[quote]

[align=center][b][color=orangered][size=200%]' 免 责 声 明 '[/size][/color][/b][/align]

======

[b][size=150%]该[/size]教材的出版时间是2009年，由于游戏与软件的迭代特性，本书的内容已经与现在你实际接触到的游戏与软件内容有所差异，有些内容可能已经过时，甚至是被淘汰“过期的”，请大家自己斟酌参考。若出现由于原文与本帖翻译有所不同而导致的错误或损失，概不负责。[/b]

[/quote]

[quote]

[align=center][b][color=orangered][size=200%]' 扇 贝 碎 碎 念 '[/size][/color][/b][/align]

======

[b][size=150%]这[/size][/b]是国外教材《Beginning Lua with World of Warcraft Addons》的第十二个章节。全书共14个章节和2个附录。这是我翻译的第四个章节，其余章节的翻译内容我会在以后陆续发帖分享。[b]也希望阅读的人不要吝啬你们的点赞&#128077;，只要点一下便能给予我莫大的支持和鼓励。[/b]

对于这本教材，我是不推荐把这本书作为新版本的插件开发指南。但若从入门或业余角度来看，这本书依然是本相当优秀的教程，可用对魔兽和插件有个整体的了解。虽然阅读过时的教程会特别令人感到不适，特别是当发现输入的代码是已经被游戏淘汰的时候。但希望大家抱着个宏观的状态来阅读，如果遇到无法进行下去的代码错误，请参考你自己正在使用的插件或最新的FrameXML或其他API。从正在使用的插件代码切入可能是最高效的方法。

我已经出了我到目前为止所翻译的几个章节的合集，有兴趣的可以自行浏览[tid=24502396][color=orangered][合集]Lua与魔兽世界插件——《Beginning Lua with World of Warcraft Addons》[/color][/tid]

[size=110%][b]最后，如果有人在阅读过程中，如果发现哪里翻译错误，或错别字需要更正，请发评论或发私信告知，谢谢大家。[/b][/size]

接下来让我们正式开始这次的阅读吧。

[/quote]

[quote]

[align=center][b][color=orangered][size=200%]' 目 录 '[/size][/color][/b][/align]

======

[list]

[\*][size=180%][b]第十二章 宏 (Macros)[/b][/size]

[list]

[\*][b][size=130%]可用命令 (Available Command)[/size][/b]

[list]

[\*][b]安全和普通斜杠命令 (Secure and Normal Slash Commands)[/b]

[\*][b]Target选项 (The Target Option)[/b]

[\*][b]安全指令列表 (List of Secure Commands)[/b]

[\*][b]施法序列 (Cast Sequences)[/b]

[\*][b]物品栏ID (Item Slot IDs)[/b]

[/list]

[\*][b][size=130%]条件 (Conditions)[/size][/b]

[list]

[\*][b]可用的条件 (Available Conditions)[/b]

[\*][b]组合条件 (Combining Conditions)[/b]

[list]

[\*]逻辑与AND (The Logical AND)

[\*]逻辑非NOT (The Logical NOT)

[\*]逻辑或OR (The Logical OR)

[/list]

[\*][b]简单的控制构造 (Simple Control Structures)[/b]

[\*][b]宏的反馈 (Macro Feedback)[/b]

[/list]

[\*][b][size=130%]突破255个字符的限制 (Circumventing the 255-Character Limit)[/size][/b]

[\*][b][size=130%]在宏中使用Lua (Using Lua in Macros)[/size][/b]

[list]

[\*][b]发送聊天消息 (Sending Chat Message)[/b]

[\*][b]计时宏 (Timers in Macros)[/b]

[\*][b]抑制错误消息 (Suppressing Errors)[/b]

[\*][b]优化选项 (Tuning Options)[/b]

[\*][b]绕过[flyable]bug (Circumventing the [flyable] Bug)[/b]

[/list]

[\*][b][size=130%]总结 (Summary)[/size][/b]

[/list]

[/list]

[/quote]

======

[align=center][color=orangered][b][size=200%]&#9632; 第十二章 宏[/size][/b]

[size=130%][b](Macros)[/b][/size][/color][/align]

[b][size=150%]对[/size][/b]于任何学习Lua以便充分体验WoW的人来说，可以肯定的是，在你们的动作条(action bar)中已经有了一些宏(macros)。例如当你要执行复杂的任务，仅需一个按钮，即可根据条件将不同的法术施放到不同的目标身上，它们会非常有用。不同的条件也可能非常的复杂，例如，你可以检测你当前的目标是敌对的(hostile)还是友方的(friendly)，是存活的(alive)还是死亡的(dead)。

如果你的动作条中有宏，你是自己创建的还是从其他地方复制粘贴的？如果是你自己创建的，你知道你在做什么吗？许多玩家认为宏只是一个由简单的斜杠命令列表组成的，但是它们非常强大。某些斜杠命令可以包含强大的条件，并由多种可以使用的选项。

宏总是逐行执行的，这意味着你必须在你想使用的斜杠命令之后插入新行。例如，一个宏发送两条消息到聊天框是这样的：

[quote]

/say Hello...

/say ...World!

[/quote]

你可以将游戏中所有可用的斜杠命令添加到宏中，从游戏中提供的斜杠命令如/say，或小的Lua脚本如/script <Lua code>，到插件创建的斜杠命令中复杂的Lua函数。唯一的限制是每个宏的字符数被限制为255。但我们会找到规避这一限制的方法。

本章是关于创建宏的，因此我们将探索宏是如何工作的，以及如何创建功能及其强大的宏。我们将从宏的可用命令概述开始。

======

[color=blue][b][size=160%]&#9679; 可 用 命 令 [/size]

[size=130%](Available Command)[/size][/b][/color]

你可以找到基本的斜杠命令，如“/cast <spell>”、“/use <item>”(内部实际上是相同的命令)，或“/castsequence <spell1>, <spell2>, ...”，但是还有更多命令，例如，你知道命令“/petautocasttoggle <pet spell>”吗？

[b][size=130%]&#9675; 安全和普通斜杠命令(Secure and Normal Slash Commands)[/size][/b]

我们将用于宏的大多数重要斜杠命令都被描述为安全斜杠命令，因为它们通常只是执行单个的受保护的函数。所有这些斜杠命令都存储在一个表中，只能从文件FrameXML\ChatFrame.lua中访问，保存局部变量的表的名称为SecureCmdList。回想一下，普通的斜杠命令(比如由插件创建的斜杠命令)存储在全局可访问的表SlashCmdList中。

在我们看来，这些普通命令相当乏味，因为它们所能做的一切也可用由Lua脚本完成。你还知道了游戏提供的所有重要的普通斜杠命令，因为你每天都在使用它们。例如，/raid <text>向团队发送一条消息，这是你在聊天框中一直想要输入的内容。还有命令/random去分配战利品。其他常用的普通斜杠命令是表情(emotes)，例如，/facepalm <some guy who just wiped the raid>(某人刚刚离开了团队)有时会非常有用。

然而，本章不涉及这些提供的普通命令。你已经知道了最重要的那些，因为你每天都在使用它们。所有其他不太常用的普通斜杠命令被用于管理公会或团队。而且，这些斜杠命令通常非常简单，它们通常只接收一个参数，比如文本或目标。我们希望编写你可以在boss战或竞技场中使用的强大的宏。

[quote]

注意：如果你需要更多关于游戏提供的普通斜杠命令的信息，暴雪在其网站上有一个列表：[url]http://www.worldofwarcraft.com/info/basics/slashcommands.html[/url]。

