决定在一起，摘下果实
- **烟花特效** 作为 **庆贺** 使用，**烟火璀璨**，**以告四方**　相守 是祝福也是期盼，进入腹中，消化吸收

注：飘花特效是根据落叶特效修改的，修改使用的图片资源与运动趋势，对于其他源代码进行了参数的调整

关于后面开花结果，模仿自然的没有想好，就没有表现在外面

第二阶段：相守部分的升华

第三季度：添加开幕词

没想好。大概就是献给xx, 或者来句人生格言，献给我十八九岁时的梦想，粒子效果

第四阶段： js文件模块化整合进入index.js

主要是为了展现整个完整的逻辑，以及便于调整参数

开发经验

1. document.getElementById 返回null

描述

语法没有错误，就是找不到东西,document.body 等也会出现问题

解决

把 导入放在后面

原因（猜测）

原因是DOM树里面还没有加载到body部分，document.getElementById放在了标签里面而var 声明变量是系统会优先执行，**结果就是** DOM读取到head时执行了document.getElementById，去找这个在body标签里面的元素

而DOM树这是还没有加载body部分。

js的变量提升什么的，很老火的，尤其是有多个文件的时候。最前面的 var 最好是一些与DOM无关的东西

2. can`t import outside module处理

描述

执行`import {xxx} form “xx.js”`浏览器报错`can not import outside module`

这是在没有使用其他框架情况之下的原生js开发

解决

在把执行了 import 语法的js文件引入html文档时,type为module

`<script type="module" src="/js/index.js"></script>`

注：在含有export的js文档就不能用 导入html

3. JavaScript 实现动作按时间顺序异步执行

使用一个递归算法，我真他妈聪明，

```
// 关键就是把下面这个列表做好，
// 关键是列表里面的 stop start里面的方法内部不能有延时setTimeout
var startStopList=[
    {
        "start":startFlowersCanvas,
        "stop":stopFlowersCanvas,
    },
    {
        "start":startRainCanvas,
        "stop":stopRainCanvas,
    }
]
// 这个index=0 放在方法体前面，很重要
var index = 0
function startStop(startStopList) {
    if (index < startStopList.length) {
        startStopList[index].start()
        setTimeout(function () {
            startStopList[index].stop()
            index++
            startStop(startStopList)// 递归
        }
            , 1e4)  // 延时部分可以优化，在最前面的数组里面加个时间，这里读取

    }
    else {
        console.log("执行完毕")
    }
}
startStop(startStopList)
```

4. 3中方法改良

数据格式——json数组

含有child的很特别，注意其父含有start,子原始的进行应该在父start条件之下执行

```
[
        {
```

```
            "start": startStarAnimation,
            "stop": stopStarAnimation,
            "timeout":1e4
        },
        {
            "start": startButterflies,
            "stop": stopButterflies,
            "timeout":1e4
        },
         {
        "start": createBg,
        "stop": stopFourQuarter, //含有child的很特别，注意其父含有start,子原始的进行应该在父start条件之下执行
        "childs": [
            {
                "start":startFlowersCanvas,
                "stop":stopFlowersCanvas,
                "timeout":1e4
            },
            {
                "start":startRainCanvas,
                "stop":stopRainCanvas,
                "timeout":1e4
            },
            {
                "start":startLeavesCanvas,
                "stop":stopLeavesCanvas,
                "timeout":1e4
            },
            {
                "start": startSnowCanvas,
                "stop": stopSnowCanvas,
                "timeout": 1e4
            },
            {
                "start": startFireworksCanvas,
                "stop": stopFireworksCanvas,
                "timeout": 1e4
            }],
        "timeout": 1e4
         },
            {
            "start": startHeartAnimation,
            "stop": stopHeartAnimation,
            "timeout":1e4
        }

]
```

js源码，在index.js之中

```
import { fourQuarterList } from "./fourQuarter.js"
import { starList } from "./star.js"
import { loveHeartList } from "./loveHeart.js"
import {butterfliesList } from "./butterfly.js"
```

```
var startStopList=starList.concat([butterfliesList[0],fourQuarterList[0],loveHeartList[0]])
console.log(startStopList)

var index = 0
var temp = 0
var sum=0  // 记录总的方法数量
var childListNum=0
var hasChild=false  //标记当前子原始是否是数组
var startStopList_copy = startStopList
// 这是我设计的按照startStopList储存的顺序执行任务的方法
// 最初的测试完成在fourQuarter.js  感兴趣可以去看，在最下面
startStop(startStopList)

function startStop(startStopList) {
    if (index < startStopList.length) {
        startStopList[index].start()
        console.log( startStopList[index].start.name)
        console.log("index: " + index)
        if (startStopList[index].childs) { // 有子任务的处理
            childListNum++;
            hasChild=true;
            temp = index;  // 储存index ,
            console.log(index + "子元素有childs  长度为："+startStopList[index].childs.length)
            index=0 // 刷新index 遍历子数组用
            startStop(startStopList[temp].childs) // 遍历childs
        }
        else {
            sum++
            setTimeout(function () {
                startStopList[index].stop()
                index++
                startStop(startStopList)// 递归
            }
                , startStopList[index].timeout)
        }

    }
    // index超出数组长度，
    //可能是执行完毕，
    //也可能只是一个又child的子数组执行完毕
    else {
        // 子元素的child遍历完成后来到这里
        // 回到父级元素
        if (hasChild===true) {
            index = temp;
            hasChild=!hasChild;
            startStopList_copy[temp].stop()
            console.log("第" + temp + "个元素迭代完毕")
            index++
            if(index<startStopList_copy.length){
                startStop(startStopList_copy)// 递归
            }
```

```
            else{
                startStopList_copy[index].stop()
                console.log("全部执行完毕，共有"+index+" 个子元素     "+sum+" 个元素")
            }

        }
        else {
            startStopList_copy[index].stop()
            console.log("全部执行完毕，共有"+index+" 个子元素     "+sum+" 个元素")
        }


    }
}

