

游戏主要功能实现

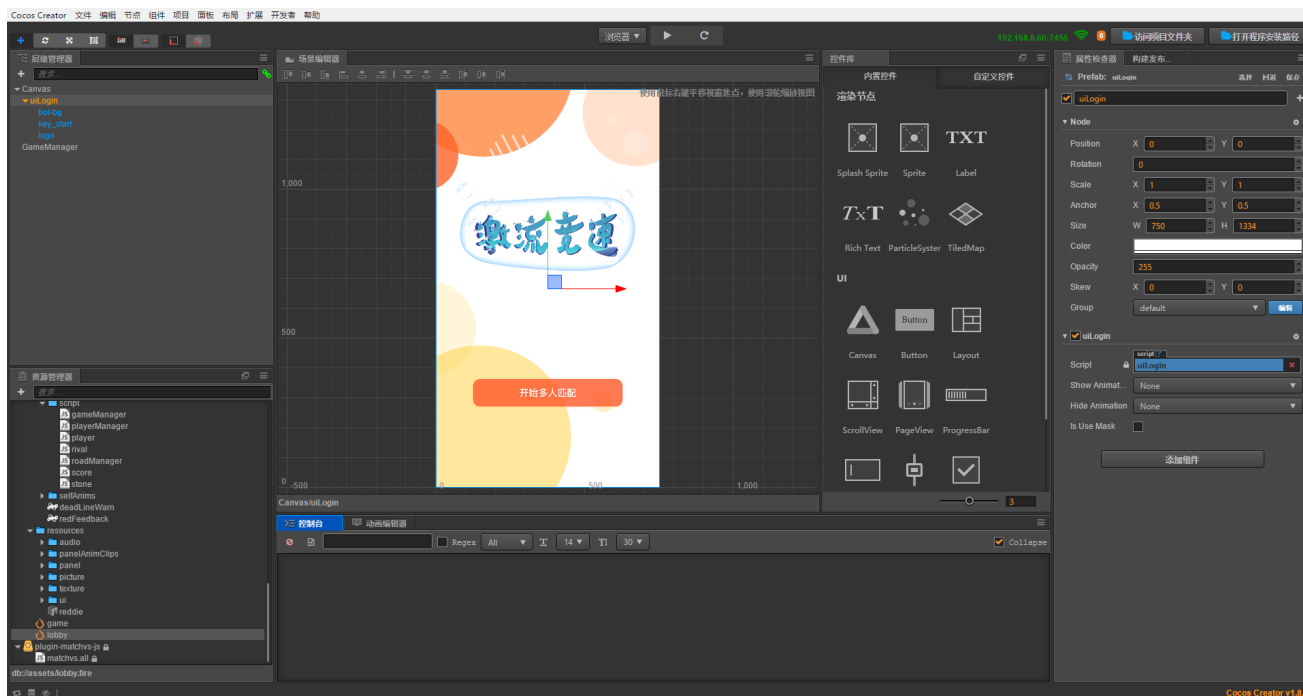
设计游戏实现可拆分为用户登录、随机匹配和创建房间与同屏游戏三个部分。

用户登录

使用Cocos Creator(以下简称CC)创建游戏登录场景

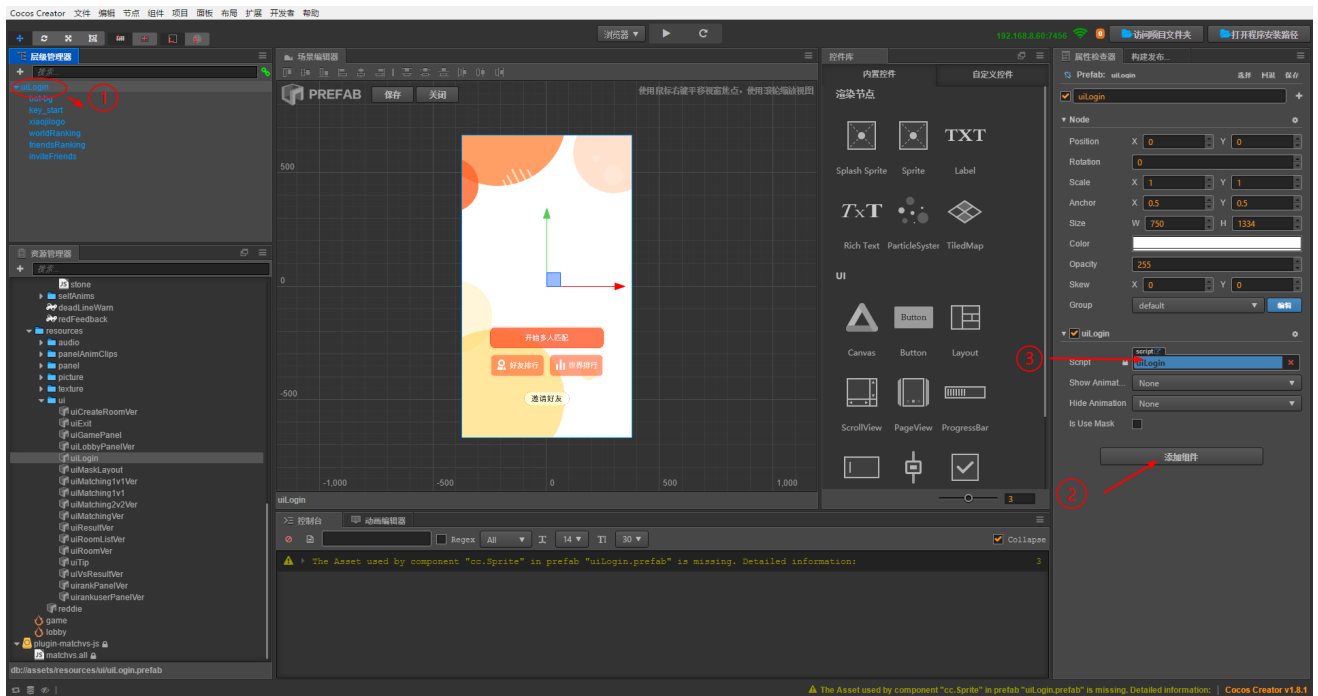
使用CC 拖动控件, 还原设计稿, 依托CC的良好的工作流,使得这部分的工作可以由游戏策划或者UI设计者来完成,程序开发者只需要在场景中挂载相应的游戏逻辑脚本. 举个例子,在登录按钮挂在一个 `uiLogin.js` 的脚本完成用户登录功能.

uiLogin.fire



1. 新建js脚本文件
2. 选中场景任一控件
3. 添加组件,选中刚新建的脚本,
4. 在脚本的 `on'Load` 函数中给按钮添加点击监听,触发登录操作

```
onLoad() {  
    this.nodeDict["start"].on("click", this.startGame, this);  
},  
startGame() {  
    Game.GameManager.matchVsInit();  
}
```



实现 `this.startGame` 函数. 登录之前需要初始化 Matchvs SDK :

uiLogin.js

```
uiLogin.js
---
var uiPanel = require("uiPanel");
cc.Class({
    extends: uiPanel,
    properties: {},

    onLoad() {
        this._super();
        this.nodeDict["start"].on("click", this.startGame, this);
    },

    startGame() {
        Game.GameManager.matchVsInit();
    }
});
```

Game.GameManager.js

```
-----
matchVsInit: function() {
    mvs.response.initResponse = this.initResponse.bind(this);
    mvs.response.errorResponse = this.errorResponse.bind(this);
    // 用户登录之后的回调
    mvs.response.loginResponse = this.loginResponse.bind(this);

    var result = mvs.engine.init(mvs.response, GLB.channel, GLB.platform, GLB.gameId);
```

```

        if (result !== 0) {
            console.log('初始化失败,错误码:' + result);
        }
    }
}

```

初始化需要的几个参数在Matchvs官网注册即可得到,注册地址 <http://www.matchvs.com>

```

channel: 'MatchVS',
platform: 'alpha',
gameId: 201372,
gameVersion: 1,
appKey: '9a93e36777004e98905a7e66d0808a42',
secret: 'daf7049c6e3e48e08bfe90abd5b77b0e',