[/quote]

现在让我们关注安全命令，乍一看它们要复杂得多。增加复杂性(adding complexity)是指你可以添加到斜杠命令中的条件和选项。在执行命令之前会检查条件，如果条件不满足，斜杠命令将被忽略。稍后你将看到关于条件的更多信息。

我们将首先研究哪些斜杠命令可以使用，以及它们提供的选项。有一个选项比较特殊，因为它对所有斜杠命令都可用，并且在内部实际上是一个条件——target选项。

[b][size=130%]&#9675; Target选项(The Target Option)[/size][/b]

让我们使用一个大家都知道的斜杠命令作为例子：/cast。这似乎是一个简单的命令，它对你当前的目标施法。但是你也可用使用/cast [target=focus] <spell>对你的焦点目标施法。在例子中这个选项的值是focus，但是它可以是任何有效的单位ID或附近的玩家或NPC的名字。

target是一个可用于所有安全命令的选项。这个选项显然不是对所有命令都有意义，但它在每个安全命令的语法上都是有效的，而不是对每个命令都有效。

[quote]

注意：只有安全命令支持target选项，普通的斜杠命令不需要。但是许多普通的斜杠命令，比如表情(emotes)，将动作的目标作为一个参数。

[/quote]

这个选项用来引出我们第一个有用的宏，对目标的目标施放一个法术。例如，治疗可以用它来治疗Boss，于是就会自动治疗它的当前目标：

[quote]

/cast [target=targettarget] <healing spell>

[/quote]

另一个对猎人或术士有用的宏是，将一个法术对准宠物的目标。这可以通过下面的斜杠命令完成：

[quote]

/cast [target=pettarget] <spell>

[/quote]

[b][size=130%]&#9675; 安全指令列表(List of Secure Commands)[/size][/b]

表12-1列出了所有可用的关于施放法术或使用装备物品的安全命令。稍后将详细解释一些一些更高级的斜杠命令，例如/castsequence

[quote]

小贴士：所有安全斜杠命令都是用Lua编写的。你可以在文件FrameXML\ChatFrame.Lua中找到他们(在文件中搜索SecureCmdList)。阅读斜杠命令后面的代码可用帮助你理解它。

[/quote]

[align=center][size=80%]表12-1 可用的法术和物品的安全斜杠命令[/size][/align]

[table]

[tr]

[td50][align=center]命令[/align][/td]

[td][align=center]描述[/align][/td]

[/tr]

[tr]

[td][align=center]/startattack <target>[/align][/td]

[td][align=center]你的角色开始攻击<目标>或者你当前的目标(如果它被省略了)。也可以使用target选项替代。[/align][/td]

[/tr]

[tr]

[td][align=center]/stopattack[/align][/td]

[td][align=center]停止自动攻击[/align][/td]

[/tr]

[tr]

[td][align=center]/cast <spell or item>[/align][/td]

[td][align=center]施放<法术>或使用<物品>。别名：/spell[/align][/td]

[/tr]

[tr]

[td][align=center]/use <spell or item>[/align][/td]

[td][align=center]等于/cast[/align][/td]

[/tr]

[tr]

[td][align=center]/castrandom <spell1>,<spell2>,...[/align][/td]

[td][align=center]从被逗号分隔列的列表中随机选择一种法术或物品并使用它。[/align][/td]

[/tr]

[tr]

[td][align=center]/castsequence <spell1>,<spell2>,...[/align][/td]

[td][align=center]当它第一次被执行时将施放<spell1>，第二次点击时将执行<spell2>，以此类推。它的参数也可以是物品。这个命令带有功能更加强大的额外重置选项(reset option)，这将在本章后面详细解释。[/align][/td]

[/tr]

[tr]

[td][align=center]/stopcasting[/align][/td]

[td][align=center]取消施法[/align][/td]

[/tr]

[tr]

[td][align=center]/cancelaura <buff>[/align][/td]

[td][align=center]取消buff <buff>[/align][/td]

[/tr]

[tr]

[td][align=center]/cancelform[/align][/td]

[td][align=center]取消你当前的变形形态。[/align][/td]

[/tr]

[tr]

[td][align=center]/equip <itemName>[/align][/td]

[td][align=center]从你的库存中装备物品<itemName>，别名：/eq[/align][/td]

[/tr]

[tr]

[td][align=center]/equipslot <slot> <item>[/align][/td]

[td][align=center]在<slot>中装备<物品>，稍后将解释槽ID(slot ID)。[/align][/td]

[/tr]

[tr]

[td][align=center]/changeactionbar <page>[/align][/td]

[td][align=center]将动作条页面更改为<page>[/align][/td]

[/tr]

[tr]

[td][align=center]/swpactionbar <page1> <page2>[/align][/td]

[td][align=center]动作条<page1>和<page2>之间的切换。如果你当前既不是<page1>，也不是<page2>，则将选择<page1>[/align][/td]

[/tr]

[/table]

这几个主要处理法术和物品的命令是宏中最重要。注意，由于全局冷却时间(global cooldown)限制，每个宏只能施放一个法术。但是有一些法术不会激活全局冷却，比如萨满和德鲁伊的自然之速(Nature’s Swiftness)。如果自然之速准备好了，以下的宏将会立即施放治疗波(Healing Wave)。如果你的自然之速正在冷却，则它将施放普通的治疗波。

[quote]

/cast Nature’s Swiftness

/cast Healing Wave

[/quote]

你也可以对德鲁伊使用这个宏，用德鲁伊的治疗之触(Healing Touch)替代治疗波。其他不能触发全局冷却时间的法术包括牧师(神圣)的心灵专注(Inner Focus)和法师(奥术)的心灵镇定(Presence of Mind)。你可以使用这两种法术构建类似的宏。

但是除了施法之外，还有更多的斜杠命令。表12-2列出了处理target和focus target的所有安全命令，这些命令也可以派上用场。

[align=center][size=80%]表12-2 你的(焦点)目标可用的安全斜杠命令[/size][/align]

[table]

[tr]

[td50][align=center]命令[/align][/td]

[td][align=center]描述[/align][/td]

[/tr]

[tr]

[td][align=center]/target <name>[/align][/td]

[td][align=center]目标的<name>。这也会把附近以<name>作为名称的玩家或NPC当作目标。你可以使用/targetexact <name>来避免这种行为。也可以使用target选项替代参数。别名：/tar[/align][/td]

[/tr]

[tr]

[td][align=center]/targetexact <name>[/align][/td]

[td][align=center]以确切的名称作为目标。你可以使用target选项替代参数。[/align][/td]

[/tr]

[tr]

[td][align=center]/targetenemy[/align][/td]

[td][align=center]以最近的敌对单位作为目标。[/align][/td]

[/tr]

[tr]

[td][align=center]/targetenemyplayer[/align][/td]

[td][align=center]以最近的敌对玩家作为目标。[/align][/td]

[/tr]

[tr]

[td][align=center]/targetfriend[/align][/td]

[td][align=center]以最近的友方单位作为目标。[/align][/td]

[/tr]

[tr]

[td][align=center]/targetfriendplayer[/align][/td]

[td][align=center]以最近的友方玩家作为目标。[/align][/td]

[/tr]

[tr]

[td][align=center]/targetraid[/align][/td]

[td][align=center]以最近的副本成员作为目标。[/align][/td]

[/tr]

[tr]

[td][align=center]/cleartarget[/align][/td]

[td][align=center]清除你的目标。[/align][/td]

