现代操作系统应用开发实验报告

学号：153311117　　**班级　　：**教务 2 班

姓名：黄楠绚　　**实验名称：**homework15

一．参考资料

请在这里列出对本实验有帮助你所参考的资料或者网站。

本次作业的 ppt；12 周自己的作业（参考了保存本地数据）；百度知道搜到的一个 GET 请求格式的回答;

Cocos2dx 引擎笔记——HttpClient session http://blog.csdn.net/yj_cs/article/details/48271977

二．实验步骤

本次作业实现的功能：1.“使用用户名登录”；“提交分数；查询最好 n 位成绩；

2.本地化保存 GAMESESSIONID 等 cookies 信息，实现自动登录；思考 HttpClient 的 enableCookies 函数的作用并写在实验报告里

在 AppDelegate.cpp 里判断 UserDefault.xml 是否有保存上次运行获得的 sessionid 值，有则直接登陆，跳转到 GameScene，没有则进入登陆界面。

```
// 判断是否保存锅sessionid
if (database->getStringForKey("sessionid") != "") {
    // 保存sessionid
    Global::gameSessionId = database->getStringForKey("sessionid");
    // 切换到game场景
    auto gameSence = GameScene::createScene();
    director->runWithScene(gameSence);
}
else {
    // create a scene. it's an autorelease object
    auto scene = LoginScene::createScene();
    // run
    director->runWithScene(scene);
}
```

LoginScene.cpp 文件中，login 按钮的触摸响应函数：

```
button->setPosition(Size(visibleWidth / 2, visibleHeight / 2));
button->addTouchEventListener(CC_CALLBACK_2(LoginScene::login, this));
this->addChild(button, 2);
```

按照 ppt，发出 POST 请求:

```
void LoginScene::login(Ref *pSender, cocos2d::ui::Widget::TouchEventType type) {
    // POST请求
    HttpRequest *request = new HttpRequest();
    request->setUrl("http://localhost:8080/login");
    request->setRequestType(HttpRequest::Type::POST);
    request->setResponseCallback(CC_CALLBACK_2(LoginScene::onHttpRequestCompleted, this));

    // 数据
    string player_name = textField->getString();
    string data = "username=" + player_name;
    const char *post_data = data.data();
    request->setRequestData(post_data, strlen(post_data));

    // 发送
    cocos2d::network::HttpClient::getInstance()->send(request);
    request->release();
}
```

处理响应的函数：

先判断是否会出现失败的情况：

```
void LoginScene::onHttpRequestCompleted(HttpClient *sender, HttpResponse *response)
    if (!response) {    // 如果没有response
        return;
    }
    if (!response->isSucceed()) {   // 要求返回是否成功
        log("response failes");
        log("error buffer: %s", response->getErrorBuffer());
        return;
    }
```

成功获取响应后：

解析 json 字符串，判断 result 的值是否为 true，若为 true，则表示登陆成功，**保存 sessionid 到本地后**，跳转到 GameScene 界面：

```
// 获得响应的字符串
string res = Global::toString(response->getResponseData());
// 解析json格式的字符串
rapidjson::Document res_d;
res_d.Parse<0>(res.c_str());
if (res_d.HasParseError()) {    // 解析错误
    CCLOG("GetParseError %s\n", res_d.GetParseError());
}
// 获得result的值
if (res_d.IsObject() && res_d.HasMember("result")) {
    // 登陆成功
    if (res_d["result"].GetBool() == true) {
        // 保存sessionid
        Global::gameSessionId = Global::getSessionIdFromHeader(Global::toString(response->getResponseHeader()));
        database->setStringForKey("sessionid", Global::gameSessionId);
        // 切换场景
        auto gameSence = GameScene::createScene();
        Director::getInstance()->replaceScene(gameSence);
    }
}
```

