设计游戏实现步骤

拆分主要功能:

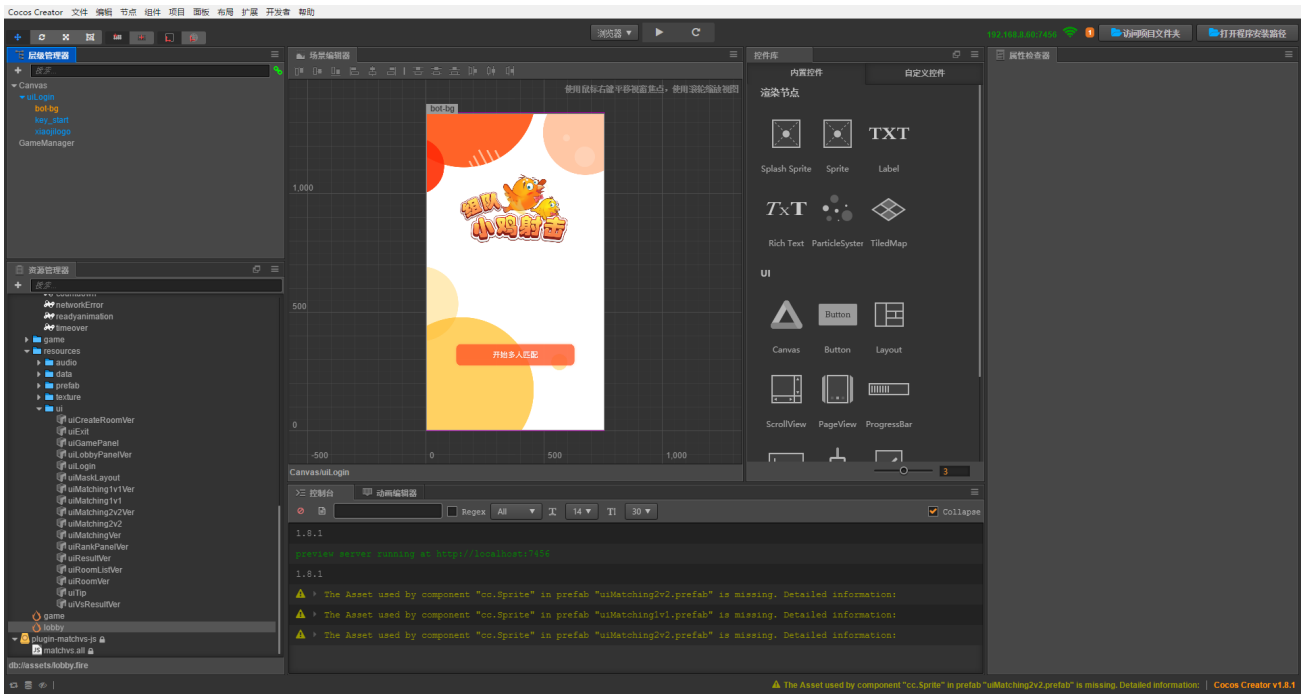
1. 用户登录
2. 随机匹配和创建房间
3. 同屏游戏

用户登录

使用Cocos Creator(以下简称CC)创建游戏登录场景

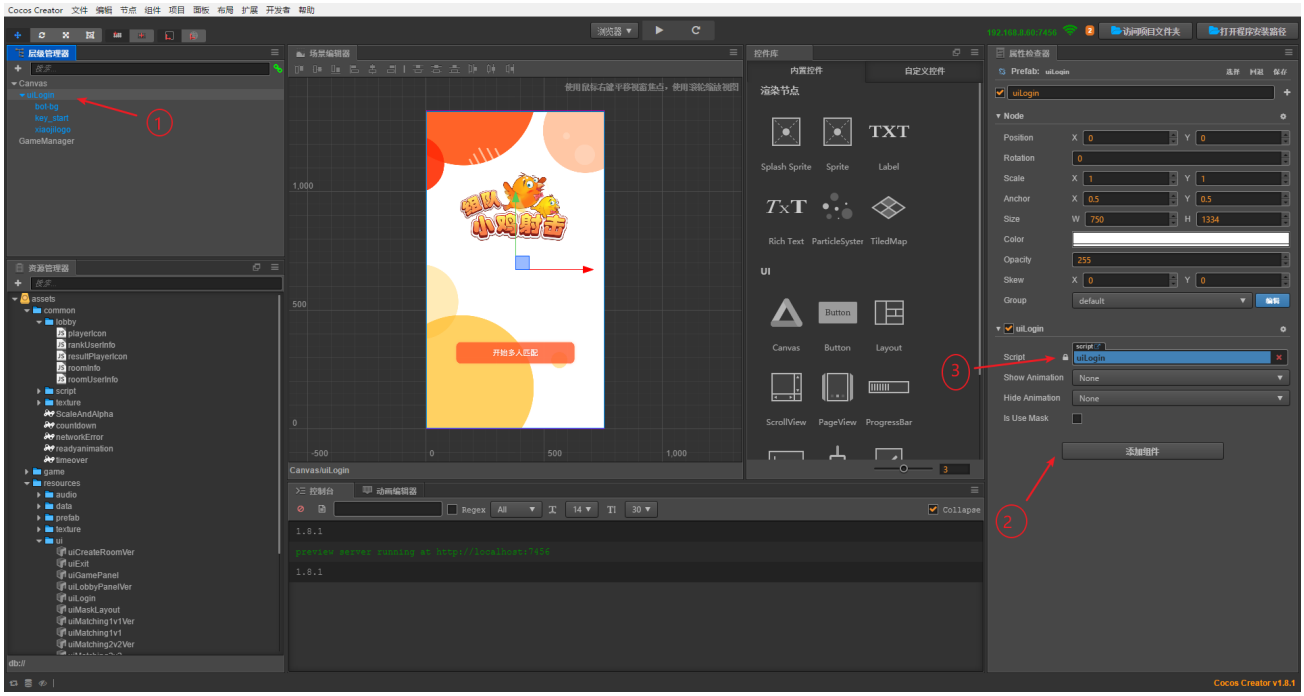
使用CC 拖动控件, 还原设计稿 , 依托CC的良好的工作流,使得这部分的工作可以由游戏策划或者UI设计者来完成,程序开发者只需要在场景中挂载相应的游戏逻辑脚本. 举个例子,在登录按钮挂在一个`uiLogin.js`的脚本完成用户登录功能.

