浙江大学

本科实验报告

课程名称:	BS体系软件设计
姓　名:	葛现隆
学　院:	计算机学院
系:	计算机系
专　业:	计算机专业
学　号:	3120102146
指导教师:	胡晓军

2015 年 6 月 28 日

浙江大学实验报告

课程名称：__BS体系软件设计__实验类型：__网页对战游戏设计__

实验项目名称：__HTML5实现一个网页对战游戏__

学生姓名：__葛现隆__专业：__计算机专业__学号：__3120102146__

同组学生姓名：__无__指导老师：__胡晓军__

实验地点：__玉泉曹光彪西-503__实验日期：__2015__年__6__月__26__日

设计报告

一、游戏基本介绍

1. 游戏名称：Patricia
2. 游戏类型：web在线双人对战游戏
3. 游戏引擎：cocos2d-js
4. 开发语言：js, html5
5. 开发IDE：WebStrom
6. 数据库搭建：SQLite
7. 游戏简介：
   Patricia是一款Web在线双人卡通对战游戏，玩家可通过访问页面，实现注册账号，角色选择，自动匹配对战，对战排行榜查询等功能；

二、游戏场景设计

游戏共有5个场景(Scene)，分别是：GameScene(包含登录界面)，SelectScene(角色选择), RankScene(排行), PlayScene(玩家对战);

各Scene下都有对应的Layer，Layer又由不同的Sprite和其他元素组成，具体如下：

GameScene/Layer:

由Account UI和Background两部分组成，其中Account UI包括了Name(用户名输入)，Password(密码输入)，Confirm(密码确认)，Login Button(登录按钮)组成；

SelectScene/Layer

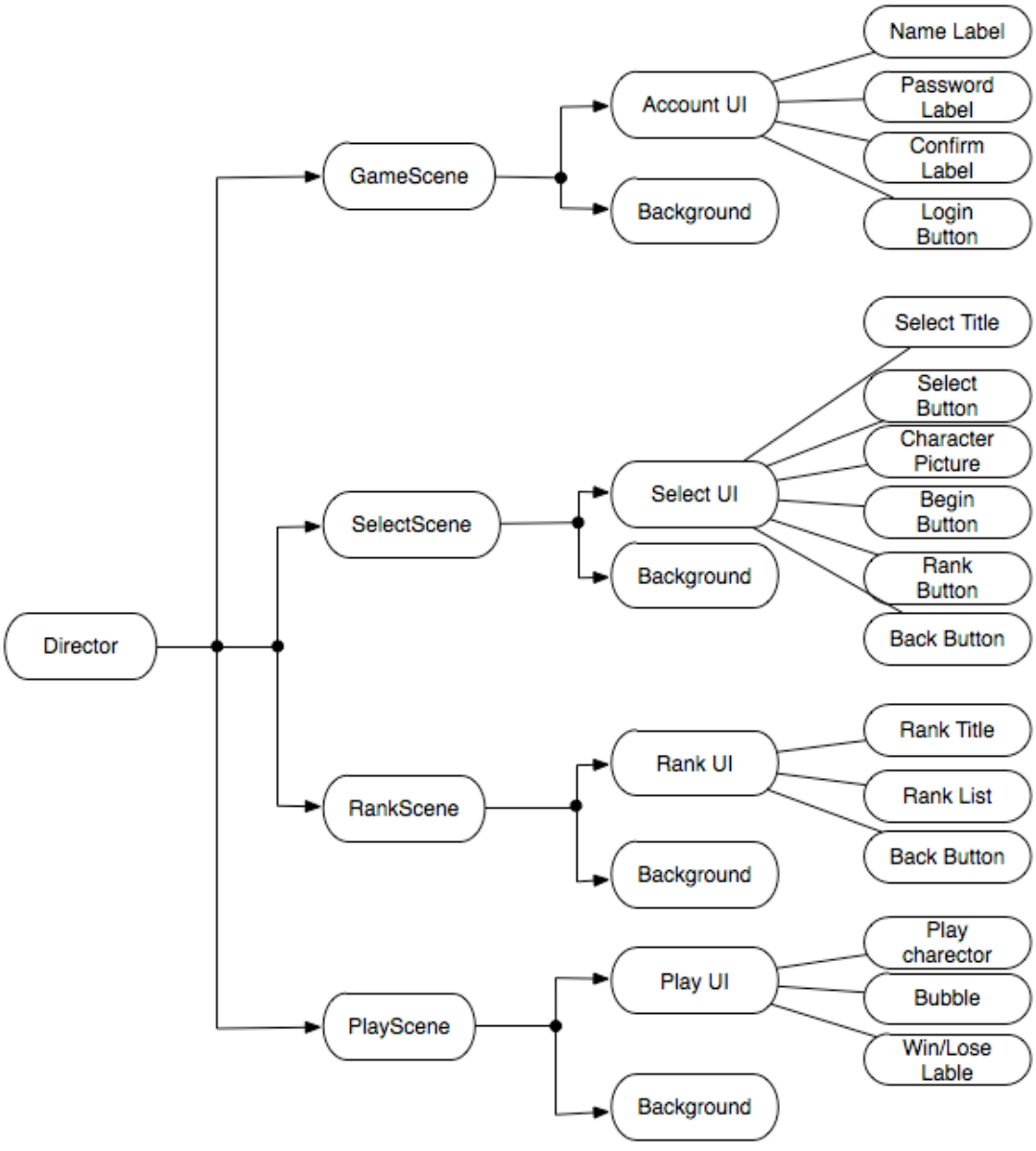
由Select UI和Background两部分组成，其中Select UI包括了(Title)标题、(Select Button)选择角色按钮、(Character Picture)角色图片、(Begin Button)开始按钮、(Rank Button)排行按钮、(Back Button)返回按钮组成；

