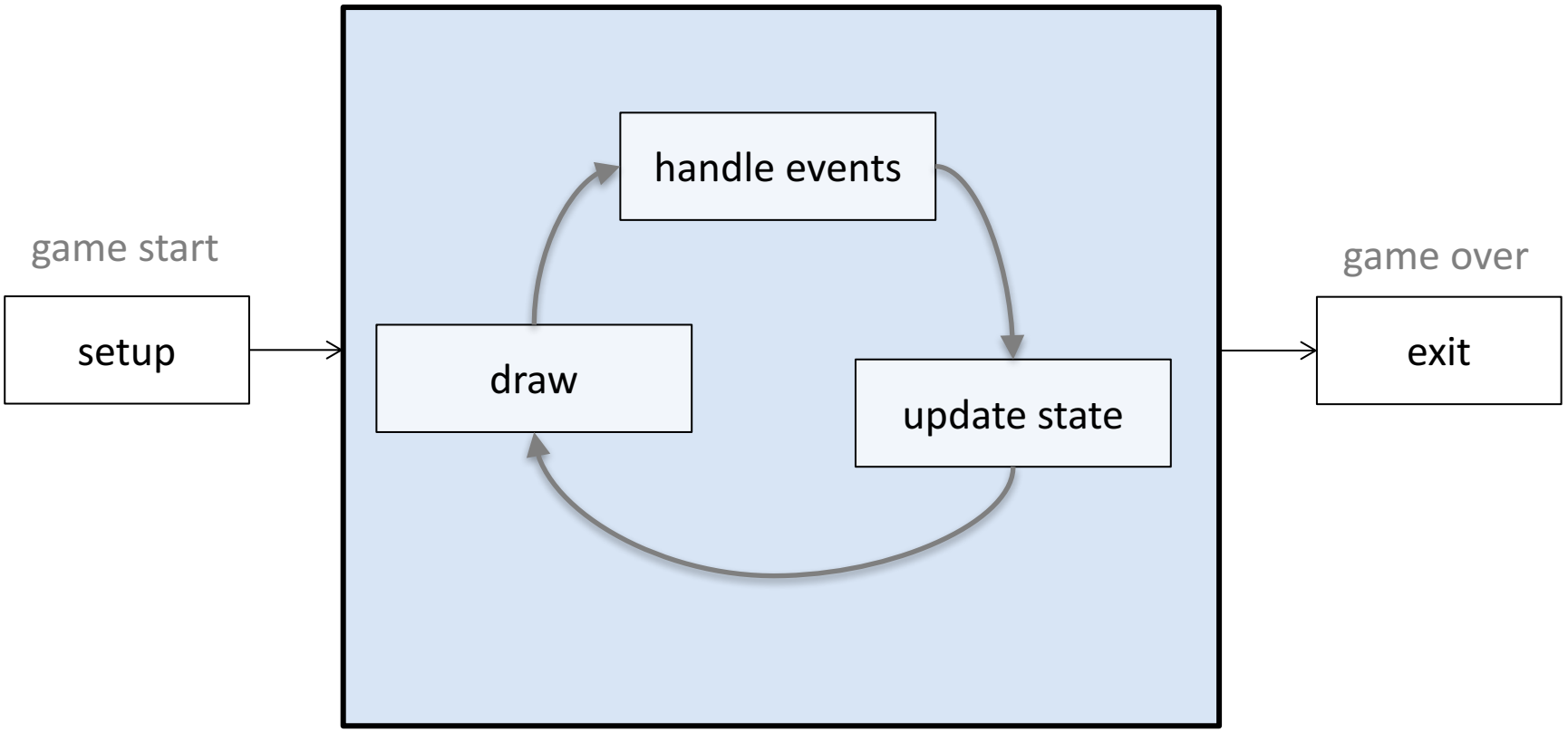
Computing Laboratory (Games) CS1830/CS1831

