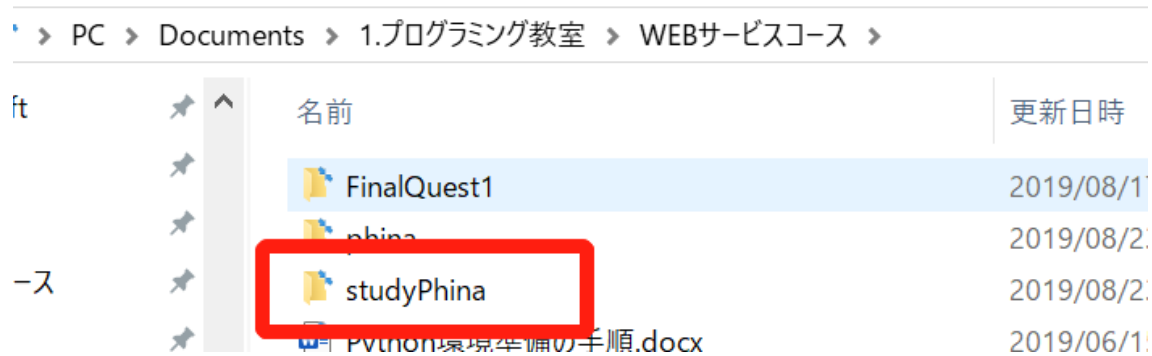
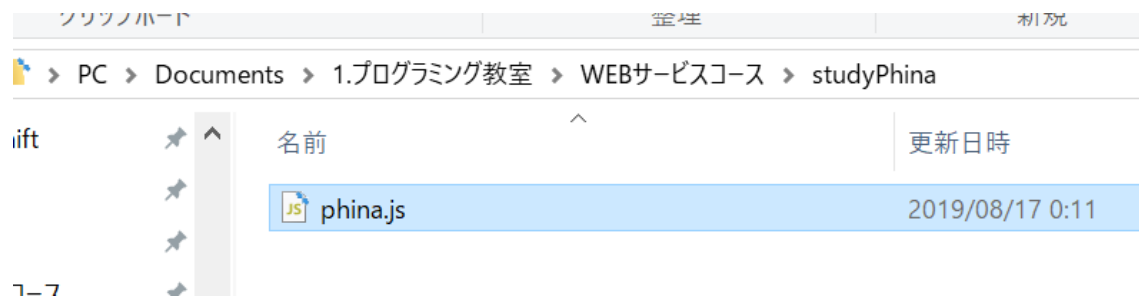


1. 创建一个新文件夹以创建游戏。 **名字和地点都能随意确定**。比如，我创建了一个文件夹，名为 "studyPhina"。



2. 在文件夹中，放置一个名为 "phina.js" 的文件，该文件从老师那里得到。
在这个 phina.js 中，有很多类，你可以很容易地使用它来制作游戏。
例如，重力、点击判断或检测键盘的触摸。



3. 接下来，我们将创建一个 HTML 页面。
phina.js 是在浏览器中运行的游戏。如果没有 HTML，在浏览器中工作不会工作。
只需使用以下源代码即可。逐一阅读它在做什么。

什么是 HTML？波汗"

"在 HTML 中加载 Javascript？波汗"

