**X System使用说明**

**X System Instructions**

**文档版本Document version V 2.1.2**

**系统版本System version V 2.1.2**

**目录 Table of Contents**

目录

[1.前言 Foreword 11](#_Toc482010192)

[1.1关于9CCN About 9CCN 11](#_Toc482010193)

[1.2关于作者Xangle13 About Xangle13, the author 11](#_Toc482010194)

[1.3关于本系统 History of this system 11](#_Toc482010195)

[1.3.1起源(Xangle13) Origins [Written by Xangle13] 11](#_Toc482010196)

[1.3.2发展(Xangle13) Development [Written by Xangle13] 11](#_Toc482010197)

[1.3.3发布(Xangle13) Release [Written by Xangle13] 12](#_Toc482010198)

[1.3.4应用(萨尤克教主)Apply[Written by HW\_Lover] 12](#_Toc482010199)

[1.3.5沉寂(萨尤克教主)Silence[Written by HW\_Lover] 12](#_Toc482010200)

[1.3.6涅槃(萨尤克教主)Nirvana[Written by HW\_Lover] 12](#_Toc482010201)

[2.授权声明 Copyright 14](#_Toc482010202)

[3.概述 System outline 15](#_Toc482010203)

[4.注意 Notice 16](#_Toc482010204)

[5.文件结构 File Structure 17](#_Toc482010205)

[6.导入系统 System Installation 19](#_Toc482010206)

[6.1文件放置 Place files 19](#_Toc482010207)

[6.2文件引用 File input 19](#_Toc482010208)

[6.3加载和启用 Load and enable 20](#_Toc482010209)

[7.功能简介 Function summary 22](#_Toc482010210)

[7.1单位特质模块 Unit Characteristics model 22](#_Toc482010211)

[7.2科技系统模块 Technology model 22](#_Toc482010212)

[8.配置文件 File Configuration 23](#_Toc482010213)

[8.1单位列表 Unit list 23](#_Toc482010214)

[Units 23](#_Toc482010215)

[Name 24](#_Toc482010216)

[Type 24](#_Toc482010217)

[Size 24](#_Toc482010218)

[LifeBarType 24](#_Toc482010219)

[Ability 24](#_Toc482010220)

[Characteristic 24](#_Toc482010221)

[Exemption 24](#_Toc482010222)

[闲话 Notes 24](#_Toc482010223)

[8.2单位特质 Units' characteristics 26](#_Toc482010224)

[8.2.1做好心理准备 Prepare yourself 26](#_Toc482010225)

[8.2.2我准备好了 I'm ready 26](#_Toc482010226)

[8.2.3通用参数 General Configurations 27](#_Toc482010227)

[8.2.4特质介绍 Characteristics introduction 43](#_Toc482010228)

[8.3地形预制 Terrain 68](#_Toc482010229)

[尘云和辐射云 Dust cloud and radiation cloud 68](#_Toc482010230)

[闲话 Notes 68](#_Toc482010231)

[8.4科技系统 Technology 70](#_Toc482010232)

[8.4.1配置格式 70](#_Toc482010233)

[8.4.2条件介绍 73](#_Toc482010234)

[9.公共函数 Public function 76](#_Toc482010235)

[9.1 API 76](#_Toc482010236)

[9.2 Overload 77](#_Toc482010237)

[10.结语 Conclusion 79](#_Toc482010238)

[11.更新日志 Version history 80](#_Toc482010239)

[1.0.3/0.9.1 80](#_Toc482010240)

[1.0.4/0.9.3 80](#_Toc482010241)

[1.0.5/0.9.6 80](#_Toc482010242)

[1.0.6/0.9.7 80](#_Toc482010243)

[1.0.7/0.9.9 80](#_Toc482010244)

[1.0.8/1.0.0 80](#_Toc482010245)

[1.0.9/1.0.0 80](#_Toc482010246)

[1.1.0/1.0.1 80](#_Toc482010247)

[1.1.1/1.0.2 80](#_Toc482010248)

[1.1.2/1.0.4 81](#_Toc482010249)

[1.1.3/1.0.5 81](#_Toc482010250)

[1.1.4/1.1.0 81](#_Toc482010251)

[1.1.5/1.1.3 81](#_Toc482010252)

[1.1.6/1.1.4 82](#_Toc482010253)

[1.1.7/1.1.5 82](#_Toc482010254)

[1.1.8/1.1.6 82](#_Toc482010255)

[1.1.9/1.1.7 82](#_Toc482010256)

[1.2.0/1.1.8 82](#_Toc482010257)

[1.2.1/1.1.9 83](#_Toc482010258)

[1.2.2/1.2.0 83](#_Toc482010259)

[1.2.3/1.2.1 83](#_Toc482010260)

[1.2.4/1.2.3 84](#_Toc482010261)

[1.2.5/1.2.4 84](#_Toc482010262)

[1.2.6/1.2.5 84](#_Toc482010263)

[1.2.7/1.2.6 84](#_Toc482010264)

[1.2.8/1.2.7 84](#_Toc482010265)

[1.2.9/1.2.8 84](#_Toc482010266)

[1.3.0/1.3.0 84](#_Toc482010267)

[1.3.1/1.3.1 85](#_Toc482010268)

[1.3.2/1.3.5 85](#_Toc482010269)

[1.3.4/1.3.6 85](#_Toc482010270)

[1.3.5/1.3.7 85](#_Toc482010271)

[1.3.6/1.3.8 85](#_Toc482010272)

[1.3.7/1.3.9 86](#_Toc482010273)

[1.3.8/1.4.0 86](#_Toc482010274)

[1.3.9/1.4.1 86](#_Toc482010275)

[1.4.0/1.4.2 86](#_Toc482010276)

[1.4.1/1.4.3 86](#_Toc482010277)

[1.4.2/1.4.4 86](#_Toc482010278)

[1.4.3/1.4.5 86](#_Toc482010279)

[1.4.4/1.4.6 87](#_Toc482010280)

[1.4.5/1.4.7 87](#_Toc482010281)

[1.4.6/1.4.8 87](#_Toc482010282)

[1.4.7/1.4.9 87](#_Toc482010283)

[1.4.8/1.5.0 87](#_Toc482010284)

[1.4.9/1.5.1 87](#_Toc482010285)

[2.0.0/2.0.0 87](#_Toc482010286)

[2.0.1/2.0.1 87](#_Toc482010287)

[2.0.2/2.0.2 88](#_Toc482010288)

[2.0.3/2.0.3 88](#_Toc482010289)

[2.0.4/2.0.4 88](#_Toc482010290)

[2.0.5/2.0.5 88](#_Toc482010291)

[2.0.6/2.0.6 88](#_Toc482010292)

[2.0.7/2.0.7 88](#_Toc482010293)

[2.0.8/2.0.8 88](#_Toc482010294)

[2.0.9/2.0.9 88](#_Toc482010295)

[2.1.0/2.1.0 89](#_Toc482010296)

[2.1.1/2.1.1 89](#_Toc482010297)

[2.1.2/2.1.2 89](#_Toc482010298)

**文档历史 Document history**

|  |  |  |  |
| --- | --- | --- | --- |
| **修订日期**  **Date** | **修订内容**  **Changes** | **文档/系统版本**  **Doc/Sys vers** | **修订人Author** |
| 2011-09-02 | 创建，内容初步编辑  **Created and basic edit** | 1.0.0/0.9.0 | xangle13 |
| 2011-09-03 | 特质编辑  **Edited characteristics** | 1.0.1/0.9.0 | xangle13 |
| 2011-09-26 | 授权声明  **Added copyright** | 1.0.2/0.9.0 | xangle13 |
| 2011-09-28 | 基于公版的修正  **Public version change** | 1.0.3/0.9.1 | xangle13 |
| 2011-09-29 | 增加触发条件  **Added trigger condition** | 1.0.4/0.9.3 | xangle13 |
| 2011-09-30 | 增加地形触发系统  **Added terrain trigger** | 1.0.5/0.9.6 | xangle13 |
| 2011-10-01 | 增加触发条件，附加文档目录  **Added trigger condition, and TOC** | 1.0.6/0.9.7 | xangle13 |
| 2011-10-02 | 增加触发条件，增加操作类特质  **Added trigger condition, Added control condition** | 1.0.7/0.9.9 | xangle13 |
| 2011-10-03 | 感谢月君的翻译  增加了简易无人机系统  **Thanks for the translation by DMare (Yue)**  **Add characteristics** | 1.0.8/1.0.0 | xangle13 |
| 2011-10-10 | 感谢澳大利亚友人煌·内森的校对  **Thanks for the collate by Nathanius** | 1.0.9/1.0.0 | xangle13 |
| 2011-10-10 | 修正了生命值条件，修正了子系统条件  **Fixed the bug health condition,**  **Fixed the bug subsystem condition** | 1.1.0/1.0.1 | xangle13 |
| 2011-10-27 | 增加ship固有能力特质  改进了无人机特质 | 1.1.1/1.0.2 | xangle13 |
| 2011-10-28 | 增加了组合特质“投送”  增加了建造速度特质 | 1.1.2/1.0.4 | xangle13 |
| 2011-11-02 | 为单位替换系统新增了超空间替换方式。 | 1.1.3/1.0.5 | xangle13 |
| 2011-11-04 | 增加特质单体护盾，并添加相应显示用UI控件。  增加特质船壳装甲，并添加相应显示用UI控件。 | 1.1.4/1.1.0 | xangle13 |
| 2011-11-06 | 增加了装甲和护盾百分比触发条件。  添加了船只经验、等级特质。  修正了显示用UI控件。  增加了相关的配置文件参数。 | 1.1.5/1.1.3 | xangle13 |
| 2011-11-06 | 紧急修正 | 1.1.6/1.1.4 | xangle13 |
| 2011-11-09 | 修正了护盾和装甲的UI显示。  修正了护盾和装甲百分比触发条件无法正常使用的问题。  改进了护盾运作机理，现在护盾能够更好的重启了。 | 1.1.7/1.1.5 | xangle13 |
| 2011-11-13 | 修正了一个可能导致游戏卡死的BUG。  改进了护盾特质的参数。  增加了UI显示相关的设置开关。 | 1.1.8/1.1.6 | xangle13 |
| 2011-11-15 | 增加了新特质 | 1.1.9/1.1.7 | xangle13 |
| 2011-11-16 | 调整了数据算法，现在数学运算更加精确了。  调整了UI显示算法，修正了护盾、装甲在数值过低的情况下出现-1%的错误显示。  开放Xsystem独有的内部函数，方便高级应用者获取或操纵Xsystem的内部信息。 | 1.2.0/1.1.8 | xangle13 |
| 2011-11-16 | 增加了特质。  增加了胜负判断列表(reslist.lua变动)。 | 1.2.1/1.1.9 | xangle13 |
| 2011-11-17 | 修正了公开函数引起的错误。  修正了照准和范围场的判定错误。  增加了新特质并调整了触发ID范围。 | 1.2.2/1.2.0 | xangle13 |
| 2011-11-18 | 调整了特质代码的运行机制。  增加了一个单位特质上限自定义参数。  增加了新特质 | 1.2.3/1.2.1 | xangle13 |
| 2011-11-24 | 增加了新特质 | 1.2.4/1.2.3 | xangle13 |
| 2011-11-25 | 重写了所有分压算法。  修正了一个时间计算上的错误。 | 1.2.5/1.2.4 | xangle13 |
| 2011-11-27 | 调整了部分分压参数，重新制定了函数加载方式。 | 1.2.6/1.2.5 | xangle13 |
| 2011-11-28 | 增加了新的触发条件和特质。 | 1.2.7/1.2.6 | xangle13 |
| 2011-11-29 | 增加了新特质  改进了特质ShipAutoAmelioration | 1.2.8/1.2.7 | xangle13 |
| 2011-11-29 | 修正了一些小BUG | 1.2.9/1.2.8 | xangle13 |
| 2011-11-30 | 改进了无人机特质  增加了停泊相关的新特质 | 1.3.0/1.3.0 | xangle13 |
| 2011-12-01 | 改进了单位停泊替换特质  修改了范围信号特质 | 1.3.1/1.3.1 | xangle13 |
| 2011-12-02 | 废弃了冲突科技模块，新增了科技系统模块。  文档增修了“注意”项，以对功能模块和特质的废弃进行说明。 | 1.3.2/1.3.5 | xangle13 |
| 2011-12-02 | 修正了科技模块无法生效的BUG  增加了新的科技限制条件  文档中添加了Xsys公开函数的相关说明 | 1.3.4/1.3.6 | xangle13 |
| 2011-12-02 | 增加了公开函数  增加了科技限制条件 | 1.3.5/1.3.7 | xangle13 |
| 2011-12-03 | 修正了一个严重的BUG  增加了新特质 | 1.3.6/1.3.8 | xangle13 |
| 2011-12-04 | 修正了一些小BUG  改良了无人机特质 | 1.3.7/1.3.9 | xangle13 |
| 2011-12-04 | 增加了新特质  增加了特质触发条件 | 1.3.8/1.4.0 | xangle13 |
| 2011-12-05 | 修正了一个由无人机导致的检索不正常 | 1.3.9/1.4.1 | xangle13 |
| 2011-12-11 | 修改了等级条件的判定 | 1.4.0/1.4.2 | xangle13 |
| 2011-12-15 | 修正了生命值触发条件无效的问题  增加了新的触发条件  增加了新特质 | 1.4.1/1.4.3 | xangle13 |
| 2011-12-18 | 简单构建了自定义资源模块  增加了新特质 | 1.4.2/1.4.4 | xangle13 |
| 2011-12-27 | 对单位替换过程逻辑进行了一些修正  重新测试了系统压力阀值 | 1.4.3/1.4.5 | xangle13 |
| 2011-12-27 | 对一些特质进行了一些小修正 | 1.4.4/1.4.6 | xangle13 |
| 2012-01-26 | 大幅修正了自定义资源系统 | 1.4.5/1.4.7 | xangle13 |
| 2012-01-26 | 为自定义资源列表增加了一个新参数 | 1.4.6/1.4.8 | xangle13 |
| 2012-02-04 | 修改了胜利判定方式 | 1.4.7/1.4.9 | xangle13 |
| XXXX-XX-XX | 不知某年有一个版本 | 1.4.8/1.5.0 | xangle13 |
| 2012-11-10 | 增加了微量特质 | 1.4.9/1.5.1 | xangle13 |
| 2015-4-8 | 移植到重置版，并利用其新特性对整个系统进行了优化，废弃了部分功能，更改了部分特质的定义方式，并增加了自定义条件ID | 2.0.0/2.0.0 | HW\_Lover |
| 2015-4-12 | 修正Ping的显示；添加了AI难度判断条件；修正说明文档的一些小错误；  增加命令信号特质和命令操作特质 | 2.0.1/2.0.1 | HW\_Lover |
| 2015-4-20 | 更改特质里条件参数D3的格式，添加若干新的判断条件；  恢复了单位状态Key开关特质并添加了单位状态Key暂时开启特质 | 2.0.2/2.0.2 | HW\_Lover |
| 2015-5-3 | 添加若干新的判断条件；  添加自定义命令代码特质 | 2.0.3/2.0.3 | HW\_Lover |
| 2015-5-11 | 感谢菲雅丽帝国MOD作者妖都镇魂曲对本文档进行了优化排版；  添加了燃料特质和相应的条件判断 | 2.0.4/2.0.4 | HW\_Lover |
| 2015-5-17 | 科技系统条件判断格式更新；  恢复X系统公共函数和全局Key判断条件；  首个重置版X系统公版 | 2.0.5/2.0.5 | HW\_Lover |
| 2015-5-26 | 修正英文版Ping文字拥挤；修正装甲特质的一个问题 | 2.0.6/2.0.6 | HW\_Lover |
| 2015-08-24 | 针对重置V1.29/V1.30版进行兼容性更新 | 2.0.7/2.0.7 | HW\_Lover |
| 2015-11-23 | 特质系统条件格式更新；添加新的特质系统判断条件；科技系统新增科技类型和脚本设置，新增若干判断条件。 | 2.0.8/2.0.8 | HW\_Lover |
| 2017-02-25 | 进行重置版2.0适配。  能力特质新增对自定义指令能力的控制ID。  命令特质新增命令目标的范围和船型筛选。 | 2.0.9/2.0.9 | HW\_Lover |
| 2017-03-04 | 实现不同规则下加载不同的数据列表。 | 2.1.0/2.1.0 | HW\_Lover |
| 2017-05-01 | 改变系统加载方式。  优化系统并修复了2.1.0的bug。 | 2.1.1/2.1.1 | Lone\_wolf  HW\_Lover |
| 2017-05-08 | 新增条件判断IsDoingAbility和CanDoAbility。  更新舰船Ability能力列表，部分ID发生改变。（影响同样波及命令特质中的命令ID）  为命令特质添加新命令：执行自定义命令。  将自定义命令特质改为自定义函式特质。  为在CustomCode.lua上添加额外的自定义函式提供更好的支持。  清理文档中无效的超链接引用。 | 2.1.2/2.1.2 | HW\_Lover |

# 1.前言 Foreword

## 1.1关于9CCN About 9CCN

9CCN最早起源与国内的HW爱好者，由几位先辈建立了论坛，9CCN由此而来。由后加入的唯一发起并成立了mod制作组。

**9CCN first formed when Chinese Homeworld enthusiasts form a forum, the 9CCN forum. A MOD production team formed under the leadership of WeiYi after he joined.**

虽然9CCN经历了数次大变故导致论坛遗失在超空间内，但9CCNmod制作组穿越了无数时空仍然坚持在常规空间内。

**The 9CCN forum itself has gone missing in hyperspace the 9CCN MOD production team managed to make it out alive.**

## 1.2关于作者Xangle13 About Xangle13, the author

Xangle13于2006年底以程序员的身份加入9CCN MOD制作组，虽然经历了无数打酱油、划水、失踪、蒸发，不过至今(2012)仍然存活。

**Xangle13 joined the 9CCN Mod production team as a programmer in the autumn of 2006.**

## 1.3关于本系统 History of this system

---------------------------------------------------------------------------------------------------------------------------------

### 1.3.1起源(Xangle13) Origins [Written by Xangle13]

一套能够完善标记所有单位并进行控制的系统早在2009年就在本人头脑中形成，只是由于各种原因一直都没有去编写代码，这种情况一直持续到2011年的5月份。

当大家真正静下心来准备大干一场的时候，这一切终于开始了。

**The idea of forming a system that allowed control and marking of all units took shape in 2009. Because of varying reasons I didn’t start coding the system until May of 2011, which is the time that we decide to bring something big.**

---------------------------------------------------------------------------------------------------------------------------------

### 1.3.2发展(Xangle13) Development [Written by Xangle13]

5月，我们制定了基本的框架和技术指标，并进行了基础架构的编写。虽然这个时候的系统还完全无法使用，在游戏中也没有任何实际的功能体现。

**In May we formulated the basic framework and technology for the system, also started the writing of that framework. Although at that point of time, the system is completely useless**

6月，进行了全部功能的编写，在之后的压力测试中我们发现：这套系统的执行效率真的是糟糕透了！这就导致了整个7月都在调整数据结构和重写算法，几乎将整个系统全部推倒重新编写了一遍。

**In June we started writing the full functions, however under pressure testing, it seemed to be very low efficiency.**

**In July, because of the low efficiency, we pretty much completely rewrote the whole system.**

7月底的时候，我做了一个重要的决定：我们将公开授权这套系统，以便更多的人能够参与到mod制作中来。我们将提供一个简单高效的平台，只需要通过简单的配置就能实现各种单位特性和特殊系统。同样的，这个决定让我蛋疼了将近两个月。修改各种触发和条件格式以让这些配置和逻辑变的更灵活和丰富，而并非像之前那样只为FXmod定制系统功能。不得不说这个过程相当令人难忘。单位特质部分的配置格式以及配套逻辑至少经过了3次大的修改。直到我在写此文档的这个部分的时候，这项工作还没有完全收尾。