Nuno Barreiro
Ionuț Țuțu

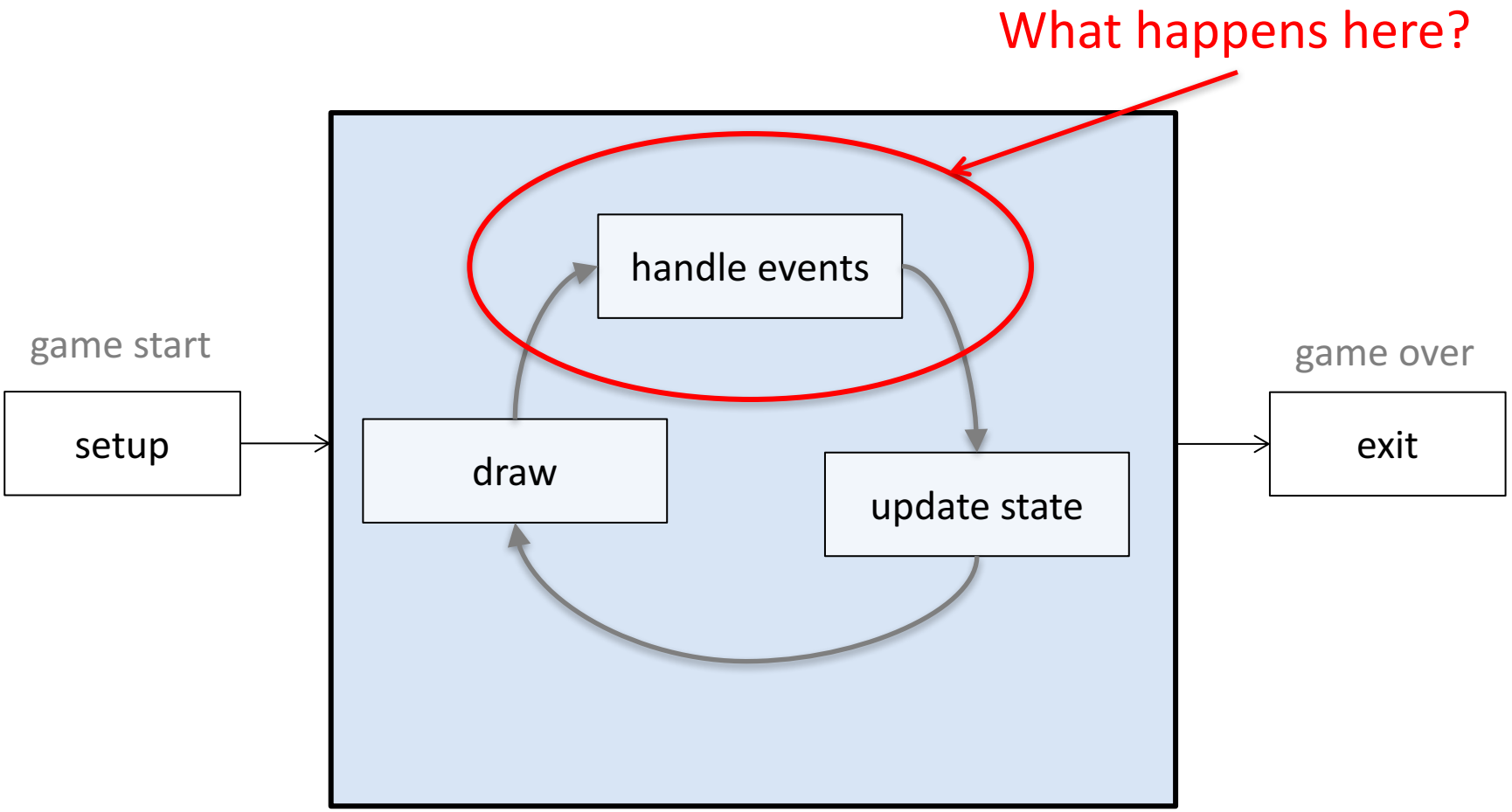
Lecture 2

Event-driven programming

Game loop


game start
setup
handle events
draw
update state
game over
exit

Handling events



`simplegui` is event-driven

- Programs written in simplegui respond to events.
- For example, when you move the mouse, or you click a button, the program reacts and does something.
- It's similar to what you have done with sensors during the robotics course.

