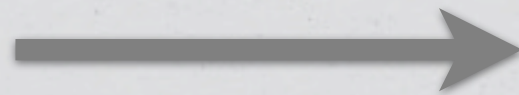# INPUT HANDLING

Finite State Machine

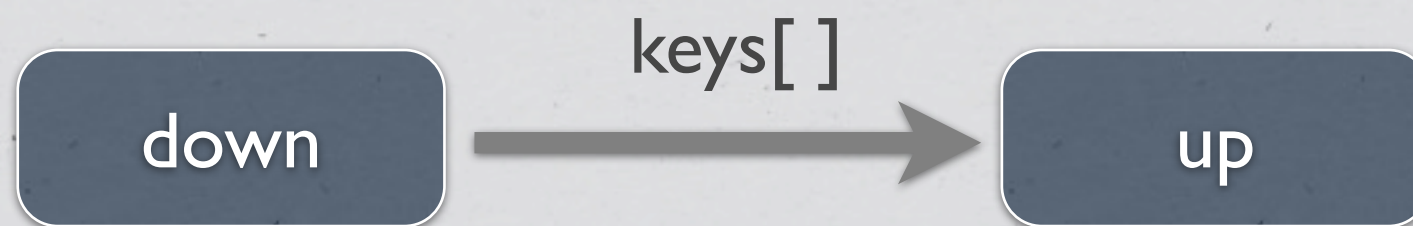# v1

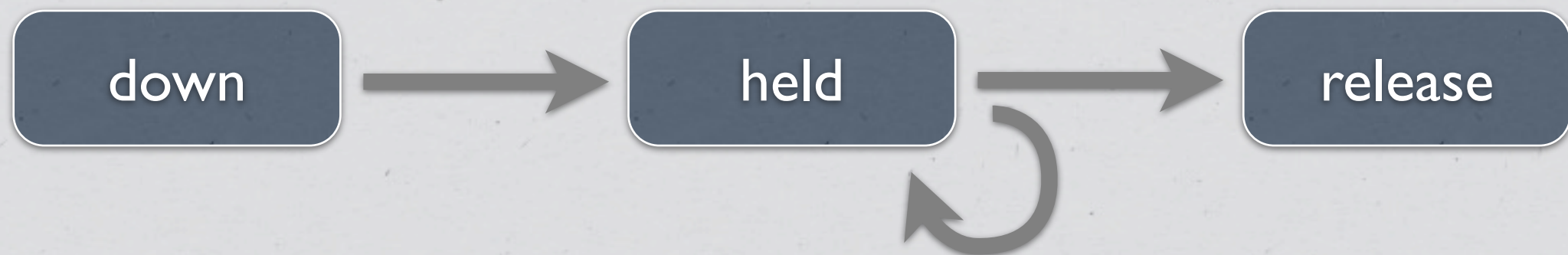press → release

v1

addEventListener(

  KeyboardEvent.KEY_DOWN, onKeyDown);

keys[ ]