**At the end of July, we decided to release the resulting high efficiency system, the only thing needing to be done is simple configuring to achieve unique units and systems.**

**This idea made me suffer for 2 months, changing the system to be much more flexible than the FX MOD's system. I have to say that this process was very remarkable, some of the code undertook at least 3 major changes.Even at this point, the job still has a long way to go.**

---------------------------------------------------------------------------------------------------------------------------------

### 1.3.3发布(Xangle13) Release [Written by Xangle13]

对于我们来说，10月是一个合适的时间。10月1日是国庆节，大家有大把的时间呆在家里做喜欢做的事。所以我们选择这个时候发布FX1.90以及这套系统，从另外的角度来说，FX1.90更像是这套系统的演示mod。

**To us, October seem to be a perfect time to release this system and FX1.9, October 1st is the Chinese national day so everyone had a lot of time to do whatever they pleased. FX1.90 is more like a demonstration of this system.**

---------------------------------------------------------------------------------------------------------------------------------

### 1.3.4应用(萨尤克教主)Apply[Written by HW\_Lover]

X System发布后，在很多MOD中得到了应用，并帮助它们实现更多的功能，除9CCN的FX：Commander MOD和Seed MOD外，还有Fairy Empire Mod、Free MOD、Dark Times MOD等，使得这些MOD的游戏性获得了大幅提升。

**After X System released, it is widely used in many MODs, and helped them add much more new functions, and much more fun. These mods are: FX:C, SEED, Fairy Empire, Free, Dark Times etc.**

---------------------------------------------------------------------------------------------------------------------------------

### 1.3.5沉寂(萨尤克教主)Silence[Written by HW\_Lover]

然而初次应用带来的新鲜感很快淡化，在X System的使用过程中，其弊端暴露无遗。使用X System后读档失效导致一局遭遇战必须从头打到尾，不能存档；对系统的压力过大导致特质单位不能大量使用，数量不能过多，否则游戏卡顿严重。这些都大大限制了X System的推广和大范围使用，不仅使其未能展示其全部的潜力，同时还在一些不了解事情真相的玩家心目中造成了不好的印象。

**However, when X System is no longer fresh to the players, some of its disadvantages became obvious. Loading issue make players cannot save during the game; too much pressure to the CPU make mods cannot add too much units' characteristics, or the game would be very laggy. All these stopped X System from spreading, and fully showing its potential. In the meanwhile, they gaveplayers who don't know the truth a bad impression of X System.**

---------------------------------------------------------------------------------------------------------------------------------

### 1.3.6涅槃(萨尤克教主)Nirvana[Written by HW\_Lover]

2015年2月25日，Gearbox Software发布了《家园重置版》，其引擎仍然基于《家园2》，脚本系统变化不大，且引入了舰船能够调用自定义代码的新特性。虽然X系统的原创者Xangle13已经不再对X系统进行维护，但是为了9CCN各MOD的移植及后续MOD开发计划的顺利实施，新一代9CCN组员仍然决心将X系统移植到《家园重置版》上，最终目标是实现原X系统的全部主要功能，并充分利用《家园重置版》的新特性对X System进行优化，努力使其能够在《家园重置版》上发挥更大的作用。

**On February 25th, 2015, Gearbox Software released Homeworld: Remastered, its engine is from classic homeworld2, with no significant difference on scripting, but a new feature that every ship can run a custom code. Years has past, and Xanlge13, the author of X System, no longer maintain his product, but in order to port mods of 9CCN to HWRM and develop them in the future, new generation of 9CCN decide to port X System first. Our final goal of this project is to restore most of the X System's main functions, and do our best to make X System more efficiency with HWRM's new feature.**

# 2.授权声明 Copyright

本作品采用知识共享署名-非商业性使用-禁止演绎 4.0 国际许可协议进行许可。要查看该许可协议，可访问 <http://creativecommons.org/licenses/by-nc-nd/4.0/> 或者写信到 Creative Commons, PO Box 1866, Mountain View, CA 94042, USA。

**This work is licensed under the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License. To view a copy of this license, visit** <http://creativecommons.org/licenses/by-nc-nd/4.0/>**or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.**

本系统是一套由9CCN制作组开发并公开授权的通用系统，任何mod制作组、个人modder、玩家都可以使用本系统进行mod制作和改良。

**This system is developed by 9CCN MOD production team; it has public license for the use of any mod production team, individual modder or player to use this system for mod making or mod modifying.**

使用本系统的modder请在发布自身mod时在显眼位置郑重声明使用了本系统。

**If this system is redistributed in any mod the modder must declare that this system has been used.**

9CCN制作组及本系统原作者Xangle、现任维护者萨尤克教主拥有本系统以及授权声明的最终解释权。

**9CCN production team, the system's author, Xangle13 and currently maintainer HW\_Lover, reserve the right of final interpretation in regards to usage of the system.**

# 3.概述 System outline

X System是一套公开授权的脚本系统，由9CCN mod制作组开发。使用HW:RM的LUA脚本API编写，是一种将脚本接口再封装，编辑固定逻辑后开放配置的功能系统。

**The X system is a public licensed MOD scripting system, developed by the 9CCN MOD production team. It is written in HW:RM's LUA script API, and is a re-packaged scripting interface.This system allows you to configure it by editing of fixed logic.**

早期系统编写者为9CCNmod制作组主程序： xangle13

**Former system author: Xangle13**

**[9CCN MOD production team main programmer]**

早期测试与配置整合者为9CCNmod制作组组长： 唯一

**Former testing and configuration: WeiYi**

**[9CCN MOD production team leader]**

目前的系统维护者为9CCNmod程序组组长：萨尤克教主

**Currently system maintainer: HW\_Lover**

**[9CCN MOD programming group leader]**

# 4.注意 Notice

随着系统开发的深入，我们逐渐**废弃**了一些原始的功能模块和特质。任何注明废弃的特质或功能模块都请不要再使用。

**As the development of this system get deeper and deeper, we abandoned some old function model and characteristics gradually. PLEASE DON'T USE ANY OF THEM.**

# 5.文件结构 File Structure

此系统的文件结构很简单，没有复杂的目录结构和文件区分。主要的十三个文件都存放于resdata文件夹之下。分别为：logics.lua、pingstring.txt、XInit.lua、XCode.lua、XConditionJudge.lua、XTechConditionJudge.lua、XFunctions.lua、XTechFunction.lua、reslist.lua、reslist2.lua、XPing.lua、CustomCode.lua、CustomCode\_Missile.lua.

**The file structure of the system is easy, all thirteen main files are in the *resdata* folder. They are: logics.lua、pingstring.txt、XCode.lua、XConditionJudge.lua、XTechConditionJudge.lua、XFunctions.lua、reslist.lua、reslist2.lua、XPing.lua、CustomCode.lua、CustomCode\_Missile.lua、CustomCode\_Switch.lua.**

|  |  |
| --- | --- |
| **XInit.lua** | XInit.lua是系统的初始化文件，需要在游戏规则脚本中首先被调用。  **XInit.lua is the system's initiating file and need to be loaded in your game rule files.** |
| **logics.lua** | logics.lua是系统的主逻辑文件，普通使用者不需要去关心这个文件的编写问题。  **logics.lua is the system's main logic file, normal users don't need to worry about this file.** |
| **Pingstring.txt** | pingstring.txt是Ping显示部分引用的特殊字符，普通使用者不需要去关心这个文件的编写问题。  **pingstring.txt stores the special chars quoted for ping-displaying, normal users don't need to worry about this file.** |
| **XCode.lua** | XCode.lua是通过重制版的新特性实现原X系统单位特质的文件，普通使用者不需要去关心这个文件的编写问题。  **XCode.lua is the file that bring back all unit characteristics by using HWRM's new feature, normal users don't need to worry about this file.** |
| **XConditionJudge.lua、XTechConditionJudge.lua** | XConditionJudge.lua和XTechConditionJudge.lua是存放X系统条件判断函式的文件，普通使用者不需要去关心这个文件的编写问题。  **XConditionJudge.lua and XTechConditionJudge.lua is the file that contains all condition judgment function, normal users don't need to worry about this file.** |
| **XFunctions.lua** | XFunctions.lua存放了X系统的所有公共函数，普通使用者不需要去关心这个文件的编写问题。  **XFunctions.lua is the file that contains all public functions of X System, normal users don't need to worry about this file.** |
| **XTechFunction.lua** | XTechFunction.lua存放科技系统的自定义函式，以供高阶modder自行编辑。  **XTechFunction.lua is the file that cantains all custom fuunctions of the technologies, this file is for experienced modders to edit.** |
| **reslist.lua、reslist2.lua** | reslist.lua和reslist2.lua是所有使用者都需要编辑和仔细维护的文件，其中包含了所有的功能配置信息。在后续的章节中我们将非常详细来说明这两个文件的每一个部分需要如何编辑。  **reslist.lua and reslist2.lua requires the editing and maintenance of all users, it includes the configuration info for all functions. We will give specific detail about how to edit each part of these files later in the manual.** |
| **XPing.lua** | XPing.lua用来存储X系统需要用到的Ping，普通使用者不需要去关心这个文件的编写问题。  **Xping.lua stores all pings that would be used by X System, normal users don't need to worry about this file.** |
| **CustomCode.lua、CustomCode\_Missile.lua、CustomCode\_Switch.lua** | CustomCode.lua文件是X系统的舰船自定义代码文件，对于需要添加特质的舰船，都要加载该文件作为其自定义代码。  CustomCode\_Missile.lua和CustomCode\_Switch.lua是舰船自定义命令的模板文件。  **CustomCode.lua is the file of ships' custom codes in X System. All kinds of ships that need an X System's characteristic should load this file as their custom codes.**  **CustomCode\_Missile.lua and CustomCode\_Switch.lua are templates of ships' custom commands.** |

# 6.导入系统 System Installation

## 6.1文件放置 Place files

将data文件夹内的内容放到你自己的mod目录下。

**Put all files in data folder into your mod directory.**

## 6.2文件引用 File input

放置好文件后，我们需要在deathmatch.lua或其他游戏规则脚本中引用文件（关于游戏规则文件的作用和基本格式请参考其他教程，本文档不做说明）。

**After placing the files, we need to input them in deathmatch.lua or any other gamerule files. (for how to make gamerule files, please find another tutor, this manual won't talk about it)**

我们可以看到在原版的deathmatch.lua文件中已经存在了一个文件引用：

**We can see that the original deathmath.lua already inputs other files as below:**

**dofilepath("data:scripts/scar/restrict.lua")**

我们在这一行之下引用XInit.lua文件：

**What we want to do is input XInit.lua of the X system under the line above:**

**dofilepath("data:leveldata/multiplayer/resdata/XInit.lua")**

类似的，打开ui\pings\pings.lua（没有则复制过来），在**文件末尾**引用XPing.lua：

**Similarly, open ui\pings\pings.lua(ctrl c+v if you don't have it), input the following at the end of this file:**

**dofilepath("data:leveldata/multiplayer/resdata/XPing.lua")**

**此外，对于每一种需要添加特质的舰船，都需要在其ship文件中引用X系统自定义代码文件如下：**

**Besides, for those ships that need X System's characteristics, we need to load X System's custom code in their .ship files:**

**addCustomCode(NewShipType,<CustomCodePath>,<FunctionLoad>,<FunctionCreate>,<FunctionUpdate>,<FunctionDestroy>,<CustomGroup>, <iRefreshTime>)**

|  |  |
| --- | --- |
| <CustomCodePath>： | 该型船只调用的自定义代码文件路径。如果船只仅使用X System提供的特质，则填入"data:leveldata/multiplayer/resdata/CustomCode.lua"即可。  **The filepath of this ship's custom code, if the ship only have characteristics that X System provides, then input:** "data:leveldata/multiplayer/resdata/CustomCode.lua" |
| <FunctionLoad>： | 该型船只在游戏中加载时呼叫的函式名称，X System目前不涉及该函式，故填""。  **The function to be called when this ship is loaded in the game. X System do not need this at the moment, so just input: "".** |
| <FunctionCreate>： | 该型船只第一次在游戏中出现时呼叫的函式名称，X System目前不涉及该函式，故填""。  **The function to be called when this ship is spawned/created in the game. X System do not need this at the moment, so just input: "".** |
| <FunctionUpdate>： | 该型船只在游戏中反复呼叫的函式名称，对应CumtomCode.lua文件中的Update\_X，对于只使用X System提供的特质的船只，填"Update\_X"。  **The function to be called again and again while this ship is in the game. For customcode.lua, it is Update\_X, if the ship only have characteristics that X System provides, then input: "Update\_X".** |
| <FunctionDestroy>： | 该型船只死亡时呼叫的函式名称，对应CumtomCode.lua文件中的Destroy\_X，对于只使用X System提供的特质的船只，填"Destroy\_X"。  **The function to be called when this ship is destroyed in the game. For customcode.lua, it is Destroy\_X, if the ship only have characteristics that X System provides, then input: "Destroy\_X".** |
| <CustomGroup>： | 以上函式中的参数CustomGroup的值，名字为该值的SobGroup包含且仅包含该船只。X system要求该值必须与船名及Units列表中的名称（见下文）一致（包括大小写）。  **Name of the SobGroup that will contain the ship. Make it identical to the ship's name, especially those in the Unit list.** |
| <iRefreshTime>： | 游戏中呼叫<FunctionUpdate>的时间间隔（单位秒）。对于特质复杂的船只，可以适当将该值设大一些以减轻系统压力。该值同时还是某些特质中的时间单位。  **Time between two calls of <FunctionUpdate>, in seconds. For those ships whose characteristics are complicated, you can make this a little bigger. It is also a unit of time in some characteristics.** |

注**Note**：

X系统的CustomCode为想要在应用了X系统的单位上添加额外的CustomCode的Modder提供了接口，该部分内容不适合Mod初学者，因此请有此需求的Modder参阅X系统高级教程。

**X System has provided an entrance for modders who want to add more custom codes to ships that have already applied X System, please check the senior tutorial of X System.**

## 6.3加载和启用 Load and enable

在之前的步骤全部完成后，我们就需要在整个游戏规则的主入口中调用并启动系统。

原版的OnInit()函数格式如下：

**When we finish all the steps above, we want to initiate the system. The original OnInit() function is as the example below.**

**function OnInit()**

**……**

**end**

我们只需要调用X System的入口函数XInit()就能够加载并启动整个系统。完成后如下：

**Now we only need to input the X system's entry function so that we can enable the system. The function should now look like the one below.**

**function OnInit()**

**……**

**XInit("reslist.lua","reslist2.lua")**

**end**

注："reslist.lua"和"reslist2.lua"可替换为任何修改后的文件名以实现不同规则下加载不同的单位信息和科技信息。

**Note:"reslist.lua" and "reslist2.lua" can be replaced by any other modified file name in order to have different information of ships and technologies in different gamemodes.**

这时系统已经被加载到游戏之中，剩下的就是填写配置文件以形成自己想要的功能逻辑。

**Now the system had been loaded into the game. All that's left is to configure files to form the function that you want.**

# 7.功能简介 Function summary

X系统包含两大主要功能模块，分别是单位特质模块和科技系统模块，简单来说，X系统能够实现对游戏中的单位行为及科技树进行更加复杂的个人定制，哪怕对毫无编程经验的modder来说都很容易。

**X system contains two main function models: the unit characteristics model and the technology model. In short, X system can make a very comlex customization of ships' behaviors and the technology tree. And it is simple even to modders without any experience of scripting.**

## 7.1单位特质模块 Unit Characteristics model

单位特质模块是本系统中最复杂的模块，包含多种功能和条件触发选项。单位特质能够让单位拥有特殊的能力或性质，这些特性大部分都是我们在ship文件中无法实现的功能。在之后的章节中我们会着重说明特质的编写方式。

单位特质模块需要单位列表支持，请务必完整填写单位列表后使用。

**The Unit Characteristics model is the most complex model in this system, it contains multiple function and trigger choices. Unit characteristics allow units to have special abilities or properties; most of these characteristics can't be achieved through editing of the "ship" files. Later in the manual we will talk about the how to write this code.**

**The Unit Characteristics model requires the units list; please configure the unit list completely before enabling this model**

## 7.2科技系统模块 Technology model

科技系统模块是一套复杂有效的对科技进行控制的功能模块。除科技本身和子系统条件外，还可基于玩家的船只、资源等综合信息定制科技树，并设置更为复杂的逻辑条件关系。使得科技冲突、开启条件更加丰富并且多样化。

**The technology model is a complex but effective technology-controlling system. It can set up a technology tree based on not just technology item and subsystems but also the informations about players'ships, RUs and many other things, making the condition of technology conflicting and unrestricting much more diversified.**

# 8.配置文件 File Configuration

## 8.1单位列表 Unit list

单位列表配置关系到reslist.lua中的Units列表。

单位列表是整个系统的核心配置，也是整个配置文件中最长最复杂的部分。

**Unit list configuration relates to Units in the file reslist.lua.**

**Unit list is the core of this configuration; it is also the most complex and longest part.**

### Units

Units列表中编写了所有具有特质的单位。格式如下：

**The Units list includes all the units that have characteristics. It is formatted as below:**

**Units=**

**{**

**{**

**Name = "Hgn\_AttackBomber",**

**Type = "Fighter",**

**Size = 5,**

**LifeBarType = 1,**

**Ability =**

**{**

**Characteristic =**

**{**

**},**

**Exemption =**

**{**

**},**

**},**

**},**

**{**

**Name = "Hgn\_Interceptor",**

**Type = "Fighter",**

**Size = 5,**

**Ability =**

**{**

**Characteristic =**

**{**

**},**

**Exemption =**

**{**

**},**

**},**

**},**

**}**

以上示例中填写了2个特质单位。

**The example above configured two characteristic units**

|  |  |
| --- | --- |
| Name | 代表单位的名称。  **Name should be the unit’s name** |
| Type | 代表单位的类型（暂时无用）。  **Type will be unit type [Not useable at the moment]** |
| Size | 代表单位编组数量，即一个单位中有多少个个体，常见于空军单位中。  **Size is how many units in a squad. eg fighter** |
| LifeBarType | 代表单位的血条状况，1为短血条，2为长血条，对于多编组单位（如各种战机），建议填0。（v1.1.3新增属性）  **LifeBarType is the type of ship's health bar, 1 is the short ones, 2 is the long ones, for squads that have more than one ship, 0 is the recommended number. (New Feature since v1.1.3)** |
| Ability | Ability参数是一个复合结构，其下有Characteristic和Exemption两个参数列表。  **Ability is a complex structure, which include Characteristic and Exemption lists** |
| Characteristic | Characteristic参数列表是我们之后要详细解说的特质系统的配置部分。  **Characteristic is the characteristic list which under the characteristic model, we will talk about it later of the manual.** |
| Exemption | Exemption参数列表目前处于废弃状态。  **Exemption is currently not used.** |

### 闲话 Notes

这里我们需要着重明确一点，凡是使用了X System提供的特质的单位，必须存在于Units列表中。

**All units that have characteristics provided by X System, must exist in the Units list.**

同时我还要强调一件事， Units列表还只是特质系统的预热，单位的特质能力将会填写到单位的Characteristic参数列表中。

**Also I want to say that the unit list is only the base of the characteristic model. The units' characteristics will be written to that units' characteristic list.**

置空方式：

**Leave blank as:**

**Units = {}**

## 8.2单位特质 Units' characteristics

### 8.2.1做好心理准备 Prepare yourself

我们终于阅读到本文档的这个部分了，事实上，这只是个开始。是的，之前那些全都只是些开胃菜，根本谈不上多么特别的功能。这些旁枝末叶的东西在单位特质模块面前全都是些渣渣。

**We are finally getting this part of the manual, in fact it is the beginning. Yes, all those before are all appetisers against this, they are nothing special, they are just some amateurs in front of the unit characteristic model.**

之所以我要先介绍那些无关紧要的功能。

**Why would we talk about stuff that are not that necessary?**

一是让大家对我所设计的配置格式有一定的了解和熟悉，能够更快的理解特质的编写方式。

**First, we would like you to have a basic understanding and familiarize of the configuration format I designed; it helps you understand the way of characteristics**' **writing more quickly.**