因此，让我们先完成 w3schools 的 HTML 课程。

https://www.w3schools.com/html/html_scripts.asp

```
<!doctype html>

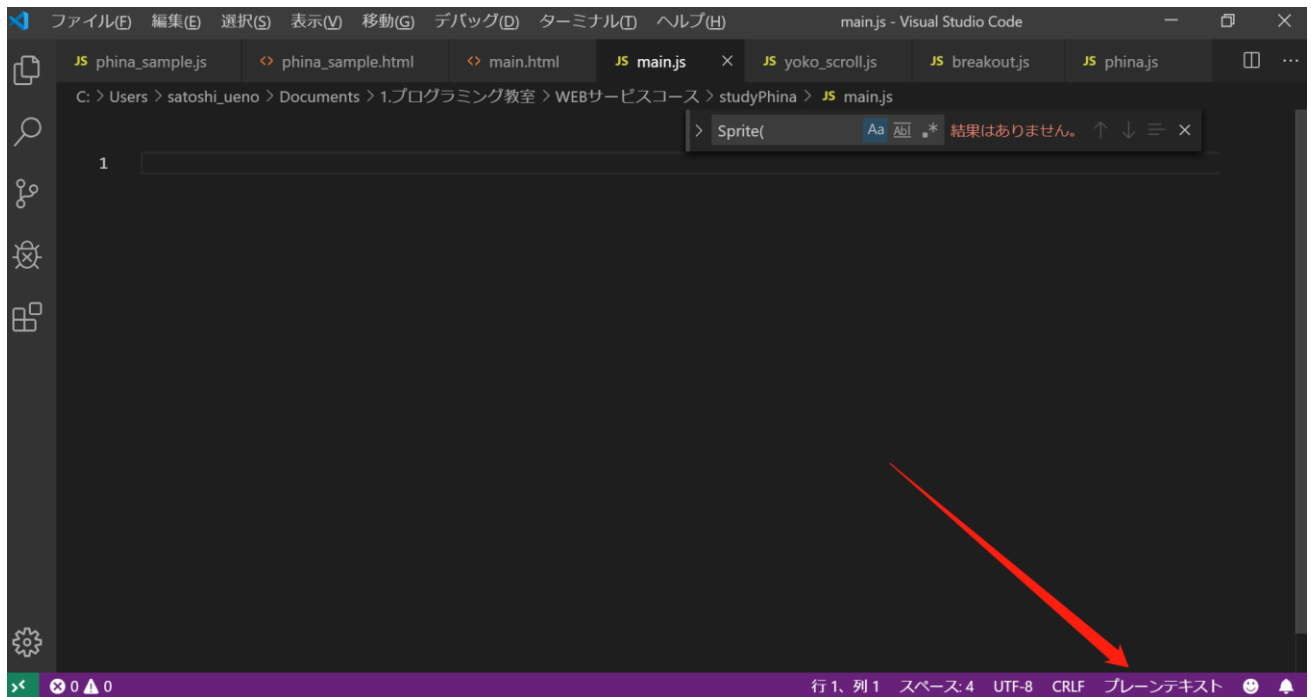
<html>
  <head>
    <meta charset='utf-8' />
    <!-- 创建与屏幕宽度相匹配的窗口。 设置为用户无法缩放 -->
    <meta name="viewport" content="width=device-width, user-scalable=no" />
    <meta name="apple-mobile-web-app-capable" content="yes" />
    <!-- 浏览器标签页中显示的字符 -->
    <title> Let's create a GAME | phina.js</title>

    <!--使用 PC 上的 phina.js -->
    <script src='phina.js'></script>

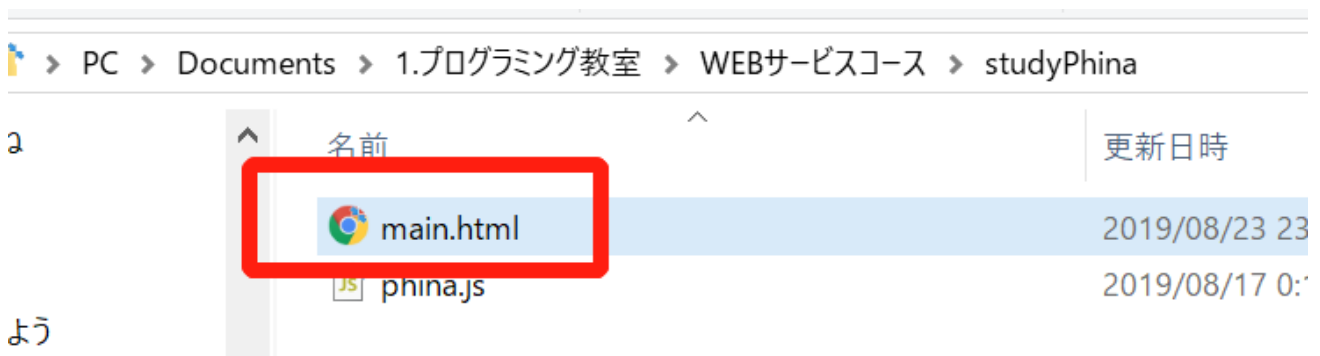
    <!-- 正在加载您在此处制作的 Javascript 文件（我们一会就做 main.js） -->
    <script src='main.js'></script>
  </head>
  <body>
    <!-- HTML 的内容其实没有什么内容 -->
    <!--你做 Javascript，游戏会动!  -->
  </body>
</html>
```

"我的 VSCode 中，并没有颜色…?"