GameScene.cpp:

submit 按钮的响应函数：

```
void GameScene::submit(Ref *pSender, cocos2d::ui::Widget::TouchEventType type) {
    // 判断score输入框内是否全是数字且不为空
    if (GameScene::isDigit(score_field->getString())) {
        HttpRequest *request = new HttpRequest();
        request->setUrl("http://localhost:8080/submit");
        request->setRequestType(HttpRequest::Type::POST);
        request->setResponseCallback(CC_CALLBACK_2(GameScene::onHttpSummitRequestCompleted, this));

        string data = "score=" + score_field->getString();
        const char *post_data = data.data();
        request->setRequestData(post_data, strlen(post_data));
        // 请求的header
        vector<string> headers;
        string cookie = "Cookie:GAMESESSIONID=" + Global::gameSessionId;
        headers.push_back(cookie);
        request->setHeaders(headers);

        // 发送
        cocos2d::network::HttpClient::getInstance()->send(request);
        request->release();
    }
```

```
// 判断score是否全是数字且不为空
bool GameScene::isDigit(string score) {
    if (score.length() > 0) {
        for (int i = 0; i < score.length(); i++) {
            if (!isdigit(score[i]))
                return false;
        }
        return true;
    }
    return false;
}
```

Submit 的请求处理函数，与登陆处理函数的区别：

解析 json 字符串后获取 info 的最高分数，并把输入框的值设置为该分数：

```
// 解析json格式的字符串
rapidjson::Document res_d;
res_d.Parse<0>(res.c_str());
if (res_d.HasParseError()) {    // 解析错误
    CCLOG("GetParseError %s\n", res_d.GetParseError());
}
if (res_d.IsObject() && res_d.HasMember("result")) {
    // 提交成功
    if (res_d["result"].GetBool() == true) {
        // 获取最高分数
        if (res_d.HasMember("info")) {
            // 获取最高分数,设置输入框数字
            score_field->setText(res_d["info"].GetString());
        }
    }
}
```

Rank 按钮的响应函数，GET 请求的发送数据形式是：网站?xxx=value ：

```
void GameScene::rank(Ref *pSender, cocos2d::ui::Widget::TouchEventType type) {
    HttpRequest *request = new HttpRequest();
    request->setUrl("http://localhost:8080/rank?top=10");
    request->setRequestType(HttpRequest::Type::GET);
    request->setResponseCallback(CC_CALLBACK_2(GameScene::onHttpRankRequestCompleted, this));

    vector<string> headers;
    string cookie = "Cookie:GAMESESSIONID=" + Global::gameSessionId;
    headers.push_back(cookie);
    request->setHeaders(headers);

    // 发送
    cocos2d::network::HttpClient::getInstance()->send(request);
    request->release();
}
```

rank 的处理函数，获取 info 的内容并将 info 字符串中的“\”替换成“\n”,在 rank 输入框显示：

```
if (res_d.IsObject() && res_d.HasMember("result")) {
    // rank成功
    if (res_d["result"].GetBool() == true) {
        if (res_d.HasMember("info")) {
            string rank_result = res_d["info"].GetString();
            // "|"用"\n"代替
            int len = rank_result.length();
            size_t pos = -1;
            for (int i = 0; i < len; i++) {
                pos = rank_result.find("|");
                if (pos == -1)
                    break;
                else
                    rank_result[pos] = '\n';
            }
            // rank输入框
            rank_field->setText("\n" + rank_result);
        }
    }
}
```

思考 HttpClient 的 enableCookies 函数的作用：

先百度：

三、sessionID的其他实现方式：

在3.x的版本中，`HttpClient::enableCookies`方法支持http会话使用cookie。 由于sessionID的保存和传递是通过cookie实现的，所以我们可以很方便的使用`HttpClient::enableCookies`方法来实现http client session。

```
cocos2d::network::HttpClient::getInstance()->enableCookies(NULL);
```

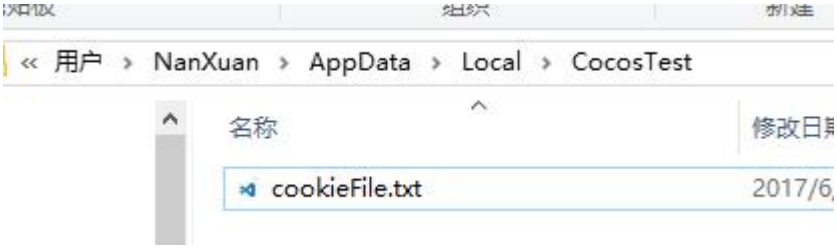
了解到 enableCookies 可以自动保存和发送；

这样还是不太明白，实践一下，把我自己原来代码的保存 cookie 的代码去掉，在三个发送 request 的代码语句上加上这一句（蓝色标注那句）：

```
request->setHeaders(headers);

// 发送
HttpClient::getInstance()->enableCookies(NULL);
cocos2d::network::HttpClient::getInstance()->send(request);
request->release();
```