二是你现在可以去吃个饭或者喝个咖啡什么的，调整精神，然后回来迎接地狱或天堂。

**Second, you could go have a cup of coffee or snacks, get a boost to face off against the heaven or hell that awaits you.**

### 8.2.2我准备好了 I'm ready

当你一切就绪，我们就开始单位特质的说明。首先我要介绍我们将要见到的第一个特质：OneselfChangePower。我将利用这个特质来全面说明整个特质配置的编写方式和逻辑。

当然，我们还需要一个具有这个特质的单位来整体展示特质在Characteristic参数列表中的编写规范。如下：

**When you are ready, we are going to talk about unit characteristics. First we are going to introduce the first characteristic, OneselfChangePower. We are using this characteristic to introduce the characteristic configuration and logic, generally and completely. Of course, we also need one unit that qualifies for this characteristic to generally show how to configure characteristic in Characteristic list, like the example below.**

**{**

**Name = "kpr\_AssaultFrigate",**

**Type = "Frigate",**

**Size = 1,**

**Ability =**

**{**

**Characteristic =**

**{**

**{**

**"OneselfChangePower", --参数D1**

**"****Other", --参数D2**

**{**

**{**

**{"Tactic",1}, --参数D3**

**}**

**},**

**{**

**{"Tech1","Tech2",}, --参数D4**

**{"Tech3",},**

**},**

**{**

**{"Subsystem1",}, --参数D5**

**},**

**{{"Madstate1",},{"Effect1","Effect2",},2,1}, --参数D6**

**{{},{},0,0}, --参数D7**

**{{},{},2,1}, --参数D8**

**{1,1,}, --参数D9**

**{"Weapon\_TurretTop", "Weapon\_TurretTop1", }, --参数D10**

**{"Weapon\_TurretTopA", "Weapon\_TurretTop1A", }, --参数D11**

**},**

**},**

**Exemption =**

**{**

**},**

**},**

**},**

这里我们省略了Unit列表，单独观察这一个单位的这一个特质。同时我给特质的每一个参数都做了标记，以方便在之后的解说中随时引用他们。

**Here we bypass the unit list and only observe this one unit's characteristics. Also I have marked every parameter of this characteristic above, so we will be easier to talk about them without quoting the whole thing.**

### 8.2.3通用参数 General Configurations

参数1到9统称为通用参数，在任何一个特质中都有这些参数并且这些参数的意义都相同，不因特质不同而改变含义和格式。

**D1 to D9 are all general configurations, every characteristic has these configurations.They have the same meanings and formats in every characteristic.**

在这里我建议你打起十二分精神，仔细阅读通用参数的使用。如果这里你无法完全理解，我建议不要再往下看了，下面的内容远没有通用参数部分来的复杂和重要。对通用参数的理解程度决定了特质功能使用的程度。

**Here I need you to give 120% of your attention to learn how to use the configuration. If you can't understand these I don**'**t recommend you continue, it only would get more complicated and important. How well you understand this configuration determines how well you can use the characteristics model.**

#### 参数1 D1

特质的名称，每一个特质对应一个不同的名称，在通用参数后会依次介绍每一种特质以及特质对应的特殊参数的使用。

**D1 is the characteristic's name, every characteristic had a different name or code. Later in this general characteristics section we will talk about how to use different characteristics.**

#### 参数2 D2

特质中与数字相关的运算方式，会在特质介绍中依次介绍具备数学运算的特质所拥有的运算方式，我们用于示例的特质OneselfChangePower只有Other一种运算方式。

**Number calculation that is related to the characteristic, we will talk about different calculation for different characteristic, but for the example we have used, OneselfChagePower, only has "other" as this calculation.**

#### 参数3 D3

这是整个特质系统中最核心最重要的参数之一，是触发特质的条件参数。

**This is the most important part of the configuration, this is the trigger.**

当特质满足了科技的前提；子系统的前提后，如何触发特质就成了重点。

**When technology and sub-system requirements are met, how to trigger the characteristic will became the point.**

我们可以去设置一个开关，让玩家能够在游戏中随时开启或关闭特质，而不像一般研究那样一旦获得就永远存在了。而这是通过X系统提供的诸多条件判断来实现的。

**We could have it enable and disable at any time unlike researched technology. This comes true thanks to many condition judgments provided by X System.**

首先，每个条件判断都有一个基本的书写格式，非常简单，如下:

**First of all, every judgment have a basic format and it's very simple:**

{<条件名称>,<条件参数1>,<条件参数2>,……},

**{<ConditionName>,<ConditionValue1>,<ConditionValue2>,……},**

条件名称决定了这是一个什么样的条件判断，是判断单位的生命值，还是单位在干什么等等，条件参数则决定条件在何种状态下得到满足。对于不同的条件判断，后面的条件满足值的格式也不同，下面我们会逐一讲解条件判断都有哪些，书写格式是怎样的。

**<ConditionName> tells the system what condition to judge, e.g. a judgment of unit's health condition or what the unit is doing. For different judgments, there's a specific format of <ConditionValue>. I'll show you what judgments do they have and what the formats look like.**

最简单的一类是布尔类型，条件的结果只有两种回答，是或否，分别用数字1和0表示。举个例子:

**The simplest among them is boolean-type, the result of the boolean-type judgment can only be yes or no. When we write them down, they become 1 or 0. For example:**

**{"IsFiring",1},**

这是一个布尔类型的条件判断式，意思是若使特质满足，则单位必须处于开火状态。是不是很容易理解呢?如果特质的触发条件是单位不能处于开火条件，则只要把1改为0就行了:

**This is a boolean-type judgment, it means the characteristic will be active when the unit is firing. Isn't it very easy to understand? If you want the characteristic to be disabled when unit is firing, just change 1 to 0:**

**{"IsFiring",0},**

注意，对于布尔类型的条件判断，条件参数不能是0和1之外的任何东西。

**Please notice that for all boolean-type judgments, the <ConditionValue> must be 0 or 1.**

所有布尔类型的条件判断及其含义如下:

**All boolean-type judgment and their meanings are:**

|  |  |
| --- | --- |
| **条件：**  **Condition:** | **合法的条件参数及其含义：**  **Possible condition values and their meanings:** |
| **IsFiring** | **1：**单位处于开火状态；**0：**单位不处于开火状态  **Means: 1: Unit is firing; 0:Unit is not firing** |
| **IsUnderAttack** | **1：**单位正遭受攻击；**0：**单位未遭受攻击  **Means: 1: Unit is under attack; 0:Unit is not under attack** |
| **DefenseFieldOpen** | **含义：1：**单位开启了防御场（T）；**0：**单位没有开启防御场  **Means: 1: Unit's defense field is open[T]; 0: Unit's defense field is not open** |
| **CloakGeneratorOpen** | **含义：1：**单位开启了隐形场（C）；**0：**单位没有开启隐形场  **Means: 1: Unit's cloak generator is open; 0:Unit's cloak generator is not open** |
| **IsMoving** | **含义：1：**单位处于移动状态（M）；**0：**单位不处于移动状态  **Means: 1: Unit is moving[M]; 0:Unit is not moving** |
| **IsAttacking** | **含义：1：**单位处于攻击状态（正在执行攻击命令）；**0：**单位不处于攻击状态  **Means: 1: Unit is attacking[given attack order]; 0:Unit is not attacking** |
| **IsGuarding** | **含义：1：**单位处于护航状态（G）；**0：**单位不处于护航状态  **Means: 1: Unit is guarding[G]; 0:Unit is not guarding** |
| **IsRepairing** | **含义：1：**单位处于维修状态（Y）；**0：**单位不处于维修状态  **Means: 1: Unit is repairing[Y]; 0:Unit is not repairing** |
| **IsRepairingAndLatched** | **含义：1：**单位处于维修状态（Y）；**0：**单位不处于维修状态  **Means: 1: Unit is repairing and latched; 0:Unit is not repairing and latched** |
| **IsBeingRepairedAndLatched** | **1：**单位处于被维修状态；**0：**单位不处于被维修状态  **Means: 1: Unit is being repaired; 0:Unit is not being repaired** |
| **IsHarvesting** | **含义：1：**单位处于采集状态（H）；**0：**单位不处于采集状态  **Means: 1: Unit is harvesting[H]; 0:Unit is not harvesting** |
| **IsLayingMines** | **含义：1：**单位处于布雷状态（N）；**0：**单位不处于布雷状态  **Means: 1: Unit is lying mines[N]; 0:Unit is not lying mines** |
| **IsCapturing** | **含义：1：**单位处于捕捉状态；**0：**单位不处于捕捉状态  **Means: 1: Unit is capturing; 0:Unit is not capturing** |
| **IsBeingCaptured** | **含义：1：**单位正在被敌人捕捉；**0：**单位不处于被敌人捕捉的状态  **Means: 1: Unit is being captured; 0:Unit is not being captured** |
| **IsCapturingAndLatched** | **含义：1：**单位处于捕捉状态并且已经附着敌舰；**0：**单位不处于附着敌舰的捕捉状态  **Means: 1: Unit is capturing and latched; 0:Unit is not capturing and latched** |
| **IsBeingCapturedAndLatched** | **含义：1：**单位正处于被捕捉状态并且对方已经附着该舰；**0：**单位不处于上述状态  **Means: 1: Unit is being captured by any ship which has already latched; 0:Unit is not in this condition** |
| **IsDocking** | **含义：1：**单位正在停泊（D）；**0：**单位没有进行停泊  **Means: 1: Unit is docking[D]; 0:Unit is not docking** |
| **IsAcceptDocking** | **含义：1：**有船只向该单位停泊；**0：**没有船只向该单位停泊  **Means: 1: At least one ship is docking to the unit; 0:No ship is docking to the unit** |
| **IsJumping** | **含义：1：**单位正在超空间跳跃（J）；**0：**单位没有进行超空间跳跃  **Means: 1: Unit is hyperspace jumping[J]; 0:Unit is not hyperspace jumping** |
| **IsRetiring** | **含义：1：**单位正在退役；**0：**单位没有退役  **Means: 1: Unit is retiring; 0:Unit is not retiring** |
| **IsCloaked** | **含义：1：**单位处于隐形状态；**0：**单位不处于隐形状态  **Means: 1: Unit is cloaked; 0:Unit is not cloaked** |
| **IsInRealSpace** | **含义：1：**单位处于常规空间；**0：**单位不处于常规空间（正在跳入/跳出或位于超空间）  **Means: 1: Unit is in real space; 0:Unit is not in real space[jumping in/out or in hyperspace]** |
| **IsInHyperspace** | **含义：1：**单位处于超空间；**0：**单位不处于超空间（正在跳入/跳出或位于常规空间）  **Means: 1: Unit is in hyperspace; 0:Unit is not in hyperspace[jumping in/out or in real space]** |
| **IsDocked** | **含义：1：**单位处于停泊状态；**0：**单位不处于停泊状态  **Means: 1: Unit is docked; 0:Unit is not docked** |

第二类条件判断是多值类型，即条件可能值多于两个，而不仅局限于0和1。

**The second judgment type is multivalue-type, whose number of possible <ConditionValue> is more than two, not just 0 or 1.**

这类条件判断的一个特点是通过取其条件满足值的相反数来实现反转解释。

**In this case, you will see that the judgement reverse their meaning by using opposite ConditionValue number.**

最直接的例子就是判断单位所采取的战术:

**The straightest example is a tactic judgment:**

**{"Tactic",-1},**

这个的意思是单位不处于主动状态则条件满足，多值类型的格式和布尔类型很像，这里不多赘述，所有多值类型的条件判断及其允许的条件满足值及其含义如下:

**This means the unit must not at aggressive tactic for the condition to trigger. Since multivalue-type and booleam-type are quite similar, I'll only list all multivalue-type judgments and their possible <ConditionValue>below without further explainations:**

|  |  |
| --- | --- |
| **条件：**  **Condition:** | **合法的条件参数及其含义：**  **Possible condition values and their meanings:** |
| **Tactic** | **3:**被动战术**passive tactic**  **2:**防御战术**defensive tactic**  **1:**主动战术**aggressive tactic**  **-1:**不处于主动战术(防御或被动战术) **not at aggressive tactic(defensive or passive tactic)**  **-2:**不处于防御战术**not at defensive tactic**  **-3:**不处于被动战术**not at passive tactic** |
| **ShieldChanged** | **3:**当前护盾值大于上一次护盾值（护盾值增加）**Current shield value is greater than last recorded value (shield value increased)**  **2:**当前护盾值大于等于上一次护盾值（护盾值增加或未变）**Current shield value is equal to or greater than last recorded value (shield value increased or not changed)**  **1:**当前护盾值等于上一次护盾值（护盾值未变）**Current shield value is equal to last recorded value (shield value not changed)**  **-1:**当前护盾值不等于上一次护盾值（护盾值改变）**Current shield value is not equal to recorded value (shield value changed)**  **-2:**当前护盾值小于等于上一次护盾值（护盾值减少或未变）**Current shield value is equal to or less than last recorded value (shield value decreased or not changed)**  **-3:**当前护盾值小于上一次护盾值（护盾值减少）**Current shield value is less than last recorded value (shield value decreased)** |
| **ArmorChanged** | **3:**当前护甲值大于上一次护甲值（护甲值增加）**Current armorvalue is greater than last recorded value (armor value increased)**  **2:**当前护甲值大于等于上一次护甲值（护甲值增加或未变）**Current armorvalue is equal to or greater than last recorded value (armor value increased or not changed)**  **1:**当前护甲值等于上一次护甲值（护甲值未变）**Current armorvalue is equal to last recorded value (armor value not changed)**  **-1:**当前护甲值不等于上一次护甲值（护甲值改变）**Current armorvalue is not equal to recorded value (armor value changed)**  **-2:**当前护甲值小于等于上一次护甲值（护甲值减少或未变）**Current armorvalue is equal to or less than last recorded value (armor value decreased or not changed)**  **-3:**当前护甲值小于上一次护甲值（护甲值减少）**Current armorvalue is less than last recorded value (armor value decreased)** |
| **HealthChanged** | **3:**当前生命值大于上一次生命值（生命值增加）**Current health value is greater than last recorded value (health value increased)**  **2:**当前生命值大于等于上一次生命值（生命值增加或未变）**Current health value is equal to or greater than last recorded value (health value increased or not changed)**  **1:**当前生命值等于上一次生命值（生命值未变）**Current health value is equal to last recorded value (health value not changed)**  **-1:**当前生命值不等于上一次生命值（生命值改变）**Current health value is not equal to recorded value (health value changed)**  **-2:**当前生命值小于等于上一次生命值（生命值减少或未变）**Current health value is equal to or less than last recorded value (health value decreased or not changed)**  **-3:**当前生命值小于上一次生命值（生命值减少）**Current health value is less than last recorded value (health value decreased)** |
| **IsControlledBy** | **5:**专家AI **Unit controlled by expert AI**  **4:**困难AI**Unit controlled by hard AI**  **3:**标准AI **Unit controlled by standard AI**  **2:**简单AI**Unit controlled by easy AI**  **1:**玩家**Unit controlled by human player**  **-1:**AI**Unit controlled by AI**  **-2:**非简单AI**Unit not controlled by easy AI**  **-3:**非标准AI**Unit not controlled by standard AI**  **-4:**非困难AI**Unit not controlled by hard AI**  **-5:**非专家AI**Unit not controlled by expert AI** |

除此之外该类型的条件判断还有：

**Other judgments of this type are:**

|  |  |  |
| --- | --- | --- |
| **条件：**  **Condition:** | **合法的条件参数：**  **Possible condition values** | **含义：**  **Meanings：** |
| **IsCharacteristicActive** | X可取任意不为0的整数。  **X can be any integer but 0.** | 当X大于0时，表示单位的第X个特质处于激活状态，小于0则表示单位的第（-X）个特质不处于激活状态。  对于两个或两个以上条件复杂且完全相同或相反的特质，使用该判断能够明显提高判断效率。  **If X is greater than 0, it means the unit's Xth characteristic is active, if X is less than 0, it means the unit's (-X)th characteristic is not active.**  **{"IsCharacteristicActive",-2,},**  该条件为：当单位的第二个特质未被激活时，条件满足。  **This means, the condition will be triggered when the unit's second characteristic is not active.** |
| **IsShipIdioStateKeyOn** | X可取任意不为0的整数。  **X can be any integer but 0.** | 当X大于0时，表示该单位ID为X的ShipIdioStateKey处于激活状态，小于0则表示该单位ID为（-X）的ShipIdioStateKey不处于激活状态。  **If X is greater than 0, it means the unit's ShipIdioStateKey whose ID equals X is active, if X is less than 0, it means the unit's ShipIdioStateKey whose ID equals (-X) is not active.**  单位的ShipIdioStateKey通过[ShipIdioStateKeyOn](#_ShipIdioStateKeyOn)，[ShipIdioStateKeyOff](#_ShipIdioStateKeyOff)和[ShipIdioStateKeyTempOn](#_ShipIdioStateKeyTempOn)特质进行控制。  **Unit's ShipIdioStateKey is controlled by characteristic "**[**ShipIdioStateKeyOn**](#_ShipIdioStateKeyOn)**","**[**ShipIdioStateKeyOff**](#_ShipIdioStateKeyOff)**" and "**[**ShipIdioStateKeyTempOn**](#_ShipIdioStateKeyOn)**".** |
| **IsGlobalStateKeyOn** | X可取-100~100的非零整数。  **X can be any integer between -100 and 100 but 0.** | 当X大于0时，表示ID为X的GlobalStateKey处于激活状态，小于0则表示ID为（-X）的GlobalStateKey不处于激活状态。  **If X is greater than 0, it means theGlobalStateKey whose ID equals X is active, if X is less than 0, it means the GlobalStateKey whose ID equals (-X) is not active.**  GlobalStateKey通过X系统公共函数[xSetGlobalStateKey()](#_xSetGlobalStateKey)进行控制。  **GlobalStateKey is controlled by** [**xSetGlobalStateKey()**](#_xSetGlobalStateKey) **function.** |
| **HealthPercentage** | X=-1~1。  **A number X between -1 and 1.** | 单位当前的生命值。对于正数，含义是生命值大于等于X%，对于0或负数，含义是生命值小于等于（-X）%。  **Means: The unit's health percentage. If X is greater than 0, it means unit's health percentage equals to or greater than X, if X equals to or less than 0, it means unit's health percentage equals to or less than(-X).** |
| **FuelPercentage** | X=-1~1。  **A number X between -1 and 1.** | 单位当前的燃料值。对于正数，含义是燃料值大于等于X%，对于0或负数，含义是燃料值小于等于（-X）%。  **Means: The unit's fuel percentage. If X is greater than 0, it means unit's fuel percentage equals to or greater than X, if X equals to or less than 0, it means unit's fuel percentage equals to or less than(-X).** |
| **ShieldPercentage** | X=-1~1。  **A number X between -1 and 1.** | 单位当前的护盾值。对于正数，含义是护盾值大于等于X%，对于0或负数，含义是护盾值小于等于（-X）%。  **Means: The unit's shield percentage. If X is greater than 0, it means unit's shield percentage equals to or greater than X, if X equals to or less than 0, it means unit's shield percentage equals to or less than(-X).** |
| **ArmorPercentage** | X=-1~1。  **A number X between -1 and 1.** | 单位当前的护甲值。对于正数，含义是护甲值大于等于X%，对于0或负数，含义是护甲值小于等于（-X）%。  **Means: The unit's armorpercentage. If X is greater than 0, it means unit's armor percentage equals to or greater than X, if X equals to or less than 0, it means unit's armor percentage equals to or less than(-X).** |
| **SquadronMember** | 任意整数X。  **X can be any integer.** | 编组单位的编组数量。对于正数，含义是编组数量大于等于X，对于0或负数，含义是编组数量小于等于（-X）。  **Means: The number of unit's squadron member. If X is greater than 0, it means squadron member equals to ormore than X, if X equals to or less than 0, it means squadron member equals to or less than (-X).** |
| **ShipLevel** | 任意整数X。  **X can be any integer.** | 单位当前的等级。对于正数，含义是等级大于等于X，对于0或负数，含义是等级小于等于（-X）。  **Means: The unit's level. If X is greater than 0, it means unit's level equals to or higher than X, if X equals to or less than 0, it means unit's level equals to or lower than (-X).** |
| **ActualSpeed** | 任意数X。  **X can be any number.** | 单位当前的移动速度。对于正数，含义是速度大于等于X，对于0或负数，含义是速度小于等于（-X）。  **Means: The unit's speed.If X is greater than 0, it means unit's speedequals to orfaster than X, if X equals to or less than 0, it means unit's speed equals to or slower than (-X).** |
| **LockingSignal/ BeingLockedSignal/ GuardingSignal/ BeenGuardedSignal/ RangeSignal/ DockingSignal/MothershipSignal** | 任意整数X。  **X can be any integer.** | 对于正数，含义是单位接收到ID等于X的信号，对于负数，含义是单位未接收到ID等于（-X）的信号。  **Means: If X is greater than 0, it means unit is receiving the specific kind of signal whose ID is X, if X is less than 0, it means unit is not receiving the specific kind of signal whose ID is (-X).** |
| **IsInsideNebula** | 任意不为0的整数X。  **X can be any integer but 0.** | 对于正数，含义是单位在ID等于X的辐射云中，对于负数，含义是单位不在ID等于（-X）的辐射云中。  **Means: If X is greater than 0, it means unit is inside nebula whose ID is X, if X is less than 0, it means unit is not inside nebula whose ID is (-X).** |
| **IsInsideDustCloud** | 任意不为0的整数X。  **X can be any integer but 0.** | 对于正数，含义是单位在ID等于X的尘云中，对于负数，含义是单位不在ID等于（-X）的尘云中。  **Means: If X is greater than 0, it means unit is inside dustcloud whose ID is X, if X is less than 0, it means unit is not inside dustcloud whose ID is (-X).** |
| **IsDoingAbility** | X可取-29~29的非零整数  **X can be any integer between -29 to 29 but 0.** | 判断单位是否在使用某项能力，X的绝对值为能力ID，见表  **Means: If the unit is using the abbility that |X| represents (see grid below) or not.**  X取负数时，表示单位没在使用该能力  **When X is less than 0, it means the unit is not using the ability.**  注：该条件判断囊括了部分布尔类型条件判断，两种判断方式均可使用  **Note: this judgement includes some of the boolean-type judgements, either of them can be used.** |
| **CanDoAbility** | X可取-29~29的非零整数  **X can be any integer between -29 to 29 but 0.** | 判断单位是否具有某项能力，X的绝对值为能力ID，见表  **Means:If the abbility that |X| represents (see grid below) of the unit is activated or not.**  X取负数时，表示单位不具有该能力  **When X is less than 0, it means that ability of the unit is not activated.** |