RankScene/Layer

由Rank UI和Background两部分组成，其中Rank UI包括了(Title)标题、(Rank List)排行列表、(Back Button)返回按钮组成；

PlayScene/Layer

由Play UI和Background两部分组成，其中Play UI包括了(Play character)双方玩家角色、(Bubble)子弹/气泡、(Win/Lose Label)赢/输结果组成；



三、游戏流程设计

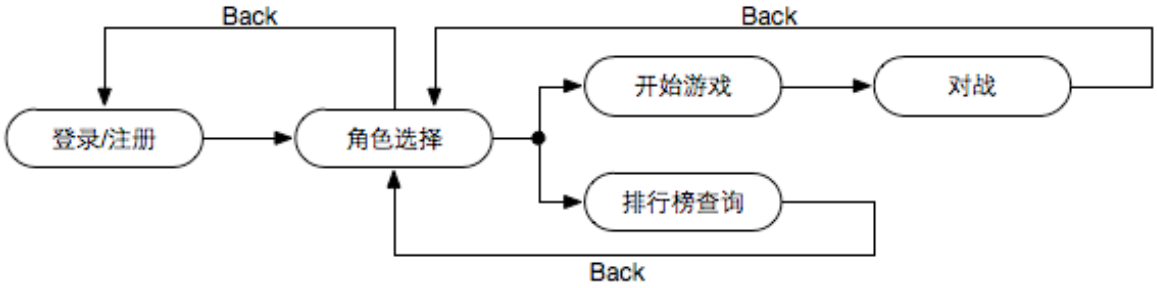
1. 游戏流程

玩家登陆界面后可通过账号密码输入登录/注册游戏；

登录后首先进入角色选择界面，此界面可进行角色选择或者进入排行榜查询界面；

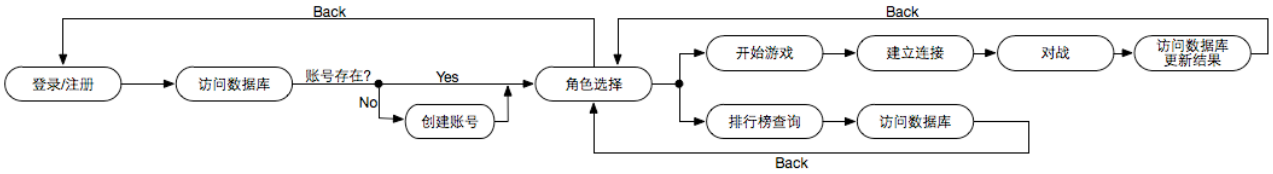
进入排行榜界面查询界面后，可查看排行榜信息，同时可返回角色选择界面；

进入游戏界面后，开操作角色与对方玩家对战，胜利或失败后可返回角色选择界面；



2. 流程设计

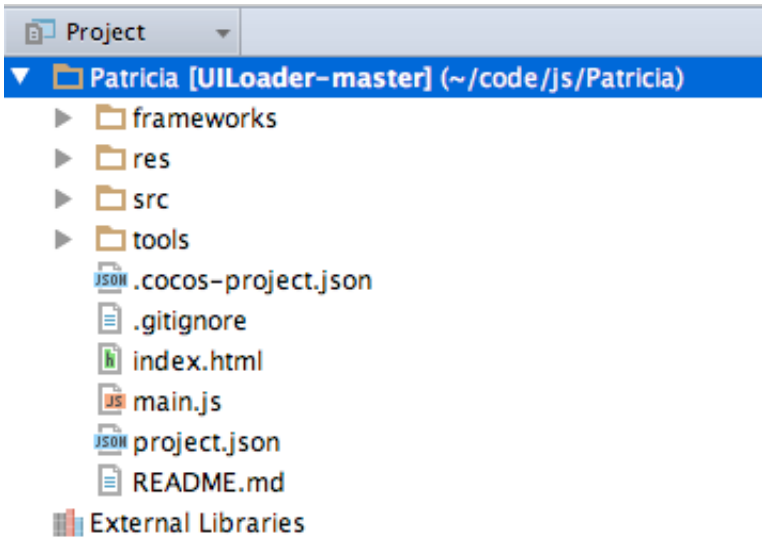
相对于游戏流程，实际流程设计中更加注重一些细节，例如登录后的数据库账号查询；访问排行榜时，排行榜数据的查询和排序；游戏开始后，游戏双方连接的建立；游戏结束后，游戏结果的记录



四、代码构架

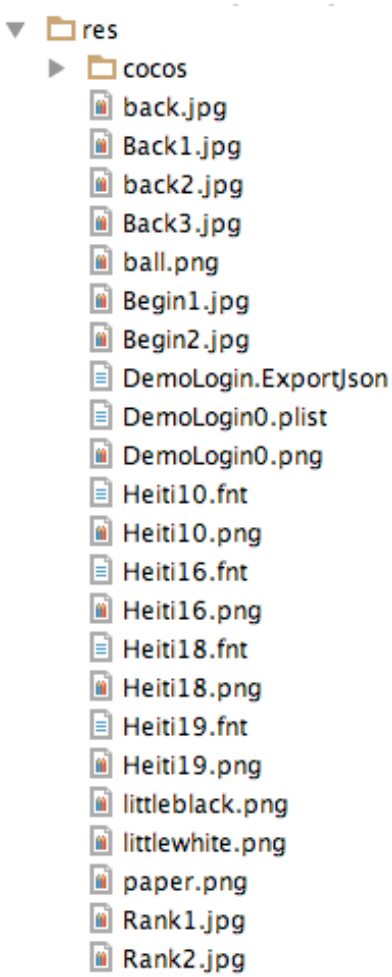
1. 基本构架

Patricia工程目录下，主要由frameworks(存放cocos2d功能文件), res(资源文件，图片jpg/png，josn，fnt等)，src(存放js)，tools，index.html(唯一的html文件)，main.js（main文件），project.json（js申明文件）；

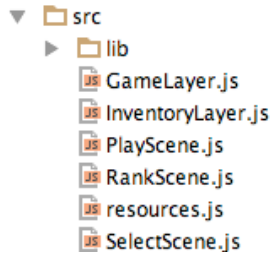


2. res文件

res文件中有各个场景所需要的图片，字体以及json文件，具体在resources.js内部申明；



3. src文件
   内部有各个Scene场景搭建所需要的js脚本文件，主要有GameLayer.js， PlayScene.js, RankScene.js, resources.js, SelectScene.js；



4. index.html
   index页面非常简单，调用了frameworks下的cocos2d-js-v3.3.js文件和main.js文件，实现程序的运行；

```
<!DOCTYPE html>
<html>
<head>
    <title>Hello Cocos2d-JS</title>
    <script type="text/javascript" src="frameworks/cocos2d-js-v3.3.js" charset="UTF-8"></script>
</head>
<body>
    <div style="...">
        <canvas id="gameCanvas" width="1136" height="640"></canvas>
    </div>

    <script type="text/javascript", src="main.js"></script>
</body>
</html>
```