运行第一次，登陆后可以发现在原来放 UserDefault.xml 的文件夹中有一个 CookieFile.txt 文件：



打开内容可以看到有保存了 sessionid:

cookieFile.txt

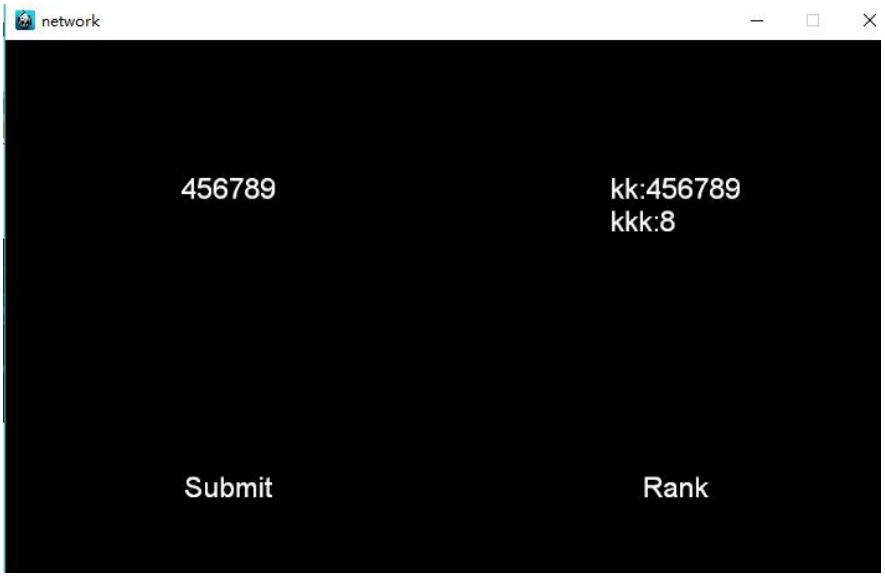
```
# Netscape HTTP Cookie File
# http://curl.haxx.se/docs/http-cookies.html
# This file was generated by libcurl! Edit at your own risk.

localhost   FALSE   /   FALSE   0   GAMESESSIONID   dc468c70fb574ebd07287b38d0d0676d
```

关掉程序，再把代码的 AppDeleGate.cpp 文件中创建第一个界面的代码改成，程序一运行即进入 GameScene：

```
//else {
    // create a scene. it's an autorelease obj
    auto scene = GameScene::createScene();
    // run
    director->runWithScene(scene);
//}
```