```

5. iframe 的调用

iframe 真的是个好东西，可以注意到以下网站

素材8　　bootstrap 模板库

在展示效果时多数都是直接调用的iframe,很方便

参数

1. frameborder:是否显示边框，1(yes),0(no)
2. height:框架作为一个普通元素的高度，建议在使用css设置。
3. width:框架作为一个普通元素的宽度，建议使用css设置。
4. name:框架的名称，window.frames[name]时专用的属性。
5. scrolling:框架的是否滚动。yes,no,auto。 控制滚动条
6. src：内框架的地址，可以使页面地址，也可以是图片的地址。
7. srcdoc , 用来替代原来HTML body里面的内容。但是IE不支持, 不过也没什么卵用
8. sandbox: 对iframe进行一些列限制，IE10+支持 相关权限

sandbox 可选参数与说明

配置	效果
allow-forms	允许进行提交表单
allow-scripts	运行执行脚本
allow-same-origin	允许同域请求,比如ajax,storage
allow-top-navigation	允许iframe能够主导window.top进行页面跳转
allow-popups	允许iframe中弹出新窗口,比如,window.open,target="_blank"
allow-pointer-lock	在iframe中可以锁定鼠标，主要和鼠标锁定有关

iframe 内js

window.parent 获取上一级的window对象，如果还是iframe则是该iframe的window对象
window.top 获取最顶级容器的window对象，即，就是你打开页面的文档
window.self 返回自身window的引用。可以理解 window===window.self(脑残

6. HTML背景音乐，自动播放失败处理，必须用户交互才能播放

百度半天，全是垃圾，都是要自己去点一下，有篇分析不错，参考

原因：谷歌给出的允许自动播放的条件

- 始终允许静音自动播放。
- 在以下情况下，允许自动播放声音：
  1、用户已与域进行了交互（单击，点击等）。
  2、在台式机上，已经超过了用户的“媒体参与度索引”阈值，这意味着该用户以前曾播放带声音的视频。
  3、用户已将该网站添加到他们在移动设备上的主屏幕，或者在桌面上 安装了PWA。
- 顶级框架可以将自动播放权限委派给其iframe，以允许自动播放声音。

个人实测

- 添加 muted　至少对于音频无效，视频好像是可以
- iframe -- 将播放资源到iframe的src中，实测无效

  `<iframe src="/sourece/1945-骆集益" allow="autoplay">`
- iframe —- 将播放资源放到src下的html, 实测无效

```
`   <iframe src="bgmusic.html" allow="autoplay">`
// bgmusic.html
<html>
    <head></head>
<body>
    <audio id="audio">
        <source id="source" src="" type="audio/mp3">
    </audio>
</body>
<script type="text/javascript" src="/js/bgMusic.js"></script>
</html>
```

- window.open -- 通过window.open打开同域名网页可以自动播放，实测无效
- mouseover -- 将audio加上mouseover 鼠标悬停在上方就能自动播放，实测无效

  Google好像非要用户自己点击，虽然可以去修改政策，但是用户谁会去干这个事情
- audioContext 一样无效的

  `The AudioContext was not allowed to start. It must be resumed (or created) after a user gesture on the page`

