key | Pygame中文文档

pygame.key

与键盘相关的 Pygame 模块。

函数

- pygame.key.get_focused() — 当窗口获得键盘的输入焦点时返回 True
- pygame.key.get_pressed() — 获取键盘上所有按键的状态
- pygame.key.get_mods() — 检测是否有组合键被按下
- pygame.key.set_mods() — 临时设置某些组合键为被按下状态
- pygame.key.set_repeat() — 控制重复响应持续按下按键的时间
- pygame.key.get_repeat() — 获取重复响应按键的参数
- pygame.key.name() — 获取按键标识符对应的名字

该模块包含处理与键盘操作相关的函数。当键盘按键被按下和释放时，事件队列将获得 pygame.KEYDOWN 和pygame.KEYUP 事件消息。这两个消息均包含 key 属性，是一个整数的 id，代表键盘上具体的某个按键。

pygame.KYEDOWN 事件还有个额外的属性 unicode 和 scancode。unicode 代表一个按键翻译后的 Unicode 编码，这包含 shift 按键和组合键。scancode 是扫描码，不同键盘间该值可能不同。不过这对于特殊按键像多媒体键的选择是有用的。

提示：当键盘按下的时候，键盘会发送一个扫描码给系统。扫描码是键盘反馈哪一个按键被按下的方式，不同类型的键盘扫描码不同。再由系统调用相应的函数将其转换为统一的 Unicode 编码。

key 属性的值是一个数字，为了方便使用，Pygame 将这些数字定义为以下这些常量：

KeyASCII	ASCII	描述
K_BACKSPACE	\b	退格键（Backspace）
K_TAB	\t	制表键（Tab）
K_CLEAR		清楚键（Clear）
K_RETURN	\r	回车键（Enter）
K_PAUSE		暂停键（Pause）
K_ESCAPE	^[	退出键（Escape）
K_SPACE		空格键（Space）
K_EXCLAIM	!	感叹号（exclaim）
K_QUOTEDBL	"	双引号（quotedbl）
K_HASH	#	井号（hash）
K_DOLLAR	$	美元符号（dollar）
K_AMPERSAND	&	and 符号（ampersand）
K_QUOTE	'	单引号（quote）

K_LEFTPAREN	(	左小括号（left parenthesis）
K_RIGHTPAREN	)	右小括号（right parenthesis）
K_ASTERISK	*	星号（asterisk）
K_PLUS	+	加号（plus sign）
K_COMMA	,	逗号（comma）
K_MINUS	-	减号（minus sign）
K_PERIOD	.	句号（period）
K_SLASH	/	正斜杠（forward slash）
K_0	0	0
K_1	1	1
K_2	2	2
K_3	3	3
K_4	4	4
K_5	5	5
K_6	6	6
K_7	7	7
K_8	8	8
K_9	9	9

K_COLON	:	冒号（colon）
K_SEMICOLON	;	分号（semicolon）
K_LESS	<	小于号（less-than sign）
K_EQUALS	=	等于号（equals sign）
K_GREATER	>	大于号（greater-than sign）
K_QUESTION	?	问号（question mark）
K_AT	@	at 符号（at）
K_LEFTBRACKET	[	左中括号（left bracket）
K_BACKSLASH	\	反斜杠（backslash）
K_RIGHTBRACKET	]	右中括号（right bracket）
K_CARET	^	脱字符（caret）
K_UNDERSCORE	_	下划线（underscore）
K_BACKQUOTE	`	重音符（grave）
K_a	a	a
K_b	b	b
K_c	c	c
K_d	d	d
K_e	e	e
K_f	f	f
K_g	g	g
K_h	h	h

K_i	i	i
K_j	j	j
K_k	k	k
K_l	l	l
K_m	m	m
K_n	n	n
K_o	o	o
K_p	p	p
K_q	q	q
K_r	r	r
K_s	s	s
K_t	t	t

K_u	u	u
K_v	v	v
K_w	w	w
K_x	x	x
K_y	y	y
K_z	z	z
K_DELETE		删除键（delete）
K_KP0		0（小键盘）
K_KP1		1（小键盘）
K_KP2		2（小键盘）
K_KP3		3（小键盘）
K_KP4		4（小键盘）
K_KP5		5（小键盘）
K_KP6		6（小键盘）
K_KP7		7（小键盘）
K_KP8		8（小键盘）
K_KP9		9（小键盘）
K_KP_PERIOD	.	句号（小键盘）
K_KP_DIVIDE	/	除号（小键盘）
K_KP_MULTIPLY	*	乘号（小键盘）
K_KP_MINUS	-	减号（小键盘）

K_KP_PLUS	+	加号（小键盘）
K_KP_ENTER	\r	回车键（小键盘）
K_KP_EQUALS	=	等于号（小键盘）
K_UP		向上箭头（up arrow）
K_DOWN		向下箭头（down arrow）
K_RIGHT		向右箭头（right arrow）
K_LEFT		向左箭头（left arrow）
K_INSERT		插入符（insert）
K_HOME		Home 键（home）
K_END		End 键（end）
K_PAGEUP		上一页（page up）
K_PAGEDOWN		下一页（page down）
K_F1		F1
K_F2		F2
K_F3		F3
K_F4		F4
K_F5		F5
K_F6		F6
K_F7		F7
K_F8		F8
K_F9		F9