舰船能力表**Ship Ability Grid**

|  |  |  |
| --- | --- | --- |
| **ID** | **名称Name** | **说明Meaning** |
| 1 | AB\_Move | 移动 |
| 2 | AB\_Attack | 攻击 |
| 3 | AB\_Guard | 保护 |
| 4 | AB\_Repair | 修理 |
| 5 | AB\_Cloak | 隐形 |
| 6 | AB\_Harvest | 采集 |
| 7 | AB\_Mine | 布雷 |
| 8 | AB\_Capture | 捕捉 |
| 9 | AB\_Dock | 停泊 |
| 10 | AB\_AcceptDocking | 接收停泊 |
| 11 | AB\_Builder | 建造 |
| 12 | AB\_Stop | 停止 |
| 13 | AB\_Hyperspace | 超空间 |
| 14 | AB\_Parade | 列队 |
| 15 | AB\_FormHyperspaceGate | 连接超空间门 |
| 16 | AB\_HyperspaceViaGate | 通过超空间门 |
| 17 | AB\_SensorPing | 感应器扫描 |
| 18 | AB\_SpecialAttack | 特殊攻击 |
| 19 | AB\_Retire | 退役 |
| 20 | AB\_DefenseField | 防御场 |
| 21 | AB\_DefenseFieldShield | 防御场2（任务中） |
| 22 | AB\_HyperspaceInhibitor | 重力井 |
| 23 | AB\_Salvage | 收集残骸 |
| 24 | AB\_Scuttle | 自毁 |
| 25 | AB\_UseSpecialWeaponsInNormalAttack | 普通攻击中使用特殊武器 |
| 26 | AB\_Steering | 姿控 |
| 27 | AB\_Targeting | 锁定 |
| 28 | AB\_Sensors | 感应器 |
| 29 | AB\_Lights | 灯光 |
| 30 | AB\_SalCap | 一代捕捉 |
| 31 | AB\_Custom | 自定义指令 |
| 32 | AB\_Kamikaze | 神风 |

至此，我们已经认识了很多条件判断了，在介绍后面两类条件判断之前，首先我想讲一下条件的组合。

**So far, we've already learned many judgments, before introducing next two types, I'd like to show you how condition judgments are combined.**

我们可以看到示例中参数3是这样的：

**Let us have a look back at D3.**

**{**

**{**

**{“Tactic",1,},**

**},**

**},**

看起来明明只有一个条件，为什么要把格式搞的这么复杂呢。这就涉及到条件的组合。

**It's just one condition, why need make it so complex? That is what I’m going to explain now.**

我们都知道，多个条件在组合的时候的关系分为与和或两种。与条件代表所有条件都必须为真整个条件才为真，或条件代表只要有一个条件为真则整个条件就为真。正是基于这种思想，我把条件设计成条件组的形式。

**We know multiple conditions could relate to one or more others. All that conditions have to be achieved at once in order to trigger, or just one only.**

从示例上来看，整个条件参数中只有一个条件组，这个条件组中只有一个条件就是{"Tactic",1}。那么这个特质的启动开关就是单位进入主动状态。现在我们对这个条件进行一些改变：

**For the example we see that there is only one condition, Tactic=1, than when unit goes into an aggressive status the condition will trigger. Now let’s change it up a little bit.**

**{**

**{**

**{"Tactic",1},**

**{"IsFiring",1,},**

**},**

**},**

可以看到添加了一个条件IsFiring=1，与原先的Tactic=1条件组合后的条件为：**单位处于主动状态并且正在开火**这个特质才会启动，这就是条件的与关系。接下来我们继续对这个条件进行一些编辑来说明条件的或关系：

**When we add condition IsFiring=1. When it go with Tactic=1, it become units need to be aggressive status AND is firing before the characteristic will active. We could change it a little more to make it better.**

**{**

**{**

**{"Tactic",1},**

**{"IsFiring",1,},**

**},**

**{**

**{"Tactic",3,},**

**},**

**},**

可以看到这时候多了一个条件组，新的条件组内有一个条件Tactic=3。整个特质开启的条件是：**（单位处于主动状态并且正在开火）或（单位处于消极状态）**

单位只要满足两个条件组其中的一组条件便可开启特质。

**We could see that we add one more condition set, this involve Tactic=3. Now it becomes so that units need to be [aggressive status and is firing] OR [at passive status]**

讲解到这里，我想大家应该对条件参数的运用有了一些了解，如果你没有看懂，我建议你回到前面继续看直到看懂为止。如果你看懂了，我们就开始讲一些更复杂的东西。

**We are going to stop here, please read through the section above a couple of times until you really get it. If you do we are going to talk about something more complicated.**

---------------------------------------------------------------------------------------------------------------------------------

第三类条件判断，其条件参数不是数值，而是和条件名称一样的字符串。请看下面的例子:

**The ConditionValue of the third kind of judgement type is a string, same as condition name, behold:**

**{"RaceName",0,"Hiigaran",},**

这是一个单位归属玩家的种族判断，除条件名称外，后面还有两个参数，其中最后一个是该条件判断的参数，即判断该单位所属玩家是否为希格拉族，那么那个0又是什么意思呢？

**This is to judge which race is the player owns this unit belongs to. It has two values after the condition name. The last one is the ConditionValue, which is a string as we explained, but what's that 0 means?**

这个0是反转参数，决定该条件是否反转解释。上面的数值判断中，布尔类型通过参数是0还是1来体现条件的反转，多值类型则通过正负，由于字符串不是数值，因此只能额外引入反转参数，该参数也是布尔类型，取值只能为0或1，为0时，条件不反转，为1时，条件反转解释。所以示例条件的意思是，判断玩家所属种族是否为希格拉。如果将反转参数从0改为1，则意思反转，变为：判断玩家所属种族是否不为希格拉。

**That 0 is the ReverseValue, it tells the system whether or not the meaning of this judgement is reversed. The ConditionValue itself is the ReverseValue in boolean-type judgements, and it goes opposite to reverse the meaning of multivalue-type judgements, but when it becomes a string, it cannot reverse the judgement's meaning any more, that's why we need a ReverseValue here. The ReverseValue is also boolean-type, when it is set to 0, the meaning is not reversed, when it is set to 1, the meaning is reversed. So the example above means: if the player owns this unit belongs to Hiigara, and if we set the ReverseValue to 1, it means: if the player owns this unit do not belong to Hiigara.**

所有字符类型的条件判断及其含义如下：

**All string-type judgments and their meanings are:**

|  |  |  |
| --- | --- | --- |
| **条件：**  **Condition:** | **合法的格式及条件参数：**  **Possible format and condition values** | **含义：**  **Meanings：** |
| **RaceTag** | **X**  任意String  **X should be a string.** | 判断单位所属玩家的种族是否拥有标签X  **If the race of player who own the unit have tag X.** |
| **RuleName** | **X**  可取任意游戏模式名  **X can be any Game Rule name.**  **[Race\_Paths]** | 判断当前游戏模式名称是否为X  **If the current gamemode is X.** |
| **RaceName** | **X**  可取任意种族名。  **X can be any possible Race Name.**  **[Symbolic\_Name]** | 判断单位所属玩家种族名是否为X  **If the owner of the unit is Race X.** |
| **IsAttackingFamily** | **"<AttackFamily1>,…< AttackFamilyN>,"**  所有可能的AttackFamily（在family.lua中定义）。  **Any possible AttackFamily**  **[define in family.lua]** | 意思是单位的攻击目标中，至少有一个单位的AttackFamily在字符串中列出  **If the unit's attacking targets have at least a ship whose AttackFamily is listed in the string.** |
| **IsAttackingShipType** | **"<ShipType1>,…< ShipType N>,"**  所有可能的舰船名称。  **Any possible ship name.** | 意思是单位的攻击目标中，至少有一个单位的名称在字符串中列出  **If the unit's attacking targets have at least a ship whose name is listed in the string.** |
| **IsBuilding** | **X**  任意舰船名  **Any possible ship name.** | 判断单位是否正在建造X  **If the unit is building ship X.** |
| **HasResearch** | **X**  研究项名称  **Any possible research item.** | 判断单位所属玩家是否研究了科技X  **If player own this unit had researched technology X.** |
| **HasUpgrade** | **X**  升级项名称  **Any possible upgrade item.** | 判断单位所属玩家是否升级了科技X  **If player own this unit had upgraded technology X.** |
| **HasSubsystem** | **X**  舰船子系统点名称  **Name of a ship's hardpoint.** | 判断单位的子系统点X上是否有子系统  **If there is a subsystem on unit's hardpoint X** |
| **IsInSobGroup** | **X**  SobGroup名  **Name of a SobGroup** | 判断单位是否在名为X的SobGroup中  **If the unit is within SobGroup X** |

除以上类型的条件判断外，还有一些条件参数多于一个，在这里统一介绍如下：

**There are more condition judgements that have more than one condition value, and we'll learn about them here:**

|  |  |  |
| --- | --- | --- |
| **条件：**  **Condition:** | **合法的格式及条件参数：**  **Possible format and condition values** | **含义：Meanings：** |
| **RaceNumber** | **{"RaceNumber",<ReverseValue>,<IndexString>,<ExpectNumber>}** | 判断单位所属玩家的种族名为<IndexString>的属性值是否为<ExpectNumber> |
| **RaceString** | **{"RaceString",<ReverseValue>,<IndexString>,<ExpectString>}** | 判断单位所属玩家的种族名为<IndexString>的属性值是否为<ExpectString> |
| **GameSettingNumber** | **{"GameSettingNumber",<ReverseValue>,<IndexString>,<ExpectNumber>}** | 判断当前模式某项设置项及设置值，数值类型。如条件为“初始RU=3000”，则应写作：{GameSettingNumber",0,"resstart",3000}, |
| **GameSettingString** | **{"GameSettingString",<ReverseValue>,<IndexString>,<ExpectString>}** | 判断当前模式某项设置项及设置值，字符串类型。如条件为“人口设置不为高人口”，则应写作：{"GameSettingString",1,"unitcap","large"}, |

到这里，所有的条件判断就都介绍完毕了。如果你不想要任何条件去触发特质，让特质变成一个被动生效的效果，置空条件参数便可：

**Now, all condition judgments have finally introduced to you. If you don't want to let any condition to achieve characteristic, let it be a passive characteristic by just leaving it blank as below.**

**{},**

条件参数的说明到此告一段落了，虽然篇幅不长，希望大家能够仔细阅读理解。当理解了条件参数之后，我接下来要说的科技和子系统前提就变的非常简单易懂了。

**The condition configure is over, it might be short, but hopefully you read it well. It will be easier later on the technology and sub-system.**

#### 参数4和参数5 D4 & D5

当我们有了条件参数后就有了开关，但是光有开关是不行的，我们还需要电力供应等先决条件。而参数4和参数5正是提供的这种电力供应（参数4为科技前提，参数5为子系统前提）。

在参数3中，我们已经有判断科技和子系统的条件判断：**HasResearch**和**HasSubsystem**，但是由于这两个条件使用频率明显更高，因此将二者单独分离出来可以简化参数3。我们一般只在必要的时候才用参数3判断科技和子系统条件。

在充分理解了条件参数后，解说科技和子系统前提就变的非常的简单了，我只需随便举几个例子大家就瞬间理解了：

**If the condition configuration is the trigger, D4 & D5 will be the power source. D4 is the technology requirement, and D5 is the sub-system requirement.**

**In D3, we already have them: HasResearch and HasSubsystem, but since they are used much more often, separating them can simplize D3 significantly. So generally, we use D3 to judge technology and subsysem only nesseary.**

**Now all we need is couple of examples.**

**{**

**{"Tech1","Tech2",},**

**{"Tech3",}**

**},**

**（拥有科技Tech1并且拥有科技Tech2）或（拥有科技Tech3）**

Tech1、Tech2等为科技的名称。

**[Research Tech 1 and Tech2] or [ Research Tech 3]**

**Tech 1 and Tech 2 are technology name**

**{**

**{"Hardpoint1",},**

**{" Hardpoint2",},**

**},**

**船只子系统点Hardpoint1或Hardpoint2上拥有子系统**

Hardpoint1、Hardpoint2为子系统点名称。

**Ships hardpoint Hardpoint1 or Hardpoint2 has subsustem**

**Hardpoint1 and Hardpoint 2 are ship's hardpoint names.**

科技和子系统也可以进行反转解释，反转的关键词为："CONTRARY"。例子如下：

**Technology and sub-system requirements could reverse too, by using "CONTRARY". It can reverse every item behind it. As shown below.**

**{**

**{"Tech1","CONTRARY","Tech2",},**

**{"Tech3",}**

**},**

**（拥有科技Tech1并且没有科技Tech2）或（拥有科技Tech3）**

**[Researched Tech1 and not researched Tech2] or [Researched Tech 3]**

**{**

**{"CONTRARY","Hardpoint1",},**

**{" Hardpoint 2",},**

**},**

**船只子系统点Hardpoint 1没有子系统或船只子系统点Hardpoint 2有子系统**

**[Ship's Hardpoint1 don't have subsystem] or [Hardpoint2 has subsystem]**

#### 参数6、参数7和参数8 D6, D7 and D8

这三个参数关系到特质开启和关闭时的特效和动画表现，是几乎纯外观性质的参数。同时这三个参数是基于同一个结构流程的参数，所以我把它们放在一起来进行说明。

首先我们要来说明一下特质在运行时经历的几个阶段：

**This 3 configuration is about how it looks, special effects, when characteristics activate and deactivate. So we put them together to be easier to understand:**

|  |  |  |  |
| --- | --- | --- | --- |
| **非激活阶段**  **Disabled stage** | **准备阶段**  **Preparatory stage** | **激活阶段**  **Active stage** | **解除阶段**  **Wind down stage** |

**非激活阶段**一般是单位经历时间最长的一个阶段，在科技、子系统、触发等前提条件未满足而使得特质无法生效的情况下特质都处于非激活阶段。

**Disabled stage is when units haven’t qualified for the characteristic yet**

**准备阶段**是特质已经具备了各种条件在实际生效前做一些预备的阶段。

**Preparatory stage means the time before the unit activates the characteristic**

**激活阶段**是特质实际生效的阶段。

**Active stage is when units have the characteristic activated**

**解除阶段**是通过触发条件关闭后进行一些收尾工作的阶段。

**Wind down stage is after the characteristic has finished**

通常来说我们并不关心非激活阶段单位该干什么，我们所关心的只有准备阶段、激活阶段和解除阶段单位会进行哪些视觉上的表现。而这三个阶段正好对应了这三个参数列表。

**Normally we don't care about the disabled stage, so we will talk about the preparatory, active, and wind down stages.**

参数6 准备阶段

**D6 preparatory stage**

参数7 激活阶段

**D7 active stage**

参数8 解除阶段

**D8 down stage**

接着来看示例中的参数6：

**Now let's look at D6:**

**{{" Madstate1",},{" Effect1"," Effect2",},2,1},**

这样书写并不方便我们在说明时进行观察，我将这个配置重新调整一下文本格式以便于观察和说明：

**This might seem to be hard to understand, why don't we format them the way they should be:**

**{**

**{**

**"Madstate1",**

**},**

**{**

**"Effect1",**

**"Effect2",**

**},**

**2,**

**1,**

**},**

第一部分为动画参数组，Madstate1为动画名称。

**The first part is the animation configuration set, Madstate1 is the animation name**

第二部分为特效参数组，Effect1、Effect2为特效名称。

**The second part is the special effect set, Effect 1 and Effect2 are effect names**

第三部分为阶段持续时长。

**The third part is how long the stage lasts**

第四部分为硬直参数（此阶段单位会无法移动保持硬直状态）。

**The fourth part is the cool down configuration, [This stage units won't move stay stiff]**

当然，光这样说根本无法把这几个参数列表和其中各种奇怪的参数组解释清楚，所以我准备借助其他一些方式来说明这些参数的填写方式。用一个大家都熟悉的单位来进行这几个阶段的模拟：Hgn\_DefenseFieldFrigate

**It might not make sense yet so why don't we use a unit that could represent this, the Hgn\_DefenseFieldFrigate.**

防御场护卫舰在启用T后会展开一个防御场，同时单位本身伴随动画表现。我们用自己的方式来描述一个心目中的防御场护卫舰在开启防御场的整个过程中应该具有的视觉上的表现。

**Defence field frigate will create a defence field when you press T, so the unit will have an animation. We could make it much more fantastic visually.**

1. 按下T后，我们想要模型有一个华丽的展开动画MadstateOpen，并且有一个排气的特效以彰显机械性和华丽性EffectOpen，展开的过程依据展开动画时长为3秒。  
**When you press T, we have an amazing animation MadstatOpen, and an awesome exhaust effect, EffectOpen, that takes about 3 seconds.**

2. 展开完成后防御场正式启动，这时我们想要一个展开部分蓝色能量环绕的特效EffectRun1，这样还不够，还要一个展开部分高能发热变红的特效EffectRun2，依据这两个特效的循环时间为2秒。  
**When it is active, we have the blue energy sphere effect, EffectRun1. This is not enough as we want an overheat effect too, EffectRun2. It takes about 2 seconds to cycle.**

