贪吃蛇游戏案例

目标

游戏的目的是用来体会 JavaScript 高级语法的使用，暂时不需要具备抽象对象的能力。

使用面向对象的方式分析问题，需要一个漫长的积累过程。

搭建页面

放一个容器盛放游戏场景 div#map，设置样式

```
#map {
  width: 800px;
  height: 600px;
  background-color: #ccc;
  position: relative;
}
```

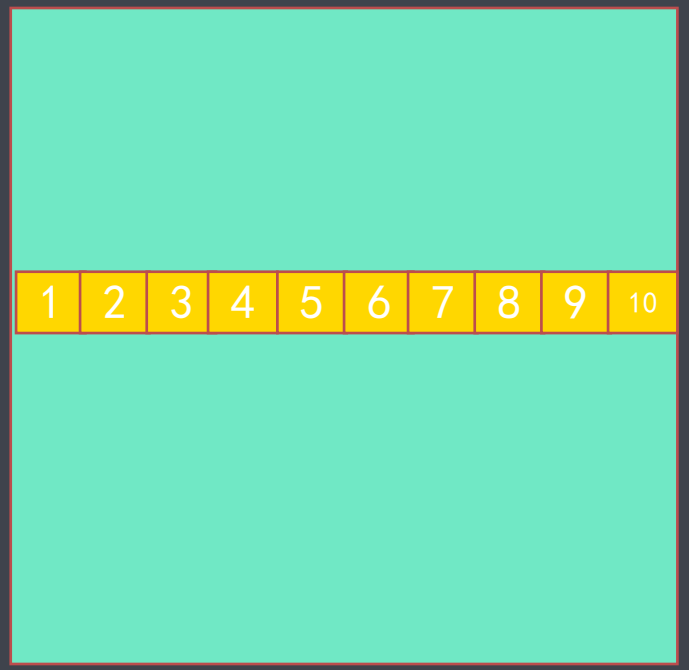
分析对象

- 游戏对象 Game
- 蛇对象 Snake
- 食物对象 Food

创建食物对象

- 创建 Food 的构造函数，并设置属性
  - x
  - y
  - width
  - height
  - color
- 通过原型设置方法
  - render　　随机创建一个食物对象，并输出到 map 上
- 通过自调用函数，进行封装，通过 window 暴露 Food 对象

创建食物对象



创建蛇对象

- 创建 Snake 的构造函数，并设置属性
  - width　　蛇节的宽度 默认20
  - height　　蛇节的高度 默认20
  - body　　数组，蛇的头部和身体，第一个位置是蛇头
  - direction　　蛇运动的方向　默认right　可以是 left　top bottom
- 通过原型设置方法
  - render　　随机创建一个蛇对象，并输出到 map 上
- 通过自调用函数，进行封装，通过 window 暴露 Snake 对象

创建游戏对象

- 创建 Game 的构造函数，并设置属性
  - food
  - snake
  - map
- 通过原型设置方法
  - start　　开始游戏（绘制所有游戏对象，渲染食物对象和蛇对象）
- 通过自调用函数，进行封装，通过 window 暴露 Game 对象

游戏逻辑 1

写蛇的 move 方法

- 在蛇对象（snake.js）中，在 Snake 的原型上新增 move 方法
- 1. 让蛇移动起来，把蛇身体的每一部分往前移动一下
- 2. 蛇头部分根据不同的方向决定 往哪里移动

游戏逻辑 2

让蛇自己动起来

- 在 snake 中添加删除蛇的私有方法，在 render 中调用
- 在 game.js 中添加 runSnake 的私有方法，开启定时器调用蛇的 move 和 render 方法，让蛇动起来
- 判断蛇是否撞墙
- 在 game.js 中添加 bindKey 的私有方法，通过键盘控制蛇的移动方向,在 start 方法中调用 bindKey

游戏逻辑 3

判断蛇是否吃到食物

- 在 Snake 的 move 方法中判断在移动的过程中蛇是否吃到食物
- 如果蛇头和食物的位置重合代表吃到食物
- 食物的坐标是像素，蛇的坐标是几个宽度，需要进行转换
- 吃到食物，往蛇节的最后加一节
- 最后把现在的食物对象删除，并重新随机渲染一个食物对象

其他处理 1

- 把 html 中的 js 代码放到 index.js 中
- 避免 html 中出现 js 代码

其他处理 2

- 自调用函数必须加分号

其他处理 3

自调用函数的参数

- 传入 window 对象：将来代码压缩的时候，可以吧 function（window） 压缩成 function（w）
- 传入 undefined
- 在将来会看到别人写的代码中会把 undefined 作为函数的参数(当前案例没有使用)
- 因为在有的老版本的浏览器中 undefined 可以被重新赋值，防止 undefined 被重新赋值

拉勾教育

—互联网人实战大学—

下载「拉勾教育App」
获取更多内容