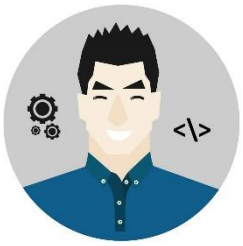

Computer Craft Challeng

功能指令参考

www.bilibili.com uid354153209

目录

说明

在本参考中，您将找到 CCC 中使用的每个功能指令的简短描述。这些功能按应用程序编程接口（API）（也称为模块）分组。该参考资料不是 ComputerCraft 中所有功能的完整列表，但是您可以在 ComputerCraft 提供的文档中（*http://computercraft.info/wiki/Category:APIs*）了解其他可用功能。

如果要了解有关 Lua 的更多信息，可以使用 Lua 5.1 参考手册，该手册可从 *https://www.lua.org/manual/5.1/*在线获得。请注意，即使有较新版本的 Lua，ComputerCraft 也会使用 Lua 5.1。

FS（文件系统）API

乌龟和游戏中的 Minecraft 计算机的文件系统与您的计算机相似。您可以使用 fs API 和文件名与这些文件进行交互：

fs.delete（*string filename*）删除一个名为 *filename 的文件*

fs.exists（*string filename*）如果存在名为 *filename* 的文件，则返回 true；否则，返回 true。否则返回 false

您编写的程序可以与乌龟或游戏计算机上加载的文件交互，但不能与运行 Minecraft 的计算机上的文件交互。

IO（输入 / 输出）API

使用 io API 时，程序可以在屏幕上显示文本，也可以通过键盘从播放器接受文本。io API 中有几个函数，但是最重要的函数是 io.read（）：

io.read（）播放器键入响应并按 ENTER 时，此函数将响应作为字符串值返回。

MATH（数学）API

该数学 API 是 Lua 中的一部分，可以从非 ComputerCraft Lua 程序调用它的函数。该 API 包括以下数字和与数学相关的功能：

math.ceil（*number num*）返回四舍五入的 *num*。

math.floor（*number num*）返回四舍五入的 *num*。

math.random（*number start*，*number end*）返回介于 *start* 和 *end* 之间的随机整数，包括 *start* 和 *end*。在开始和结束的参数都是可选的。如果未传递任何参数，则该函数返回 0.0 到 1.0 之间的小数点。如果未使用 *start* 参数，则该函数返回 1 到 *end* 之间的整数。

OS（操作系统）API

ComputerCraft 操作系统提供以下功能，海龟和游戏中的计算机可以使用这些功能：

os.getComputerLabel（）返回乌龟标签的字符串（即其名称）。

os.loadAPI（*string filename*）将名为 *filename* 的程序作为模块加载，以便当前程序可以调用

其函数。

os.setComputerLabel（*string / nil label*）设置乌龟的标签为 *label*。如果 *label* 为 nil，该函数将删除乌龟的标签。

os.sleep（*time*）暂停的 *time* 为秒数。

SHELL（外壳）API

乌龟可以从 CLI（命令行界面）运行命令，就像播放器可以从 CLI 运行命令一样。您可以使用 Shell API 在 turtle 程序中运行 CLI 命令：

shell.run（*string command*）运行*命令*，就像播放器在 CLI shell 上输入了字符串一样。如果*命令*由于不存在，崩溃或调用 error（）而终止，则返回 false；否则返回 true。

STRING（字符串） API

该字符串 API 是 Lua 中的一部分，可以从非 ComputerCraft Lua 程序调用它的函数。

string.find（*string haystack*，*string needle*）在*干草堆*字符串中查找*针线*，并返回两个整数：找到*针线*的位置和该字符串结束的位置。例如，string.find（'hello'，'el'）返回 2 和 3，因为'el'在'hello'中的第二个字符处找到并在第三个字符处结束。如果*针*串不在*干草堆*串中，则函数返回 nil。该*针*字符串也可以找到文本模式，这超出了本书的范围。您可以在 *https://www.lua.org/manual/5.1/manual.html#5.4.1/*上了解有关文本模式的更多信息。

string.sub（*string bigstring*，*number start*，*number length*）返回 *bigstring* 的子字符串或一部

分，从*起始*位置*开始*并返回下一个*长度的*字符。所述*长度*参数是可选的。如果未传递 *length*，则子字符串从 start 开始，并继续到 *bigstring* 的结尾。例如，string.sub（'hello'，3，2）返回'll'，而 string.sub（'hello'，2）返回'ello'。

TEXTUTILS（打字机）API

使用 textutils，可以使程序一次显示一个字符的文本，以创建精美的打字机效果：

textutils.slowPrint（*string text*，*number rate*）与 print（）相似，只不过它一次将一个字符写入文本中的字符。它的速率参数是可选的，并且指定有多少个字符每秒被打印。