3. 当能量耗完时，我们需要一个模型展开部分关闭合拢为原始状态的动画MadstateClose，动画时长为4秒。  
**We want it closed as it starts, so we set MadstateClose, the animation takes about 4 seconds**

4. 并且我们不想让防御场护卫舰在启动防御场的时候还能乱跑，这是个强力的能力，必然会抽取引擎大量的能量，哪还有剩余用来溜街呢。  
**This is so awesome, let’s take a break, so the frigate won't move when the ability is active.**

当设计好这一切后反映到配置文件上时会如何呢：

**So what is it over? As the example below:**

**{{"MadstateOpen",},{"EffectOpen",},3,0},**

**{{},{"EffectRun1","EffectRun2",},2,1},**

**{{"MadstateClose",},{},4,0},**

现在我们可以通过这个例子很好的了解参数的每个部分的作用了。

当你打算给一个具有加速特质的单位在加速飞行时一个漂亮的拉烟拖尾，或者一个自我修复能力特质的单位一个修理动画时，就是这些参数发挥作用的时候了。

**Now we know how to use it, any time you want to add some crazy awesome ability, you know where to look.**

#### 参数9 D9

参数9是一些额外参数罗列的参数列表。一共有两个数字参数：特质受限参数和状态保持参数。前一个为特质受限参数，后一个为状态保持参数。

**D9 is a number set, it has two numbers: first is characteristic restriction, second is status configuration.**

特质受限参数是指单位在转让、被捕获等导致能力前提条件丧失的情况下仍然保有已获的能力。如果为1，则即使科技等条件因为各种原因无法满足时，特质的触发能力不会丧失，依然可以自由的使用触发来进行启用和关闭特质。如果为0则在前提丧失的情况下无法再使用触发启用或关闭特质。

**Characteristic restriction is for when a unit has been captured, should it be able to continue to use its ability. 1 is for yes, because technology race conflict, 0 is no.**

如果状态保持参数为1，则这个单位仍然可以保持加速状态（即特质丧失后仍能保持已经进入的高级状态），如果状态保持参数为0，则该单位会强制还原到原始状态（即特质丧失后无法保持已经进入的高级状态）。

**Status configuration is when the unit is using the ability, when it is given to another player could it stay active or not. 1 is for yes still active, 0 is for no, sending it back to its original status.**

### 8.2.4特质介绍 Characteristics introduction

#### OneselfChangePower

**{**

**"OneselfChangePower",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**{"Weapon1", "Weapon2", }, --特有参数C1**

**{"Weapon3", "Weapon4", }, --特有参数C2**

**},**

OneselfChangePower是武器切换特质，激活状态下使用特有参数1中的武器，非激活状态下使用特有参数2中的武器。

**OneselfChangePower changes weapon characteristics, before it is active it uses C2 and after activation it uses C1**

**------------------------------------------------------------------------------------------**

#### OneselfSpeedChange

**{**

**"OneselfSpeedChange",**

**"Add",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**0.8, --特有参数A1**

**},**

OneselfSpeedChange是速度变化特质，非激活状态下为单位原始速度，激活状态下根据运算方式和特有参数1中的数值变化速度。运算方式有Add方式和Multiplication方式。

**OneselfSpeedChange is a speed change characteristic, deactivate is the original speed, active is based on a formula. Which are adding and multiplying.**

Add方式：原速度\*(1+参数)

**Add = (original speed) x (1+A1)**

Multiplication方式：原速度\*参数

**Multiply = (original speed) x A1**

当同一单位受到多个速度变化特质影响运算方式的遵循先加后乘的原则，即：原速度\*（1+加参数1+加参数2+...）\*（乘参数1\*乘参数2\*...）

**The unit that get influence by two types will be: Speed= original X (1+A1) X A1**

**------------------------------------------------------------------------------------------**

#### OneselfBuildingSpeedChange

**{**

**"OneselfBuildingSpeedChange",**

**"Add",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**0.8, --特有参数N1**

**},**

OneselfBuildingSpeed能够改变船只的建造速度。

**OneselfBuildingChange is a building speed change characteristic.**

Add方式：原速度+参数

**Add = (original speed) x (1+N1)**

Multiplication方式：原速度\*参数

**Multiply = (original speed) x N1**

特有参数1 建造速度参数

**------------------------------------------------------------------------------------------**

#### OneselfDamageChange

**{**

**"OneselfDamageChange",**

**"Add",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**0.5, --特有参数B1**

**},**

OneselfDamageChange是伤害变化特质，非激活状态下单位受到正常的伤害，激活状态下根据运算方式和特有参数1中的数值变化所受伤害。运算方式有Add方式、Multiplication方式和Other方式。

**OneselfDamageChange changes damage taken, deactivated is normal, active reduces by adding, multiplying, and other.**

Add方式：所受伤害\*（1-参数）

**Add = (Damage taken) x (1-B1)**

Multiplication方式：所受伤害\*参数

**Multiply = (Damage taken) x B1**

Other方式：单位所受秒伤>船体装甲上限\*参数，否则伤害无效，硬减免

**Other = (Damage taken per second) > ((Ship armour max) x B1), or else no damage taken.**

**------------------------------------------------------------------------------------------**

#### OneselfRepair

**{**

**"OneselfRepair",**

**"Add",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**0.05, --特有参数R1**

**},**

OneselfRepair是装甲变化特质，非激活状态下不生效，激活状态下根据运算方式和特有参数1中的数值变化装甲值。运算方式有Add方式和Multiplication方式。

**OneselfRepair is changing the armour value, same as before with adding and multiplying.**

Add方式：装甲+参数[每秒n%最大装甲值]

**Add= armour + R1 [0.05%]**

Multiplication方式：当前装甲值\*（1+参数），以当前装甲值为基础进行倍率恢复，当前装甲值越高则效果越强。

**Multiply= (present armour) x (1+R1). More HP it will repair more.**

**------------------------------------------------------------------------------------------**

#### OneselfInvulnerability

**{**

**"OneselfInvulnerability",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**},**

OneselfInvulnerability是无敌特质，激活后无敌（能承受攻击但不受伤害）。

**OneselfInvulnerability is the immortal characteristic, when it is active, the unit won't take damage [It can be attacked but no damage will be inflicted]**

**------------------------------------------------------------------------------------------**

#### OneselfHardpointInvulnerability

**{**

**"OneselfHardpointInvulnerability",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**"Engine", --特有参数I1**

**},**

子系统无敌特质，激活后子系统点上的子系统无敌（能承受攻击但不受伤害）。

**OneselfHardpointInvulnerability is the immortal characteristic for subsystem, when it's active the subsystem on the specific hardpoint won't take damage [It can be attacked but no damage will be inflicted]**

特有参数1 子系统点名称。

**I1 is the name of the hardpoint**

**------------------------------------------------------------------------------------------**

#### OneselfUntargeted

**{**

**"OneselfUntargeted",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**},**

OneselfUntargeted是不可被锁定特质，激活后任何敌对单位都无法锁定拥有此特质的单位（仍然会受到范围伤害，非无敌）。

**OneselfUntargeted makes a unit unable to be locked onto, when it's active it won't be lock onto, but will still take range (area) damage.**

**------------------------------------------------------------------------------------------**

#### OneselfShipAbility

**{**

**"OneselfShipAbility",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**1, --特有参数S1**

**1, --特有参数S2**

**},**

开启或屏蔽船只固有能力。

**Enable/Disable unit's ability.**

特有参数1 能力ID，可用能力ID见下表：

**S1 is the ID of ship ability(see** [**ship ability grid**](#SAG)**).**

特别地，设置为0代表该单位被鼠标选中的能力。

**Particularly, when set to 0, it means the ability of being selected.**

特有参数2 1为激活时拥有能力，0为激活时屏蔽能力。**1 means ability enabled when activated, 0 means ability disabled when activated.**

**------------------------------------------------------------------------------------------**

#### OneselfIdioFuel

**{**

**"OneselfIdioFuel",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**50000, --特有参数F1**

**100, --特有参数F2**

**1, --特有参数F3**

**},**

OneselfIdioFuel是燃料特质，拥有该特质的单位航行时会消耗燃料，燃料耗尽后无法移动。燃料必须停泊才能补充，一旦耗尽就必须打捞后才能再次投入使用。

**OneselfIdioFuel gives unit a fuel tank, unit with this characteristic will consume fuel while moving, and it cannot move after the fuel is exhausted. Fuel can only be supplied by docking, once the fuel is exhausted, unit must be salvaged to be reactivated.**

使用该特质的单位必须为能够停泊其他单位的小型单位，必须有打捞点，并在ship文件中加入CanBeSalvaged能力，且编组数量不能大于1。

**Unit with this characteristic must be a small unit that can dock with other units, it must have a salvage point, and a CanBeSalvaged ability defined in .ship file, squadron size must be 1.**

特有参数1 燃料总量。**Fuel tank capacity.**

特有参数2 移动时每秒消耗的燃料量。单位只要移动，就按照固定的速率消耗燃料。**Fuel consumed per second.**

特有参数3 移动时每米/秒²消耗的燃料量。如0.1是指单位以100m/s的速度前进时每秒消耗10单位，以200m/s的速度前进时每秒消耗20单位。**Fuel consumed per m/s², e.g. 0.1 means a unit moving at 100m/s speed will consume 10 unit of fuel per second, and 20 if at a speed of 200m/s.**

参数2和3对燃料消耗的方式不同，但均同时消耗燃料，因此可同时使用，也可择一使用。同时使用时，燃料消耗为二者之和，择一使用时，不使用的一项设为0即可。

**F2 and F3 is different, but they consume fuel at the same time, so either or both of them can be used. When use them togather, fuel consume is their sum, when use one of them, set the other to 0.**

**------------------------------------------------------------------------------------------**

#### OneselfIdioShield

**{**

**"OneselfIdioShield",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**10000, --特有参数F1**

**5000, --特有参数F2**

**20, --特有参数F3**

**1, --特有参数F4**

**45, --特有参数F5**

**2000, --特有参数F6**

**},**

单体护盾是一种可以吸收攻击伤害的个人防御场，在护盾能量耗尽失效前，护盾将会吸收所有对船体造成的伤害。

**OneselfIdioShield is a defense field that can absorb damage. Before its energy exhausted, it can absorb all damage to the ship.**

数值化护盾特质必须由结构累计特质提供基础结构数值，否则无效。

**This characteristic must have a basic structure value provided by AssistantStructure characteristic, otherwise it won't work.**

特有参数1 护盾强度。**Max energy of the shield.**

特有参数2 默认护盾量拥有值。**Default value of energy.**

特有参数3 护盾的每秒恢复速度。**How much energy can be recoverd per second.**

特有参数4 破盾时是否过载导致护盾停机一段时间，过载结束后护盾会重新启动并回充初始护盾值。0为否，1为是。**Whether the shield crash for a while after the energy exhausted. After that the shield will restore and have a basic value of energy.**

特有参数5 护盾停机到恢复重启所需要等待的时间（秒）。**How long before the shield restore after crashing(in seconds).**

特有参数6 恢复重启瞬间护盾所能恢复的护盾量。**Basic value of energy to be given when the shield restored.**

该特质需要配合Ping进行显示，只有LifeBarType=1或2且编组数量为1（Size=1）的单位能够正常显示单位护盾，其他类型的单位请谨慎使用该特质。

**Shield's energy will be displayed by ping, only units whose LifeBarType is 1 or 2 and squadron size is 1(Size=1) can display shield energy properly.**

**------------------------------------------------------------------------------------------**

#### OneselfIdioArmor

**{**

**"OneselfIdioArmor",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**10000, --特有参数W1**

**10000, --特有参数W2**

**10, --特有参数W3**

**0.6, --特有参数W4**

**},**

船壳装甲是一种可以部分吸收攻击伤害的附加装甲层，在船体受到攻击时，船壳装甲会吸收部分伤害并自身损耗。

**OneselfIdioArmor is an additional armor that can absorb some to the ship.**

数值化装甲特质必须由结构累计特质提供基础结构数值，否则无效。

**This characteristic must have a basic structure value provided by AssistantStructure characteristic, otherwise it won't work.**

特有参数1 装甲强度。**Max health of the armor.**

特有参数2 默认装甲拥有量。**Default value of health.**

特有参数3 装甲的默认每秒恢复速度。**How much health value can be recovered per second.**

特有参数4 装甲的伤害吸收率，0.6代表能够吸收60%对船体造成的伤害。**Damage absorb persentage. 0.6 means the armor can absorb 60% damage to the ship.**

该特质需要配合Ping进行显示，只有LifeBarType=1或2且编组数量为1（Size=1）的单位能够正常显示单位护盾，其他类型的单位请谨慎使用该特质。

**Armor's health will be displayed by ping, only units whose LifeBarType is 1 or 2 and squadron size is 1(Size=1) can display armor health properly.**

**------------------------------------------------------------------------------------------**

#### AssistantStructure

**{**

**"AssistantStructure",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**50000, --特有参数V1**

**},**

配合数值护盾和数值装甲的结构累计特质，用于脚本获取船只实际装甲数量值的辅助特质。

**Provide a basic structure value to OneselfIdioShield and OneselfIdioArmor.**

特有参数1 结构数值（单位血量）。**Structure value(max health of the unit),**

**------------------------------------------------------------------------------------------**

#### OneselfHardPointUnderDefend

**{**

**"OneselfHardPointUnderDefend",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**"Engine", --特有参数V1**

**20000, --特有参数V2**

**0, --特有参数V3**

**},**

OneselfHardPointUnderDefend特质用于定义船只上的子系统点（子系统点而非子系统），被定义的子系统点将能够根据设置受到护盾和装甲的保护，其上的子系统在受到打击时伤害会优先由护盾或装甲吸收。

**OneselfHardPointUnderDefend can define hardpoint, subsystem on the defined hardpoint can be protected by shield or armor.**

特有参数1 子系统点名称。**Name of the hardpoint.**

特有参数2 子系统点上子系统的血量数值。**Max health of the subsystem.**

特有参数3 防御层级，0代表在装甲之下，1代表在护盾之下装甲之上。**Protected by who. 0 means protected by shield and armor, 1 means only protected by shield.**

**------------------------------------------------------------------------------------------**

#### OneselfHardPointDamageChange

**{**

**"OneselfHardPointDamageChange",**

**"Add",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**0.5, --特有参数B1**

**},**

OneselfHardPointDamageChange是子系统伤害变化特质，能够影响由OneselfHardPointUnderDefend所申明的子系统遭受攻击时所受的伤害。运算方式有Add方式和Multiplication方式。

**OneselfHardPointDamageChange can change the damage taken by subsystems on the hardpoint defined by OneselfHardPointUnderDefend.**

Add方式：所受伤害\*（1-参数）

**Add = (Damage taken) x (1-B1)**

Multiplication方式：所受伤害\*参数

**Multiply = (Damage taken) x B1**

**------------------------------------------------------------------------------------------**

#### SheepShip

**{**

**"SheepShip",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**5000, --特有参数W1**

**1, --特有参数W2**

**},**

SheepShip是类似帝国时代中绵羊特性的特质，拥有此特质的单位在一定范围内有某一玩家的单位时会被强制转换阵营，被这一玩家所获得。直到另一个玩家接近这一范围并且周围无此单位所有者的其他单位时才被新靠近的玩家获得。

特有参数1 单位获取范围。

特有参数2 转换所需最少单位数量。

**------------------------------------------------------------------------------------------**

#### ShipExperience

**{**

**"ShipExperience",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**400, --特有参数C1**

**1, --特有参数C2**

**},**

船只经验值，能够累积经验值并使船只等级提升，而船只等级可以作为特质触发条件使用。

**An exp characteristic, unit can gain exp and get higher level, which can trigger other characteristics.**

特有参数1 能够获取的经验值上限，达到上限后便无法累加经验值。**Max exp to gain. Exp cannot gained when this limit reached.**

特有参数2 每秒获取的经验值。**Exp gain per second.**

为配合经验值特质，reslist.lua中新增经验值、等级列表：LevelExperienceList

**LevelExperienceList added in reslist.lua for this characteristic.**

格式如下：

**LevelExperienceList =**

**{**

**200,**

**300,**

**400,**

**600,**

**800,**

**1000,**

**}**

每一项代表升级所需要的总经验值。即第一级需要200点，第二级需要300点。

**Numbers inside means the total exp needed for each level. e.g. 200exp for level 1, 300exp for level 2.**

该特质需要配合Ping进行显示，只有LifeBarType=1或2且编组数量为1（Size=1）的单位能够正常显示单位护盾，其他类型的单位请谨慎使用该特质。

**Units' level will be displayed by ping, only units whose LifeBarType is 1 or 2 and squadron size is 1(Size=1) can display armor health properly.**

**------------------------------------------------------------------------------------------**

#### DroneShip

**{**

**"DroneShip",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},1,0},**

**{{},{},0,0},**

**{1,1,},**

**"kpr\_Drone", --特有参数R1**

**2, --特有参数R2**

**{0,15}, --特有参数R3**

**3000, --特有参数R4**

**1, --特有参数R5**

**1, --特有参数R6**

**{0,0,1}, --特有参数R7**

**},**

特有参数1 无人机名称 **Drone name**

特有参数2 无人机数量 **Drone number**

特有参数3 依次为无人机首机补充等待时间、后续机补充间隔时间。**How long to wait until the first wave of drones reached and after.**

特有参数4 无人机作战半径，作战半径大于0时为战斗无人机，无人机会搜寻作战半径内的敌人进行攻击。作战半径为0时为采集无人机，无人机会自动进行采集作业。**Drone operational radius of action. Drones will attack enemies within this radius, 0 means the drones are harvesters, they will harvest resources automatically.**

特有参数5 无人机补充方式（0：超空间出现；1：母舰内发射；2：停泊母舰但不发射）**Drones supplied in which way(0: through hyperspace; 1:launch from mothership;2:dock with mothership).**

特有参数6 无人机被动状态行为（0：跟随；1：停泊母舰）**Drones' passive action(0:follow mother ship;1:dock with mothership).**

特有参数7 依次为母舰的F2、F3、F4状态是否为载机主动出击状态。**Whether drones are activated or not when their mothership is at passive,defensive and aggressive tactic.**

**------------------------------------------------------------------------------------------**

#### AreaSemaphore

**{**

**"AreaSemaphore",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**1601, --特有参数S1**

**2000, --特有参数S2**

**{1,1,1,}, --特有参数S3**

**},**

AreaSemaphore是范围发信特质，所发送信号为范围信号。

**AreaSemaphore is range signal characteristic, this characteristic will send signal to nearby ships as a trigger.**

特有参数1为范围信号辨识ID，编号可自定义(需为大于0的整数)。

**S1 is the range signal ID, it could be personalised, but needs to be a integer greater than 0.**

特有参数2为信号的最大发送半径。

**S2 is the radius of the signal range**

特殊参数3为接受信号的单位阵营，本阵营是否接收；同盟阵营是否接收；敌对阵营是否接收。

**S3 determines the sides that receive the signal, first is own side, then friendly, then enemy, 1 is for yes, 0 is for no.**

A单位具有以下特质

**Unit A has the following characteristics:**

**{**

**"AreaSemaphore",**

**"Other",**

**{**

**{-2,},**

**},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**1001,**

**2000,**

**{1,1,0,},**

**},**

B单位具有以下特质

**Unit B has the following characteristics:**

**{**

**"OneselfSpeedChange",**

**"Add",**

**{**

**{**

**{"RangeSignal",1001,},**

**},**

**},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**0.8,**

**},**

当某玩家的A单位范围发信特质处于激活状态时，其半径2000内的所有本阵营以及同盟阵营的B单位都将激活速度变化特质获得80%的移动速度提升。

**When unit A activates AreaSemaphore, within a radius of 2000m all friendly units' of type unit B will have an 80% speed increase.**

