HoldTap Keycodes
The HoldTap module lets keys do double duty:
tap the key to do one thing,
hold it longer than the configurable `tap_time` to do another.

HoldTap is often used with modifier keys.
For example `KC.HT(KC.ESCAPE, KC.LCTRL)` configures
a key that sends Escape when tapped and
left control when held.
It can be used with regular keys as well
like `KC.HT(KC.SPACE, KC.ENTER)` to send space on tab
and enter on hold.

Simply import HoldTap and add it to the modules list.
This lets you use `KC.HT` actions like those below.

```python
from kmk.modules.holdtap import HoldTap
holdtap = HoldTap()
# optional: set a custom tap timeout in ms
# holdtap.tap_time = 300
keyboard.modules.append(holdtap)
```

Keycodes

New Keycode	Description
`LCTL = KC.HT(KC.SOMETHING, KC.LCTRL)`	`LCTRL` if held `kc` if tapped
`LSFT = KC.HT(KC.SOMETHING, KC.LSFT)`	`LSHIFT` if held `kc` if tapped
`LALT = KC.HT(KC.SOMETHING, KC.LALT)`	`LALT` if held `kc` if tapped
`LGUI = KC.HT(KC.SOMETHING, KC.LGUI)`	`LGUI` if held `kc` if tapped
`RCTL = KC.HT(KC.SOMETHING, KC.RCTRL)`	`RCTRL` if held `kc` if tapped
`RSFT = KC.HT(KC.SOMETHING, KC.RSFT)`	`RSHIFT` if held `kc` if tapped
`RALT = KC.HT(KC.SOMETHING, KC.RALT)`	`RALT` if held `kc` if tapped
`RGUI = KC.HT(KC.SOMETHING, KC.RGUI)`	`RGUI` if held `kc` if tapped
`SGUI = KC.HT(KC.SOMETHING, KC.LSHFT(KC.LGUI))`	`LSHIFT` and `LGUI` if held `kc` if tapped
`LCA = KC.HT(KC.SOMETHING, KC.LCTRL(KC.LALT))`	`LCTRL` and `LALT` if held `kc` if tapped
`LCAG = KC.HT(KC.SOMETHING, KC.LCTRL(KC.LALT(KC.LGUI)))`	`LCTRL` and `LALT` and `LGUI` if held `kc` if tapped
`MEH = KC.HT(KC.SOMETHING, KC.LCTRL(KC.LSFT(KC.LALT)))`	`CTRL` and `LSHIFT` and `LALT` if held `kc` if tapped
`HYPR = KC.HT(KC.SOMETHING, KC.HYPR)`	`LCTRL` and `LSHIFT` and `LALT` and `LGUI` if held `kc` if tapped

Custom HoldTap Behavior
The full HoldTap signature is as follows:
```python
KC.HT(KC.TAP, KC.HOLD, prefer_hold=True, tap_interrupted=False, tap_time=None,
repeat=HoldTapRepeat.NONE)
```
* `prefer_hold`: decides which keycode the HoldTap key resolves to when another

  key is pressed before the timeout finishes. When `True` the hold keycode is
  chosen, the tap keycode when `False`.
* `tap_interrupted`: decides if the timeout will interrupt at the first other
  key press/down, or after the first other key up/release. Set to `True` for
  interrupt on release.
* `tap_time`: length of the tap timeout in milliseconds.
* `repeat`: decides how to interpret repeated presses if they happen within
  `tap_time` after a release.
  * `TAP`: repeat tap action, if previous action was a tap.
  * `HOLD`: repeat hold action, if previous action was a hold.
  * `ALL`: repeat all of the above.
  * `NONE`: no repeat action (default), everything works as expected.
  The `HoldTapRepeat` enum must be imported from `kmk.modules.holdtap`.

Each of these parameters can be set for every HoldTap key individually.