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`onLoad`函数中给按钮添加点击监听,触发登录操作

```
uiLogin.js
-----

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

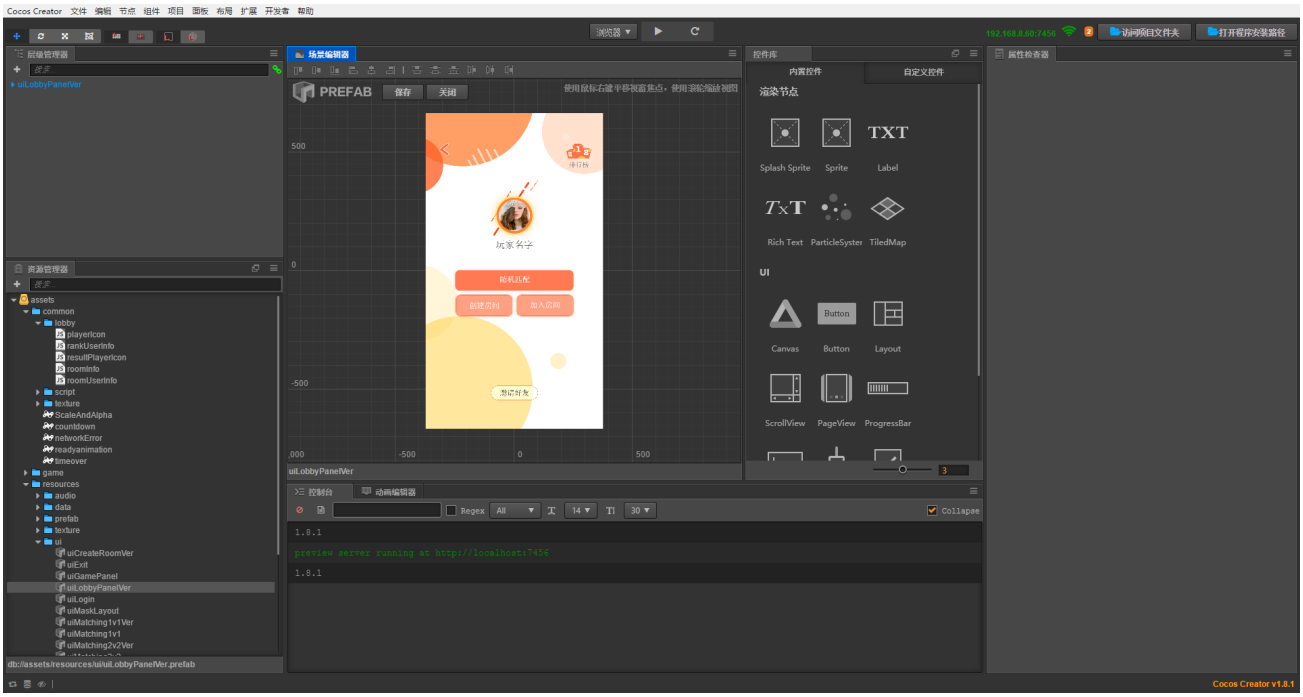
```
    channel: 'MatchVS',
    platform: 'alpha',
    gameId: 201330,
    gameVersion: 1,
    appKey: '7c7b185482d8444bb98bc93c7a65daaa',