```

登录Matchvs游戏云,返回UserID,登录成功.

```

registerUserResponse: function(userInfo) {
    var deviceId = 'abcdef';
    var gatewayId = 0;
    GLB.userInfo = userInfo;

    console.log('开始登录,userId:' + userInfo.id)

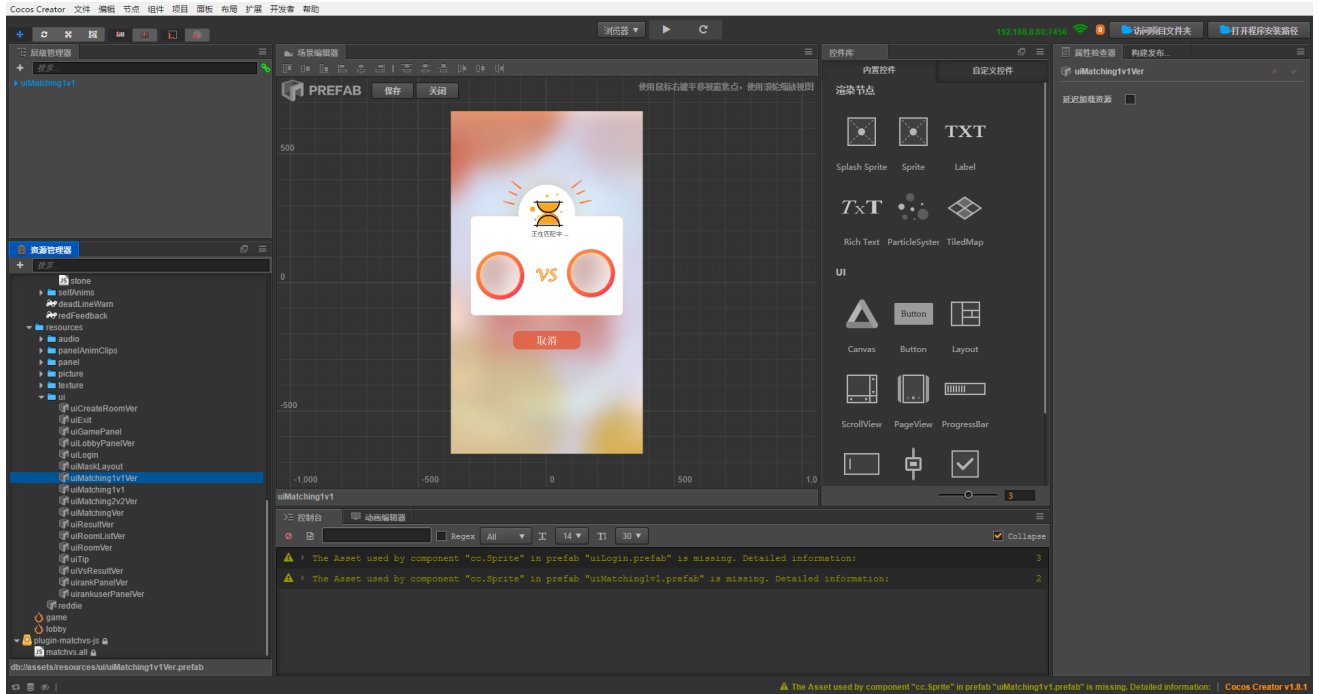
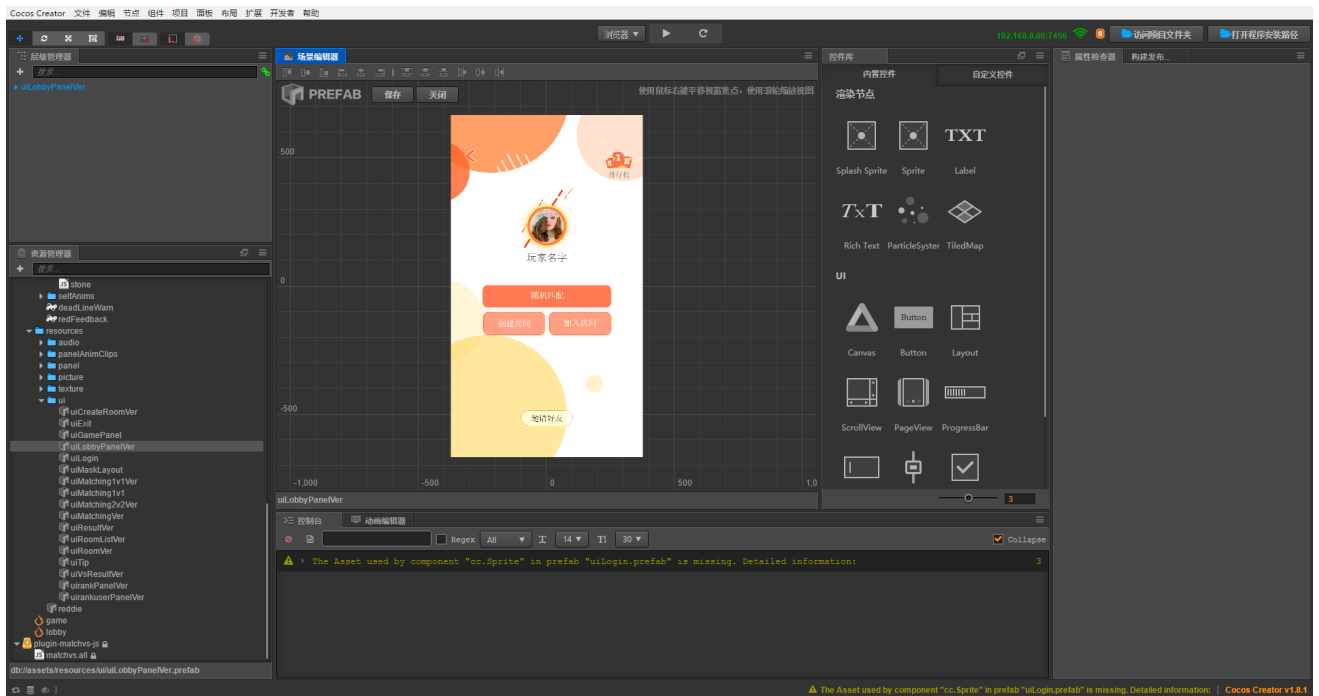
    var result = mvs.engine.login(
        userInfo.id, userInfo.token,
        GLB.gameId, GLB.gameVersion,
        GLB.appKey, GLB.secret,
        deviceId, gatewayId
    );
    if (result !== 0) {
        console.log('登录失败,错误码:' + result);
    }
},

loginResponse: function(info) {
    if (info.status !== 200) {
        console.log('登录失败,异步回调错误码:' + info.status);
    } else {
        console.log('登录成功');
        this.lobbyShow();
    }
},