5. main.js
   main.js对视图大小进行了调整，并创建了GameLayer图层；

```
cc.game.onStart = function(){

    cc.view.adjustViewPort(true);
    cc.view.setDesignResolutionSize(960, 640,cc.ResolutionPolicy.SHOW_ALL);
    cc.view.resizeWithBrowserSize(true);
    cc.LoaderScene.preload(g_resources, function () {
        var scene = new cc.Scene();
        scene.addChild(new GameLayer());
        cc.director.runScene(scene);
    }, this);
};
cc.game.run();
```

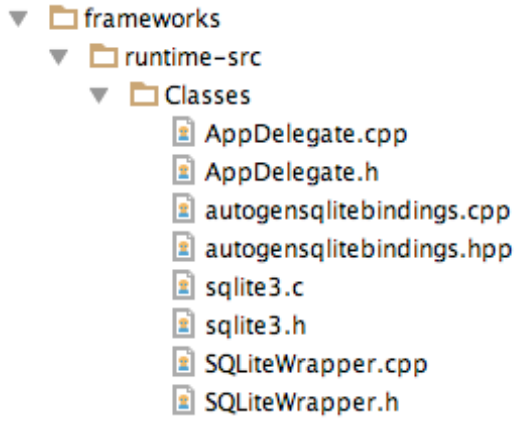
6. project.json

内部申明了所有会被使用到的js文件；

```
{
    "debugMode"     : 1,
    "frameRate"     : 60,
    "id"            : "gameCanvas",
    "renderMode"    : 1,
    "jsList"        : [
        "src/resources.js",
        "src/RankScene.js",
        "src/PlayScene.js",
        "src/SelectScene.js",
        "src/lib/UILoader.js",
        "src/GameLayer.js",
        "src/InventoryLayer.js"
    ]
}
```

7. sqlite

frameworks/runtime-src/Classes目录下定义了sqlite数据库使用所需文件(为C/C++文件)，需要配置JSB实现Js对C++的调用



五、实现细节

1. 按钮的实现

使用Sprite精灵体创建按钮，并通过Menu类进行绑定，使用listen对按钮点击事件进行监听；

```
var back1 = new cc.Sprite(res.Back1_jpg);
var back2 = new cc.Sprite(res.Back3_jpg);
var back = new cc.MenuItemSprite(back1,back2);
back.x=size.width/2;back.y=size.height/2-230;
back.setOpacity(200);
//begin.setScale(0.5);

var listener5 = cc.EventListener.create({
    event:cc.EventListener.TOUCH_ONE_BY_ONE,
    onTouchBegan:function(touch,event) {
        if(cc.rectContainsPoint(event.getCurrentTarget().getBoundingBox(),touch.getLocation())){
            console.log(">>>");
            //cc.eventManager.removeListener(listener);
            cc.director.runScene(new SelectScene());
        }
    }
});
cc.eventManager.addListener(listener5, back);

var menu = new cc.Menu(back);
menu.x=0;menu.y=0;
this.addChild(menu);
```

2. 子弹优化

使用pool对发射的子弹进行内存优化处理；

```
Ball.reCreate = function (x, y, z,w) {
    if(cc.pool.hasObject(Ball)){
        return cc.pool.getFromPool(Ball,x,y,z,w);

    }
    else{
        return new Ball(x,y,z,w);
    }
}
```

3. 使用SQLite进行数据库管理

```
if(!cc.sys.isNative){
    cc.log("只能在Native下使用");
    return true;
}
this._db = new sql.SQLiteWrapper();
this._dbPath = this._db.initializing("data.db","res","");

this._isOpen = this._db.open(this._dbPath);

cc.log("数据库打开结果:" + this._isOpen?"已打开...":"未打开...");

if(this._isOpen){
    var st = this._db.statement("select * from equip");

    var ary = [];
    while(st.nextRow()){
        var equipVO = new CEquipVO();
        equipVO.wid = parseInt(st.valueString(0));
        equipVO.name = st.valueString(1);
        equipVO.desc = st.valueString(2);
        equipVO.level = st.valueString(3);
        equipVO.icon = st.valueString(4);
        equipVO.quality = st.valueString(5);
        ary.push(equipVO);
    }

    for(var vo in ary){
        cc.log("equipData:" + ary[vo].toString());
    }
}
return true;
```