    secret: 'f469fb05eee9488bb32adfd85e4ca370',
```

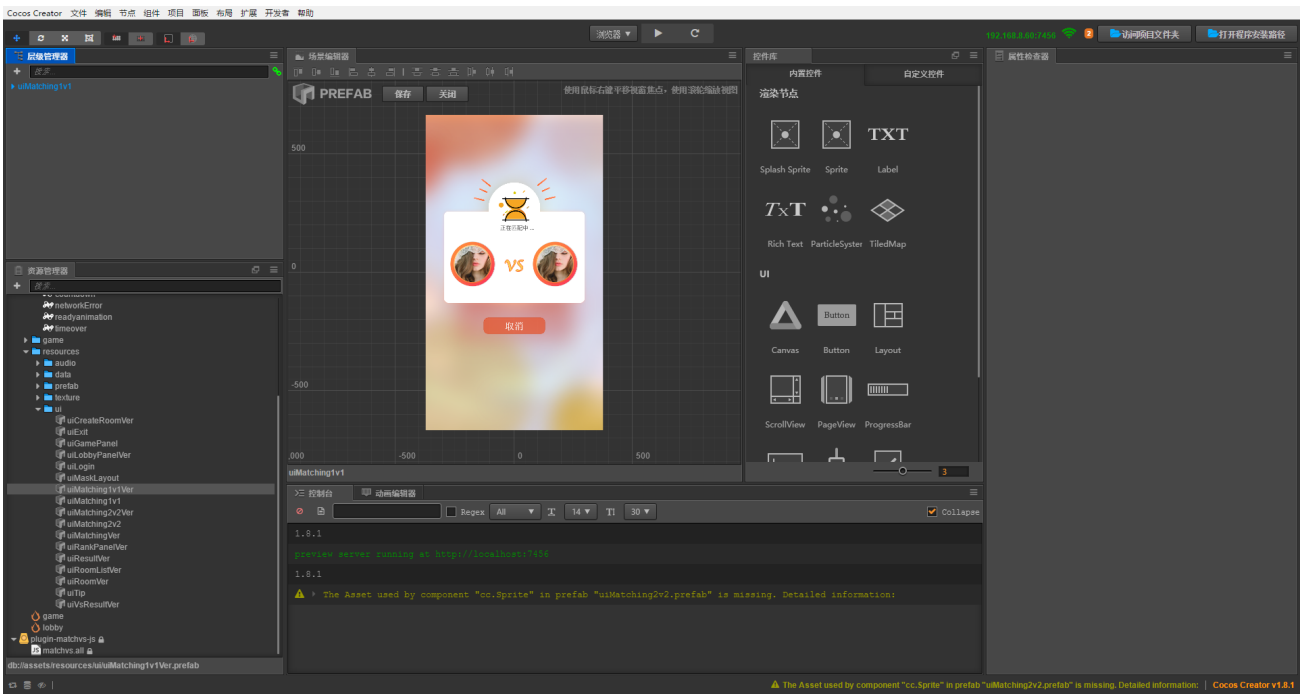
注册成功后,登录Matchvs游戏云,返回UserID,登录成功.

```
gameManager.js
-----

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

    console.log('开始登录,用户Id:' + 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) 给用户反馈比配进度





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.