down → up

addEventListener(
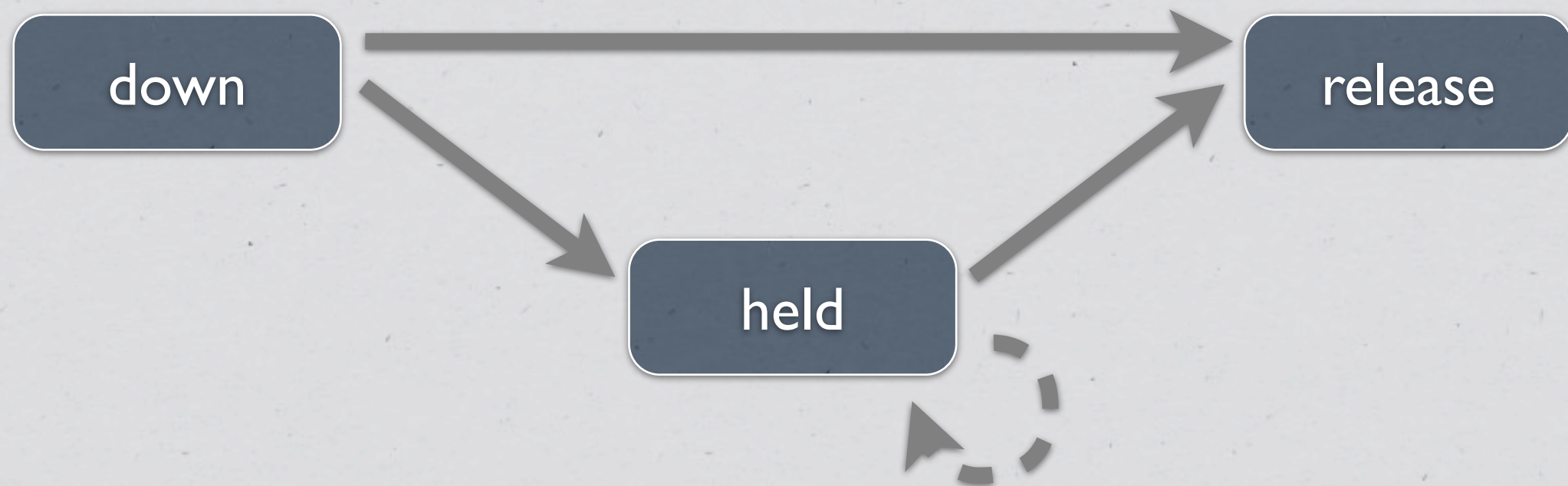
  KeyboardEvent.KEY_UP, onKeyUp);

sample with Flash: using down instead of press (up instead of release)

v2

down → held → release
(held loops back to itself)

hold will always occur at least once
multiple holds may be sent until release

v3

down → release

down → held

held → release
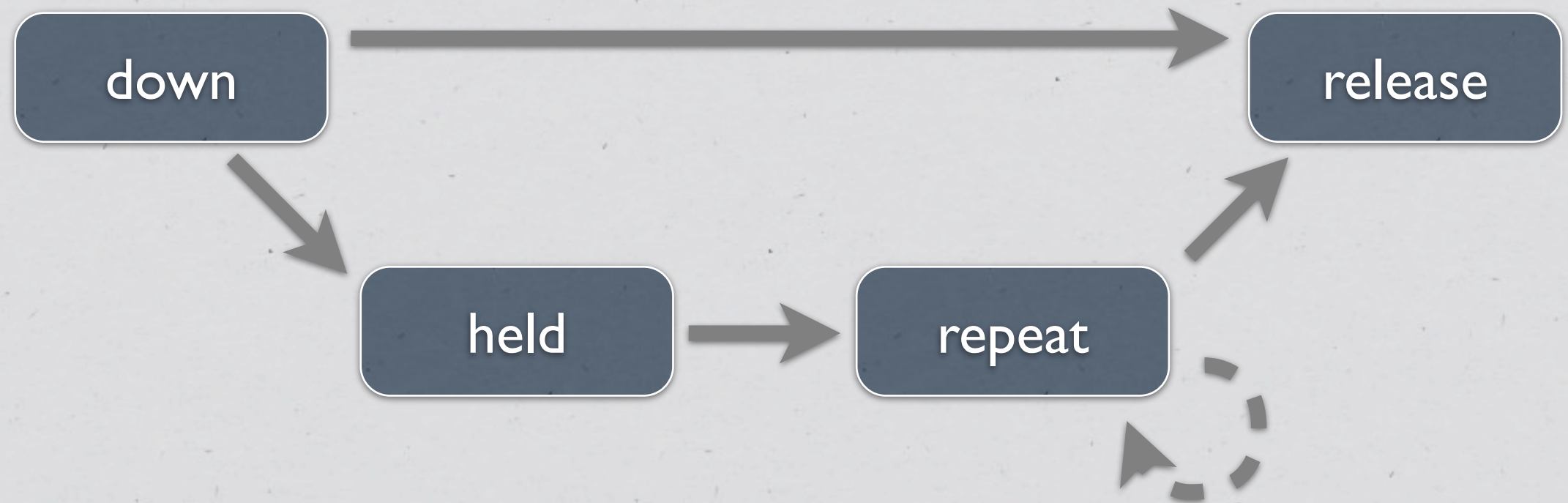
hold is initially delayed, may not be fired
by implementation: hold may fire x1 or may fire xN times (repeatedly)