单击 VSCode 右下角的"Plane Text"以切换到 HTML。



4. 完成后，将其保存在文件夹中。






5. 现在，您就准备好了。

基本上，您刚刚生成的 HTML 文件以及 phina.js 文件位于其中，因此您不必再访问该文件。

您只需在 Javascript 文件中编写下一个编程即可。

首先，让我们创建一个空的 Javascript 文件，如下所示：

> PC > Documents > 1.プログラミング教室 > WEBサービスコース > studyPhina		
名前		更新日時
 main.html		2019/08/23
 main.js		2019/08/23
 phina.js		2019/08/17

6. 让我们创建 main.js。

首先，编写最小、最基本的 Javascript 代码。

```
phina.globalize(); // phina.js 现已可用。
```

在 phina 中，我们定义类。

如果你不太了解类，让我们说，"我们在这里做一个屏幕。"

在游戏行业，屏幕称为 Scene。开始屏幕、游戏屏幕、菜单屏幕等

从"整个屏幕的老板"称为 DisplayScene，我创建了一个"游戏屏幕"称为"主场景的孩子"

```
phina.define('MainScene', {
  superClass: 'DisplayScene',
  init: function() { // 这是构造函数。 首先只做一次的过程。
    this.superInit(); // 调用父类 (DisplayScene) 的构造函数。
```

当你做你想做的事时，我会在这里添加它。

```
}
});
```

这是游戏开始的入口。

现在只有一个屏幕叫 GameScene，所以你只需要显示一个屏幕。

```
phina.main(function() {
  var app = GameApp({
    startLabel: 'main' // 各种屏幕时，我们将在此处添加它们。 我们来做吧。
  });
  app.run();
});
```