```

随机匹配和创建房间

使用CC创建大厅场景(uiLobbyPanel.fire)给用户选择匹配方式,创建匹配场景(uiMatching1v1.fire) 给用户反馈匹配进度



和登录功能的实现步骤类似:写一个 `uiMatching1v1.js` 脚本挂在到场景中的控件上.

`uiMatching1v1.js`

```

-----
joinRandomRoom: function() {
    var result = mvs.engine.joinRandomRoom(GLB.MAX_PLAYER_COUNT, '');
    if (result !== 0) {
        console.log('进入房间失败,错误码:' + result);
    }
},

```

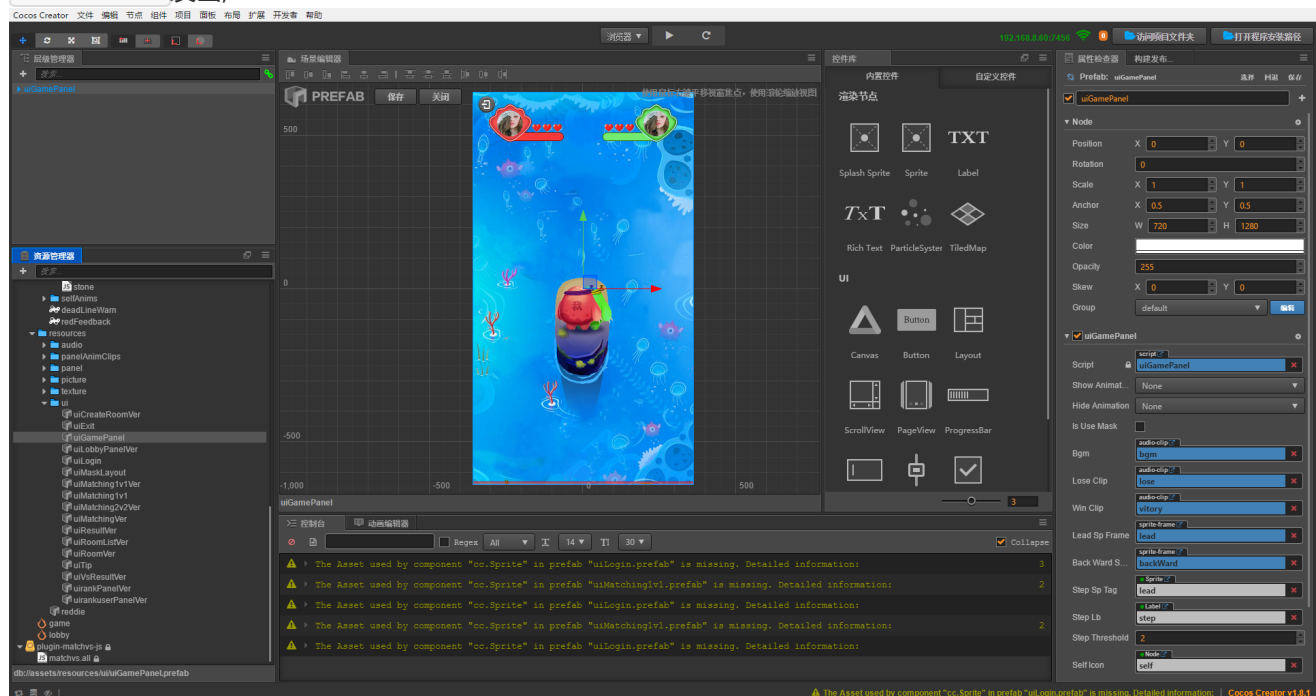
通过监听 `joinRoomResponse` 和 `joinRoomNotify` 匹配结果

```
gameManager.js
---
joinRoomResponse: function(status, roomUserInfoList, roomInfo) {
    if (status !== 200) {
        console.log("失败 joinRoomResponse:" + status);
        return;
    }
    var data = {
        status: status,
        roomUserInfoList: roomUserInfoList,
        roomInfo: roomInfo
    }
    clientEvent.dispatch(clientEvent.eventType.joinRoomResponse, data);
},

joinRoomNotify: function(roomUserInfo) {
    var data = {
        roomUserInfo: roomUserInfo
    }
    clientEvent.dispatch(clientEvent.eventType.joinRoomNotify, data);
},
}
```

同屏游戏，实现游戏同步

还是按照上面的套路,新建场景(uiGamePanel.fire),挂在脚本(uiGamePanel.js). 攻击的动作使用Matchvs 的 sendEventEx 发出,



```
onClick: function() {
    if (Game.GameManager.gameState !== GameState.Play) {
        return;
    }

    if (this.data.ID === Game.PlayerManager.player.jumpRecordId) {
```

```

var exist = Game.PlayerManager.player.jumpPos.some(function (x) {
    return this.data === x;
}).bind(this));
if (!exist) {
    cc.audioEngine.play(this.clickAudio, false, 1);
    this.stoneSp.spriteFrame = this.clickSf;
    var msg = {
        action: GLB.PLAYER_STEP_DATA,
        data: this.data
    };
    Game.GameManager.sendEventEx(msg);
}
},

```

另一方的客户端收到后处理加分,播放击中动画等事情;

```

// 玩家行为通知--
sendEventNotify: function(info) {
    var cpProto = JSON.parse(info.cpProto);
    if (info.cpProto.indexOf(GLB.GAME_START_EVENT) >= 0) {
        GLB.playerUserIds = [GLB.userInfo.id]
        var remoteUserIds = JSON.parse(info.cpProto).userIds;
        remoteUserIds.forEach(function(id) {
            if (GLB.userInfo.id !== id) {
                GLB.playerUserIds.push(id);
            }
        });
        this.startGame();
    }
}

```

开发完成后, 再通过CC的微信小游戏一键发布功能上线微信。