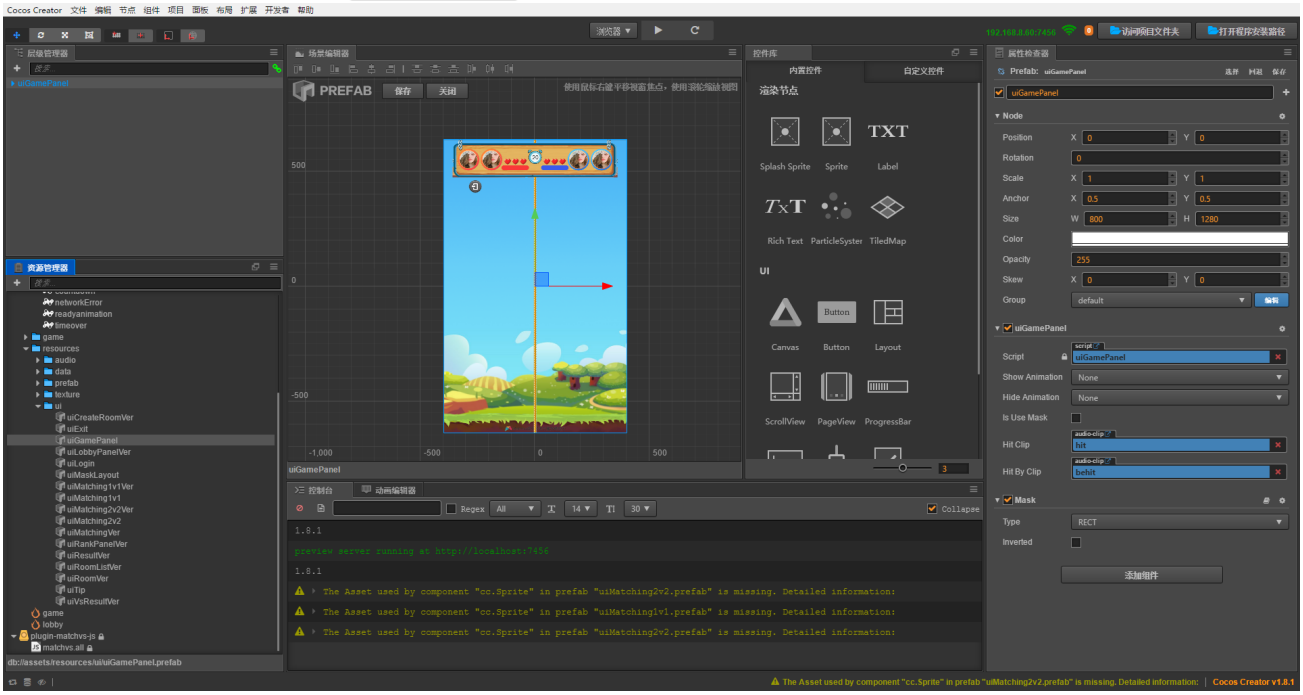
```
uiMatching1v1.js
-----

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
    }
    // 把事件发给关心这个事件的节点脚本
    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),在playerManager.js中,加载了player.js.在player.js中,攻击的动作使用Matchvs 的 `sendEventEx`发出,

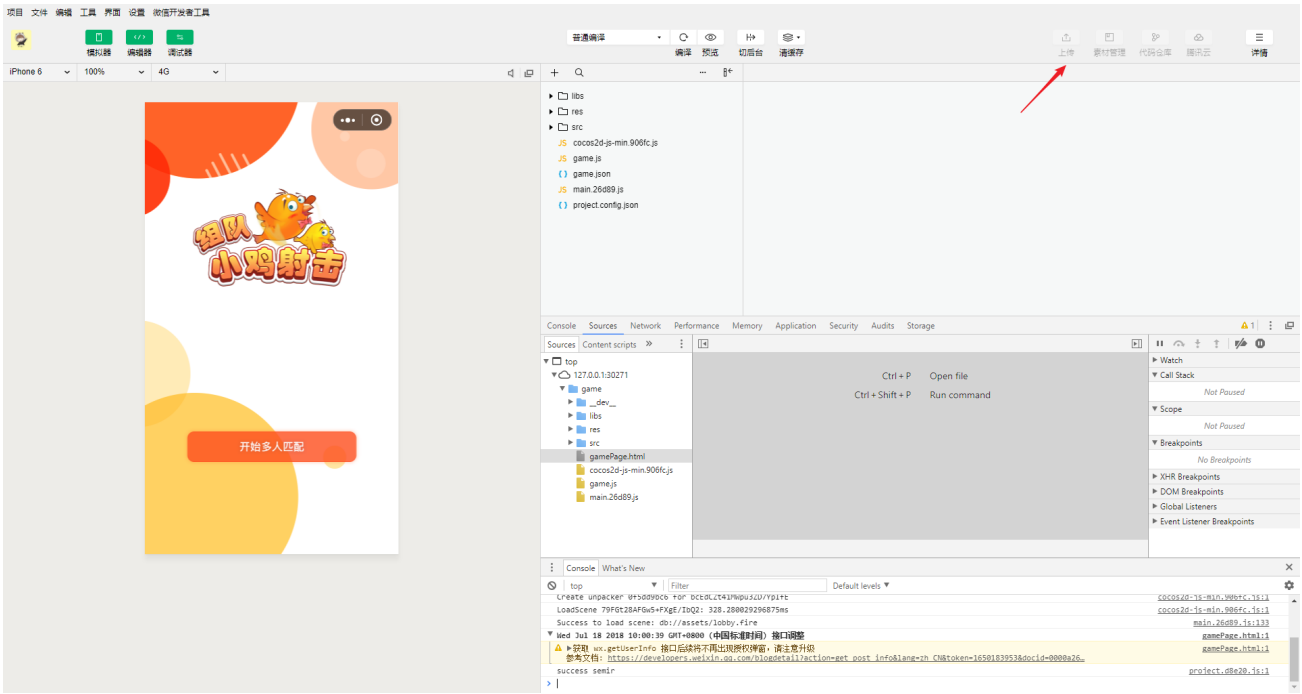