- 如何定义函数

通常,

```
function init() {}
```

但只要在对象中定义,

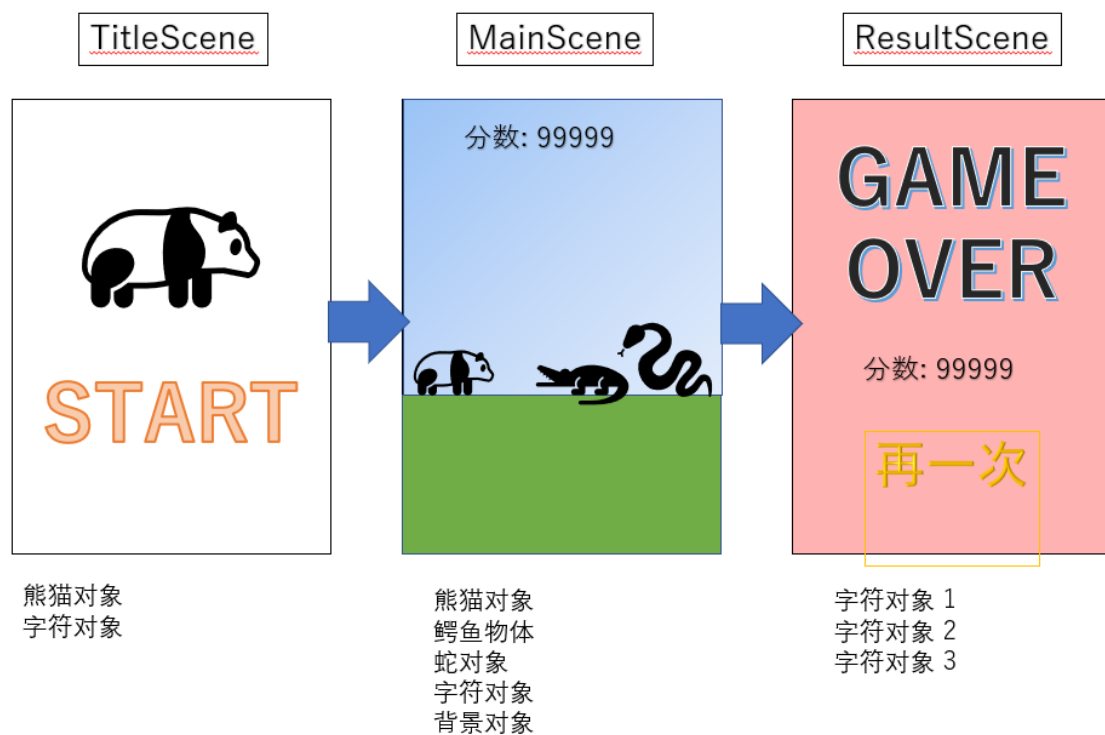
```
init : function() {}
```

看起来像这样。

7. 了解场景和对象之间的关系。

要制作游戏，你需要先做一个场景。

图像，其中每个场景将放置所需的对象。

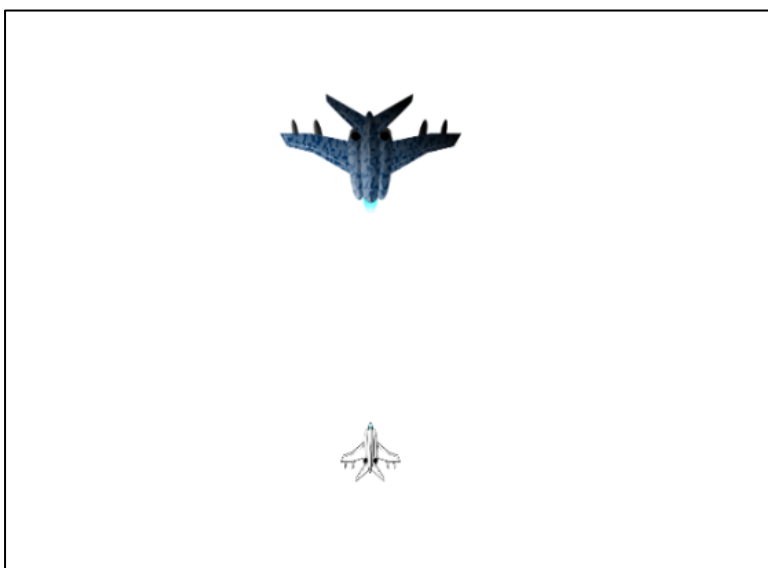


8. 鉴于上述情况，让我们在 MainScene 中放置一个字符对象。

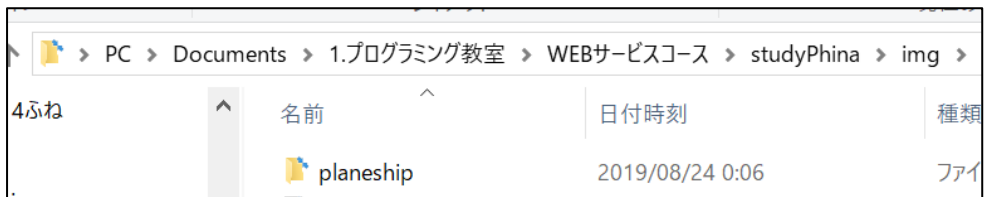
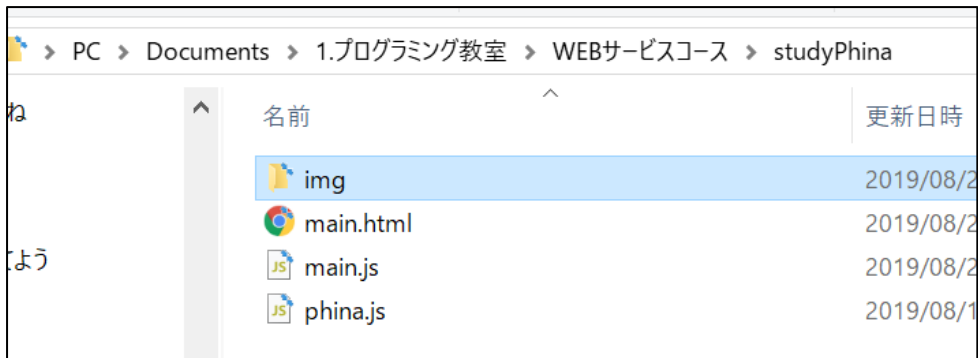
```
phina.define('MainScene', {
  superClass: 'DisplayScene',
  init: function() { // 这是构造函数。 底线是，首先只做一次。
    this.superInit(); // 调用父类（DisplayScene）的构造函数。

    // 或更少的增编
    // 不需要新的
    // 直接编写所需的对象类型将创建该对象。
    // 字符： Label ()， 字符： Sprite ()， 圆形： CircleShape () 等。
    // 最后一个 .addChildTo (this) 是使创建的 Label 对象成为 MainScene 的子级。
    // 首先，有一个屏幕，并把必要的对象在那里。
    var label = Label('Hello, I'm phina.js!').addChildTo(this)
    // 稍后对标签执行各种操作
    // 「.通过连接（点），您可以对对象进行各种连接。