结论，还是向官方屈服吧，我累了

7. 背景音乐一曲放完歌曲切换

1. 随机从曲库选择一首歌曲
2. 一曲终了，自动播放下一首与当前不同的歌曲
3. 用户选择开始播放后，选悬浮一个小球供用户选择暂停或者下一首

```
var music = ['1945-骆集益.mp3', 'kiss the rain.mp3', 'river flows in you.mp3', '卡农.mp3', '告白の夜.mp3', '梦中的婚礼（钢琴版）.mp3', '给你们.mp3', '菊次郎的夏天.mp3', '贝加尔湖畔.mp3', '风中的蒲公英.mp3']
function getRandomBGM() {
    var playing = music[Math.floor(Math.random() * music.length)]
    console.log("随机选择背景音乐" + playing)
    return "/sourece/"+playing
}
function canpaly() {
    console.log("可以播放" + playing)

}
var playing = getRandomBGM()
var audio = document.createElement("audio");
audio.id="audio";
document.body.appendChild(audio)
audio.setAttribute("src", playing);
audio.setAttribute("loop", '');//  执行  loop  就不会触发ended
audio.setAttribute("autoplay", 'autoplay');
audio.setAttribute("muted", '');
// audio.setAttribute("controls", ''); // 显示控制面板 调试用
audio.setAttribute("preload", 'auto');
audio.addEventListener("canplay", canpaly, false)
audio.onended=function () {
    audio.pause()
    let playing = getRandomBGM()
    audio.setAttribute("src", playing);
    audio.play()
}
audio.addEventListener("ended",
    function () {
        audio.pause()
        let playing = getRandomBGM()
        audio.setAttribute("src", playing);
        audio.play()
    }
    , false)

function playBGM(){
    let audio = document.getElementById("audio");
    if(audio.paused){
        console.log("开始播放")
        audio.play()
    }
    else{
        console.log("暂停播放")
        audio.pause()
    }

```

```
}

function addBGMControler(){
    var addButton=document.getElementById("addBGMControler");
    addButton.parentElement.removeChild(addButton)
    var control_div=document.createElement("div");
    var control_txt=document.createElement("div")
    control_txt.innerText=playing.replace("/sourece/","").replace(".mp3","")
    control_div.appendChild(control_txt)
    control_div.className="BGMcontroler";
    control_div.setAttribute("onclick","playBGM()")
    var control_img=document.createElement("img")
    control_img.id="control_img"
    control_img.src="/img/a-flower.png"
    control_div.appendChild(control_img)
    document.body.appendChild(control_div)

}
```

8。THREE.TextureLoader(); 不支持for 外部嵌套 处理蝴蝶特效

解释 错误示例

后面执行的 load 会覆盖前面的load

```
var load=THREE.TextureLoader();
for(var j=0;j<num;j++){
// 这个j 不会生效 load方法只会记录最后的一个值，前后会覆盖
load("src",function(){
})
}
```

解决方案

先执行 多个 load(src) 将生成的 texture 存到数组 随机悬着一个

```
var butterflies_texture=[]
  for(var image_index=0;image_index<butterfly_imagees.length;image_index++){
    butterflies_texture.push(loader.load(butterfly_imagees[image_index]))
  }
  for (var i = 0; i < BUTTERFLY_NUM; i++) {
    var texture=
butterflies_texture[Math.floor(Math.random()*butterflies_texture.length)]
    texture.magFilter = THREE.NearestFilter;
    texture.minFilter = THREE.NearestFilter;
    butterflies[i] = new Butterfly(i, texture);
    butterflies[i].obj.position.set(((i + 1) % 3 - 1) * i * 50, 0, 1800 /
BUTTERFLY_NUM * i);
    scene.add(butterflies[i].obj);
    console.log(i)
  }
```

9. js打字机 与自动滚动 scrollIntoView（）

1. 打字机

```
var print_speed = 150
//heart_div  dom元素
//heart_txt  txt文本
function printer(heart_div, heart_txt) {
    let n = 0
    var clock = setInterval(() => {
        n += 1
        heart_div.innerHTML = heart_txt.substring(0, n)   // 打字机
        heart_div.scrollIntoView({behavior: "instant", block: "end", inline: "nearest"})  // 实现自动滚动
        if (n >= heart_txt.length) {
            window.clearInterval(clock)
        }
    }, print_speed)

}
```

2. scrollIntoView（） 需要给css 不然不会自动滚动

```
// 隐藏 滚动条
#text_div::-webkit-scrollbar {
  display: none;
}
//  比较重要的是
//   display:flex
//    align-items:flex-end;    对齐位置
#text_div {
  display: flex;
  justify-content: center;
  align-items:flex-end;
  position: fixed;
  top: 50vh;
  left: 50vw;
  transform: translate(-50%, -50%);
  background: transparent;
  overflow-y: scroll;
  text-align: center;
  vertical-align: bottom;
}
```

10.requestAnimationFrame（） 一直在后台跑，移除dom元素不能停止 requestAnimationFrame

描述

偶然发现,因为控制台一直在输出 render 在我把相关canvas 移除之后依然在执行，所以白白消耗算力

```
function render() {
    console.log("render")// 这个输出一直在shu'c，也就是整个方法一直在执行
    var time = clock.getDelta();
    for (var i = 0; i < butterflies.length; i++) {
        butterflies[i].render(renderer, time);
    }
    renderer.render(scene, camera);
};
function renderLoop() {
    render();
    requestAnimationFrame(renderLoop);
};
```

解决

window.cancelAnimationFrame(id); 确实取消了 但是存在滞后，大概几帧吧

```
var  id=null
function render() {
    console.log("render")
    var time = clock.getDelta();
    for (var i = 0; i < butterflies.length; i++) {
        butterflies[i].render(renderer, time);
    }
    renderer.render(scene, camera);
};
function renderLoop() {
    render();
    id= requestAnimationFrame(renderLoop);
};

stop(){// 确实取消了 但是存在滞后，大概几帧吧
    window.cancelAnimationFrame(id)
}
```