[/tr]

[tr]

[td][align=center]/targetlasttarget[/align][/td]

[td][align=center]以你的最后一个目标为单位。[/align][/td]

[/tr]

[tr]

[td][align=center]/targetlastenemy[/align][/td]

[td][align=center]以你的最后一个敌对目标作为目标[/align][/td]

[/tr]

[tr]

[td][align=center]/targetlastfriend[/align][/td]

[td][align=center]以你的最后一个友方目标作为目标[/align][/td]

[/tr]

[tr]

[td][align=center]/assist <target>[/align][/td]

[td][align=center]协助<target>，也可以使用target选项替代参数。别名：/a <target>[/align][/td]

[/tr]

[tr]

[td][align=center]/focus <target>[/align][/td]

[td][align=center]把<target>设为焦点，你也可以使用target选项。[/align][/td]

[/tr]

[tr]

[td][align=center]/clearfocus[/align][/td]

[td][align=center]清除焦点。[/align][/td]

[/tr]

[/table]

表12-3列出了对猎人和术士来说特别重要的斜杠命令，因为他们可以命令你的宠物。

[align=center][size=80%]表12-3 宠物的可用安全斜杠命令[/size][/align]

[table]

[tr]

[td50][align=center]命令[/align][/td]

[td][align=center]描述[/align][/td]

[/tr]

[tr]

[td][align=center]/petattack <target>[/align][/td]

[td][align=center]使你的宠物开始攻击<目标>。也可以使用target选项。[/align][/td]

[/tr]

[tr]

[td][align=center]/petfollow[/align][/td]

[td][align=center]设置你的宠物跟随。[/align][/td]

[/tr]

[tr]

[td][align=center]/petstay[/align][/td]

[td][align=center]设置你的宠物停止。[/align][/td]

[/tr]

[tr]

[td][align=center]/petpassive[/align][/td]

[td][align=center]设置你的宠物为被动。[/align][/td]

[/tr]

[tr]

[td][align=center]/petdefensive[/align][/td]

[td][align=center]设置你的宠物防守。[/align][/td]

[/tr]

[tr]

[td][align=center]/petautocaston <spell>[/align][/td]

[td][align=center]启用自动施放宠物的法术<spell>[/align][/td]

[/tr]

[tr]

[td][align=center]/petautocastoff <spell>[/align][/td]

[td][align=center]禁用自动施放宠物的法术<spell>[/align][/td]

[/tr]

[tr]

[td][align=center]/petautocasttoggles <spell>[/align][/td]

[td][align=center]切换自动施放宠物的法术<spell>[/align][/td]

[/tr]

[/table]

表12-4列出了所有处理副本角色的命令，如主坦克和主辅助的分配(assignments)。你可能想知道为什么要保护下面的函数，因为他们似乎不处理可以自动执行游戏的功能。原因是，如果他们没有被保护，你可以写一个插件，总是将你团队中生命值最低的玩家设置为主坦克。一个单位框体插件可以使用一个安全的组页眉(secure group header)来显示你所有的主坦克。这将总是显示最低生命值的玩家，而安全/受污染代码系统(secure/tainted code system)的目的是防止这样的插件。

[align=center][size=80%]表12-4 副本角色可用的安全斜杠命令[/size][/align]

[table]

[tr]

[td50][align=center]命令[/align][/td]

[td][align=center]描述[/align][/td]

[/tr]

[tr]

[td][align=center]/clearmaintank[/align][/td]

[td][align=center]清除你的主坦克分配。别名：/clearmt[/align][/td]

[/tr]

[tr]

[td][align=center]/maintank <target>[/align][/td]

[td][align=center]将目标设定为新的主坦克。也可以使用target选项。别名：/mt[/align][/td]

[/tr]

[tr]

[td][align=center]/maintankoff <target>[/align][/td]

[td][align=center]移除<taget>的主坦克角色。别名/mtoff[/align][/td]

[/tr]

[tr]

[td][align=center]/clearmainassist[/align][/td]

[td][align=center]清除主要协助分配。别名/clearma[/align][/td]

[/tr]

[tr]

[td][align=center]/mainssist <target>[/align][/td]

[td][align=center]设置<target>为主协助。别名/ma[/align][/td]

[/tr]

[tr]

[td][align=center]/mainassistoff <target>[/align][/td]

[td][align=center]移除<target>的主协助状态。别名/maoff[/align][/td]

[/tr]

[/table]

这是四种主要的斜杠命令。但是还有一些其他的命令可以使用。表12-5列出了所有剩余的安全命令。

[align=center][size=80%]表12-5 其他可用的安全斜杠命令[/size][/align]

[table]

[tr]

[td50][align=center]命令[/align][/td]

[td][align=center]描述[/align][/td]

[/tr]

[tr]

[td][align=center]/duel <target>[/align][/td]

[td][align=center]请求与<target>进行决斗。这里不可以使用target选项。[/align][/td]

[/tr]

[tr]

[td][align=center]/forfeit[/align][/td]

[td][align=center]取消决斗。别名：/yield和/concede[/align][/td]

[/tr]

[tr]

[td][align=center]/stopmacro[/align][/td]

[td][align=center]停止宏的执行。[/align][/td]

[/tr]

[tr]

[td][align=center]/click <secure frame> <mouseButton>[/align][/td]

[td][align=center]在全局变量<secure frame>与<mouseButton>中执行存储的安全框体的OnClick处理程序。这是一个非常强大的命令，因为它允许我们点击任意的安全按钮模板，稍后你将看到如何使用它。[/align][/td]

[/tr]

[/table]

在本节的表中有两个命令我们没有完全解释：/castsequence和/equipslot。现在让我们仔细看看。

[b][size=130%]&#9675; 施法序列(Cast Sequences)[/size][/b]

施法序列命令一个额外选项：reset选项，它定义了何时重置为施法序列中第一个法术的标准。该选项不是必须的，默认情况下，施法序列在到达列表中最后一个法术时重新启动。你可能希望reset选项和target选项一样，以方括号的方式添加到命令参数的开头，但事实并非如此。重置选项必须放在可选条件(即target选项)和法术列表之间。下面的例子会说明这一点：

[quote]

/castsequence [target=focus] reset=30 <spell1>,<spell2>,...

[/quote]

每次宏被执行的时候都会启动一个30秒的计时器，当计时器到期时就会重置为第一个法术。这意味着如果30秒内没有施法序列，它将重置。但reset接收的不仅仅是数字，它也可以是定义重置操作的触发器和字符串。你可以使用以下触发器。

[quote]

1、当你改变你的目标时，目标重置序列。

2、当你离开战斗，战斗触发。

3、当你在执行宏时，按住键盘修饰键ctrl、alt和shift重置序列。

[/quote]

斜杠命令处理程序后面的代码只是在这个选项上使用string.find，如果它找到匹配的触发器，就会重置序列，因此你可以设置多个触发器，当其中一个触发器发生时重置。例如，reset=ctrltarget30，当你按下Ctrl再点击该宏时、当你更改你的目标时，或当你30秒没有使用宏时，将重置它。

[b][size=130%]&#9675; 物品栏ID(Item Slot IDs)[/size][/b]

安全命令/equipslot <slot> <item>带有一个物品(item)将要被装备到的物品栏ID(item slot ID)。命令/use <item>和/cast <item>也接收这样的ID代替物品名称。

有两种类型的清单ID(inventory IDs)：字符串(比如你头部的头部装备栏(HeadSlot))和数字(比如你的头部是1)。宏只能使用数字标识符，而插件也可以(而且应该)在API函数中使用字符串标识符：

