

太阳神三国杀 · 国战

2.0.0 发布说明



Mogara Team

2015.8.10

更新内容

下载地址: <http://pan.baidu.com/s/1qW7k3rl>
请在游戏之前认真阅读本文档, 祝您游戏愉快。

问题修复

- 修复 Room::doDragonPhoenix 函数计算体力时的错误。(Core)
- 修复被【断肠】的武将暗置后明置其他角色显示技能的错误。(Core)
- 一定程度上修复内存泄漏(Core)
- 修复装备上的装备牌可以连横的问题(Core)
- 完全修复【红颜】预亮问题。(Skill)
- 修复拥有【帷幕】技能的角色使用过河拆桥或顺手牵羊拿走黑色【闪电】的问题。(AI)
- 一定程度上修复夏侯渊无脑【神速】的问题。(AI)
- 一定程度上修复无脑使用【度势】的问题。(AI)
- 修正一些 AI 检测国战不存在技能的问题。(AI)
- 修正【固政】、【无双】不使用的的问题。(AI)
- 修正 SmartAI:askForCardChosen 在选牌时不考虑 disable_list 的问题。(AI)
- 修复【水淹七军】对同伙的不正常问题(已考虑孙尚香)(AI)
- 修复给空城诸葛亮【木牛流马】的问题。(AI)
- 修复使用【杀】的一点 AI 问题。(AI)
- 修复【横征】的函数嵌套问题。(AI)
- 修复【火烧连营】等牌仍然【连横】给其他角色的问题。(AI)

API 方案变化

- 移植身份的【化身】动画函数。(API)
- 为 Room::askForYiji 增加 expand_pile 函数, 在触发函数时会展开 pile。(API)
- 增加手动添加可见牌的函数。(API)
- 更新 Room::askForExchange 函数, 使其拥有更灵活的使用方式。(API)
- 新增 Room::askForCardsChosen 函数, 用于对一名角色多次选

牌。(API)

- 加入对蛊惑框按钮是否可用的手动判断。(API)
- 增加技能无效的 **property** 和明置技能无效的 **property**。(API)
- 增加 **Room::askForMoveCards** 函数，用于将一定数量的牌分成两堆 (**Up&Bottom**) 并调整顺序。(API)
- 给视为技能添加 **limit_mark**。(API)
- 增加触发技的动态优先权 (**DynamicPriority**) 的支持。(API)
- 增加体力流失 (**HpLost**) 和卡片确定生效 (**CardEffectConfirmed**) 事件。(API)
- 增加 **Room::notifyChooseCards** 函数，用于解决【固政】、【傲才】等技能。(API)
- 新增 **Room::notifyMoveToPile** 函数，是其可以配合视为+技能卡 (API)
- 加入隐藏的类似于【木流牛马】的 **Pile (^pile)** (API)
- 增加 **Room::askForPlayersChosen** 函数，用于选择多名角色。(API)
- 扩充 **Room::askForChoice** 函数，可以多列。(API&ui)
- 给触发技能添加 **record** 函数，用于记录一些需要的信息。(API)
- 修改【断肠】属性的储存方式。(从 **QStringList** 变为 **QString**) (API)
- 增加 **QVariant** 的 **canConvert** 函数。(API)
- 修复创建 **Lua** 宝物时的闪退问题。(LuaAPI)
- 新增 **Lua** 对 **QMap** 的支持。(LuaAPI)
- 移除 **VS2010** 和 **VS2012** 的工程文件。(VCProject)

优化&新特征

- 增加男性角色【水淹七军】的配音。(audio)
- 优化技能结算顺序，从当前回合的角色开始结算。(Core)
- exppattern** 对隐藏 **pile** 的支持。(Core)
- 优化【礼让】的结算，一定程度上减少插入结算带来的问题。(Skill)
- 优化【放逐】和【缔盟】的代码结构。(Skill)
- 【固政】、【恂恂】、【突袭】等技能使用新的 **API** 函数。(Skill)
- 优化观星框的界面。(ui)
- expand_pile** 中的隐藏 **pile** 改为从右侧展开。(ui)
- 优化【玉玺】与各种摸牌阶段技能（突袭、恂恂）的 **AI** 判断。(AI)
- 优化【天妒】和【屯田】同时存在时对判定牌的判断。(AI)
- 优化【无懈可击】和【无懈可击·国】的使用策略。(AI)