运行程序，直接进入 Game 界面，发现可以发送提交和排序请求：

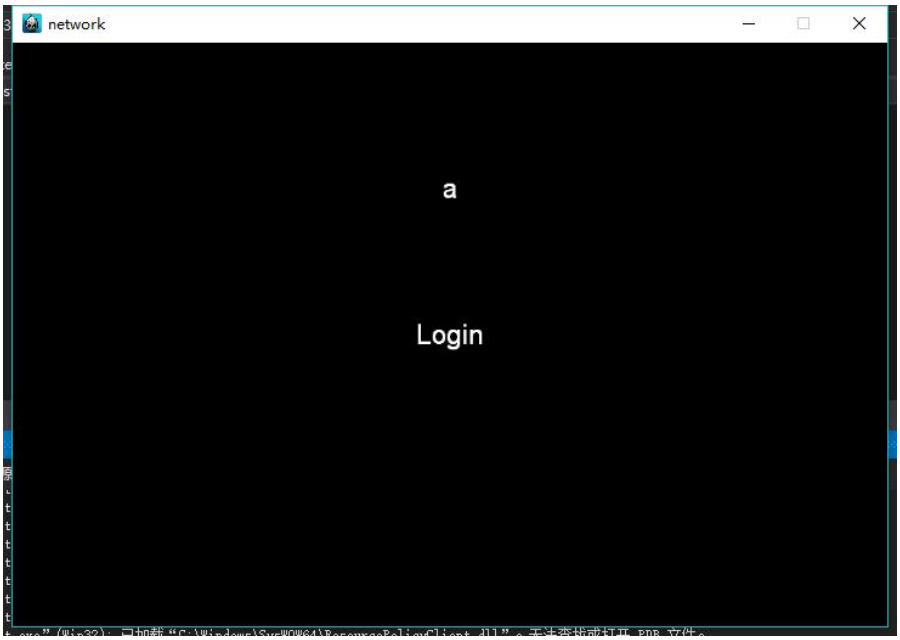


因此，**enableCookies** 函数的作用就是可以本地化保存 cookie，还可以自动发送，只需要加上一句代码，

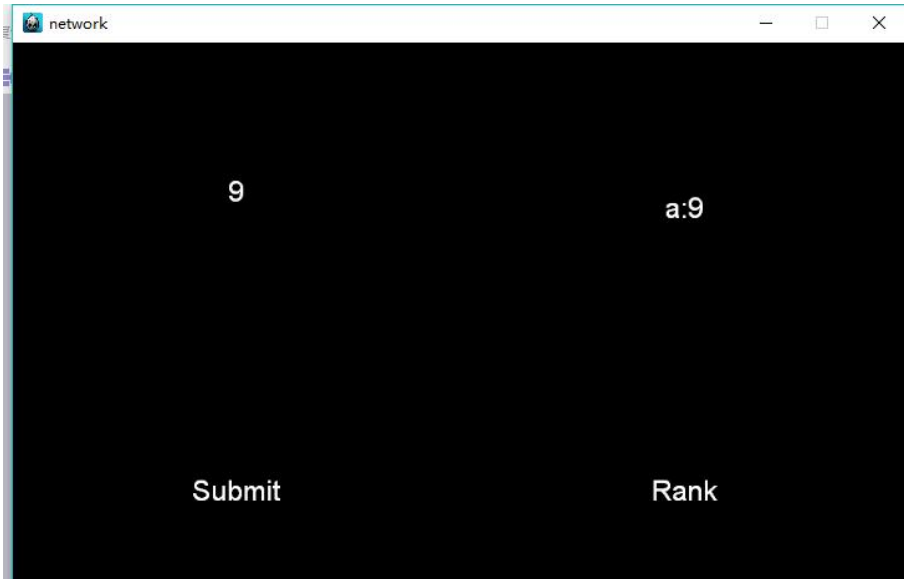
比我用 UserDefault.xml 方便多了~

三．**实验结果截图**

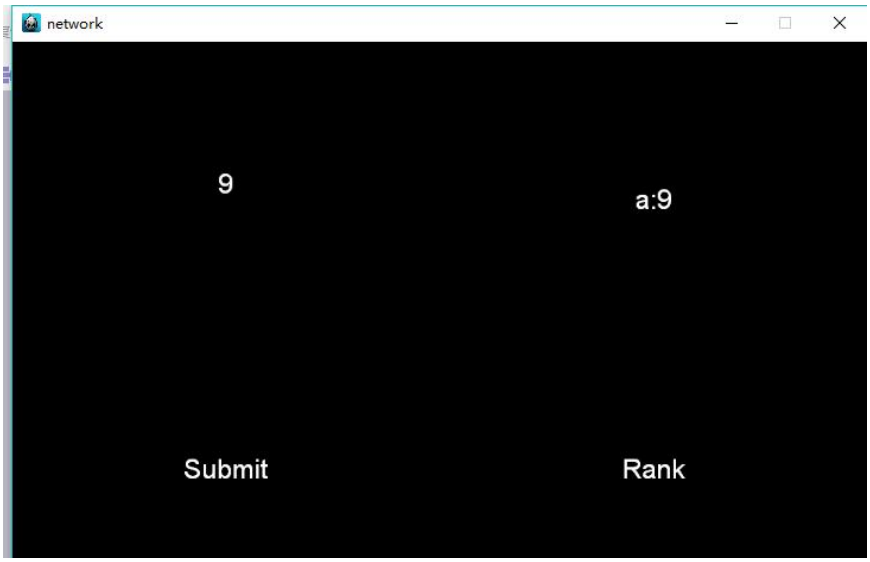
登陆界面：



登陆后，在 score 输入 9，先点击 submit 后点击 rank：

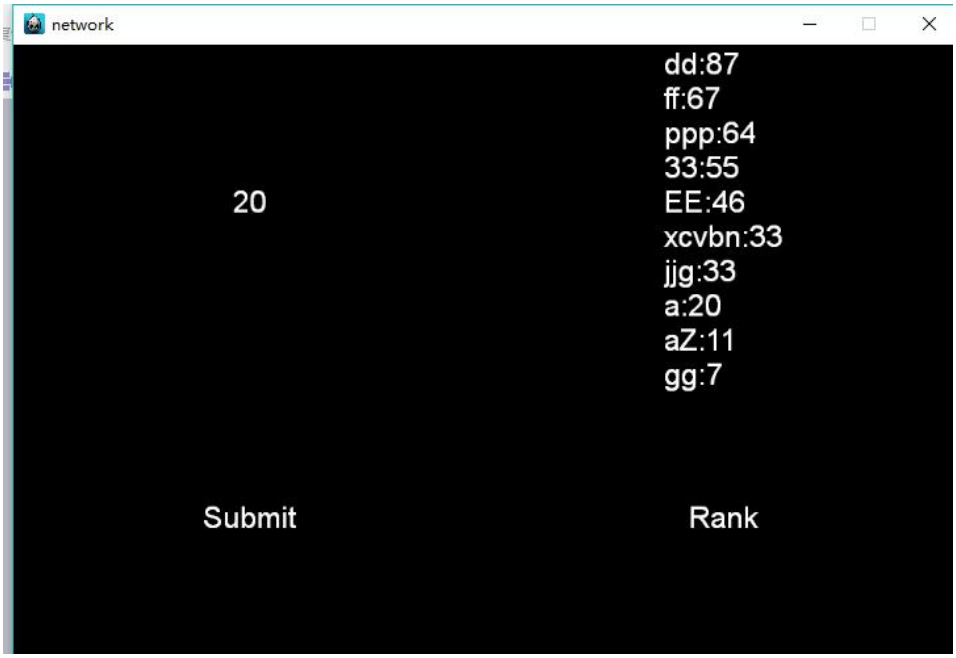


输入更小数字 5，点击提交和排序：score 输入框和 rank 文本框都是 9：

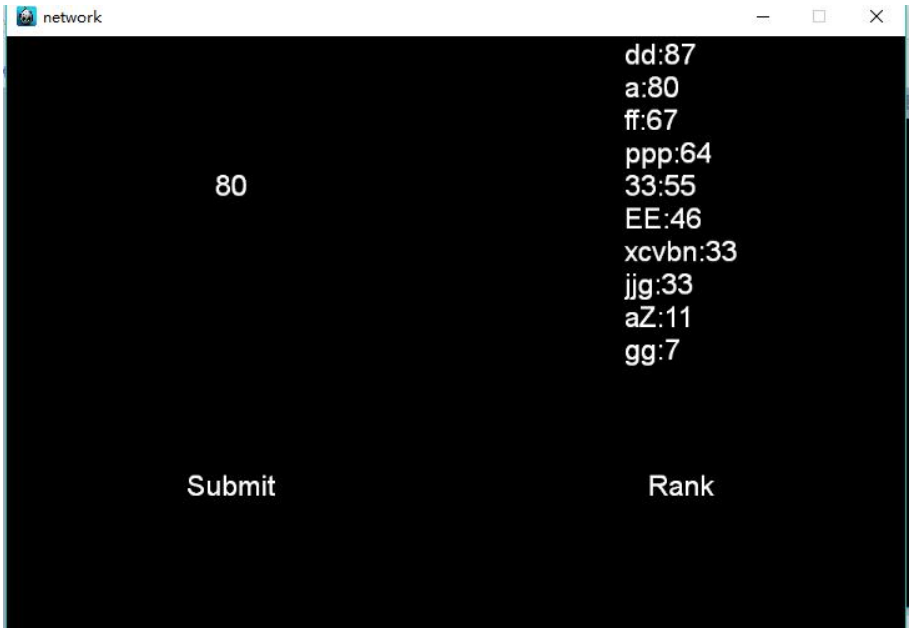