[quote]

id, texture = GetInventorySlotInfo(slotName)

[/quote]

图12-1 显示了所有装备栏的槽位标识符(slot identifiers)

[align=center][img]./mon\_202011/27/-10er0nQ5-6294ZlT3cScw-et.png.medium.jpg[/img][/align]

[align=center][size=80%]图12-1 装备栏ID[/size][/align]

这些数字对我们很重要，因为我们不能字宏命令中使用字符串。我们现在可以编写简单且强大的物品使用宏。想象你装备了两个“在Y秒内产生X额外伤害”的花哨饰品。他们通常共享一个短暂的冷却时间，因此你不能同时激活他们。有一个宏可以激活当前没有冷却时间的饰品是非常有用的。

这里用/use(或/cast)命令会非常简单，我们可以尝试在同一个宏中激活这两个饰品。

[quote]

/use 13

/use 14

[/quote]

如果其中一个处于冷却状态，它将被跳过，并显示通常的“该物品尚未准备好”消息。稍后你将看到如何屏蔽(suppress)此错误消息。

到目前为止，我们尝试的宏非常简单，你可能已经知道了如何使用大多数命令。让我们创建一些更高级的带有条件的宏。

======

[color=blue][b][size=160%]&#9679; 条 件[/size]

[size=130%](Conditions)[/size][/b][/color]

条件以方括号的形式添加到宏中，就像target选项一样(从语法上讲，它实际上也是一个条件)。一个简单条件的好例子是exists，如果命令的目标存在，则赋值(evaluates)为true，如果目标不存在则赋值为false。如果你有一个目标并且不做其他事情，下面的斜杠命令对你的目标施放<spell>。

[quote]

/cast [exists] <spell>

[/quote]

你还可以使用target选项来修改命令的目标。exists条件也指向被目标选项指向的单位。如果你有目标，以下命令对你的目标施放<spell>。

[quote]

/cast [target=focus, exists] <spell>

[/quote]

在一个命令中，多个条件总是用逗号分隔。我们现在可以向这个宏添加其他条件，但是首先让我们对现有的条件进行概述。

[quote]

小贴士：使用红色问好作为宏的图标，因为它会根据条件自动选择适当的图标。

[/quote]

[size=130%][b]&#9679; 可用的条件(Available Conditions)[/b][/size]

表12-6列出了所有可用的条件，并进行了简短的描述。一些更复杂的条件，如装备，将在本节后面详细解释。

[align=center][size=8%]表12-6 可用条件[/size][/align]

[table]

[tr]

[td50][align=center]条件[/align][/td]

[td][align=center]描述[/align][/td]

[/tr]

[tr]

[td][align=center]bar:x[/align][/td]

[td][align=center]如果动作条x当前被选中，则为true。[/align][/td]

[/tr]

[tr]

[td][align=center]btn:x[/align][/td]

[td][align=center]如果宏被鼠标按钮x点击则为true，就像安全模板属性按钮后缀一样。x也可用是任何作为按钮传递给/click命令的字符串。[/align][/td]

[/tr]

[tr]

[td][align=center]channeling:<spell>[/align][/td]

[td][align=center]如果你当前正在引导<spell>则为true。该法术是可选的，如果你当前正在使用任何法术，只要是引导而不是法术，则赋值为true。[/align][/td]

[/tr]

[tr]

[td][align=center]combat[/align][/td]

[td][align=center]如果你在战斗中则为true[/align][/td]

[/tr]

[tr]

[td][align=center]dead[/align][/td]

[td][align=center]如果命令目标死亡则为true[/align][/td]

[/tr]

[tr]

[td][align=center]equipped:<itemType>[/align][/td]

[td][align=center]如果你一个<tiemType>类型的物品被装备，则为true。这个参数稍微复杂一些，稍后会详细解释。别名：worn:<itemType>[/align][/td]

[/tr]

[tr]

[td][align=center]exists[/align][/td]

[td][align=center]如果目标存在则为true[/align][/td]

[/tr]

[tr]

[td][align=center]flyable[/align][/td]

[td][align=center]如果你在外域(Outlands)或诺森德(Northrend)并且可以飞行(mount)，则为真。警告：这个条件像在冬拥湖(Wintergrasp)或达拉然(Dalaran)这样的区域中赋值为true(这是一个使这个条件几乎完全没用的bug)[/align][/td]

[/tr]

[tr]

[td][align=center]flying[/align][/td]

[td][align=center]如果你是正在飞行则为true[/align][/td]

[/tr]

[tr]

[td][align=center]group:<type>[/align][/td]

[td][align=center]有效的群组类型(group type)是队伍和团队，如果你在这样的群组中，则赋值为true。[/align][/td]

[/tr]

[tr]

[td][align=center]harm[/align][/td]

[td][align=center]如果你的目标是敌对则为true[/align][/td]

[/tr]

[tr]

[td][align=center]help[/align][/td]

[td][align=center]如果你的目标是友方则为true[/align][/td]

[/tr]

[tr]

[td][align=center]indoors[/align][/td]

[td][align=center]如果不能骑乘则为true[/align][/td]

[/tr]

[tr]

[td][align=center]mod:<modifiler>[/align][/td]

[td][align=center]如果宏被带有修饰符的按键点击，则为ture。有效的<modifier>值为ctrl、alt和shift。[/align][/td]

[/tr]

[tr]

[td][align=center]mounted[/align][/td]

[td][align=center]如果你在一个坐骑上则为true[/align][/td]

[/tr]

[tr]

[td][align=center]outdoors[/align][/td]

[td][align=center]如果可以骑乘坐骑则为true[/align][/td]

[/tr]

[tr]

[td][align=center]party[/align][/td]

[td][align=center]如果你的目标在你的队伍中则为true[/align][/td]

[/tr]

[tr]

[td][align=center]pet:<name>[/align][/td]

[td][align=center]如果你当前的宠物是<name>则为true。该参数是可选的，也可以只使用pet来检查任何宠物。[/align][/td]

[/tr]

[tr]

[td][align=center]stance:x[/align][/td]

[td][align=center]如果你目前以ID为x的姿态(stance)或变形形态，则为真。姿态ID将在稍后详细说明。ID是可选的，你可以只使用stance来检查任何姿态或变形形态是否处于激活状态。别名：form:x[/align][/td]

[/tr]

[tr]

[td][align=center]stealth[/align][/td]

[td][align=center]如果你处于潜行则为true[/align][/td]

[/tr]

[tr]

[td][align=center]swimming[/align][/td]

[td][align=center]如果你当前正在游泳则为true[/align][/td]

[/tr]

[/table]

这些条件可以用于除了决斗命令/duel和/forfeit之外的所有安全命令。

这个列表中较为复杂的条件之一是equipped，它接受物品的类型而不是名称。每个物品都有一个类型和一个子类型，例如，可以使用他们对拍卖行重点物品进行分类。在这个宏条件中可以使用他们的标识符是拍卖行所显示的物品类型的本地化字符串。例如你可以使用条件equipped:Clot来检查你当前是否穿着布甲装备。在这里使用装备类型没有太大意义，有趣的选择是使用武器类型。例如，盗贼可以使用以下宏，在一个简单的宏中结合背刺(Backstab)(需要一把匕首)和影袭(Sinister Strike)，具体取决于当前装备的武器。

[quote]

/cast [equipped:Daggers] Backstab

/stopmacro [equipped:Daggers]