v4

down → release

down → held → repeat → release
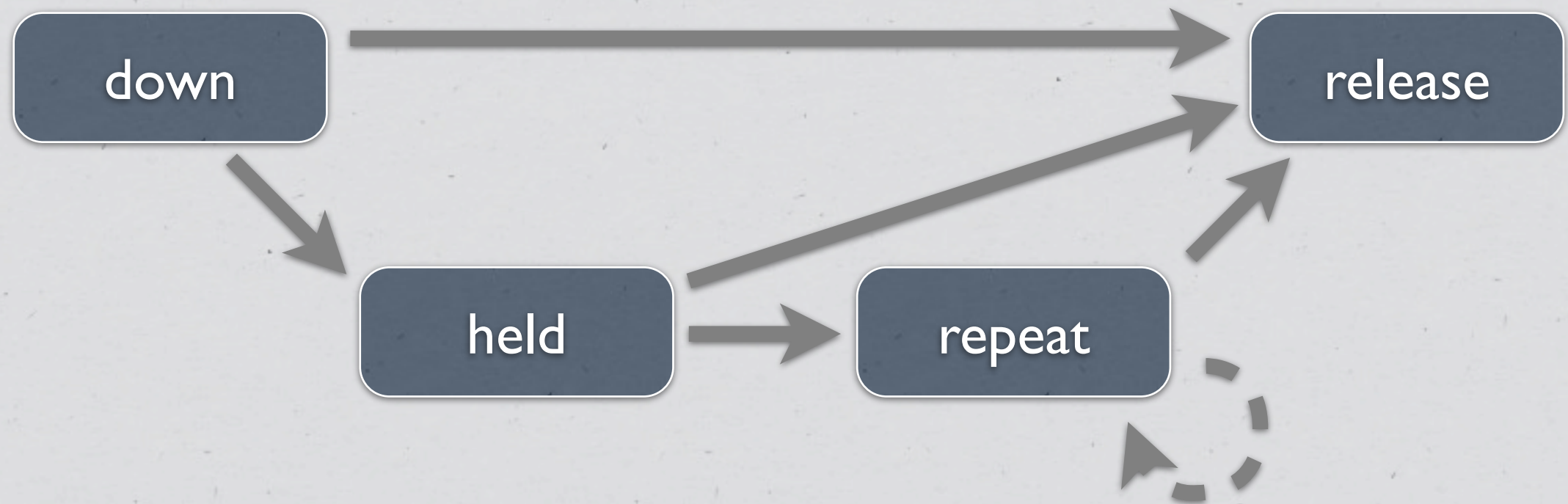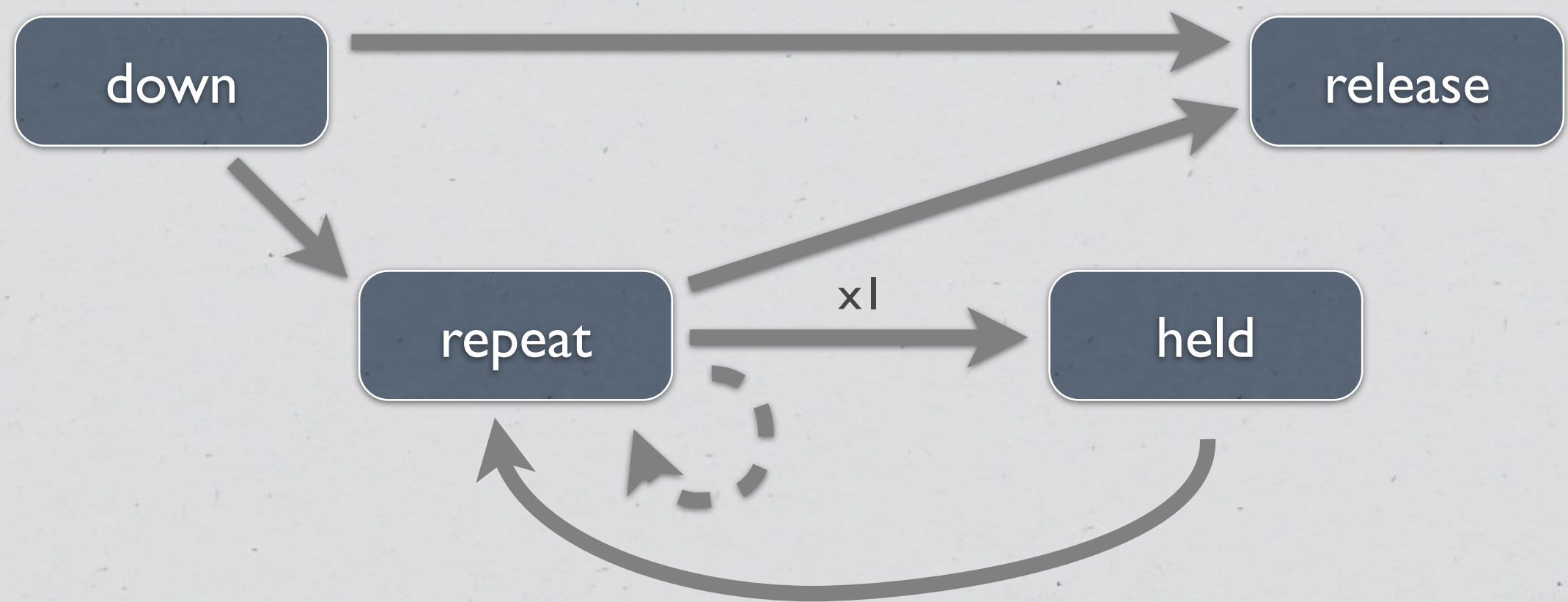
Now an input is continued to be pressed, a single hold will occur
At least x1 repeat will occur too; which may not be desired