输入更大数字 20 先点击 submit 后点击 rank：：

用 ta 给的 demo 运行登陆并提交分数后，在我自己的程序上重新点击 rank，可以看到用户排名：



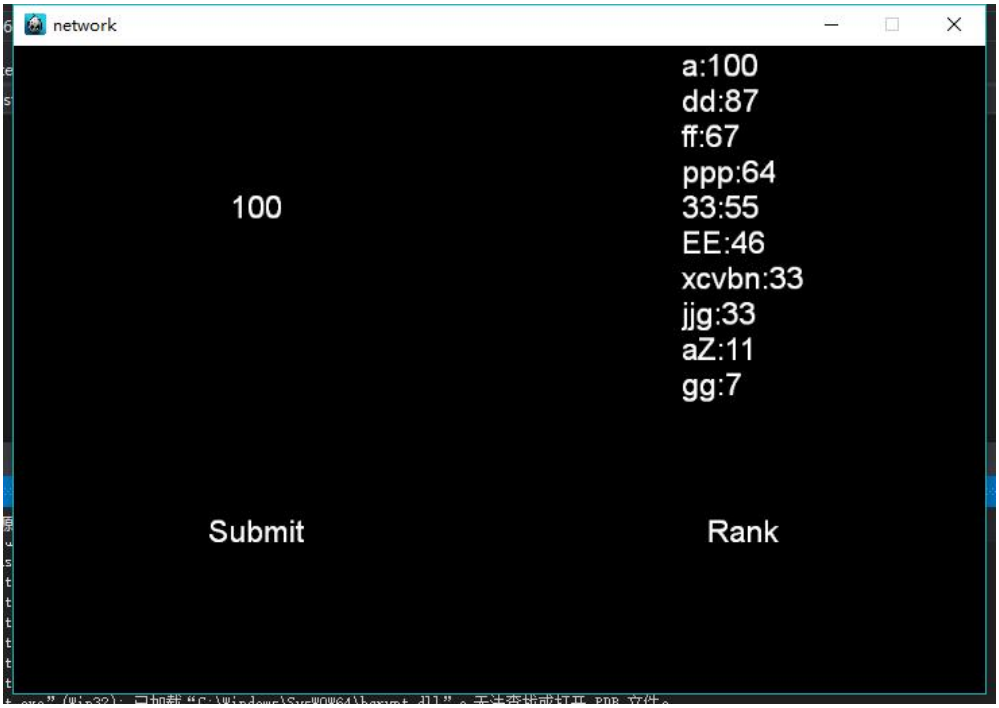
在 score 输入 80 后，点击提交和排名，可以看到用户 a 的排名变化：



并且，可以看到 UserDefalut.xml 文件中保存了 sessionid 的值：

关闭我的程序，重新运行：

没有看到 LoginScene，直接进入 GameScene，输入 100 后点击提交和排序，看到用户 a 排名变化：



四．实验过程遇到的问题

请在这里写下你在实验过程中遇到的问题以及解决方案。

问题：在 GET 请求发送数据使用了 POST 的方法，返回的 json 结果总算说我 No provide top

解决：上网找到了 GET 请求发送数据的方法

五．思考与总结

请在这里写下你本次试验的心得体会以及所思所想。

这次作业不难，由于没有学过 web，也没有深入了解计网知识，有些东西还是懵懵懂懂，想看一下 localhost:8080 网页是什么样的，用自己的浏览器发现打不开 localhost:8080 和 127.0.0.1 网页，上网搜了好久还是没有解决，问了现操项目组里的一个大三的师兄，他说”在 java 里面每个 url 都有对应处理的逻辑代码“，”可能 localhost:8080/hello 可以显示什么但是 localhost:8080/不一定能显示“。自己还有很多不明白，可能上网查资料会查到一点东西，但觉得还是自己有空去尝试写一个服务器出来就能明白啦。学无止境，学的越多，发现自己不懂的越多，自己还需要努力。

1. 实验报告提交格式为 pdf。

2. 实验内容不允许抄袭，我们要进行代码相似度对比。如发现抄袭，按 0 分处理。