/cast Sinister Strike

[/quote]

如果你正在使用匕首，这将施放背刺并取消该宏。否则它施展邪恶打击。在下一节中，你将看到使用一个cast命令编写这样一个宏的方法。

[quote]

注意：物品类别总是复数，所以使用[equipped:Dagger]不起作用。

[/quote]

另外两个重要的分类是单手武器(One-Hand)和双手武器(Two-Hand)，如果你使用单手武器或双手武器，则它们会被赋值为真。

表中完全没有解释的另一个条件是stance:x。姿态是战士的不同姿态、还有德鲁伊变形形态、牧师的暗影形态和救赎之魂、盗贼潜行、萨满的幽灵狼和术士的恶魔形态。姿态ID 1指的是你在变形/姿态条中可见的第一个姿态或变形，2指的是第二个，以此类推。例如姿态1：指战士的战斗姿态或德鲁伊的熊姿态。

姿态ID可以指向不同的形态或姿态，取决于德鲁伊的天赋(talent)和专精(specialization，spec)。对德鲁伊来说，stance:5指的是平衡专精下的枭兽(Moonkin)形态，和恢复专精下的生命树(Tree of Life)形态。牧师稍微复杂一点，如暗影形态可用的话，stance:1表示暗影形态，如果救赎之魂可用但暗影形态不可用，则表示救赎之魂。如果两种天赋都点了，救赎之魂是stance:2。

[quote]

注意：圣骑士的光环和猎人的守护灵不是姿态，即使它们出现在变形/姿态栏上。这意味着你不能使用姿态条件来检查哪一个光环或守护灵当前处于激活状态。

[/quote]

我们已经讨论了所有可用使用的条件。让我们结合它们来构建复杂的表达式。

[b][size=130%]&#9675; 组合条件(Combining Conditions)[/size][/b]

检查两个条件最简单的方法实际上并不是结合两个不同的条件。所有条件以condition:argument的形式用斜杠分隔开来，这些参数将用逻辑或OR组合在一起。如果你按下Shift或Alt键，下面的命令可用施放<spell>：

[quote]

/cast [mod:shift/alt] <spell>

[/quote]

但是我们也可用在一个命令中组合两个不同的条件。最简单的组合是连词(用逻辑和AND连接条件)。

[b]· 逻辑与AND(The Logical AND)[/b]

你已经在上一节中已经看到了一个示例，因为target选项在语法上是一个条件：

[quote]

/cast [target=focus,exists] <spell>

[/quote]

我们可用添加另一个用逗号分隔的条件来检查它们。以下示例在你的目标存在且死亡时对其施展复活：

[quote]

/cast [exists,dead] Resurrection

[/quote]

设置条件以检查目标是否存在或是否死亡也非常的有用。我们需要逻辑非NOT来完成这件事。

[b]· 逻辑非NOT(The Logical NOT)[/b]

条件的否定也可用轻松完成。只需向条件添加前缀no来否定它(应用逻辑NOT)。因此，没有死亡意味着你的目标是或者的。如果你的目标存在并且还活着，以下的宏可用对他施放<spell>：

[quote]

/cast [exists,nodead] <spell>

[/quote]

逻辑NOT被局限于一个单一的条件，不能再宏条件中添加括号。让我们试着否定上一个例子中的条件。想象你把这个否定的表达式写成Lua表达式，就像这样：

[quote]

not (exists and not dead)

[/quote]

在宏中不可能使用像[no(exists,nodead)]这样的条件。我们需要将这个表达式转换成可用在宏中表示的等价表达式。我们可用通过分析条件的含义在这样一个简单的例子中做到这一点。这意味着我们的目标要么不存在，要么已经死亡了。在Lua语法中可用将其写成(不存在)或死亡。稍后你将看到如何使用逻辑或宏。

这两种形式之间的转换并不总是那么明显，特别是当你有更复杂的表达式时。但是布尔代数中有一个简单的法则可用用来做这个变换。德摩根定律是这样的：

[quote]

not (A and B) = (not A) or (not B)

[/quote]

如果我们把这个应用到我们的表达式not (exists and not dead)，我们会得到(not exists)或(not not dead)。显然，not not dead的意思和dead是一样的，所以我们得到(not exists)或dead。

本法则也适用于用or替代and。

[quote]

not (A or B) = (not A) and (not B)

[/quote]

但我们仍不能在宏中使用表达式，因为我们还没有讨论逻辑，让我们看看它是如何工作的。

[b]· 逻辑或OR(The Logical OR)[/b]

若要用逻辑OR组合两个条件，请在方括号中添加另一个表达式块。最后一节中的示例如下所示。

[quote]

/cast [noexists][dead] <spell>

[/quote]

如果你的目标不存在或已经死亡，此技能将会使用<spell>。OR操作符在某种程度上也受到了限制。假如你有一个(A or B) and (C or D)形式的表达式，你可能认为可用把宏条件写成[A][&#65314],[C][D]，但这是不对的。你还必须将表达式转换成析取范式(disjunctive normal form)，将表达式分解为多个子表达式，这些子表达式可用像正常条件一样简单，并与逻辑OR组合在一起。

对于这种表达式的变换还有一个有用的定律——分配律

[quote]

A and (B or C) = (A and B) or (A and C)

A or (B and C) = (A or B) and (A or C)

[/quote]

让我们试着将这个定律应用于表达式(A or B)and(C or D)，将其转换为适合于宏的形式。

[quote]

(A or B) and (C or D)

= ((A or B) and C) or ((A or B) and D)

= ((A and C) or (B and C)) or ((A and D) or (B and D))

= (A and C) or (B and C) or (A and D) or (B and D)

[/quote]

表达式的这种形式与原始形式具有相同的含义，可用用作如下所示的宏条件：

[quote]

/cast [target=player][target=focus] <spell>

[/quote]

第一个代码块总是为true，因为它不包含条件，所以第二个块永远不会执行，它的target选项没有影响。但我们依然可以向代码块添加条件，以根据这些条件设置target选项。我们可用使用这个技术来扩展本章的第一个宏，这个宏可用对你目标的目标施放治疗法术。

[quote]

/cast [target=targettarget] <healing spell>

[/quote]

你可以用这个宏以一个怪物为目标，并且治疗该当前拉住怪物的坦克玩家，但当我们的目标是一个友方玩家时，它是没有用的。下面的宏会检查我们当前的目标是否是友方，如果是，会对他施放治疗；否则它对目标的目标施放治疗。

[quote]

/cast [target=target,help] [target=targettarget] <healing spell>

[/quote]

注意：第一个target选项实际上不是必须的，因为target默认设置为target。

我们可以进一步扩展宏，首先检查鼠标悬停单位ID。这个单位ID与安全单位模板(secure unit templates)一起工作，它允许你用鼠标悬停在你的副本框体来治疗玩家。接下来的检查可以是目标，然后是目标的目标。这些单位ID上可能会出现help fails的情况，但在这种情况下，我们能然可以对自己施放这个魔法。以下是懒惰治疗者的终极宏，它执行前面提到的所有检查，并选择一个可以治愈的目标：

[quote]

/cast [target=mouseover,hepl][target=target, help][target=targettarget,help] [target=player] <healing sepll>

[/quote]

注意，第二个代码块中需要target=target，因为第一个块将目标设置更改为mouseover(鼠标悬停)。仅仅使用[help]作为第二个是没有意义的，因为它检查鼠标悬停目标是否是友方，事实并非如此。我们知道，因为如果它是友方的话，就永远不会到达第二个块。