v5

down → release
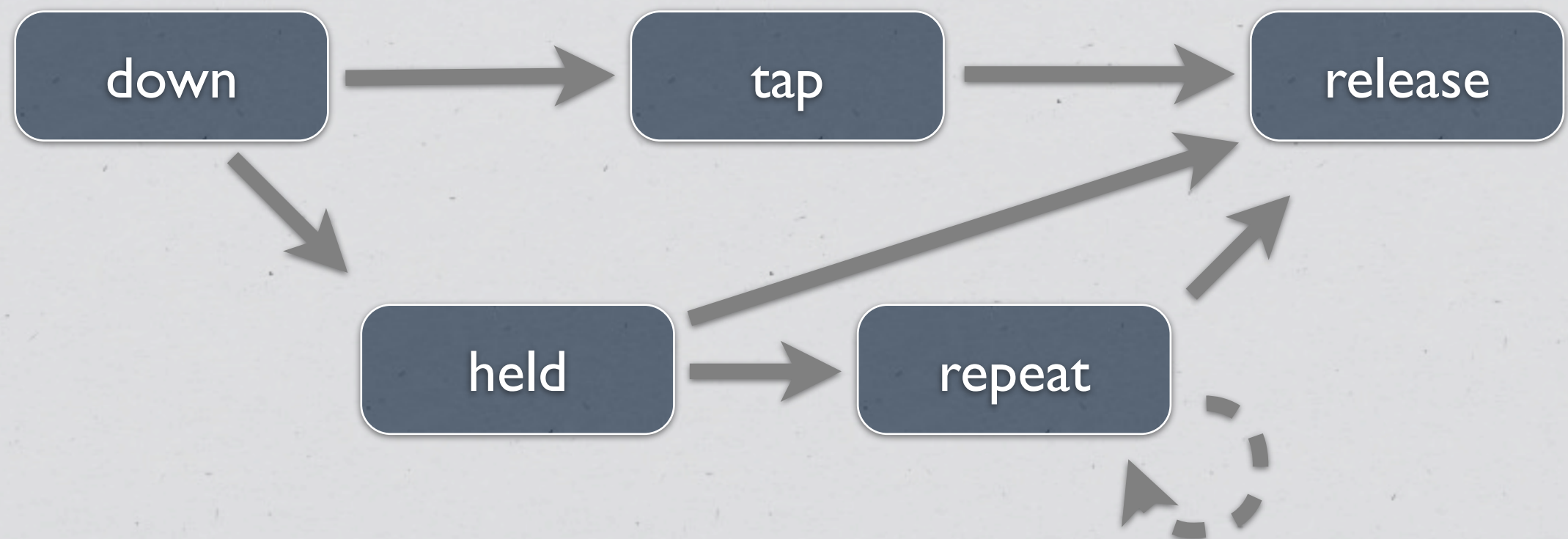
down → held

held → repeat

held → release

repeat → release (with self-loop)

Slight modification to skip any repeats from occurring if held just long enough

v5b

down → release

down → repeat

repeat →(xl)→ held

repeat → release

held → repeat

repeat ↺ (fire hold once)

Modification of previous where repeat fires before hold
Requires logic to only fire hold once