完善【戮力同心】对目标的判断。(AI)
细节优化(Global)
Android Version 初步支持(Global)
【一统天下】场景稍微进行修改。(Xmode)

历史遗留问题（待修复）

改变一览颜色只对武将一览有用，卡牌一览无用(ui)
【援军盛宴】、【远交近攻】的AI未进行细致判断。(AI)
【驱虎】目标判断可能出现失误。(AI)
【天香】角色选择可能还有问题。(AI)
【乱击】杀死队友。(AI)
余下两个野心家的时候AI会无所事事？？(AI)

关于版本问题

Android Version

鉴于Qt的跨平台特性和玩家们的要求

(<http://tieba.baidu.com/p/3955181780>)

我们暂时发布安卓预览版本版本。

强调：在新架构完成之前，开发组不会受理任何有关Android Version的界面问题，毕竟这只是PC版的一个副产品。（原谅开发组没有那么多精力）如果您是一位Qt移动端大师并有兴趣改善我们的版本，请参见[其他](#)中的电子邮件地址来加入开发组。

Android 版本的配置方法

首先下载安卓版本的神杀·国战，并安装之。

然后下载PC版的国战，将这些文件（夹）放入SD卡目录下/android/data/org.qsgrara.qsanguosha中：

ai-selector audio diy etc extensions font
hero-skin image lang lua rule skins
style-sheet ui-script qt_zh_CN.qm sanguosha.qm

然后就可以开始游戏了。

如果想自己编译Android Version，请参考：

<http://mogara.org/build-tutorial/>

Windows XP

由于使用 v120 平台工具，神杀·国战已经完全不支持 Windows XP 系统。解决方案请参考 [MSVC2010](#) 和 [MSVC2012](#) 的编译技巧

API 说明部分（开发）

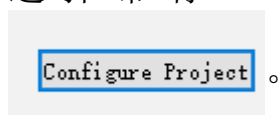
MSVC2010 和 MSVC2012 的编译技巧

在太阳神三国杀·国战（以下简称神杀·国战）2.0.0 的版本中，由于技术需要，Mogara Team（以下简称开发组，即我们）删除了 VS2010 和 VS2012 的配置文件。当然，肯定有许多使用 VS2010 或者 VS2012 来编译神杀·国战的朋友。我们可以通过 QtCreator 来配置神杀·国战。

打开 QSanguosha.pro（如果你使用的是官方 Qt 库的话 QtCreator 会包括在内，否则你需要到 qt.io 下载 QtCreator 并手动配置环境）然后会提示配置环境：



（如果没有这个就需要手动配置环境了，具体的请百度吧。）一定要选择带有 MSVC2010（或者 2012）的这一项，然后点



对于 XP 用户，你可以使用每夜版（nightly.mogara.org）或者自己配置环境。（下载 Windows SDK 7.0，里面包括了 MSVC2010，然后用 QtCreator 来配置；当然，你可以使用 MinGW，简单粗暴。关于 MinGW，请参考：<http://mogara.org/build-tutorial/>）

新函数介绍

可以看出神杀·国战 2.0.0 对 API 的改动非常大，这里我们大体讲解几个函数，具体的功能和灵活使用就请大家在实践中自己摸索吧。

Room::askForExchange

先来看这个函数的定义：

```
const Card *askForExchange(ServerPlayer *player, const char
*reason, int exchange_num, int min_num = 0, const char *prompt = "",
const char *expand_pile = "", const char *pattern = "")
```