[quote]

注意：不需要添加检查目标是否还活着，因为条件帮助的作用不仅仅在于检查目标是否友好。它会检查目标是否是有益法术的有效目标，而死亡的目标不是这类法术的有效目标(复活术除外，复活术不能与help检查同时使用。)

[/quote]

我们现在可以用条件构建复杂的表达式，但是整个命令的形式仍然非常简单：使用if <expression>，之后do一些操作，最后end构造。有一个elseif或else块将是一个很好的补充。

[b][size=130%]&#9675; 简单的控制构造(Simple Control Structures)[/size][/b]

宏中唯一可用的控制结构是if-else-elseif块。在宏中没有循环这样的东西。

宏中else或selseif块以分号开始。然后是要使用的条件和宏参数。我之前展示了一个宏，如果你装备了一把匕首，它就会使用背刺，否则就会使用影袭。

[quote]

/cast [equipped:Daggers] Backstab

/stopmacro [equipped:Daggers]

/cast Sinister Strike

[/quote]

这个宏使用stopmacro命令，正如我所说的，我们将尝试它的另一个版本，它只包含一行代码。我们只需要添加一个分号，后面跟着else块的内容，这里只有使用Sinister Strike(影袭)作为参数施放。

[quote]

/cast [equipped:Daggers] Backstab; Sinister Strike

[/quote]

elseif-block在分号后面使用额外条件，就像下面的宏，如果你有敌方目标，它会施放伤害法术，如果你的目标是友方目标，它会施放治疗法术：

[quote]

/cast [harm] <damage spell>;[help] <healing spell>

[/quote]

控制结构的另一个有趣的用法是使用btn或mod条件来根据所使用的鼠标按钮或键盘修饰符执行不同的操作。盗贼可用使用下面的宏，先用鼠标左键点击，然后再用鼠标右键点击，对两种武器都施加毒药。

[quote]

/cast Wound Poison VII

/cast [button:1]; [button:2] 17

[/quote]

第一个命令使用毒药，然后要求你选择一个目标。如果你左键点击宏，第二个命令就会在你的主手上使用放置的技能，如果你右击宏，第二个命令就会在右手上使用放置的法术。

另一个非常强大的宏是为了试图打断一个法术的战士准备的。战士可以在战斗中打断法术并且还拥有带盾牌猛击(Shield Bash)(需要盾牌)的防御姿态。拳击(Pummel)可以用于狂暴姿态，这个法术不需要特定的武器类型。以下宏在你处于狂暴姿态时施放拳击，否则它会检查你是否装备了盾牌，然后施放盾牌猛击。如果这个检查也失败了，也就是说你处于战斗姿态或防御姿态，并且没有装备盾牌，它就会释放狂暴姿态，所以你可以第二次执行这个宏。

[quote]

/cast [stance:3] Pummel; [equipped:Shields] Shield Bash; Berserker Stance

[/quote]

注意不可能一下就施放狂暴姿态和拳击，因为狂暴姿态会触发全局冷却时间。

你可能已经注意到的一件事是，如果你选择红色问好作为宏图标，它们会自动显示正确的法术。但是我们也可以控制显示哪些法术和鼠标提示。

[b][size=130%]&#9675; 宏的反馈(Macro Feedback)[/size][/b]

当你在当前环境下执行宏时，它总是使用法术的图标和冷却时间。但这并不足够，并且可以在宏的开始使用命令#show或#showtooltip来控制这种行为。这两个命令还需要宏的问号图标。

这两个命令都很简单。它们带有一个单独的参数，然后显示在该宏的动作条中。下面的宏在动作条中显示为攻击，其行为就像在动作条中有普通攻击，而不是宏一样。但是当你单击它时，它会施放Flash Heal

[quote]

#show Attack

/cast Flash Heal

[/quote]

命令#showtooltip遵循完全相同的语法。唯一的不同是，将鼠标放在宏上时，还会获得该法术的鼠标提示，它还设置图标。#show命令仅设置图标，鼠标提示宏的名称。

也可以从安全命令中使用已知的条件，并且分号用于简单的if-then-else结构。下面的宏创建了一个按钮，如果你有匕首，则显示背刺，否则显示影袭。

[quote]

#showtooltip [equipped:Daggers] Backstab; Sinister Strike

[/quote]

但是以前使用相同条件的宏已经具有此功能(鼠标提示除外，该提示仅显示了宏的名称)。宏的图标已经正确，并且你几乎不需要操作按钮的鼠标提示。因此你可能向知道这些命令的真正用途是什么。答案是，你不能始终在宏中使用带有条件的简单斜杠命命令。例如，当实际工作由Lua代码完成时，你将需要此功能，因为/script命令没有被解析确定要显示的图标。在本章的最后，当我们创建一个挂载宏时，你将看到一个这样的例子。

你还需要使用#showtooltip，如果你的宏有一个stopmacro命令，你想要显示一个不同的图标，如果命令的条件是满足的，因为当确定图标/鼠标提示显示时，游戏不检查停止条件。但是，你很少需要根据stopmacro条件显示不同的图标，因为你通常可以通过使用控制结构来避免使用stopmacro。

showtooltip的另一个可能用途是当你的宏调用其他宏或执行安全按钮的单击操作时。例如，假设在第一个动作中有两个宏，两个动作槽，ActionButton1和ActionButton2。你现在想创建一个宏，如果你有一个敌对的目标，它会执行宏1，否则会执行宏2。

[quote]

#showtooltip [harm] <damage spell>; <heal spell>

/click [harm] ActionButton1; ActionButton2

[/quote]

如果我们不在这里使用#showtooltip，按钮根本不会显示图标，因为在确定要显示图标时，WoW不会解析/click所引用的按钮。但是，我们为什么要编写这样一个简单调用另一个宏呢？因为其他宏的长度也可以达到255个字符，这意味着我们已经成功地绕过了宏只有255个字符的限制。但还有其他方法可以做到这一点，而不必将宏分割成许多小的部分。让我们看看如何结合/click和安全动作按钮模板来创建非常长的宏。

======

[color=blue][b][size=160%]&#9679; 突 破 255 个 字 符 的 限 制[/size]

[size=130%](Circumventing the 255-Character Limit)[/size][/b][/color]

一个宏很少需要超过255个字符，但有一种情况是，你想要完成“杀死所有稀有怪物”的成就。它是一个真正有用的宏，试图以所有稀有怪物为目标，和当你以它为目标时显示一条消息。之后你可以在西游怪物的刷新点到处飞，并一直点击宏。

我们需要使用Lua代码创建一个安全动作按钮，并将其类型设置为macro，将其属性macrotext设置为如下内容。

[quote]

/targetexact Loque’nahak

/targetexact High Thane Jorfus

...etc with all 23 Northrend rare spawns

/stopmacro [noexists]

/script ChatFrame1:AddMessage(“Found a rare spawn: ”..UnitName(“target”))

[/quote]

这显然不是一个普通的宏，所以让我们为它构建一个安全操作按钮。创建在宏中使用的按钮的最佳位置是在单独的插件中。我有一个名为RandomCrap的小插件，它由一个XML和一个Lua文件组成，其中包含我经常使用的一些函数和模板。但如果你只是想测试它，你也可以在一个插件如(TinyPad)中输入以下代码。

创建这样一个宏最简单的方法是使用23个/targetexact命令编写这样一个字符串。注意，用方括号分隔字符串允许在字符串中使用新行，因此可以在代码中逐字地编写长宏。但是，更聪明的做法是创建一个包含所有boss名称的表，并从这个字符串自动宏文本。它更短，更灵活，如果你想允许部分匹配，你可以很容易地将target命令从targetexact更改为/target：

