

deceit_in_detail

```

entry /
# load up on arrows and start tracking time within this tactic
archer.arrows = HorseArcher.MAXIMUM_ARROW_CAPACITY
archer.ticks = 0
archer.post_fifo(Event(signal=signals.Second, times=0, period=archer.to_time(1.0), deferred=True))

second / archer.ticks += 1

Senior_Advance_War_Cry / archer.post_fifo(Event(signal=signals.Advance_War_Cry))
Senior_Skirmish_War_Cry / archer.post_fifo(Event(signal=signals.Skirmish_War_Cry))
Senior_Retreat_War_Cry / archer.post_fifo(Event(signal=signals.Retreat_War_Cry))

Other_Ready as e / archer.dispatch_to_empathy(e)
Other_Retreat_Ready as e / archer.dispatch_to_empathy(e)

exit / archer.cancel_event(Event(signal=signals.Second))
    
```

```

HorseArcher

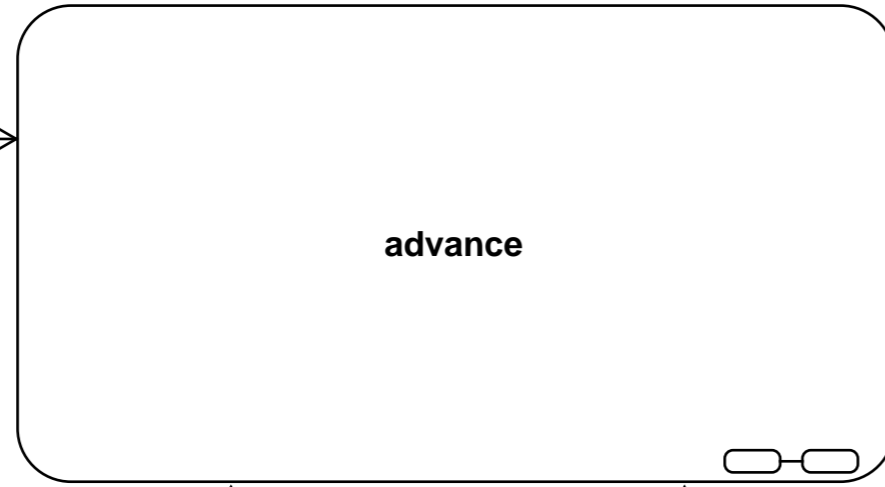
def dispatch_to_all_empathy(self, event):
    for name, other in self.others.items():
        other.dispatch(event)

def dispatch_to_empathy(self, event, other_archer_name=None):
    if other_archer_name is None:
        other_archer_name = event.payload
    if other_archer_name is not None:
        self.add_member_if_needed(other_archer_name)
        self.others[other_archer_name].dispatch(event)
    
```

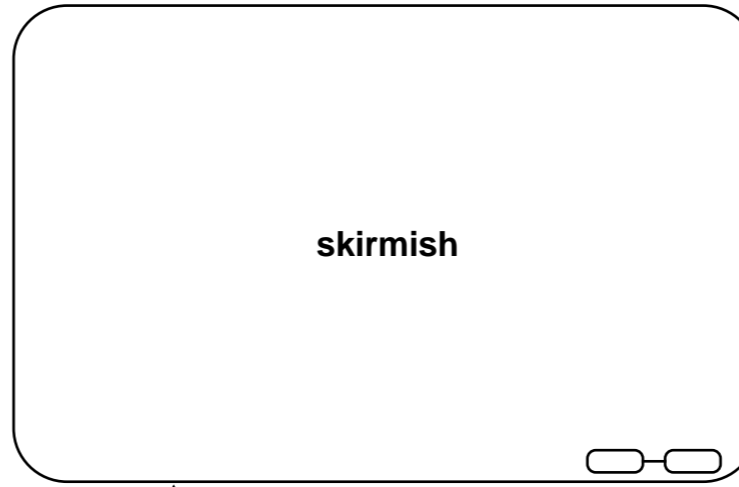
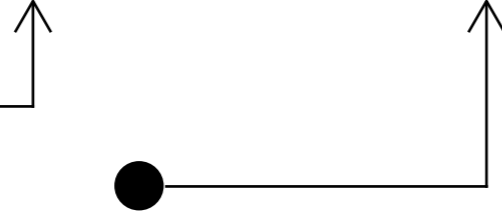
Pattern Hint
local events
ex: Advance_War_Cry

Pattern Hint
remote events
ex: Other_Advance_War_Cry

Advance_War_Cry as e /
archer.dispatch_to_all_empathy(e)

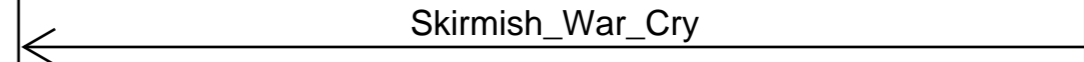


Other_Advance_War_Cry as e /
archer.post_fifo(
Event(
signal=signals.Advance_War_Cry))
archer.dispatch_to_empathy(e)

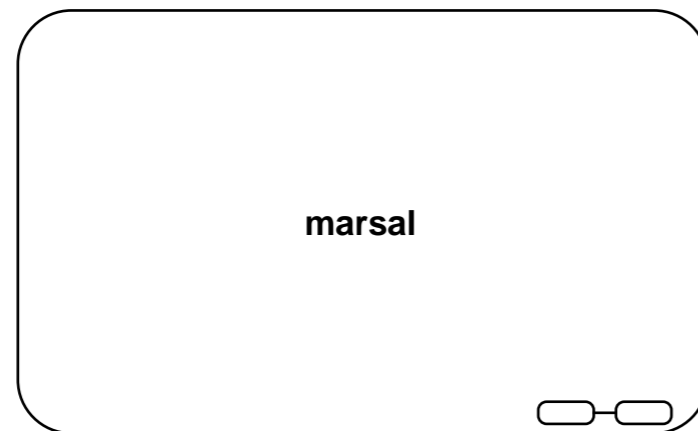


Other_Skirmish_War_Cry as e \
archer.dispatch_to_empathy(e)

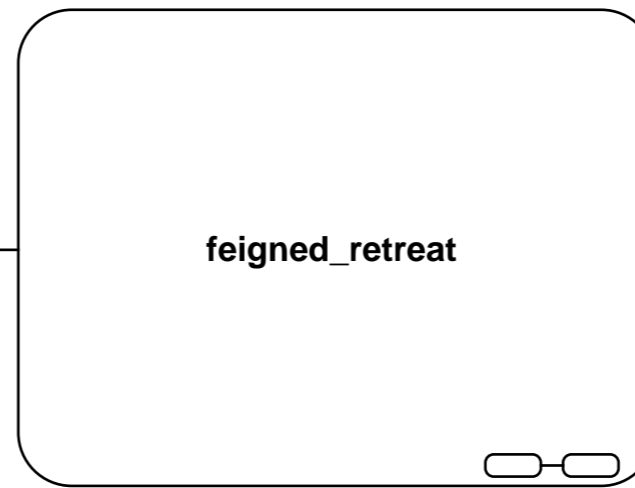
Skirmish_War_Cry



Retreat_War_Cry as e /
archer.dispatch_to_all_empathy(e)



Out_Of_Arrows



Other_Retreat_War_Cry as e /
archer.dispatch_to_empathy(e)