    // 使用 .addChildTo 将坐标放在屏幕上，或在 setPosition 中放置坐标。
    // Javascript 是行的末尾，通过放置";(分号)，能够结束处理
    // 就是说，从“var label =“开始，到” gridY.center();“为止算一行。
    .setPosition(this.gridX.center(), this.gridY.center()); // 设置坐标
  }
});
```

9. 接下来，让我们把这两架飞机，你见过在某个地方。



首先，将两个图像文件放在 studyPhina 文件夹中。



10. 将代码添加到 main.js。

```
phina.globalize();// phina.js 现已可用。
```

```
//像这样指定图像和声音文件。
```

```
let ASSETS = {  
  image:{  
    white:"img/planeship/white.png",  
    black:"img/planeship/black.png"  
  }  
};
```

```
phina.define('MainScene', {
  superClass: 'DisplayScene',
  init: function() { // 这是构造函数。 首先只跑一次。
    this.superInit(); // 调用父类 (DisplayScene) 的构造函数。
```

像这样，我创建一个飞机对象，并把它放在屏幕上。

// 白色飞机

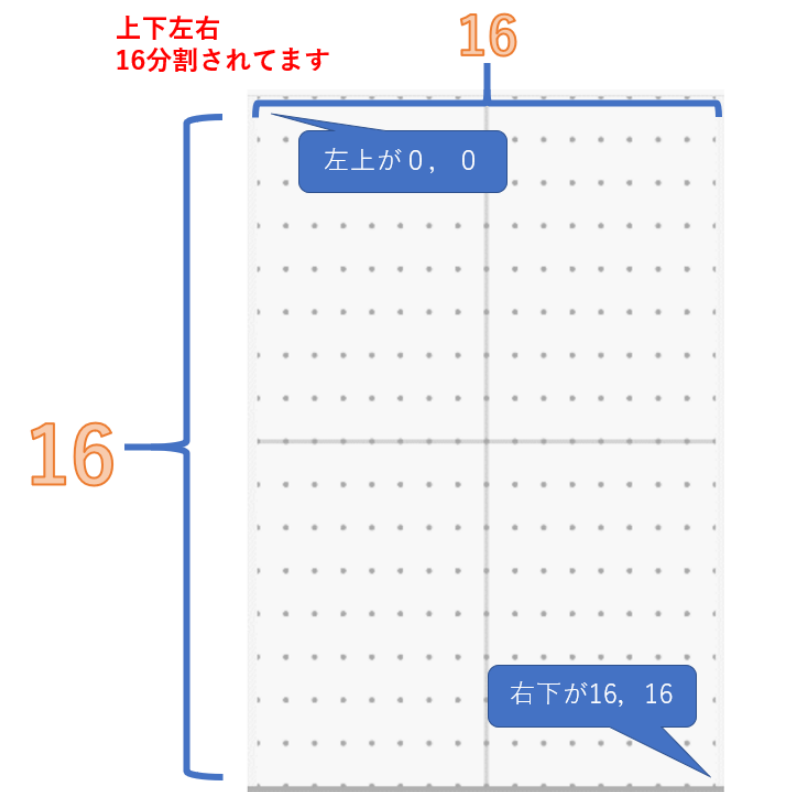
```
let white = Sprite("white", 48, 48).addChildTo(this);
white.x = this.gridX.center();
white.y = this.gridY.center();
```