[code=lua]

local rarespawns = {“Loque’nahak”, “Hildana Deathstealer”, “Fumblub Gearwind”, “Perobas the Bloodthirster”, “King Ping”, “Crazed Indu’le Survivor”, “Grocklar”, “Syreian the Bonecarver”, “Griegen”, “Aotona”, “Vyragosa”, “Putridus the Ancient”, “High Thane Jorfus”, “Old Crystalbark”, “Icehorn”, “Vigdis the War Maiden”, “Tukemuth”, “Scarlet Highlord Daion”, “Seething Hate”, “Zul’drak Sentinael”, “Terror Spinner”, “King Krush”, “Dirkee”}

local targets = “”

for i, v in ipairs(rarespawns) do

targets = targets..”/targetexact ”..v..”\n”

end

local frame = CreateFrame(“Button”, “RareSpawns”, nil, “SecureActionButtonTemplate”)

frame:SetAttribute(“type”, “macro”)

frame:SetAttribute(“macrotext”, target..[[

/stopmacro [noexists]

/script ChatFrame1:AddMessage(“Found a rare spawn: ”..UnitName(“target”))]])

[/code]

该代码首先生成宏文本的/targetexact部分，然后创建一个安全操作按钮，并将其类型设置为macro。然后将属性macrotext设置为前面生成的字符串，然后是停止条件和显示消息的代码。

这项技术适用于所有需要255个字符以上的宏，但它也有一个小缺点：你必须使用#show(或#showtooltip)命令来在动作条中获得图标和工具提示。我们现在可以构建一个宏，它可以单击带有长宏属性的不可见的安全操作按钮。这个宏非常简单，因为/click命令需要一个包含按钮的全局变量。

[quote]

/click RareSpawns

[/quote]

这与单击我们创建的安全按钮具有相同的效果。

======

[color=blue][b][size=160%]&#9679; 在 宏 中 使 用 Lua[/size]

[size=130%](Using Lua in Macros)[/size][/b][/color]

还有最后一个主题，它实际上与传统的宏没有什么关系——在宏中使用Lua。这非常简单，只需要使用斜杠命令/script。但是在宏中由上面Lua可用呢？请记住，不能从Lua中施展法术或使用物品，因为这些函数受保护，因此只能从安全代码中调用。来自宏的代码永远不安全。

[b][size=130%]&#9675; 发送聊天消息(Sending Chat Message)[/size][/b]

在上一节中，你在宏中看到了一个简单的Lua聊天消息例子。我们调用ChatFrame1:Addmessage(msg)来在聊天框中显示一个通知。另一个类似的用途是发送聊天消息，但这不一定需要Lua代码。简单的聊天消息可用发送与常见的斜杠命令，如/say和/raid。甚至可用通过在消息中使用%t将目标包含在聊天消息中，它将被替换成为当前目标的名称。这不是一个特定于宏的函数，你可用在你的聊天信息中使用%t来指向你在《魔兽世界》中的目标。以下的宏对德鲁伊很有用，它施放复生，并发送复活玩家的名字到团队聊天中。

[quote]

/cast Rebirth

/raid Battle res on %t!

[/quote]

这个宏根本不使用任何Lua。但是这个宏的另一个有用功能是向复活玩家发送一个私聊，这样它就可用准备回到战斗中。你可能会认为这里有可能使用/w %t Battle res incoming。但它只适用于聊天信息，而不是私聊的目标。这意味着我们需要Lua代码啊向当前目标发送一个私聊。

[quote]

/cast Rebirth

/raid Battle res on %t!

/script SendChatMessage(“Battle res incoming!”, “WHISPER”, nil, UnitName(“target”))

[/quote]

在宏中使用Lua的一个明显缺陷是很快就会达到255个字符的限制。解决方案是将实际的Lua代码外包到一个小插件中，然后提供一个简单的斜杠命令或从宏调用Lua函数。下面的代码创建了斜杠命令/wtarget <msg>，它将发送消息<msg>到你的当前目标：

[code=lua]

SLASH\_WHISPERTARGET1 = “/wtarget”

SLASH\_WHISPERTARGET2 = “/wtar” -- alias /wtar <msg>

SlashCmdList[“WHISPERTARGET”] = function(mag)

if UnitName(“target”) then

SendChatMessage(msg, “WHISPER”, nil, UnitName(“target”))

end

end

[/code]

你现在只需在宏中使用以下命令：

[quote]

/wtar Battle res incoming!

[/quote]

除了更简短和更容易使用之外，当你没有目标时，它不会产生错误消息。

另外一个有用的功能是安排在特定时间之后发送消息。着允许你创建倒计时宏，这是非常有用的，所以让我们看看如何完成它。

[b][size=130%]&#9675; 计时宏(Timers in Macros)[/size][/b]

实现这个最简单的方法是从你的宏中使用SimpleTimingLib。但是创建一个以秒为单位的斜杠命令比在某段时间之后执行的斜杠命令更容易。下面的代码创建了一个斜杠命令，该命令在给定时间后执行另一个斜杠命令。它需要安装SimpleTimingLib库，要么作为独立运行的版本，要么嵌入到插件中。

[quote]

local function runCmd(cmd)

local old = ChatFrameEditBox:GetText()

ChatFrameEditBox:SetText(cmd)

ChatEdit\_SendText(ChatFrameEditBox)

ChatFrameEditBox:SetText(old)

end

SLASH IN1 = “/in”

SlashCmdList[“IN”] = function(msg)

local time, cmd = msg:match(“(%d+)(.+)”)

if cmd:sub(1,1) ~= “/” then

cmd = “/”..cmd

end

SimpleTimingLib\_Schedule(time, runCmd, cmd)

end

[/quote]

函数runCmd通过使用一个技巧来执行实际的命令——它修改默认聊天框中包含的文本，并调用函数ChatEdit\_SendText(editBox)，该函数在文件FrameXML\ChatFrame.lua中定义。它还可以在这里解析命令并找到与之关联的斜杠命令处理程序或表情，但这将是一项相当大的工作(需要大约30行代码)。如果你想找到斜杠命令在默认UI中式如何处理的，可以在文件frameXML\ChatFrame.lua中读取函数ChatEdit\_ParseText(editBox, send)。

我们现在可以在宏中使用/in命令，例如这样：

[quote]

/raid Pull in 3 seconds!

/in 1 /raid Pull in 2 seconds!

/in 2 /raid Pull in 1 second!

/in 3 /raid Pull now!