参数如下：

参数说明	意义	备注
<i>player</i>	被询问的玩家	NULL
<i>reason</i>	原因（技能名）	NULL
<i>exchange_num</i>	交换的最大数目	NULL
<i>min_num</i>	交换的最少数目	如果为 0 返回值为 NULL
<i>prompt</i>	提示	如果没有则会默认提示
<i>expand_pile</i>	相应展开的 pile	如果没有则不展开 pile
<i>pattern</i>	要求的 pattern	如果 expand_pile 为空字符串，则默认为手牌，否则默认为 expand_pile 里的手牌
<i>return</i> 返回值	一张 Card	这张 Card 的子卡就是选择的牌。可以用 Card::getSubcards() 获得装有 id 的 QList。

这个函数已经使用了新的 AI 函数，具体的请参考 [SmartAI.lua](#) 中的 [SmartAI:askForExchange](#) 函数。

Room::askForPlayersChosen

这个函数的作用是选择多名角色（必须是连续数目，比如说 0~3），在【突袭】中使用了这个函数。

函数定义如下：

```
QList<ServerPlayer *> askForPlayersChosen(ServerPlayer *player, const
QList<ServerPlayer *> &targets, const char *reason, int min_num = 0,
int max_num = 2, const char *prompt = "", bool notify_skill = false);
```

参数如下

参数	意义	备注
<i>player</i>	被询问的玩家	
<i>targets</i>	待选择的玩家列表	
<i>reason</i>	原因（技能名）	
<i>min_num</i>	选择玩家的最小数目	
<i>max_num</i>	选择玩家的最大数目	默认为 2
<i>prompt</i>	提示	有默认提示
<i>notify_skill</i>	是否提示发动技能	如果是在 cost 里请填写 true
<i>return</i>	选择的玩家	没有进行 sortByActionOrder ，请手动排序。

这个函数在原有的AI函数上进行了修改，可以返回一个table（里面必须全部是ServerPlayer），也可以返回一名选定的玩家。

Room::askForCardsChosen

从字面上看，这个函数的作用是对一名角色选择多张牌。

函数定义如下：

```
QList<const Card*> askForCardsChosen(ServerPlayer *chooser,
ServerPlayer *choosee, const char *handle_string, const char
*reason);
```

参数如下：

参数	意义	备注
<i>chooser</i>	选择牌的玩家	
<i>choosee</i>	被选择牌的玩家	
<i>handle_string</i>	操作字符串	请参考附录 A
<i>reason</i>	原因（技能名）	
<i>return</i>	选择的卡片	可能为空

函数的实质就是调用askForCardChosen，所以ai和askForCardChosen是一样的。上次选择的牌会被添加到disable_list中。

Room::askForMoveCards

这是一个多功能的函数，【恂恂】、【称象】甚至【落英】、【固政】都

可以使用这个函数。神杀·国战 2.0.0 中的【恂恂】已经使用了这个函数。技能【称象】请参考附录 E。

函数的定义如下：

```
AskForMoveCardsStruct askForMoveCards(ServerPlayer *zhuge, const
 QList<int> &upcards, const QList<int> &downcards, bool visible, const
 char *reason, const char *pattern, const char *skillName, int
 min_num, int max_num, bool can_refuse = true, bool moverestricted =
 false, const QList<int> &notify_visible_list = QList<int>());
```

嗯~有些长，不过我们慢慢来看参数：

参数	意义	备注
<i>zhuge</i>	调整牌的玩家	
<i>upcards</i>	出现框时放在上面的牌	可以为空的 IntList()
<i>downcards</i>	出现框时放在下面的牌	可以为空的 IntList()
<i>visible</i>	其他角色能不能看见牌	和最后一个参数有关
<i>reason</i>	原因（技能名）	和翻译有关
<i>pattern</i>	Lua 全局函数名或""	必须是全局函数名
<i>skill_name</i>	高亮的技能名	无特殊的话同 reason
<i>min_num</i>	最少选择牌的数目	
<i>max_num</i>	最大选择牌的数目	
<i>can_refuse</i>	能否点击取消	
<i>moverestricted</i>	不符合 pattern 的牌能否移动，如果为 true 则不能移动	默认为 false
<i>notify_visible_list</i>	通知显示的 Card 的 id	和 visible 函数有关
<i>return</i>	一个结构体	具体在附录 B 中

