设计思路

本项目是一个训练敏捷度的小游戏，玩家需要躲避障碍物收集到更多的宝石。

基本说明：

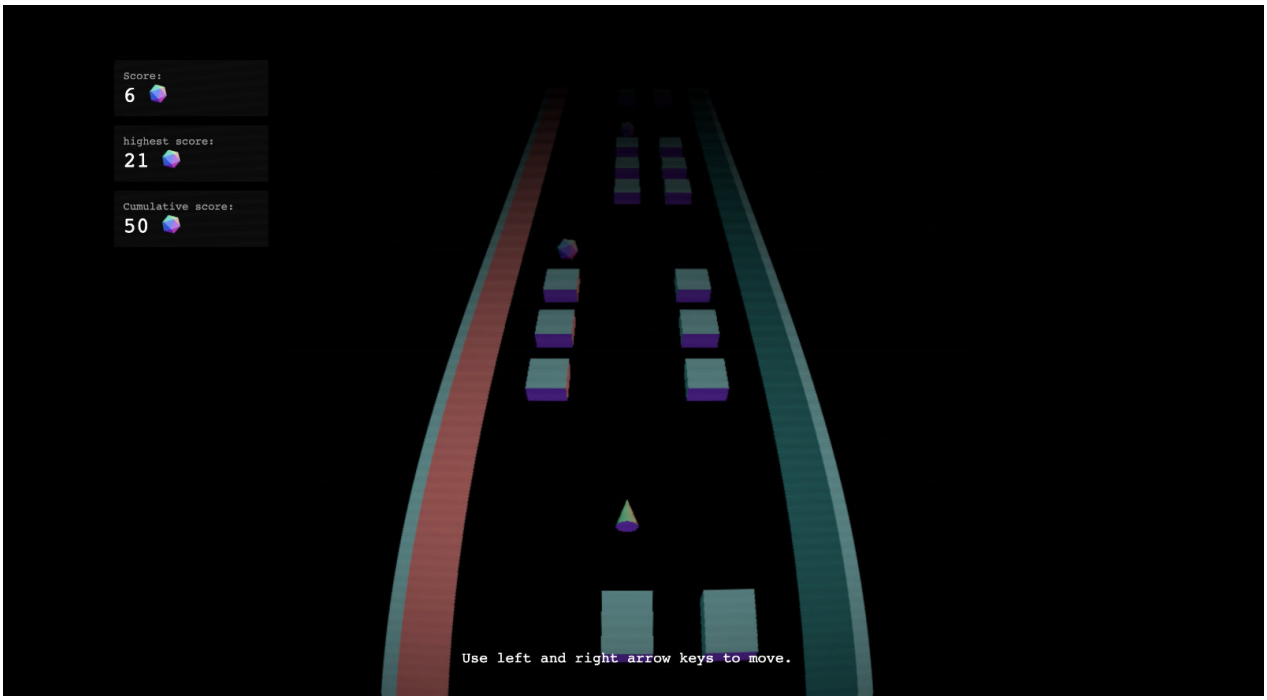
- 进入网站，游戏即开始。
- 玩家通过「敲击键盘上的左右键」，或「点按手机屏幕的左右侧」来操控玩家向左或向右移动。
- 游戏界面左右两侧各设置了一堵高墙，玩家的移动范围只能在两堵墙之间。
- 两墙之间设置了障碍物和宝石，玩家碰到障碍物，游戏即结束。
- 在不碰到障碍物的前提下，玩家需要尽可能多的收集宝石。

附加说明：

- 左侧的三个面板分别表示：「本次游戏得分」「历次最高得分」「历次总计得分」。
- 为了提高游戏的难度，玩家向前移动的速度会越来越快，即留给玩家左右移动的反应时间会越来越短。

功能介绍

1. 玩家通过左右移动来躲避障碍物和收集宝石
2. 左侧面板记录玩家「本次游戏得分」「历次最高得分」「历次总计得分」

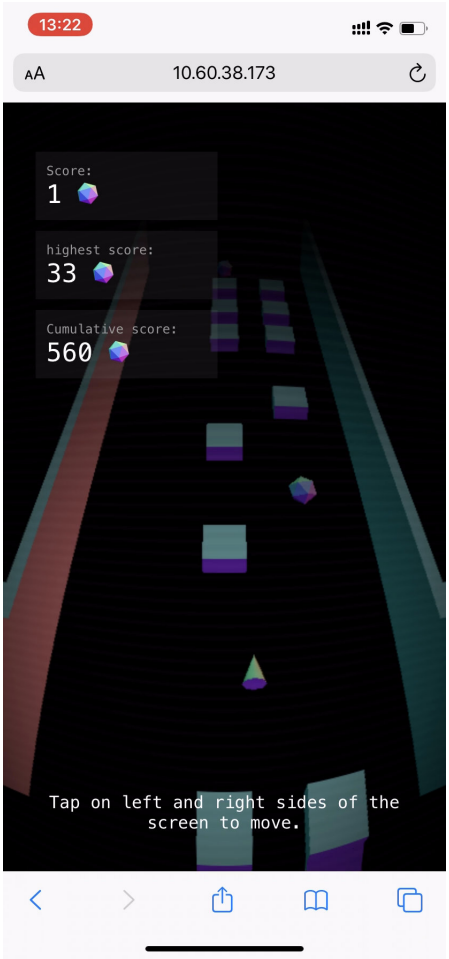


项目特点

1. **同时考虑了网页端和手机端**

   玩家可以分别通过「敲击键盘上的左右键」，或「点按手机屏幕的左右侧」来操控玩家向左或向右移动。

   ```
   window.addEventListener('keydown', (event) => {});
   window.addEventListener('touchstart', (event) => {});
   ```



2. **使用state来管理游戏的状态**

   比「直接修改某些关键变量」要更有组织性和条理性，不会造成意外修改

```
let savedState: SavedState = {};
const initialState: GameState = {};
export default function gameState(state = initialState, action: Action)
{
    const newState = {...state};
    switch (action.type) {
        case ActionType.SET_BEGIN:
            newState.isbegin = action.value as boolean;
            break;
        default:
            return state;
    }
    return newState;
};
```

3. **可扩充性较好**

   游戏当前只有三条可移动的轨道，但通过下面的函数，可以轻松铺设更多、更复杂的路线。

```
//铺设道路的函数
//参数中的'lane'表示轨道，负数为左，正数为右
function spawnWall(camera: PerspectiveCamera, scene: Scene, lane:
number) {
    const geometry = new BoxGeometry(0.5, 0.5, 0.05);
    spawnGeometry(material2, geometry, boxMeshes, camera, scene, lane);
}

//尝试在「左2」和「右1」铺设两个方块
function spawn(camera: PerspectiveCamera, scene: Scene) {
    spawnWall(camera, scene, 1);
    spawnWall(camera, scene, -2);
}
```

本项目只设置了三条轨道是因为时间有限，而设计路线使其不出现死胡同需要耗费大量精力

开发环境

语言：typescript

前端：react+three

部署：nginx+docker