TURTLE（海龟）API

该 turtle API 包含所有常用的功能，你的程序可以调用，使海龟执行某些操作。接下来让我们根据功能可以触发的动作来依次查看这些功能。

建筑功能

您可以调用函数以通过放置块来操控海龟。但是相同的功能也会导致海龟执行其他操作，具体取决于海龟当前插槽中的项目：

turtle.place（）对海龟当前插槽中的项目执行操作。对于方块，此功能会将方块放置在 Minecraft 世界中。但是，如果表 1 列出的特殊项目之一在当前插槽中，则该项目将以表 1

指定的方式使用。如果海龟无法放置该方块或对该方块执行操作，则该函数返回 false。

turtle.placeDown（）与 turtle.place（）类似，但对海龟下方的空间的动作。

turtle.placeUp（）与 turtle.place（）类似，但对海龟上方的空间的动作。

表 1：场所功能可以使用的特殊项目

项目	**行动**
盔甲	将装甲放置在装甲架上。
船	将船放在水上。
染料类	染羊。
空桶	收集熔岩或水。也可以从牛身上收集牛奶。
烟花	引燃烟花。
火石和钢	着火可燃块或激活虚空传送门。
矿车	将矿车放到轨道上。
幼树，花朵或种子	将对象种植在泥土或草块中。
剪刀	剪羊并收集羊毛。
标志	在上面放置带有文字的标志。要在标牌上写文本，请向 turtle 函数传递一个字符串。例如，turtle.place（'This \ nis a \ nsign。'）。
生成卵	产生暴民。

燃料功能

海龟每次移动都会消耗一单位（一方格）燃料，但是如果燃料用完了，它们就不会移动。因此，给海龟加油并了解其加油功能很重要：

turtle.getFuelLevel（）返回当前海龟已存储的燃料量。如果 *ComputerCraft.cfg* 配置文件已

禁用燃料需求，则返回“无限制”。

turtle.getFuelLimit（）返回海龟可以存储的最大燃料量。对于大多数海龟，限制为 20,000 单位；对于其他类型的海龟，限制为 100,000 单位。如果 *ComputerCraft.cfg* 配置文件已禁用燃料需求，则返回“无限制”。

turtle.refuel（*number*）消耗当前的燃料项。该 *num* 参数可选。如果 *num* 没有给出，该功能会消耗当前的插槽中的所有物品。

库存功能

每只海龟都有一个带有 16 个编号的插槽。您可以使用各种功能指令使海龟对其执行操作。这些功能通常使用一个数字来指示要对哪个编号的插槽执行操作。

turtle.compareTo（*number slot*）如果当前插槽中的物品与 *slot* 中的物品相同时，则返回 true，否则返回 false。

turtle.drop（*number*）从当前槽位取出物品放入容器内。该 *number* 参数可选。如果 *number* 没有给出，该功能取出当前槽位的所有物品。如果删除了任何物品，则返回 true；否则返回 false。

turtle.dropDown（*number amount*）与 turtle.drop（）类似，不同之处在于此函数将物品放置到海龟 *下方*的空间或容器 *中*。

turtle.dropUp（*number amount*）与 turtle.drop（）类似，不同之处在于此函数将物品放置到海龟 *上方*的空间或容器中。

turtle.equipLeft（）在乌龟的左侧取消装备该工具（如果有），并在当前插槽中装备该工具。如果配备，则返回 true；否则返回 false。

turtle.equipRight（）在海龟右侧取消装备该工具（如果有），并在当前插槽中装备该工具。如果配备，则返回 true；否则返回 false。

turtle.getItemCount（*number slot*）返回的*插槽中*项目数。使用当前插槽，如果*插槽*没有给出。

turtle.getItemDetail（*number slot*）返回有关*插槽中*项的信息的表值，如果插槽为空，则返回 nil。使用当前插槽，如果*插槽*没有给出。

turtle.getItemSpace（*number slot*）返回的剩余空间的*槽位*。

turtle.getSelectedSlot（）返回当前插槽编号（1 到 16）。

turtle.select（*number slot*）将当前插槽更改，从 1 到 16 的数字。

turtle.suck（*number*）海龟*前面*的空间或容器中取出物品。（大多数项目的完整堆栈为 64，尽管某些项目（例如鸡蛋，雪球或空桶）最多只能堆栈 16。）如果已取出任何项目，则返回 true；否则返回 false。

turtle.suckDown（*number amount*）与 turtle.suck（）相似，除了此函数从海龟*下面*的空间或容器*中*取出物品。

turtle.suckUp（*number amount*）与 turtle.suck（）相似，除了此函数从海龟*上方*的空间或容器中取出物品。

turtle.transferTo（*number slot*，*number amount*）将 *number* 项从当前插槽传输到 *slot*。该 *number* 参数可选。如果 *number* 没有给出，该函数传输所有*槽位*项目。如果有任何项目转移，则返回 true；否则返回 false。