[/quote]

还可以省略要调用的命令中的前斜杠，因为/in处理程序会在命令不是以斜杠开头的情况下添加一个斜杠。语法/in 3 raid Pull now！也会执行。

[quote]

注意：/in命令无法执行任何安全命令，因为执行路径被提供斜杠命令的插件污染了。

[/quote]

Deadly Boss Mods(DBM)附带的另一个计时器斜杠命令式所谓的披萨计时器(pizza timer)，可以通过斜杠命令/dbm timer xx:yy <text>来开始。它显示了一个带有名字<text>的DBT计时器。你可以在一个宏中使用它来跟踪某个技能或法术的冷却时间。但是还有另一个更强大的斜杠命令：/dbm broadcast timer xx:yy <text>，它广播一个dbm计时器给你的队伍或团队群组。此命令需要团队助手(raid assistant)或团长(leader)状态。

团长可以使用如下的宏来为团队宣布一个短暂的休息时间。

[quote]

/rw Have a short break! (5 min)

/dbm broadcast timer 5:00 Break

[/quote]

[b][size=130%]&#9675; 抑制错误消息(Suppressing Errors)[/size][/b]

为什么会有人想要禁止错误消息呢？答案是有一些宏总是生成错误消息，即使它们被成功执行。这方面的一个例子是我们前面使用的饰品宏。

[quote]

/use 13

/use 14

[/quote]

这个宏试图同时使用两个饰品，这总是会生成一个错误消息。如果第一个/use命令成功，第二个/use命令会因为全局冷却而失败。去掉红色的“物品还没有准备好”消息的一个简单方法是清除显示该消息的框体。这个框体存储在全局变量UIErrorsFrame中，它的类型是MessageFrame。这种消息框体的工作原理类似于聊天框体(滚动消息框体)，但没有滚动功能。你可以使用方法frame:AddMessage(msg)向这样的框架添加消息，最重要的是使用methodframe: clear()消除消息。此框体类型(以及所有其他框体类型)的完整参考可在附录A中获得。

下面的宏在尝试执行两个/use命令后清除UIErrorsFrame。

[quote]

/use 13

/use 14

/script UIErrorsFrame:Clear()

[/quote]

当然，也可以将此代码放在一个插件中。但是这里的Lua代码相对较短，因此不值得这样做。

[b][size=130%]&#9675; 优化选项(Tuning Options)[/size][/b]

一些常用的宏不同于其他宏，因为它们基本上只是执行一次Lua脚本。这些“宏”简单地设置了一个所谓的cvar(控制台变量(console variable)，一个存储游戏设置的变量，比如你的视频选项)，这也可以通过编辑文件World of Warcraft\WTF\config.wtf来完成。这样的选项可以通过调用SetCVar(“option”, “value”)来设置，并使用GetCVar(“option”)来获取。

下面是一个常见的宏示例，它允许你通过设置选项进一步缩小camerDistanceMaxFactor到2.5。在界面选项菜单的距离滑块(The distance slider)最多到2.0。

[quote]

/script SetCVar(“camerDistanceMaxFactor”, ”2.5”)

[/quote]

注意这个选项的指保存在config.wtf中，这意味着执行了这个“宏”一次，并且你可能根本不会将这个简单的Lua脚本存储在一个宏中。

还有许多其他可用选项，可以将他们设置为比选项菜单所允许的更高的值。控制“地面场景(Ground Clutter)”密度和据伦理的视频选项就是一个很好的例子。下面的宏将这两个选项设置为视频效果菜单中滑块之外的值。

[quote]

/script SetCvar(“groundEffectDensity”, “256”)

/script SetCVar(“groundEffectDist”, “140”)

[/quote]

这增加了杂草和其他植被的数量和可见范围。然而，如果你没有一个高端的配置，这可能会大大降低你的FPS。请注意，配置菜单中的选项被限制为较低的值是有原因的。

[b][size=130%]&#9675; 绕过[flyable]bug(Circumventing the [flyable] Bug)[/size][/b]

另一个非常有用的宏是在允许飞行坐骑或普通坐骑的区域使用飞行坐骑。如果条件flyable正常工作，这看起来是一个非常简单的宏。你可以简单地使用以下宏来做到这一点：

[quote]

/cast [flyabel] <flying mount>

/cast [noflyable] <normal mount>

[/quote]

但是飞行不能正常工作。如果你在外域或诺森德允许飞行的区域，则此条件的值为true。在《巫妖王之怒》之前，在外域任何地方，普通坐骑允许的地方飞行坐骑也是被允许的。这在巫妖王之怒中不在正确，因为你不能在达拉然(除了克拉瑟斯的着陆点)或冬拥湖，但是那里仍然是true。这意味着宏在这些区域中失效。

但是使用坐骑不是一个受保护的动作，即使它可以通过安全命令/cast完成。在3.0补丁中引入的函数CallCompanion(type, id)可以用来召唤非战斗宠物或坐骑。类型可以是“CRITTER”来召唤非战斗宠物，也可以是“MOUNT”来使用普通坐骑。坐骑的ID可以从默认的宠物和坐骑菜单获得，第一个坐骑(或宠物)的ID为1，第二个为2，以此类推。

下面的代码创建一个斜杠命令/mount，它使用基于当前所在区域的可用坐骑列表中随机飞行坐骑或普通坐骑。

[code=lua]

local groundMounts = {1, 16, 26}

local flyingMounts = {5, 10, 22}

SLASH\_MOUNT1 = “/mount”

SlashCmdList[“MOUNT”] = function(msg)

local zone = GetRealZoneText()

local subzone = GetSubZoneText()

if IsMounted() then

Dismount()

elseif IsFlyableArea() and zone ~= “WWintergrasp”

and (zone ~= “Dalaran” or subzon == ”Krasus’ Landing”) then

CallCompanion(“MOUNT”, flyingMounts[math.random(#flyingMounts)])

else

CallCompanion(“MOUNT”, groundMounts[math.random(#flyingMounts)])

end

end

[/code]

注意，这里有必要检查我们是否以及上了坐骑了，如果上了，就解除，因为CallCompanion (type, id)不像/cast命令那样，它不会解除坐骑。之后，斜杠命令使用一个良好的条件来检查是否真的允许在该区域飞行，如果是这种情况，则使用来自flyingmounth表的随机坐骑。否则，它使用表groundMounts中的随机坐骑。如果你想使用基于此脚本的宏，就必须用你喜爱的坐骑的ID来填充这些表。

使用提供这个斜杠命令的插件的一个宏可能是这样的：

[quote]

#showtooltip [flyable] <flying mount>; <normal mount>

/mount

[/quote]

#showtooltip只是为了得到宏的图标和鼠标提示，这个图标在达拉然和冬拥湖中是错误的并不重要。

======

[color=blue][b][size=160%]&#9679; 总 结[/size]

[size=130%](Summary)[/size][/b][/color]

[b][size=150%]本[/size][/b]章节演示了一种与《魔兽世界》API交互的不同方式：通过斜杠命令，乍一看似乎相当简单。但是这些安全的斜杠命令可以与复杂的条件结合使用，这使得它们非常强大。我们查看了所有可用的安全斜杠命令，它们可以执行通常只能从安全代码(即从默认UI)或使用安全模板执行的操作。

我们讨论了如何使用逻辑运算符组合多个条件来构建复杂的表达式。然后我们了解了如何在斜杠命令中的简单 if-then-else-end 块中使用这些表达式。

本章的下一个主题是在宏中使用Lua，我们尝试了许多可用在宏中使用的有用的Lua脚本。你还了解了如何将宏与安全模板结合使用以规避255个字符的限制，如果你希望使用长宏，这一解决方案可能非常重要。

安全斜杠命令和安全模板是执行某些动作的唯一方式，比如施法。在本章中，你看到了如何结合这两种技术来创建及其强大的宏，以帮助你进行副本或PVP对抗。

如果你正在寻找宏，请一定去WoWWiki页面[url]http://ww.wowiki.com/Useful\_macros[/url]逛逛。这个页面有一个相当强大的宏编译，可以用于各种目的，你说不定会在那里找到你要找的东西。你可能已经找到了这个宏列表并从那里复制了一些宏。但是你现在知道了这些宏是如何工作的，并且可用根据自己的需要定制它们。理解一段代码并能够对其进行定制总是比只是复制和粘贴要好。