**------------------------------------------------------------------------------------------**

#### LockingSemaphore

**{**

**"LockingSemaphore",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**501, --特有参数L1**

**},**

LockingSemaphore为照准发信特质，所发信号为攻击照准信号（即攻击锁定）。

**LockingSemaphore characteristic will send a signal as a trigger to the ship being attacked by this unit.**

特有参数1为照准信号辨识ID，编号可自定义（需为大于0的整数）。

**L1 is the signal ID, it has to be a integer greater than 0.**

照准信号辨识ID作为其他特质的条件参数，示例：

A单位具有以下特质

**This ID's characteristic configuration is as below.**

**Unit A has this characteristic:**

**{**

**"LockingSemaphore",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**111,**

**},**

B单位具有以下特质

**Unit B has this characteristic**

**{**

**"OneselfSpeedChange",**

**"Add",**

**{**

**{**

**{"LockingSignal",111,},**

**},**

**},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**-0.5,**

**},**

当某玩家的A单位照准发信特质处于激活状态时，被此玩家的单位A攻击的任何B单位都会激活变速特质受到50%减速影响。

**When unit A's LockingSemaphore active, all unit B that are under attack by this unit will reduce speed by 50%.**

**------------------------------------------------------------------------------------------**

#### PassiveSemaphore

**{**

**"PassiveSemaphore",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**801, --特有参数Y1**

**},**

被动信号特质由被攻击者发出，攻击者接收信号ID并作为特质的启动条件。

**PassiveSemaphore characteristic will send a signal as a trigger to any ships attacking this unit.**

特有参数1 为被动信号辨识ID，需为大于0的整数。

**Y1 is the signal ID, it has to be a integer greater than 0.**

**------------------------------------------------------------------------------------------**

#### GuardingSemaphore

**{**

**"GuardingSemaphore",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**1001, --特有参数Z1**

**0, --特有参数Z3**

**},**

保护信号特质在单位保护某个其他单位时将信号ID传递给对方，被保护的单位能够根据保护者的信号ID触发其他特质。

**GuardingSemaphore characteristic will send a signal as a trigger to the ships that are being guarded by this unit.**

特有参数1 保护信号ID，需为大于0的整数。

**Z1 is the signal ID, it has to be a integer greater than 0.**

特有参数2 盟友单位是否能接收保护信号，0为否，1为是。

**Z2 determines whether or not the friendly units will receive the signal. 0 means no, 1 means yes.**

**------------------------------------------------------------------------------------------**

#### PunySemaphore

**{**

**"PunySemaphore",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**1301, --特有参数A1**

**0, --特有参数A2**

**},**

受保护信号特质在单位被其他单位保护时将信号ID传递给对方，保护的单位能够根据被保护者的信号ID触发其他特质。

**PunySemaphore characteristic will send a signal as a trigger to any ships that are guarding this unit.**

特有参数1 被保护信号ID，需为大于0的整数。

**A1 is the signal ID, it has to be a integer greater than 0.**

特有参数2 盟友单位是否能接收被保护信号，0为否，1为是。

**A2 determines whether or not the friendly units will receive the signal. 0 means no, 1 means yes.**

**------------------------------------------------------------------------------------------**

#### MotherShip

**{**

**"MotherShip",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**2001, --特有参数Q1**

**},**

MotherShip是一种将信号发送给停泊在母舰内的船只的特质，母舰内停泊的船只可将信号ID作为触发条件触发其他特质。

特有参数1 停泊信号ID，需为大于0的整数。

**Q1 is the signal ID, it has to be a integer greater than 0.**

**------------------------------------------------------------------------------------------**

#### DockingMotherShip

**{**

**"DockingMotherShip",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**1, --特有参数Q1**

**0.2, --特有参数Q2**

**},**

DockingMotherShip是一种由停泊船只发送给被停泊母舰的一种信号特质，被停泊母舰通过MotherShipFilter特质来接收并汇总这些信号。

**DockingMotherShip characteristic will send a signal as a trigger to the ship that this unit is docking with. The signal will be given to mothership's MotherShipFilter characteristic.**

特有参数1 停泊信号ID。（可为任意数字。因为不是作为条件判定的数字ID，只是供给MotherShipFilter特质对应的ID，所以可以为任意整数。）

**Q1 is the signal ID, it can be any integer (used in MotherShipFilter, not a ConditionValue).**

特有参数2 停泊信号强度。（用于MotherShipFilter的累计，每个停泊单位会发出与该参数等大小的信号强度，小队单位计算其总数。）

**Q2 is the signal's strength (used in MotherShipFilter, every single docking ship's signal strength will be added in total).**

**------------------------------------------------------------------------------------------**

#### MotherShipFilter

**{**

**"MotherShipFilter",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**2501, --特有参数W1**

**1, --特有参数W2**

**2, --特有参数W3**

**},**

MotherShipFilter是母舰接收停泊信号的特质，此特质会接收停泊在船只上单位的停泊信号DockingMotherShip并分析汇总。最终发送一个母舰信号ID，此信号ID可以作为母舰本身其他特质的触发条件。（即可制作停泊了N艘A型船后触发能力X）

**MotherShipFilter characteristic will calculate all DockingMotherShip signals sent to this ship and decide whether or not it send a signal to the ship itself as a trigger.**

特有参数1 母舰激发信号ID，作为其他特质触发条件的数字ID，需为大于0的整数。

**Z1 is the signal ID, it has to be a integer greater than 0.**

特有参数2 所接收的停泊信号ID。与相应的DockingMotherShip停泊信号对应。

**Z2 is the DockingMotherShip signal ID to calculate.**

特有参数3 发送母舰信号ID所需要的停泊信号ID强度，即达到该强度后才激发母舰信号ID。2代表停泊的拥有1停泊信号的船只发出的信号强度大于等于2才会激发母舰信号。

**Z3 is the total signal strength need to reach for this characteristic to send its signal.**

**------------------------------------------------------------------------------------------**

#### CommandSemaphore

**{**

**"CommandSemaphore",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**1201, --特有参数C1**

**},**

CommandSemaphore是一种发出命令信号的特质，发出的命令信号由执行命令操作特质的单位接收，具体作用见下文命令操作特质的说明。

**CommandSemaphore is a command signal characteristic. See OneselfCommand to know how it works.**

特有参数1 命令信号ID，需为大于0的任意整数。

**A1 is the signal ID, it has to be a integer greater than 0.**

**------------------------------------------------------------------------------------------**

#### OneselfCommand

**{**

**"OneselfCommand",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},1,0},**

**{{},{},0,0},**

**{{},{},4,0},**

**{1,1,},**

**2, --特有参数C1**

**-1, --特有参数C2**

**{0,0,1}, --特有参数C3**

**1000, --特有参数C4**

**{"hgn\_scout","hgn\_interceptor"}, --特有参数C5**

**1, --特有参数C6**

**},**

OneselfCommand为命令操作特质，触发时会向单位发送一个命令，由通用参数6中的时长数值决定命令发送的延迟时间；由通用参数8中的时长数值决定命令的发送频率。

**OneselfCommand gives order to unit, D6 define how long it delays, D8 define order frequency.**

特有参数1 参数1为命令ID，决定发送的是什么命令，见下表：

**C1 is order ID, define which order to give:**

|  |  |  |
| --- | --- | --- |
| **ID** | **命令Command** | **说明Meanings** |
| 1 | Move | 向发出指定命令信号的船只移动 |
| 2 | Attack | 攻击发出指定命令信号的船只 |
| 3 | Guard | 保护发出指定命令信号的船只 |
| 4 | Repair | 修理发出指定命令信号的船只 |
| 6 | Harvest | 采集附近的资源 |
| 8 | Capture | 捕捉发出指定命令信号的船只 |
| 9 | Dock | 向发出指定命令信号的船只停泊 |
| 12 | Stop | 停止当前执行的命令 |
| 14 | Parade | 以发出指定命令信号的船只为基准列队 |
| 23 | Salvage | 回收发出指定命令信号的船只 |
| 24 | Scuttle | 自毁 |
| 31 | Custom | 自定义命令 |
| 32 | Kamikaze | 向发出指定命令信号的船只发动自杀攻击 |

特有参数2 参数2为命令对象参数。0代表对象为所有场上单位，大于0则与命令信号特质中的ID参数对应，即对象为发出指定ID命令信号的单位。小于0代表对象为所有不发出(-ID)命令信号的单位。

**C2 determines targets of the order. 0 means all units in the map, greater than 0 means units have command signal ID equals C2, less than 0 means all units in the map except units have command signal ID equals (-C2).**

特有参数3 参数3是命令对象的阵营定义，其与参数2共同决定命令的对象。其中第一个参数定义己方单位是否纳入命令对象（0为否，1为是），第二个参数为盟友，第三个参数为敌军。

**C3 determines targets sides, first is own side, then friendly, then enemy, 1 is for yes, 0 is for no.**

特有参数4 参数4是命令对象的距离定义，在参数2、3的基础上，只有距离该单位小于参数4的单位会作为命令的对象。设0则不作限制。

特有参数5 参数5是命令对象的类型定义，在参数2、3的基础上，只有在参数5中列出的单位会作为命令的对象。设空则不作限制。

特有参数6 参数6为命令对象反转参数。0为不反转，1为反转，反转后，命令对象变为命令的执行者，命令执行者变为命令的对象。

**C6 is a order-reverse var. 0 means order do not reverse, 1 means reverse.Once order is reversed, order targets will become order takers, and order taker become order target.**

如示例中所示，触发后所有1000范围内不发出ID=1的命令信号的敌军希格拉侦察机和希格拉拦截机单位向该单位发动攻击，命令发送后4秒整个动作完成一次循环，然后再次延迟1秒发送下一次命令，直到触发条件不满足。建议整个命令间隔为5~10秒。

**In this example, all hgn\_scout and hgn\_interceptor that within 1000 range and don't have command signal ID=1 will be given an order to attack this unit after the characteristic activated. Order will be given every (4+1) seconds.**

**------------------------------------------------------------------------------------------**

#### CycExplode

**{**

**"CycExplode",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},1,0},**

**{{},{},0,0},**

**{{},{},8,0},**

**{1,1,},**

**1000, --特有参数E1**

**500, --特有参数E2**

**{1,1,1,}, --特有参数E3**

**},**

CycExplode是循环触发的范围伤害特质。

**CycExplode is the cycle of explode range damage.**

特有参数1为爆炸是产生的伤害数值。

**E1 is damage value**

特有参数2为爆炸是产生伤害的范围。

**E2 is damage range**

特有参数3为受到伤害的单位阵营，本阵营是否受到伤害；同盟阵营是否受到伤害；敌对阵营是否受到伤害。

**E3 is damaged units, the order is own, friendly, enemy with 1 = yes and 0 = no.**

如示例中所示，触发后爆炸伤害延迟1秒产生，伤害产生后9秒整个动作完成一次循环，然后再次延迟1秒进行下一次爆炸伤害，直到触发条件不满足。建议整个循环间隔为10~20秒。

**In the example, when is active the explode will happen after 1second, it will take 9 seconds to complete a cycle, and a second later it will send again . I recommend it have a gap of 10~20 seconds.**

配合生命值0%条件可制作殉爆伤害。

**You could use a trigger 0% of HP to create the explosion damage.**

**------------------------------------------------------------------------------------------**

#### ResourceUnitBuildAward

**{**

**"ResourceUnitBuildAward",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**0.1, --特有参数K1**

**},**

ResourceUnitBuildAward特质能对玩家建造花费进行记录并按比例奖励RU。

特有参数1 建造奖励比例，0.1代表10%。

**------------------------------------------------------------------------------------------**

#### ResourceUnitLossesCompensate

**{**

**"ResourceUnitLossesCompensate",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**0.1, --特有参数K1**

**},**

ResourceUnitLossesCompensate特质能对玩家损失单位RU价值进行记录并按比例补偿RU。

特有参数1 损失补偿比例，0.1代表10%。

**------------------------------------------------------------------------------------------**

#### ResourceUnitKillsBounty

**{**

**"ResourceUnitKillsBounty",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**0.1, --特有参数K1**

**},**

ResourceUnitKillsBounty特质能对玩家猎杀单位RU价值进行记录并按比例奖励RU。

特有参数1 猎杀赏金比例，0.1代表10%。

**------------------------------------------------------------------------------------------**

#### ResourceUnitFabricated

**{**

**"ResourceUnitFabricated",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**100, --特有参数K1**

**},**

ResourceUnitFabricated特质能够凭空产生RU。

特有参数1 每秒产生的RU数。

**------------------------------------------------------------------------------------------**

#### RestrictShipBuildingOption

**{**

**"RestrictShipBuildingOption",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**{"Hgn\_BattleCruiser","Hgn\_Scout","Hgn\_Interceptor",}, --特有参数G1**

**},**

RestrictShipBuildingOption特质能够限制或解锁船只的建造项。特质触发时限制特有参数1中所列的建造项，特质停止时解锁特有参数1中所列的建造项。

特有参数1 所需限制的建造项名称。

**------------------------------------------------------------------------------------------**

#### ShipDockAmelioration

**{**

**"ShipDockAmelioration",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**{**

**{“Hgn\_Scout", "Hgn\_Interceptor", }，**

**}, --特有参数U1**

**},**

ShipDockAmelioration特质在激活时能将停泊在该船中满足条件的指定类型的船只替换为其他船只。

特有参数1 替换单位列表。每行第一个参数是需要被替换的船只类型，第二个是替换后的船只类型。**对于第一个参数的舰船，必须在Units列表中进行定义并正确输入其编组数量**。

**------------------------------------------------------------------------------------------**

#### ShipAutoAmelioration

**{**

**"ShipAutoAmelioration",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**"Hgn\_BattleCruiser", --特有参数B1**

**-1, --特有参数B2**

**},**

ShipAutoAmelioration特质在条件满足激活后立刻使当前船只通过超空间进行跳跃离开，一艘新的改进后的船只会在相同的位置超空间跳跃出现替换之前的旧舰。

特有参数1 希望替换为的目标单位类型。

特有参数2 旧单位销毁延迟时间，-1代表必须进入超空间后才进行销毁。

**同时请着重注意，在想要被替换的原始单位的ship文件中添加以下属性：**

**NewShipType.paradeData = "accordantparade"**

**这是用于替换单位后新单位与原始单位进行朝向对齐的必要参数，如果无此参数会导致游戏崩溃。**

**------------------------------------------------------------------------------------------**

#### ShipInstantAmelioration

**{**

**"ShipInstantAmelioration",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**"Hgn\_BattleCruiser", --特有参数R1**

**-1, --特有参数R2**

**},**

ShipInstantAmelioration特质在条件满足激活后瞬间将原船只替换为目标船只，所以请确保你所替换的目标船型和原始船型类似，避免不必要的”视觉冲击”。

特有参数1 希望替换为的目标单位类型。

特有参数2 旧单位销毁延迟时间，-1代表必须进入超空间后才进行销毁。

**同时请着重注意，在想要被替换的原始单位的ship文件中添加以下属性：**

**NewShipType.paradeData = "accordantparade"**

**这是用于替换单位后新单位与原始单位进行朝向对齐的必要参数，如果无此参数会导致游戏崩溃。**

**------------------------------------------------------------------------------------------**

#### EffectOnly

**{**

**"EffectOnly",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**},**

EffectOnly特质无任何实际功能，仅用于根据触发播放动画和特效，纯视觉调整性的特质。

**EffectOnly has no specific functions, it only play animations and effects when activated.**

适用于依据触发条件中的护盾、装甲、结构增减，船只移动等条件播放所需的特效和动画，以增强视觉表现。

**------------------------------------------------------------------------------------------**

#### SubsystemBuild

**{**

**"SubsystemBuild",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**{"Hgn\_C\_Sensors\_DetectHyperspace",}, --特有参数J1**

**{"Sensors 1",}, --特有参数J2**

**},**

用于建造和拆解子系统，激活时会自动建造所列子系统，解除时将指定子系统点上的子系统清除。

**Build and destroy subsystem, subsystem(s) will be built when activated, and destroyed when deactivated.**

特有参数1 所需建造的子系统列表（子系统名）**Names of subsystems to build.**

特有参数2 所需拆除的子系统点列表（子系统点名）**Names of hardpoints on which subsystems to destroy.**

**------------------------------------------------------------------------------------------**

#### ShipIdioStateKeyOn

**{**

**"ShipIdioStateKeyOn",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**10101, --特有参数F1**

**},**

ShipIdioStateKeyOn特质能够在激活时使属于当前单位的一个全局Key开启，且该Key不会因该特质的条件不再满足而关闭。因此这个特质能被用于记录单位进入过某种状态，从而作为单位其他特质的触发条件。

特有参数1所需打开的Key的ID编号，可为任意大于0的整数。此ID可使用IsIdioStateKeyOn的条件进行判断。

**------------------------------------------------------------------------------------------**

#### ShipIdioStateKeyOff

**{**

**"ShipIdioStateKeyOff",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**10101, --特有参数F1**

**},**

ShipIdioStateKeyOff特质能够在激活时使属于当前单位的一个全局Key关闭，因此这个特质能被用于记录单位进入过某种状态，从而作为单位其他特质的触发条件。

特有参数1所需关闭的Key的ID编号，可为任意大于0的整数。此ID可使用IsIdioStateKeyOn的条件进行判断。

**------------------------------------------------------------------------------------------**

#### ShipIdioStateKeyTempOn

**{**

**"ShipIdioStateKeyTempOn",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**10101, --特有参数F1**

**10, --特有参数F2**

**20, --特有参数F3**

**},**

ShipIdioStateKeyTempOn与ShipIdioStateKeyOn一样，可以使属于当前单位的一个全局Key开启，但这种开启是暂时的，只能持续一段时间，时间耗尽后该Key就会关闭，关闭后该Key需要冷却一段时间才能再次被开启。（请勿使ShipIdioStateKeyTempOn与ShipIdioStateKeyOn一起控制同一个Key）。

特有参数1该特质所控制的Key的ID编号，可为任意大于0的整数。此ID可使用IsIdioStateKeyOn的条件进行判断。

特有参数2满足条件后Key开启的时长。

特有参数3时间耗尽后该Key冷却的时长，冷却完毕后该Key才能再次开启。

**------------------------------------------------------------------------------------------**

#### CustomCode

**{**

**"CustomCode",**

**"Other",**

**{},**

**{},**

**{},**

**{{},{},0,0},**

**{{},{},0,0},**

**{{},{},0,0},**

**{1,1,},**

**Start\_X, --特有参数F1**

**Do\_X, --特有参数F2**

**Finish\_X, --特有参数F3**

**},**

CustomCode是自动执行自定义函式的特质，用于运行Modder为单位编写的CustomCode。

特有参数1特质激活瞬间运行的函式名称。

特有参数2特质激活期间运行的函式名称。

特有参数3特质关闭瞬间运行的函式名称。

## 8.3地形预制 Terrain

地形预制是为单位特质系统准备的依据地图设计进行触发的环境条件。由尘云和辐射云组成。

**Terrain is for units' characteristic system, to create a map that will create trigger from dust clouds and nebula clouds.**

### 尘云和辐射云 Dust cloud and radiation cloud

地形预制关系到reslist.lua中的两个配置列表：DustCloudList和NebulaCloudList。

制作过或观察过地图文件的同学们都知道，HW2中有两种云团，一种是具有遮蔽作用的友善的dustcloud，另一种是具有伤害作用有害的nebula。

**Terrain is related to reslist.lua file, DustCloudList and NebulaCloudList.**

**If you have made some maps for HW2, you will know that there are two type of cloud. Clouds for hiding, dust clouds, and clouds that do damage, nebulas.**

HW2本身赋予了简单的地形效果，现在我们利用这些云团作为特质的一种触发条件，以达到模拟地形的效果。

**HW2 has simple terrain effects, we are going to use them as a trigger.**

首先需要将地图中的尘云和辐射云编辑至列表中，格式如下：

**First we need to put them into the list as format below.**

**DustCloudList =**

**{**

**“DustCloud\_1”,**

**“DustCloud\_2”,**

**}**

**NebulaCloudList =**

**{**

**“Nebula\_1”,**

**}**

以上示例中我们添加了两个尘云名字分别为DustCloud\_1和DustCloud\_2以及一个辐射云Nebula\_1。

**In the example above we added two dust clouds, DustCloud\_1 and DustCloud\_2, and a nebula cloud Nebula\_1.**

我们在之前给出了单位在尘云内的判断条件是IsInsideDustCloud，单位在辐射云内的判断条件是IsInsideNebula，如何对应到我们定义的云上去呢？