// 黑色飞机

```
let black = Sprite("black", 146, 96).addChildTo(this);
black.x = this.gridX.center();
black.y = this.gridY.span(4);
black.scaleY *= -1; // 方向的"大小"，则外观将反转。
}
});
```

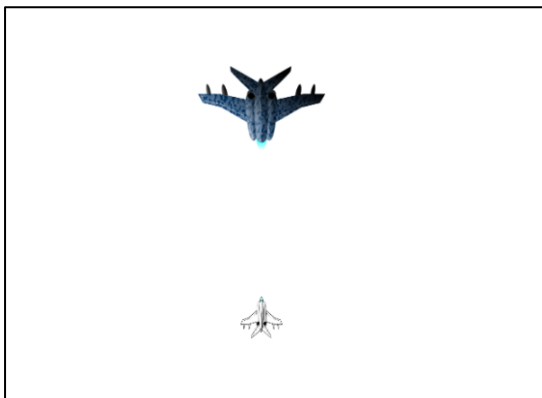
什么是 gridX, gridY? ?

请看下文。事实上，屏幕是以这种方式划分的。此外，




```
phina.main(function() {
  var app = GameApp({
    startLabel: 'main', // 当您想要增加各种屏幕时，我们将在此处添加它们。
    assets: ASSETS,      // 读取顶部指定的图像或
  });
  app.run();
});
```

完成后，保存并在浏览器中打开 main.html，您将看到如下所示。



11. 让我们移动白色飞机。

```
black.scaleY *= -1; // 外观的Y的方向大小逆转就外观也会逆转
}, // 别忘记放这逗号，init 函数和 update 函数是在 MainScene。
// scratch 里的 forever 一样
update:function(app) {
  const key = app.keyboard; // app 是整个游戏，那里面有键盘的状态，叫 keyboard

  if (key.getKey("right")) { // 按了 right 按钮，getKey 会成 True
    white.x += 1;
  }

  if (key.getKey("left")) { // 按了 left 按钮，getKey 会成 True
    white.x += -1;
  }
}
```

这么做还差一步。

让我们记住一个叫做“变量范围”。

在 function 中定义的变量不能在 function 之外使用。

换句话说，白色变量在 `init : function()` 中定义，
在下面的 `update : function()` 中不可用。

那么，我们要做的就是创建一个通用变量，可以在 `init` 和 `update` 中使用，然后通过它就行。

换句话说，就是这样。

```
// 白色飞机
let white = Sprite("white", 48, 48).addChildTo(this);
white.x = this.gridX.center();
white.y = this.gridY.center();
this.white = white; // 把 init 里的 white 变量放在 MainScene 的 white 变量里
```

在 Scratch，这是 forever。

```
update:function(app) {
  let white = this.white; // 把 MainScene 里的 white 变量放在 update 的 white 变量里

  const key = app.keyboard; // app 是整个游戏, 那里面有键盘的状态, 叫 keyboard

  if (key.getKey("right")) { // 按了 right 按钮, getKey 会成 True
    white.x += 1;
  }

  if (key.getKey("left")) { // 按了 left 按钮, getKey 会成 True
    white.x += -1;
  }
}
```

修复后，保存它，并确保白色飞机移动。

12. 挑战挑战

让黑色飞机自动从一边到另一边移动。

[提示 (1)] : phina 坐标为左上角 0, 0。向右移动时, X 越大, 越向下移动, Y 越大。与 Scratch 不同, 请小心。

提示 (2) ;black 的 X 坐标只需更改, 因此让我们更新 black.x 的值。

提示 (3) ; " or " 是在 Javascript 为 " || "