v6

down → tap → release

down → held → repeat

held → release

repeat → release

Like version 5 but a tap is explicitly sent if release occurs without a hold

v6b

Tap may be a special case that is fired only after a quick release has occurred

v7

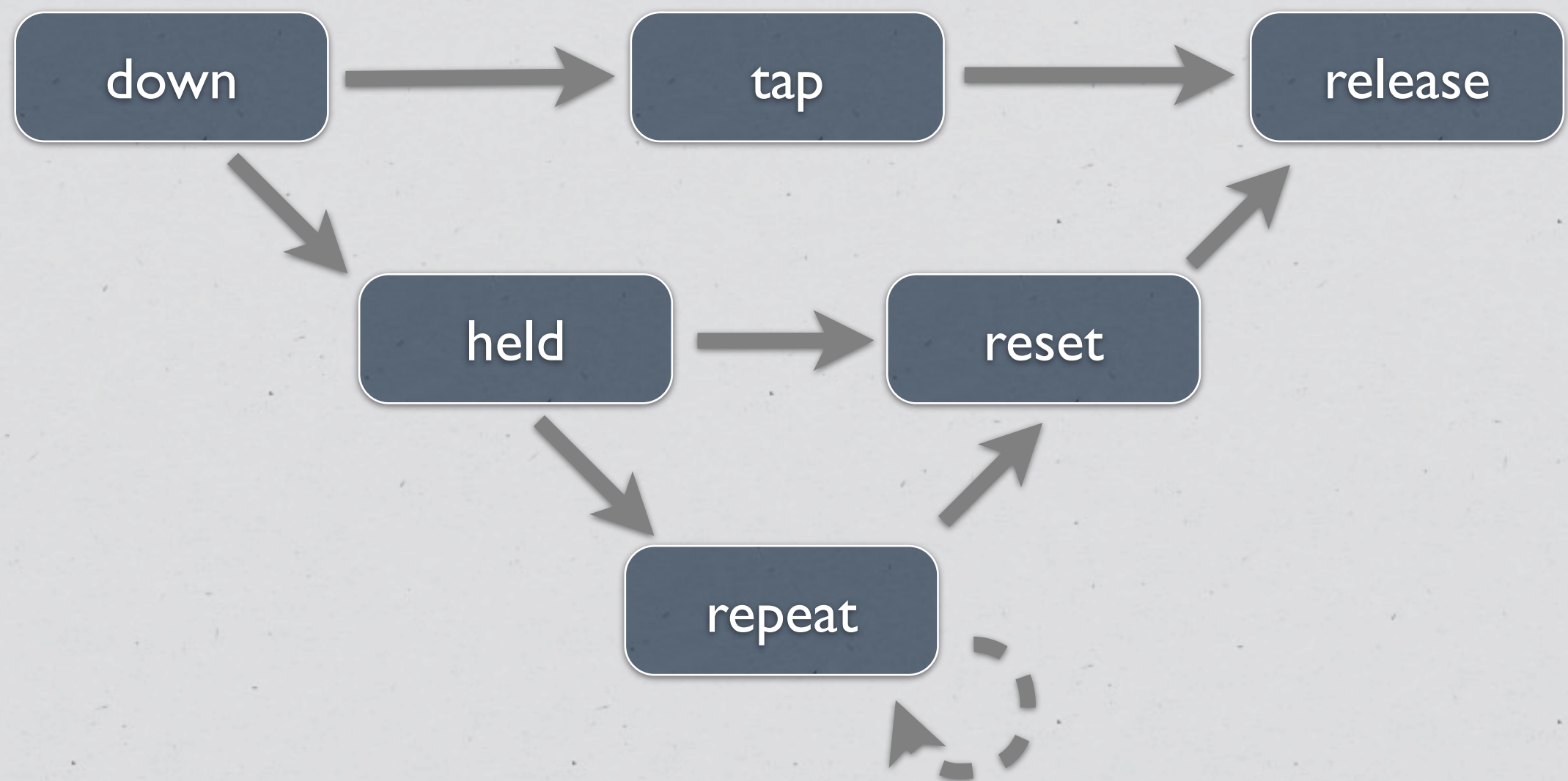reset state only fires when heading to a release from either a hold or repeat
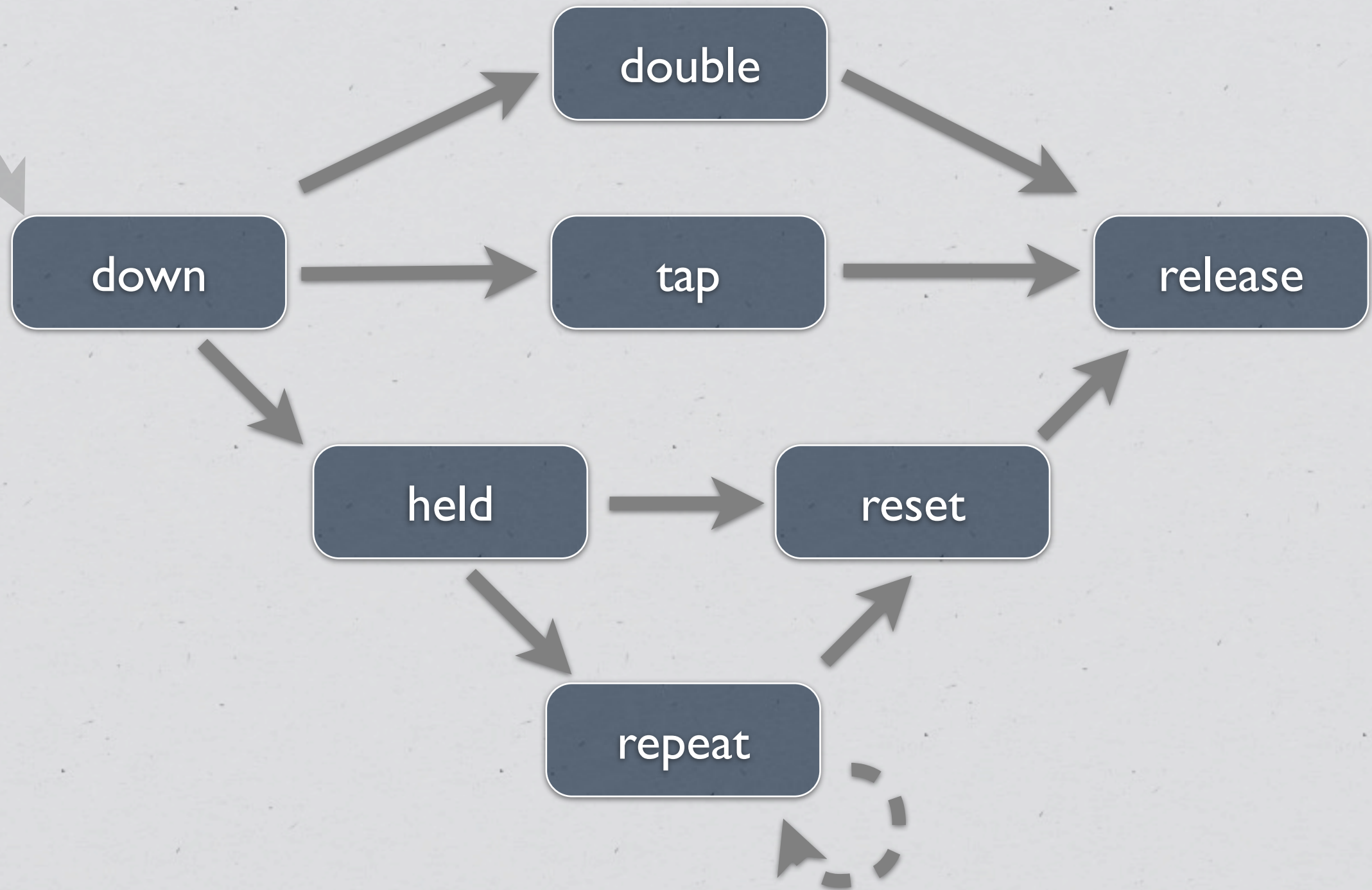
v8