运动功能

海龟可以向任何方向移动，只要它们尝试移动的空间尚未被另一个方块占据。您可以使用以

下功能来控制海龟向各个方向移动。如果海龟能够移动，则所有运动功能都返回 true；否则返回 false：

turtle.back（）将海龟后移动一格

turtle.down（）将海龟向下移动一格

turtle.forward（）将海龟向前移动一格

turtle.turnLeft（）将海龟左移；不消耗燃料

turtle.turnRight（）将海龟向右转；不消耗燃料

turtle.up（）将海龟向上移动一个空格

感知功能

乌龟可以使用以下功能检查在它们前面，上面或下面的一个空间中的方块：

turtle.compare（）如果 turtle 前面的块与当前插槽中的方块的类型相同，则返回 true；否则返回 false。

turtle.compareDown（）与 turtle.compare（）类似，但比较的是海龟*下面项目*。

turtle.compareUp（）与 turtle.compare（）类似，但比较的是海龟*上面项目*。

turtle.detect（）如果在海龟前面有一个方块，则返回 true；否则返回 false。

turtle.detectDown（）与 turtle.detect（）类似，但检查的是海龟*下面*的方块。

turtle.detectUp（）与 turtle.detect（）类似，但检查的是海龟*下面*的方块。

turtle.inspect（）返回两个值：true 和一个表值，其中包含有关海龟前面的方块信息。如果海龟前面没有物品，则返回 false。

turtle.inspectDown（）与 turtle.inspect（）类似，但返回海龟*下面*方块的信息。

turtle.inspectUp（）与 turtle.inspect（）类似，但返回海龟*下面*方块的信息。

工具功能

海龟可以使用配备的工具执行操作，并且海龟可以执行的每个操作都有与工具相关的功能。您可以为海龟配备镐、锹、斧、锄、剑、工作台，之后以下功能可用：

turtle.attack（）如果配备了剑，海龟会攻击前面的任何东西。如果暴徒受到攻击，则返回 true；否则，返回 false。

turtle.attackDown（）与 turtle.attack（）类似，但海龟攻击的空间*低于*它。

turtle.attackUp（）与 turtle.attack（）类似，但海龟攻击的空间*高于*它。

turtle.craft（*number*）需要海龟装备一个工作台。如果制作了某些东西，则返回 true；否则返回 false。

turtle.dig（）开采或耕种海龟前面的方块。海龟必须装备有镐才能使用此功能来挖方块。如果已开采了东西，则返回 true；否则返回 false。

turtle.digDown（）与 turtle.dig（）类似，开采或耕种海龟下面的方块。

turtle.digUp（）与 turtle.dig（）类似，开采或耕种海龟上面的方块。请注意，此功能无法阻止污垢。

LUA 功能

Lua 语言随附以下函数，因此在调用这些函数时，您无需在函数名之前键入模块名：

error（*字符串消息*）终止程序并显示*消息*（如果给出）。

exit（）退出交互式外壳。您只能在交互式外壳程序中使用它。

print（*字符串/数字值*）在屏幕上显示*字符串/数字值*，后跟换行符。该参数可选。如果未传递任何内容，则该函数仅显示换行符。

HARE（野兔） API

如果您在编程中用到了 hare API。与本参考中列出的其他 API 不同，ComputerCraft 不附带 hare，因此您必须首先从 CLI 外壳程序运行 pastebin get wwzvaKuW hare 来下载它。您编写的使用 hare API 的每个程序都必须包含代码 os.loadAPI（'hare'），以便该程序调用野兔模块的功能：

hare.buildFloor（*number length*，*number width*）使用海龟库存中的物品构建地板*长度为 length* 的方块，*宽度为 width 的方*块。

hare.buildRoom（*长度*，*宽度*，*高度*）构建一个立方体。

hare.buildWall（*number length*，*number height*）使用海龟库存中的物品来建造长而高的墙体。

hare.countInventory（）返回所有海龟库存槽中的物品总数。

hare.digUntilClear（）继续挖掘海龟前面的空间，直到该空间不包含任何块。当砾石或沙子可能会落在海龟前方时，可以使用此功能。

hare.digUpUntilClear（）与 hare.digUntilClear（）类似，清除海龟*上方*的空间。

hare.findBlock（*string name*）旋转乌龟，并在乌龟面对名为 *name* 的方块时停止。如果乌龟找不到该块，则在功能代码运行完后，乌龟将朝其原始方向结束。如果找到该块，则返回

true；否则返回 false。

hare.selectAndPlaceDown（）选择一个非空的库存槽并将物品放在该槽中。

hare.selectEmptySlot（）选择一个空的库存槽。如果找到插槽，则返回 true；否则返回 false。

hare.selectItem（*string name*）选择一个包含名称为 *name* 的项目的库存槽。如果找到该项目，则返回 true；否则返回 false。

hare.sweepField（*number length*，*number width*，*function sweepFunc*）将海龟移到矩形空间上，并在每个空间处调用 *sweepFunc*。