在这里论述一下 *visible* 和 *notify_visible_list* 的关系：如果 *visible* 为 true 且 *notify_visible_list* 为空，那么会对所有角色展示你当前的牌，例如【称象】，如果 *visible* 为 false 的话 *notify_visible_list* 参数无效，表现为只有自己能看见。如果 *visible* 为 true 且 *notify_visible_list* 不为空，那么只有 *notify_visible_list* 里的 id 对应的牌能被其他角色看见。举个例子来说，如果 *visible* 为 true，*notify_visible_list* 里面有一个 25，那么 25 这张牌被移动时会被其他角色看见。如果 *notify_visible_list* 里面装的是 -1，那么其他角色只能看见牌的背面（类似观星）；如果 *visible* 为 false，其他角色根本不会显示这个框。具体的请在使用中观察。

这个函数的 AI 请参考 *SmartAI.lua* 的 *SmartAI:askForMoveCards* 函数。正常情况下，这个返回两个 table，第一个 table 里面装的是在上面牌的 id，第二个 table 装的是在下面牌的 id。具体的问题我们在附录 B 中阐述。

Room::notifyMoveToPile 和 Room::notifyChooseCards

从字面上看，这两个函数都是 notifyMove，也就是假 move。这类函数是专门处理 OL 类似从手牌打出功能的。

先看看 notifyMoveToPile 函数，定义如下：

```
void notifyMoveToPile(ServerPlayer *player, const QList<int>
&cards,const char *reason, Player::Place place, bool in, bool
is_visible);
```

参数如下：

参数	意义	备注
player	被操作的玩家	
cards	要移入（移出）pile 里牌的 id	
reason	移动的原因 通常为技能名	pile 的名字为#加上 reason
place	这些牌移入前（移出后）应有的位置	
in	移入还是移出	
is_visible	是否正面向上的移入（移出）	技巧： 一般和 in 保持一致

一般是先移入然后再移出，两次的 in 要注意不同。

再来看 notifyChooseCards，这个函数是用来从一对牌里面选择一张牌，而形式更加友好（在手里选），其定义是这样：

```
QList<int> notifyChooseCards(ServerPlayer *player, const QList<int>
&cards, const char *reason, Player::Place notify_from_place,
Player::Place notify_to_place, int max_num, int min_num = 0, const
char *prompt = "",const char *pattern = "");
```

参数如下：

参数	意义	备注
player	被询问的玩家	
cards	被操作的牌	
reason	同 notifyMoveToPile	
notify_from_place	移入前牌的位置	
notify_to_place	移出后牌的位置	和 from 可能不一样
max_num	选择牌的最大数量	
min_num	选择牌的最小数量	
prompt	提示	有默认提示
pattern	选择牌的要求	默认为在#reason 中
return	选择的牌	

其实 notifyChooseCards 就是 notifyMoveToPile+askForExchange，所以大部分参数都是通用的。

DynamicPriority

动态优先权这个概念在神杀·身份中就接触过。我们来看看定义：

```
if spec.dynamic_priority and type(spec.dynamic_priority) == "table"
then
    for e,v in pairs(spec.dynamic_priority) do
        skill:insertPriority(e,v)
    end
end
```

也就是说在触发技能里放入一个 dynamic_priority 的 table，用事件作为 index，后面加上值就行了，比如说：

```
dynamic_priority = {
    [sgs.EventPhaseStart] = 3,
    [sgs.DamageCaused] = 2,
    [sgs.Dying] = -1
},
```

对于没有在这里列举的事件，其优先权就是 priority 属性设置的。
这个功能有一些小问题，希望大家在使用的时候能够反馈信息。

on_record

作为触发技的一个函数，这个函数在 can_trigger 前执行，并用于记录一些信息，比如说落英（参见附录 D）。其参数和 can_trigger 是一样的，同样需要判断。

Room::askForChoice

目前对这个函数做了一点改进，可以搞多列选项。
观察下面一段代码：

```
local room = R
local player = P
room:askForChoice(player,"someSkill","choice1+choice2|
choice3+choice4+choice5+choice6|choice7+choice8+choice9|choice10")
```

它的运行结果是：