double

down

tap

release

held

reset

repeat
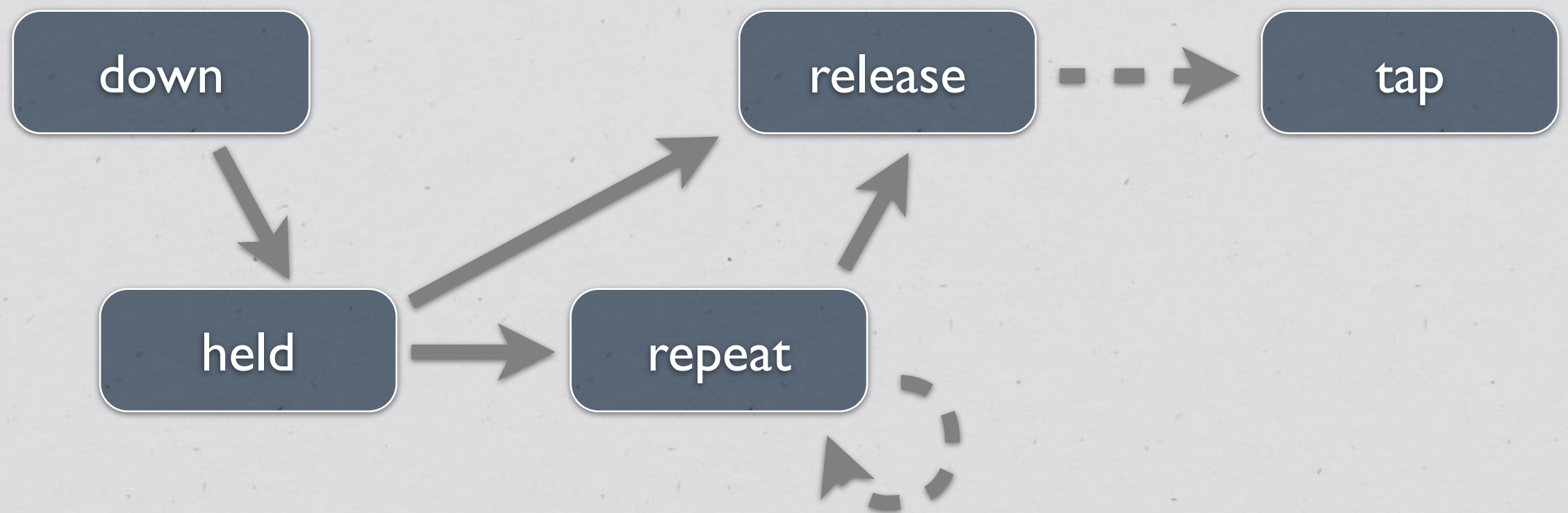
a double [tap] is sent if state is maintained across calls
and in isolated state the 2nd press/release would have created a tap

*fin