```
player.js
-----

hurt: function(murderId) {
    var msg = {
        action: GLB.PLAYER_HURT_EVENT,
        playerId: this.userId,
        murderId: murderId
    };
    Game.GameManager.sendEventEx(msg);
}
```

另一方的客户端收到后处理事情;

```
gameManager.js
-----

 // 玩家行为通知--
sendEventNotify: function(info) {
    if (info.cpProto.indexOf(GLB.PLAYER_HURT_EVENT) >= 0) {
        if (Game.GameManager.gameState !== GameState.Over) {
            player = Game.PlayerManager.getPlayerByUserId(cpProto.playerId);
            if (player) {
                player.hurtNotify(cpProto.murderId);
            }
            // 检查回合结束--
            var loseCamp = Game.PlayerManager.getLoseCamp();
            if (loseCamp != null) {
                Game.GameManager.gameState = GameState.Over
                if (GLB.isRoomOwner) {
                    this.sendRoundOverMsg(loseCamp);
                }
            }
        }
    }
}
```

开发完成后， 再通过CC的一键发布微信小游戏功能上线微信。


项目 文件 编辑 工具 界面 设置 微信开发者工具
模拟器 编辑器 调试器
普通编译 编译 预览 切后台 清缓存
上传 素材管理 代码仓库 腾讯云 详情
iPhone 6 100% 4G
组队小鸡射击
开始多人匹配
libs
res
src
cocos2d-js-min.906fc.js
game.js
game.json
main.26d89.js
project.config.json
Console Sources Network Performance Memory Application Security Audits Storage
Sources Content scripts
top
127.0.0.1:30271
game
__dev__
libs
res
src
gamePage.html
cocos2d-js-min.906fc.js
game.js
main.26d89.js
Ctrl + P Open file
Ctrl + Shift + P Run command
Watch
Call Stack
Not Paused
Scope
Not Paused
Breakpoints
No Breakpoints
XHR Breakpoints
DOM Breakpoints
Global Listeners
Event Listener Breakpoints
Console What's New
top
Filter
Default levels
LoadScene 79FGt28AFGw5+FXgE/IbQI: 328.280029296875ms
Success to load scene: db://assets/lobby.fire
Wed Jul 18 2018 10:00:39 GMT+0800 (中国标准时间) 接口调整
▶获取 wx.getUserInfo 接口后续将不再出现授权弹窗，请注意升级
参考文档: https://developers.weixin.qq.com/blogdetail?action=get_post_info&lang=zh_CN&token=1650183953&docid=0000a26…
success semir