**The judgments that we give before are IsInsideDustCloud and IsInsideNebula. But how are we going to use these?**

我们可以看到DustCloud\_1在列表DustCloudList中位于第一个位置，DustCloud\_2位于第二个位置。所以DustCloud\_1对应的触发条件ID是1，DustCloud\_2对应的触发条件ID是2。同理Nebula\_1对应的触发条件ID为1。

**As we can see, DustCloud\_1 is first and DustCloud\_2 is second, in which DustCloud\_1 is 1 and DustCloud\_2 is 2. Same as Nebula\_1 would be 1.**

当我们为某个特质填上触发条件{"IsInsideDustCloud",1,},时，就意味着这个特质将在进入尘云DustCloud\_1后触发。

**Adding {"IsInsideDustCloud",1,}, to one of the characteristics means it needs to enter DustCloud\_1 in order to active.**

### 闲话 Notes

如果我们在地图制作中将一种尘云编辑为绿色的外观并命名为DustCloud\_Go，并将这种尘云名称添加至地形预制列表中。

**If we put dust cloud as DustCloud\_Go, and put it into the list.**

再给单位一个进入尘云DustCloud\_Go触发的加速特质，那么我们所有地图中的DustCloud\_Go尘云都能成为高速公路。

**Also we are going to make a unit that could increase speed when it enters the dust cloud, then all the DustCloud\_Go will become a super high way.**

如果换成减伤特质，则尘云会成为一种掩体。

**If we make it reduce damage it will become a cover.**

当然，做一点小手脚就能让某些单位具有破除掩体的功能。

**However, if there are units that have the characteristic of reverse, it will make covers become graves.**

## 8.4科技系统 Technology

科技系统配置关系到reslist2.lua中的配置列表：Technologies

**Technology list configuration relates to Technologies in the file reslist2.lua.**

### 8.4.1配置格式

Technologies配置格式如下：

**Technologies =**

**{**

**{**

**Name = "ScoutEMPAbility",**

**RaceName = "Hiigaran",**

**Type = "Normal",**

**Length = 0,**

**ColdTime = 0,**

**Function = {},**

**Opener = {},**

**Lock = 0,**

**},**

**{**

**Name = "AttackBomberImprovedBombs",**

**RaceName = "Hiigaran",**

**Type = "Normal",**

**Length = 0,**

**ColdTime = 0,**

**Function = {},**

**Opener = {},**

**Lock = 0,**

**}**

**{**

**Name = "ImprovedTorpedo",**

**RaceName = "Hiigaran",**

**Type = "Normal",**

**Length = 0,**

**ColdTime = 0,**

**Function = {},**

**Opener = {},**

**Lock = 0,**

**}**

**}**

|  |  |
| --- | --- |
| **Name** | 科技名称。**Name of the technology.** |
| **RaceName** | 科技所属的种族名。**Name of the race that owns the techonology.** |
| **Type** | 科技的类型。**Type of the techonology.**  共有"Instant","InstantReset","AutoReset","FunctionReset"和"Normal"五类。**All four types are: "Instant", "InstantReset", "AutoReset", "FunctionReset" and "Normal".**  "Instant"类科技是瞬时免费科技，玩家点击该科技则立即研究完成，且不消耗RU。  **"Instant" type technology is instantly free, it is granted immediately after player click to research it and it costs 0 RU.**  "InstantReset"为瞬时重置科技，玩家点击后会立即取消研究。  **"InstantReset" type technology however, will stop researching itself after clicked.**  "AutoReset"类科技是自动重置科技。玩家点击后会取消研究并禁止研究，过一段时间（参数ColdTime）后可重新点击研究。  **"AutoReset" type technology will also stop researching itself after clicked, but it will also be restricted for a while and be unrestricted later.**  "FunctionReset"类科技是脚本重置科技。玩家点击后会取消研究并禁止研究，直到脚本将之解锁后方可重新点击研究。  **"FunctionReset" type technology will stop researching itself after clicked, and be restricted until the script unrestricts it.**  "Normal"即为普通科技类型。  **"Normal" type technology is, well, just normal.** |
| **Length** | 科技持续时间，仅针对"AutoReset"类科技，作用见下文Function参数。**Time of the "AutoReset" type techonology's effect lasts.** |
| **ColdTime** | 科技冷却时间。仅针对"AutoReset"类科技，作用见上文Type参数。**Cold down time of the "AutoReset" type techonology.** |
| **Function** | 科技完成（或持续）时脚本执行的指令，需要Modder在XTechFunction.lua中自行添加，格式参照文件中的示例和说明（仅适合有经验的脚本编写者）。**Function to run when the technology is researched (or active). Need to be added manually in XTechFunction.lua, check that file to learn how to do it (for experienced scripters).** |
| **Opener** | 科技开启的前提条件，此参数是科技系统中最为复杂的参数。**Condition that unrestrict the technology.** |
| **Lock** | 科技锁定参数，1为条件满足开启科技可研究后不会因为条件出现变化而封禁科技；0则会因为条件的再次不满足而封禁科技；-1代表即使此科技因条件不满足而不可研究，科技项仍然可见，但是无法进行研究。 **Whether the technology will be locked. 1 means once the condition reached, the technology will be unrestricted; 0 means the technology may be restricted again when condition changed; -1 means the technology will always be visible, but it can only be researched when the condition reached.** |

#### 条件逻辑

Opener参数控制着科技是否能被研究。即能实现冲突科技，又增加了新的开启条件，能够关联到玩家建造的船只数量等数据。

**Opener determines whether the techonology can be researched.**

**{**

**"ShipCount", --参数D1**

**"Player", --参数D2**

**"ALL", --参数D3**

**0, --参数D4**

**"Hgn\_Carrier",**

**2,**

**},**

**参数D1** 条件判断名称，表明这是一个什么条件。

**D1 is the condition name, indicate what condition this is.**

**参数D2** 条件统计范围，统计范围有"Player"，"Alliance"，"Enemy"，"AllAlliance"，"AllEnemy"可选。分别对应玩家自身，任意一个同盟（最高数值拥有者），任意一个敌人（最高数值拥有者），所有同盟总和，所有敌人总和。

**D2 determines what value to calculate. "Player" is the value of current player being calculated, "Alliance" is the highest value among player's allies, "Enemy" is the highest value among all player's enemies, "AllAlliance" is the total value of player's all allies and "AllEnemy" is the total value of player's all enemies.**

**参数D3** 种族条件，统计指定名称的种族，"ALL"代表所有种族。

**D3 determines which race to count. "All" means all races.**

**参数D4** 反转参数，条件是否反转解释，1为是，0为否。

**D4 is reverse parameter, 1 is reverse, 0 is no.**

**------------------------------------------------------------------------------------------**

示例**Example:**

**Technologies =**

**{**

**{**

**Name = "ScoutEMPAbility",**

**RaceName = "Hiigaran",**

**Type = "Normal",**

**Length = 0,**

**ColdTime = 0,**

**Function = {},**

**Opener =**

**{**

**{**

**{"ShipCount","Player","Hiigaran", 0,"Hgn\_Scout",2,},**

**{"ResUnit","Player","Hiigaran", 0,5000,},**

**},**

**{**

**{"ShipCount","Player","Hiigaran", 0,"Hgn\_Scout",1,},**

**{"ResUnit","Player","Hiigaran", 0,10000,},**

**},**

**},**

**Lock = 1,**

**},**

**}**

所定义的科技名为ScoutEMPAbility

科技所属种族是1，希格拉

**条件1**为：统计单位数量，统计范围是玩家自身单位，种族为希格拉，单位类型是Hgn\_Scout，需要数量为2。

**条件2**为：统计RU，统计范围是玩家自身单位，种族为希格拉，需要5000RU。

**条件3**为：统计单位数量，统计范围是玩家自身单位，种族为希格拉，单位类型是Hgn\_Scout，需要数量为1。

**条件4**为：统计RU，统计范围是玩家自身单位，种族为希格拉，需要10000RU。

整体条件为（条件1与条件2）或（条件3与条件4）。

通常解释：拥有两队侦察机并拥有5000RU，或，拥有一队侦察机并拥有10000RU。

科技为锁定科技。

**------------------------------------------------------------------------------------------**

### 8.4.2条件介绍

#### ShipCount

**{**

**"ShipCount",**

**"Player",**

**"ALL",**

**0,**

**"Hgn\_Carrier", --特有参数1**

**2, --特有参数2**

**},**

ShipCount是船只统计条件，统计指定类型的船只数量，只有当指定玩家拥有的船只数量达到所设数值，才开启科技。

特有参数1 统计的船只类型

特有参数2 需要达到的数量要求

**------------------------------------------------------------------------------------------**

#### TechnologyState

**{**

**"TechnologyState",**

**"Player",**

**"Hiigaran",**

**1,**

**"ScoutEMPAbility", --特有参数1**

**5, --特有参数2**

**},**

TechnologyState是科技检测，检测指定科技的状态。

特有参数1 目标科技名称。

特有参数2 目标科技状态。

-2:已满足研究列表解锁条件，将始终可见(Lock=-1)

-1:已满足研究列表解锁条件，但被X系统禁止研究(Lock=0 or 1)

0:未在研究列表中解锁，尚未受X系统控制

1: 已满足研究列表和X系统解锁条件，但仍受X系统控制(Lock=0 or -1)

2: 已满足研究列表和X系统解锁条件，将始终可见(Lock=1)

3: 正在研究，且仍受X系统控制(Lock=0 or -1)

4: 正在研究，且不受X系统控制(Lock=1)

5:已研究完成

6:激活中(AutoReset/FunctionReset)

7:冷却中(AutoReset/FunctionReset)

**------------------------------------------------------------------------------------------**

#### ResUnit

**{**

**"ResUnit",**

**"Player",**

**"Hiigaran",**

**0,**

**10000, --特有参数1**

**},**

ResUnit是资源数量检测，需达到足够的资源积累才能开启科技的条件。

特有参数1 资源持有数量需求。

**------------------------------------------------------------------------------------------**

#### ShipKills

**{**

**"ShipKills",**

**"Player",**

**"Hiigaran",**

**0,**

**20, --特有参数1**

**},**

杀敌数量检测，需达到足够的杀敌积累才能开启科技的条件。

特有参数1 杀敌数量需求。

**------------------------------------------------------------------------------------------**

#### ShipKillsInRU

**{**

**"ShipKillsInRU",**

**"Player",**

**"Hiigaran",**

**0,**

**5000, --特有参数1**

**},**

杀敌价值检测，需达到足够的杀敌价值积累才能开启科技的条件。

特有参数1 杀敌价值累计需求。

**------------------------------------------------------------------------------------------**

#### ShipLosses

**{**

**"ShipLosses",**

**"Player",**

**"Hiigaran",**

**0,**

**20, --特有参数1**

**},**

损失数量检测，需达到足够的损失积累才能开启科技的条件。

特有参数1 损失数量需求。

**------------------------------------------------------------------------------------------**

#### ShipLossesInRU

**{**

**"ShipLossesInRU",**

**"Player",**

**"Hiigaran",**

**0,**

**5000, --特有参数1**

**},**

损失价值检测，需达到足够的损失价值积累才能开启科技的条件。

特有参数1 损失价值累计需求。

**------------------------------------------------------------------------------------------**

#### ShipBuild

**{**

**"ShipBuild",**

**"Player",**

**"Hiigaran",**

**0,**

**20, --特有参数1**

**},**

建造数量检测，需达到足够的建造积累才能开启科技的条件。

特有参数1 建造数量需求。

**------------------------------------------------------------------------------------------**

#### ShipBuildInRU

**{**

**"ShipBuildInRU",**

**"Player",**

**"Hiigaran",**

**0,**

**5000, --特有参数1**

**},**

建造价值检测，需达到足够的建造价值积累才能开启科技的条件。

特有参数1 建造价值累计需求。

**------------------------------------------------------------------------------------------**

#### TributeDonated

**{**

**"TributeDonated",**

**"Player",**

**" Hiigaran ",**

**0,**

**5000, --特有参数1**

**},**

供金送出检测，需达到足够的供金送出积累才能开启科技的条件。

特有参数1 供金送出累计需求。

**------------------------------------------------------------------------------------------**

#### TributeReceived

**{**

**"TributeReceived",**

**"Player",**

**" Hiigaran ",**

**0,**

**5000, --特有参数1**

**},**

供金获得检测，需达到足够的供金获得积累才能开启科技的条件。

特有参数1 供金获得累计需求。

**------------------------------------------------------------------------------------------**

# 9.公共函数 Public function

公共函数是Xsystem开放给高级使用者的一系列接口函数，类似HW2原版API，提供Xsystem与其他脚本程序的交互渠道。其他外部脚本可以通过调用Xsystem的API来获取Xsystem的内部数据，并能在一定程度上对Xsystem的运行进行数据注入和影响。

## 9.1 API

API是Xsytem提供的一系列公共函数，通过调用这些公共函数，能够获取Xsystem中的一些数据，并且可以在一定程度上通过外部脚本提供Xsystem所需的运行参数。

公共函数只能在船只的自定义代码文件中被调用。

**iShip**为单位在数据表中的下标。在自定义代码文件中，应为shipID。

**xSetGlobalStateKey(iKeyNumber,iKey)**

用于关闭或开启全局条件。Key值可以作为单位特质的触发条件。

**iKeyNumber:**全局Key的编号ID。

**iKey:**开启或关闭参数，可用的参数值为"On"和"Off"，分别为开和关。

如果你想要制作一段小剧情：“所有船只向敌人全速突击。”，你只需要为你所想要出场的船只添加一个加速特质，并把这个特质的触发条件设置为全局Key。当你在外部脚本中设置开启这个全局Key时，所有船只的这些特质都会被触发，同样的，关闭这个全局Key就能停止这些特质的运行。

**xGetGlobalStateKey(iKeyNumber)**

用于获取当前全局Key的状态，当全局Key处于开启状态时返回"On"，关闭状态返回"Off"。

**iKeyNumber:**全局Key的编号ID。

**xSetShipIdioStateKey(iShip,iKeyNumber,iKey)**

除了全局Key，单位还可以拥有私有Key，私有Key是单位自身所有，互相间不干扰。这些私有Key值也可以作为船只特质的触发条件。

**iKeyNumber:**私有Key的编号ID。

**iKey:**开启或关闭参数，可用的参数值为"On"和"Off"，分别为开和关。

**xGetShipIdioStateKey(iShip,iKeyNumber)**

用于获取当前私有Key的状态，当私有Key处于开启状态时返回"On"，关闭状态返回"Off"。

**iKeyNumber:**私有Key的编号ID。

**xGetShipName(iShip)**

获取船只的类型。返回船只类型字符串，无效或船只阵亡则返回空字符串。

**xGetShipShield(iShip)**

获取船只当前护盾值百分比。返回值是一个0~1的浮点数。

**xGetShipShieldValue(iShip)**

获取船只当前护盾数值。返回船只当前护盾实际数值量。

**xGetShipArmor(iShip)**

获取船只当前装甲值备份比。返回一个0~1的浮点数。

**xGetShipArmorValue(iShip)**

获取船只当前装甲数值。返回船只当前装甲实际数值量。

**xGetShipLevelValue(iShip)**

获取船只当前等级。单位阵亡或无效返回-1。

**xGetShipExpValue(iShip)**

获取船只当前经验值。单位阵亡或无效返回-1。

**xAddShipExp(iShip,iExp)**

增加船只经验值。

**iExp:**增加的经验值数值。

**xShipOffline(iShip)**

用于停止船只特质的计算，船只特质全部离线脱机。

**xShipOnline(iShip)**

用于重新启动船只特质的计算。

## 9.2 Overload

Overload是对函数的重载，在Xsystem中申明并调用了一些未实现过程的空函数，这些空函数在一些特定的运行阶段中被调用。使用者可按照自身需求实现这些空函数，并对函数进行重载

注意：

使用者实现的函数必须先于Xsystem加载，否则函数重载无效。

重载函数只能在船只的自定义代码文件中被调用。

**function xOverloadLevelUp(iShip)**

**end**

此函数在单位升级时被调用，Xsystem将触发升级事件的单位下标作为参数传递给此函数，使用者可根据自身需要对此函数进行实现。

**function xOverloadLevelDown(iShip)**

**end**

此函数在单位降级时被调用，Xsystem将触发升级事件的单位下标作为参数传递给此函数，使用者可根据自身需要对此函数进行实现。

# 10.结语 Conclusion

X System的核心内容是单位特质系统的运用，重点在于触发条件的组合；科技、子系统前提的组合；特质之间的互相影响，并配合附加的冲突科技系统和单位替换系统，能够变化出各种有趣的单位和科技设计。

**X system is mainly about the unit characteristic system, the point is how to combine them into something interesting and awesome, by adding a little sub-system or research conflict technology.**

我希望通过这个平台，让更多的人能够把自己的创意展现在HW2的mod中，让我们所爱的HW2变的更为有趣，让更多人喜欢上HW并进入到HW的圈子中来。

**I want to use this platform to make many more people able to express their ideas through HW2 MODs, make the game we love much more lovable and let more people to fall in love and become involved in the HW industry.**

目前9CCN正在全力进行X System和FX1.90的制作冲刺。

**Now 9CCN is doing whatever we can to produce X system and FX 1.90.**

**By xangle13 around 2011**

在家园：重制版下，X系统最重要，最强大，也是最独一无二的单位追踪能力被更加强大的CustomCode取代，至此，任何有丰富编程经验的Modder都能够为单位写出十分复杂的行为模式，相比之下，X系统毕竟只是运行在家园：重制版上的一套脚本，在功能上并不能提供更多，但其简洁明晰的书写格式以及对脚本功能的系统化整理仍然起着不可替代的作用，尤其是对CustomCode的大面积应用或种族科技树的复杂设定来说，X系统更是那些缺少编程经验的Modder的福音。

**In Homeworld:Remastered, the most important, powerful and unique ability of X System, the unit tracking ability, is replaced by ships' custom codes, with which experienced modders can write scripts for units to give them extreme complex mode of behavior. X System is no more than a script running in HW:RM, so the function it can provide is limited. However, its simple and clear format and its systematization of the HW script functions are still playing irreplaceable roles, makes it a blessedness especially for modders who lacks scripting experience but want to heavily apply custom codes or customize a complex technology tree.**

**By 萨尤克教主 in 2017**

# 11.更新日志 Version history

## 1.0.3/0.9.1

基于公版系统的细微修正，没有功能上的变化。

**Public system detail fixes, nothing big.**

## 1.0.4/0.9.3

添加了新的触发条件

## 1.0.5/0.9.6

新增尘云、辐射云相关的地形触发条件。Realist.lua中新增尘云列表和辐射云列表，触发条件参数序列调整

**Added dust cloud and nebula cloud terrain trigger. Realist.lua adds cloud lists, trigger become**

## 1.0.6/0.9.7

添加了新的触发条件

增加了一组多机单位的编组内数量触发，触发条件参数序列调整

**Added new trigger**

**Also added a new set of multiple unit.**

## 1.0.7/0.9.9

添加了新的触发条件

增加了新的操纵类特质

**Added new trigger**

**Added new characteristic**

## 1.0.8/1.0.0

感谢月君的翻译。

增加了新的操纵类特质：

[**配置文件——单位特质——特质介绍——DroneShip**](#_DroneShip_)