K_F10		F10
K_F11		F11
K_F12		F12
K_F13		F13
K_F14		F14
K_F15		F15
K_NUMLOCK		数字键盘锁定键（numlock）
K_CAPSLOCK		大写字母锁定键（capslock）
K_SCROLLOCK		滚动锁定键（scrollock）
K_RSHIFT		右边的 shift 键（right shift）

K_LSHIFT		左边的 shift 键（left shift）
K_RCTRL		右边的 ctrl 键（right ctrl）
K_LCTRL		左边的 ctrl 键（left ctrl）
K_RALT		右边的 alt 键（right alt）
K_LALT		左边的 alt 键（left alt）
K_RMETA		右边的元键（right meta）
K_LMETA		左边的元键（left meta）
K_LSUPER		左边的 Window 键（left windows key）
K_RSUPER		右边的 Window 键（right windows key）
K_MODE		模式转换键（mode shift）
K_HELP		帮助键（help）
K_PRINT		打印屏幕键（print screen）
K_SYSREQ		魔术键（sysrq）
K_BREAK		中断键（break）
K_MENU		菜单键（menu）
K_POWER		电源键（power）
K_EURO		欧元符号（euro）

还有一个 mod 属性，用于描述组合键状态。

以下是组合键的常量定义：

KeyASCII	描述
KMOD_NONE	木有同时按下组合键
KMOD_LSHIFT	同时按下左边的 shift 键
KMOD_RSHIFT	同时按下右边的 shift 键
KMOD_SHIFT	同时按下 shift 键
KMOD_CAPS	同时按下大写字母锁定键
KMOD_LCTRL	同时按下左边的 ctrl 键
KMOD_RCTRL	同时按下右边的 ctrl 键
KMOD_CTRL	同时按下 ctrl 键
KMOD_LALT	同时按下左边的 alt 键
KMOD_RALT	同时按下右边的 alt 键
KMOD_ALT	同时按下 alt 键
KMOD_LMETA	同时按下左边的元键
KMOD_RMETA	同时按下右边的元键
KMOD_META	同时按下元键
KMOD_NUM	同时按下数字键盘锁定键
KMOD_MODE	同时按下模式转换键

提示：如果 mod & KMOD_CTRL 是真的话，表示用户同时按下了 Ctrl 键。

函数详解

pygame.key.get_focused()

当窗口获得键盘的输入焦点时返回 True。

get_focused() -> bool

当窗口获得键盘的输入焦点时返回 True，如果窗口需要确保不失去键盘焦点，可以使用 pygame.event.set_grab(True) 独占所有

的输入接口。

提示：注意，这样做你就无法将鼠标移出窗口客户区了，但你仍然可以通过 Ctrl - Alt - Delete 热键“解围”。

pygame.key.get_pressed()

获取键盘上所有按键的状态。

get_pressed() -> bools

返回一个由布尔类型值组成的序列，表示键盘上所有按键的当前状态。使用 key 常量作为索引，如果该元素是 True，表示该按键被按下。

使用该函数获取一系列按钮被按下的状态，并不能正确的获取用户输入的文本。因为你无法知道用户按键的被按下的顺序，并且快速的连续按下键盘可能无法完全被捕获（在两次调用 pygame.key.get_pressed() 的过程中被忽略），也无法将这些按下的按键完全转化为字符值。实现此功能可以通过捕获 pygame.KEYDOWN 事件消息来实现。

pygame.key.get_mods()

检测是否有组合键被按下。

get_mods() -> int

返回一个包含所有组合键位掩码的整数。使用位操作符 & 你可以检测某个组合键是否被按下。

提示：假如 pygame.key.get_mods() 返回值存放在 mods 变量中，如果 mods & KMOD_CTRL 为 True，表示 ctrl 键正被按下。

pygame.key.set_mods()

临时设置某些组合键为被按下状态。

set_mods(int) -> None

创建一个位掩码整数，包含你需要设置为被按下状态的组合键。

小甲鱼温馨提示：比如我们需要设置 ctrl 和 alt 组合键为按下状态，则可以 mods = KMOD_CTRL | KMOD_ALT，然后调用 pygame.key.set_mods(mods)，这样尽管用户没有按下 ctrl 和 alt 组合键，它们依然是显示被按下状态。

pygame.key.set_repeat()

控制重复响应持续按下按键的时间。

set_repeat() -> None

set_repeat(delay, interval) -> None

当开启重复响应按键，那么用户持续按下某一按键，就会不断产生同一 pygame.KEYDOWN 事件。delay 参数设置多久后（单位是毫秒）开始发送第一个 pygame.KEYDOWN 事件。interval 参数设置发送两个事件之间的间隔。如果不传入任何参数，表示取消重复响应按键。

pygame.key.get_repeat()

获取重复响应按键的参数。

get_repeat() -> (delay, interval)

当开启重复响应按键，那么用户持续按下某一按键，就会不断产生同一 pygame.KEYDOWN 事件。返回值是一个二元组，第一个元素 delay 表示多久后（单位是毫秒）开始发送第一个 pygame.KEYDOWN 事件。第二个元素 interval 表示发送两个事件之间的间隔。

默认情况下重复响应按键是没有开启的。

pygame.key.name()

获取按键标识符对应的名字。

name(key) -> string

获取一个按键标识符对应的字符串描述。