**Thanks for the translation by DMare.**

**Added new characteristic,** [**DroneShip**](#_DroneShip)

## 1.0.9/1.0.0

感谢澳大利亚友人煌·内森的校对。

**Thanks for the collation by Nathanius.**

## 1.1.0/1.0.1

修正了生命值条件，修正了子系统条件。

**Fixed the bug of health condition; fixed the bug of subsystem condition.**

## 1.1.1/1.0.2

增加了Ship固有能力特质。

OneselfShipAbility

详见：

[**配置文件——单位特质——特质介绍——OneselfShipAbility**](#_OneselfShipAbility)

改进了无人机特质

详见：

[**配置文件——单位特质——特质介绍——DroneShip**](#_DroneShip_)

## 1.1.2/1.0.4

增加了单位投送特质。单位投送可将预设的单位以停泊投送器单位并发射的形式，通过超空间投送至投送器单位保护（G）的目标单位处。这是首个需要两种特质配合生效的功能。

详见：

[**配置文件——单位特质——特质介绍——DeliverShip**](#_DeliverShip)

[**配置文件——单位特质——特质介绍——DeliverGate**](#_DeliverGate)

增加建造速度特质：OneselfBuildingSpeedChange

详见：

[**配置文件——单位特质——特质介绍——OneselfBuildingSpeedChange**](#_OneselfBuildingSpeed)

## 1.1.3/1.0.5

为单位替换系统新增了一种替换方式，超空间跳跃替换。需要替换的单位进行一次超空间跳跃后旧单位被删除，新单位在旧单位进入超空间的位置出现。

详见：

[**配置文件——单位替换**](#_单位替换_Unit_replacement)

## 1.1.4/1.1.0

增加单体护盾特质，护盾在能量耗尽前能够吸收所有船体所受的伤害。

详见：

[**配置文件——单位特质——特质介绍——OneselfIdioShield**](#_OneselfIdioShield)

增加船壳装甲特质，装甲能够在破裂前部分吸收船体受到的伤害。

详见：

[**配置文件——单位特质——特质介绍——OneselfIdioArmor**](#_OneselfIdioArmor)

以上两特质需要，添加了用于显示的UI控件，文件目录有所变化。

## 1.1.5/1.1.3

增加了装甲和护盾百分比触发条件

优化了UI控件，调整了鼠标悬停显示护盾和装甲百分比的位置，并增加鼠标悬停显示船只当前实时速度。

根据1.1.0版本的百分比护盾和装甲特质衍生出数值化的护盾和装甲系列特质。百分比护盾和装甲特质废弃。

详见：

[**配置文件——单位特质——特质介绍——AssistantStructure**](#_AssistantStructure)

增加了经验值、等级特质ShipExperience。详见：

[**配置文件——单位特质——特质介绍——ShipExperience**](#_ShipExperience)

为了配合显示，UI进行了一些调整和美化。

**并在reslist.lua中添加了两个显示相关的参数：**

BarDisplayOnOff = 1

BarDisplayOnOff是UI显示的开关参数，设为0则不显示护盾和装甲等参数。

SPingStringVersion = "CS"

SPingStringVersion是语言选择参数，由于中文和英文版本游戏所使用的字体不同，排版和美化方式也不同，需要根据自己mod所使用的语言调整此参数。”CS”为简体中文，”EN”为英文。

## 1.1.6/1.1.4

紧急修正，单位列表中为单位新增一条属性：LifeBarType

详见：

[**配置文件——单位列表**](#_单位列表_Unit_list)

## 1.1.7/1.1.5

修正了护盾和装甲UI显示部分两种计算方式显示不统一的问题。

修正了护盾和装甲百分比触发条件无法正常使用的问题。

调整了护盾重启的作用机理，现在护盾能够更好的过载和重启了。

## 1.1.8/1.1.6

修正了一个可能导致游戏卡死的BUG。

改进了护盾特质参数，将过载重启时间和重启回充量两个参数分离单独设置。

详见：

[**配置文件——单位特质——特质介绍——OneselfIdioShield**](#_OneselfIdioShield)

调整了UI显示并增加了一系列用于控制显示的开关（**reslist.lua内容新增**）：

**OverBarShieldDisplayOnOff = 1**

**OverBarArmorDisplayOnOff = 1**

**OverBarSpeedDisplayOnOff = 1**

**OverBarLevelDisplayOnOff = 1**

**SelectedBarShieldDisplayOnOff = 1**

**SelectedBarArmorDisplayOnOff = 1**

OverBarShieldDisplayOnOff用于控制鼠标悬停时护盾百分比显示，0为不显示。

OverBarArmorDisplayOnOff用于控制鼠标悬停时装甲百分比显示，0为不显示。

OverBarSpeedDisplayOnOff用于控制鼠标悬停时实时速度显示，0为不显示。

OverBarLevelDisplayOnOff用于控制鼠标悬停时船只等级显示，0为不显示。

SelectedBarShieldDisplayOnOff用于选中船只时护盾条显示，0为不显示。

SelectedBarArmorDisplayOnOff用于选中船只时装甲条显示，0为不显示。

## 1.1.9/1.1.7

增加了特质OneselfHardPointUnderDefend2，用于定义船只上的子系统点并使其受到护盾和装甲的保护。

增加了特质OneselfHardPointDamageChange，用于修正由特质OneselfHardPointUnderDefend2所定义的子系统点在遭受打击时受到的伤害。

详见：

[**配置文件——单位特质——特质介绍——OneselfHardPointUnderDefend2**](#_OneselfHardPointUnderDefend2)

[**配置文件——单位特质——特质介绍——OneselfHardPointDamageChange**](#_OneselfHardPointDamageChange)

## 1.2.0/1.1.8

调整了数据算法，现在数学运算更加精确了。

调整了UI显示算法，修正了护盾、装甲在数值过低的情况下出现-1%的错误显示。

开放Xsystem独有内部函数，方便高级用户获取或操纵Xsystem的内部数据。有需要的可以单密我获取函数列表。

## 1.2.1/1.1.9

增加新特质子系统无敌：OneselfHardpointInvulnerability

详见：

[**配置文件——单位特质——特质介绍——OneselfHardpointInvulnerability**](#_OneselfHardpointInvulnerability)

添加玩家胜负判定，失去所有指定类型船只的玩家将判负。**reslist.lua中添加胜负判定列表**，格式如下：

ShipForSurvive =

{

{ **--种族1**

"Hgn\_MotherShip", **--船只类型1**

"Hgn\_Carrier", **--船只类型1**

},

{ **--种族2**

"Vgr\_MotherShip",

"Vgr\_Carrier",

},

}

增加序列以适应用户自身mod的种族设置。默认的reslist.lua文件中附带了适用于原版的存活单位预制序列。

增加一个开关，用于开启使用或不使用Xsystem的胜负判定检测。

**PlayerSurviveJudge = 1**

PlayerSurviveJudge为0时不启用Xsystem的胜负判定，为1时启用。

## 1.2.2/1.2.0

修正了一个公开函数引起的错误。

修正了一个照准和范围场判定错误。

增加了一个被动信号源特质PassiveSemaphore。与照准特质相反，被动信号由被攻击者发出，攻击者接收信号ID并作为某些特质的触发条件。

详见：

[**配置文件——单位特质——特质介绍——PassiveSemaphore**](#_PassiveSemaphore)

**条件ID范围调整：**

501~800 照准信号ID

801~1000 被动信号ID

## 1.2.3/1.2.1

分离并调整了特质检测和运行函数。

增加一个新的参数，能够自行调整单位可预定的特质数量上限(**reslist.lua新增**)：

**ScheduledShipAbilityMaxNumber = 10**

以上表示默认每个单位最多能够拥有10个特质。

增加新特质：建造奖励、损失补偿、猎杀赏金、资源生成

详见：

[**配置文件——单位特质——特质介绍——ResourceUnitBuildAward**](#_ResourceUnitBuildAward)

[**配置文件——单位特质——特质介绍——ResourceUnitLossesCompensate**](#_ResourceUnitLossesCompensate)

[**配置文件——单位特质——特质介绍——ResourceUnitKillsBounty**](#_ResourceUnitKillsBounty)

[**配置文件——单位特质——特质介绍——ResourceUnitFabricated**](#_ResourceUnitFabricated)

## 1.2.4/1.2.3

增加了两个新特质(**调整了触发条件ID序列**)，详见：

[**配置文件——单位特质——特质介绍——GuardingSemaphore**](#_GuardingSemaphore)

[**配置文件——单位特质——特质介绍——PunySemaphore**](#_PunySemaphore)

[**配置文件——单位特质——特质介绍——RestrictShipBuildingOption**](#_RestrictShipBuildingOption)

[**配置文件——单位特质——特质介绍——ShipDockAmelioration**](#_ShipDockAmelioration)

[**配置文件——单位特质——特质介绍——ShipAutoAmelioration**](#_ShipAutoAmelioration)

**使用新增的ShipDockAmelioration和ShipAutoAmelioration可以替代原始的单位替换模块功能。**

## 1.2.5/1.2.4

重写了所有分压算法。修正了一个时间计算上的错误。

## 1.2.6/1.2.5

调整了部分分压参数，重新制定了函数加载方式。

## 1.2.7/1.2.6

增加了新的触发条件

增加了用于播放特效的无实际功能特质：

[**配置文件——单位特质——特质介绍——EffectOnly**](#_EffectOnly)

## 1.2.8/1.2.7

增加了特质，详见：

[**配置文件——单位特质——特质介绍——ShipInstantAmelioration**](#_ShipInstantAmelioration)

改进了特质ShipAutoAmelioration，使得超空间出现的替换单位能够与原始单位朝向进行同步对齐。

详见：

[**配置文件——单位特质——特质介绍——ShipAutoAmelioration**](#_ShipAutoAmelioration)

## 1.2.9/1.2.8

修正了一些意想不到的小BUG。

## 1.3.0/1.3.0

修正了保护特质和被保护特质中存在的一些BUG。

改进了无人机特质，增加了一个母舰控制载机状态，改良了无人机补充间隔时间定义。

详见：

[**配置文件——单位特质——特质介绍——DroneShip**](#_DroneShip_)

增加了新特质，详见：

[**配置文件——单位特质——特质介绍——MotherShip**](#_MotherShip)

[**配置文件——单位特质——特质介绍——DockingMotherShip**](#_DockingMotherShip)

[**配置文件——单位特质——特质介绍——MotherShipFilter**](#_MotherShipFilter)

**调整了触发ID序列，尘云ID和辐射云ID后移。增加了母舰信号和母舰激发信号。**

详见：

[**配置文件——单位特质——通用参数——参数3**](#_参数3)

## 1.3.1/1.3.1

改进了单位停泊替换特质，添加了停泊目标类型参数。必须停泊在有效的母舰内才能进行单位的更新替换。

详见：

[**配置文件——单位特质——特质介绍——ShipDockAmelioration**](#_ShipDockAmelioration)

修改了范围信号特质，现在范围信号特质中信号范围数值如果为0代表信号范围无限大。

## 1.3.2/1.3.5

新增科技系统模块，**reslist.lua中增加科技系统列表：**

**Technologies =**

**{**

**}**

配置格式见：

[**功能简介——科技系统模块**](#_科技系统模块_Technology_model)

[**配置文件——科技系统**](#_科技系统_Technology)

文档中追加了对旧系统废弃的说明，详见：

[**注意**](#_注意_Notic)

## 1.3.4/1.3.6

修正了科技模块无法生效的BUG。

修正了一些文档上的小错误。

增加了新的科技系统条件，详见：

[**配置文件——科技系统——条件介绍——ResUnit**](#_ResUnit)

添加了对公共函数的说明部分，详见：

[**配置文件——公共函数**](#_公共函数_Public_function)

改进了单位投射特质，增加了单位投射时出现方式参数，允许超空间出现和停泊至目标机库中两种方式。详见：

[**配置文件——单位特质——特质介绍——DeliverShip**](#_DeliverShip)

## 1.3.5/1.3.7

增加了公共函数，详见：

[**配置文件——公共函数——xGetShipExpValue**](#_xGetShipExpValue)

[**配置文件——公共函数——xAddShipExp**](#_xAddShipExp)

增加了新的科技系统条件，详见：

[**配置文件——科技系统——条件介绍——ShipKills**](#_ShipKills)

[**配置文件——科技系统——条件介绍——ShipKillsInRU**](#_ShipKillsInRU)

[**配置文件——科技系统——条件介绍——ShipLosses**](#_ShipLosses)

[**配置文件——科技系统——条件介绍——ShipLossesInRU**](#_ShipLossesInRU)

[**配置文件——科技系统——条件介绍——ShipBuild**](#_ShipBuild)

[**配置文件——科技系统——条件介绍——ShipBuildInRU**](#_ShipBuildInRU)

[**配置文件——科技系统——条件介绍——TributeDonated**](#_TributeDonated)

[**配置文件——科技系统——条件介绍——TributeReceived**](#_TributeReceived)

## 1.3.6/1.3.8

修正了一个因初始化错误导致的单位特质中科技和子系统条件无效的BUG。

修正了公共函数xGetShipLevelValue获取的等级非船只当前等级的错误。

增加了新特质，详见：

[**配置文件——单位特质——特质介绍——SubsystemBuild**](#_SubsystemBuild)

## 1.3.7/1.3.9

修正了无人机特质在周围无敌人时也会弹射无人机的问题。

改良了无人机特质，新增了采集无人机特性。

详见：

[**配置文件——单位特质——特质介绍——DroneShip**](#_DroneShip_)

## 1.3.8/1.4.0

增加了新特质，详见：

[**配置文件——单位特质——特质介绍——ShipIdioStateKeyOn**](#_ShipIdioStateKeyOn)

[**配置文件——单位特质——特质介绍——ShipIdioStateKeyOff**](#_ShipIdioStateKeyOff)

增加了特质触发条件

## 1.3.9/1.4.1

修正了一个由无人机导致的单位检索延迟问题。

延长了单位销毁的等待时间，现在所有需要销毁的单位都必须等待20秒才进行实际销毁。

## 1.4.0/1.4.2

修改了等级判定条件中反转条件的判定。

## 1.4.1/1.4.3

修正了生命值触发条件无效的问题。

增加了新的触发条件

增加一条重要的配置编写原则提示：

**任何因为没有完整编写排除列表和单位列表而导致的运行效率低下、卡顿现象，本人概不负责。**

详见：

[**配置文件——单位列表——闲话**](#_闲话_Notes)

增加绵羊单位特质，利用单位靠近的方式转换其他单位阵营。

详见：

[**配置文件——单位特质——特质介绍——SheepShip**](#_SheepShip)

## 1.4.2/1.4.4

增加了自定义资源模块并根据显示需要修改了资源显示界面。在资源显示界面中公开了游戏记时显示功能以及FXmod先祖使用的能量系统。

详见：

[**配置文件——自定义资源**](#_自定义资源_Custom_resource)

公开了特质，虽然一公开就被废弃。

详见：

[**配置文件——单位特质——特质介绍——EnergyMachinery**](#_EnergyMachinery（废弃Abandon）)

新增了特质。

详见：

[**配置文件——单位特质——特质介绍——CustomResourceUnitFabricated**](#_CustomResourceUnitFabricated)

## 1.4.3/1.4.5

略微修正了单位替换的过程逻辑。

重新测试并修改了系统压力阀值，略微降低了阀值（虽然在运行表现上差别不大，依然保持流畅）。

## 1.4.4/1.4.6

修正了单位停泊替换特质在母舰爆炸损毁单位被抛出时仍然执行替换的问题。

修正了单位停泊替换后新单位停留在母舰内不自动发射的问题。

改进了部分特质，详见：

[**配置文件——单位特质——特质介绍——ShipDockAmelioration**](#_ShipDockAmelioration)

[**配置文件——单位特质——特质介绍——ShipAutoAmelioration**](#_ShipAutoAmelioration)

[**配置文件——单位特质——特质介绍——ShipInstantAmelioration**](#_ShipInstantAmelioration)

修正了一个自定义资源的显示问题。

## 1.4.5/1.4.7

对自定义资源部分进行了大幅度修正。

详见：

[**配置文件——自定义资源**](#_自定义资源_Custom_resource)

同时略微修正了科技系统中关于Lock参数的定义。

详见：

[**配置文件——科技系统**](#_科技系统_Technology)

## 1.4.6/1.4.8

对自定义资源进行了部分调整，增加了一项参数：默认资源上限。用于定义默认的自定义资源上限和区分有上限资源和无上限资源。

详见：

[**配置文件——自定义资源——基本配置**](#_基本配置)

## 1.4.7/1.4.9

修改了胜利判定方式，现在无定义胜利单位的种族将不会因为单位剩余状态而失败。

## 1.4.8/1.5.0

未知

## 1.4.9/1.5.1

新增了特质

详见：

[**配置文件——单位特质——特质介绍——ControlStop**](#_ControlStop)

修改了特质**OneselfShipAbility**，为其添加了一个新的参数0。

[**配置文件——单位特质——特质介绍——OneselfShipAbility**](#_OneselfShipAbility)

## 2.0.0/2.0.0

移植到重制版。

利用重制版的新特性对整个系统进行了优化。

废弃了部分功能。

更改了部分特质的定义方式。

增加了自定义条件ID。

## 2.0.1/2.0.1

修正了Ping的显示。

添加了AI难度判断条件。

修正了说明文档的一些小错误。

增加命令信号特质和命令操作特质。

详见：

[**配置文件——单位特质——特质介绍——CommandSemaphore**](#_CommandSemaphore)

[**配置文件——单位特质——特质介绍——OneselfCommand**](#_OneselfCommand)

## 2.0.2/2.0.2

更改特质里条件参数D3的格式，添加若干新的判断条件。

详见：

[**配置文件——单位特质——通用参数——参数3**](#_参数3)

恢复了单位全局Key特质并增加了单位全局Key暂时开启特质。

详见：

[**配置文件——单位特质——特质介绍——ShipIdioStateKeyOn**](#_ShipIdioStateKeyOn)

[**配置文件——单位特质——特质介绍——ShipIdioStateKeyOff**](#_ShipIdioStateKeyOff)

[**配置文件——单位特质——特质介绍——ShipIdioStateKeyTempOn**](#_ShipIdioStateKeyTempOn)

## 2.0.3/2.0.3

添加若干新的判断条件；添加自定义命令代码特质。

## 2.0.4/2.0.4

感谢菲雅丽帝国MOD作者妖都镇魂曲对本文档进行了优化排版。

添加了燃料特质和相应的条件判断。

详见：

[**配置文件——单位特质——特质介绍——OneselfIdioFuel**](#_OneselfIdioFuel)

## 2.0.5/2.0.5

科技系统条件判断格式更新。

恢复X系统公共函数和全局Key判断条件。

首个重置版X系统公版。

## 2.0.6/2.0.6

修正英文版Ping文字拥挤。

修正装甲特质的一个问题。

## 2.0.7/2.0.7

针对重置V1.29/V1.30版进行兼容性更新——科技系统和胜利判定条件中的种族ID定义改为前缀定义。

## 2.0.8/2.0.8

添加新的特质系统判断条件；科技系统新增科技类型和脚本设置，新增若干判断条件。

## 2.0.9/2.0.9

进行重置版2.0适配。

能力特质新增对自定义指令能力的控制ID。

命令特质新增命令目标的范围和船型筛选。

## 2.1.0/2.1.0

实现不同规则下加载不同的数据列表。

## 2.1.1/2.1.1

改变系统加载方式。

详见：

[**导入系统——加载和启用**](#_6.3加载和启用_Load_and)

优化系统并修复了2.1.0的bug。

## 2.1.2/2.1.2

新增条件判断IsDoingAbility和CanDoAbility。

更新舰船Ability能力列表，部分ID发生改变。（影响同样波及命令特质中的命令ID）

详见：

[**舰船Ability能力表**](#SAG)

为命令特质添加新命令：执行自定义命令。

详见：

**配置文件——单位特质——特质介绍——**[**OneselfCommand**](#_OneselfCommand)

将自定义命令特质改为自定义函式特质。

详见：

**配置文件——单位特质——特质介绍——**[**CustomCode**](#_CustomCode)

为在CustomCode.lua上添加额外的自定义函式提供更好的支持。

清理文档中无效的超链接引用。