Events in `simplegui`

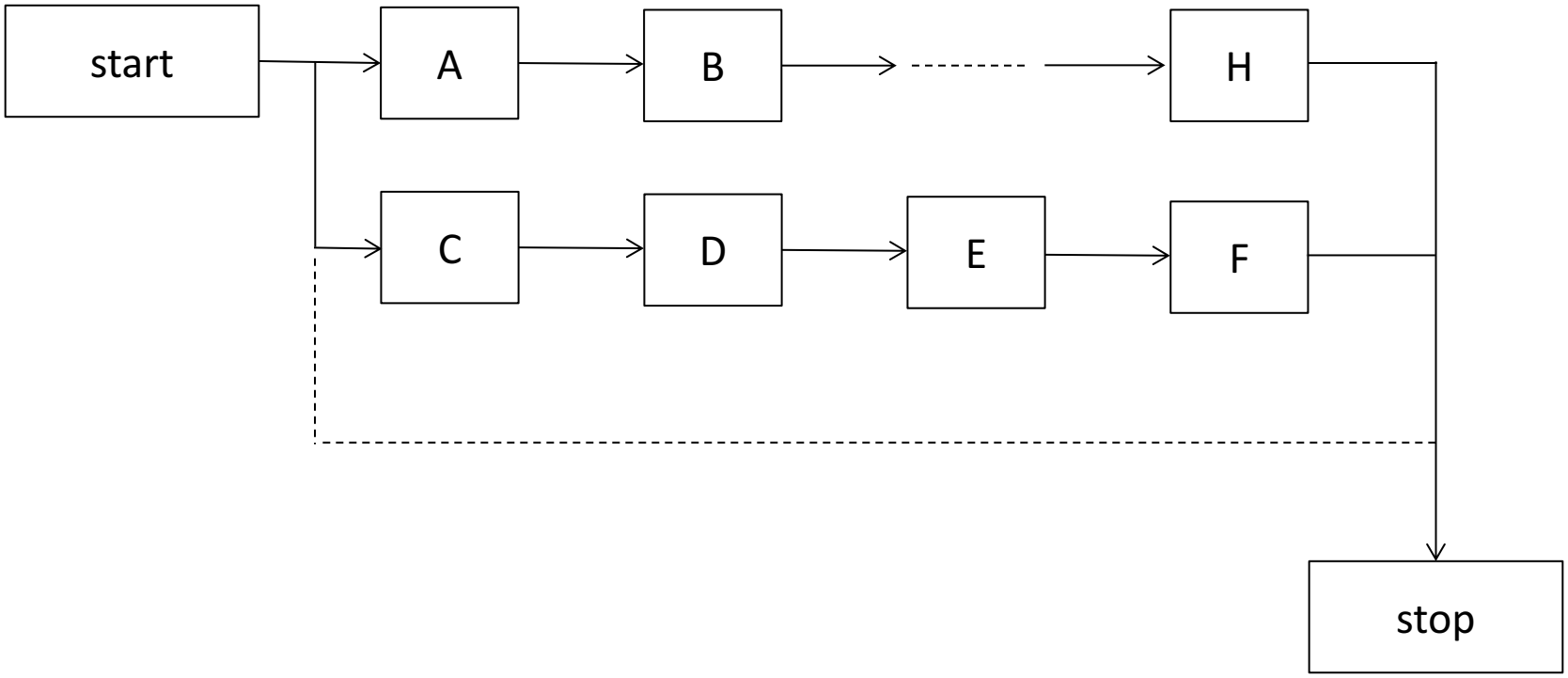
User-triggered

- Input (control panel)
  - Button
  - Text box
- Keyboard
- Mouse
  - Click (left, mouse-up)
  - Drag

System-triggered

- Timer
- Draw

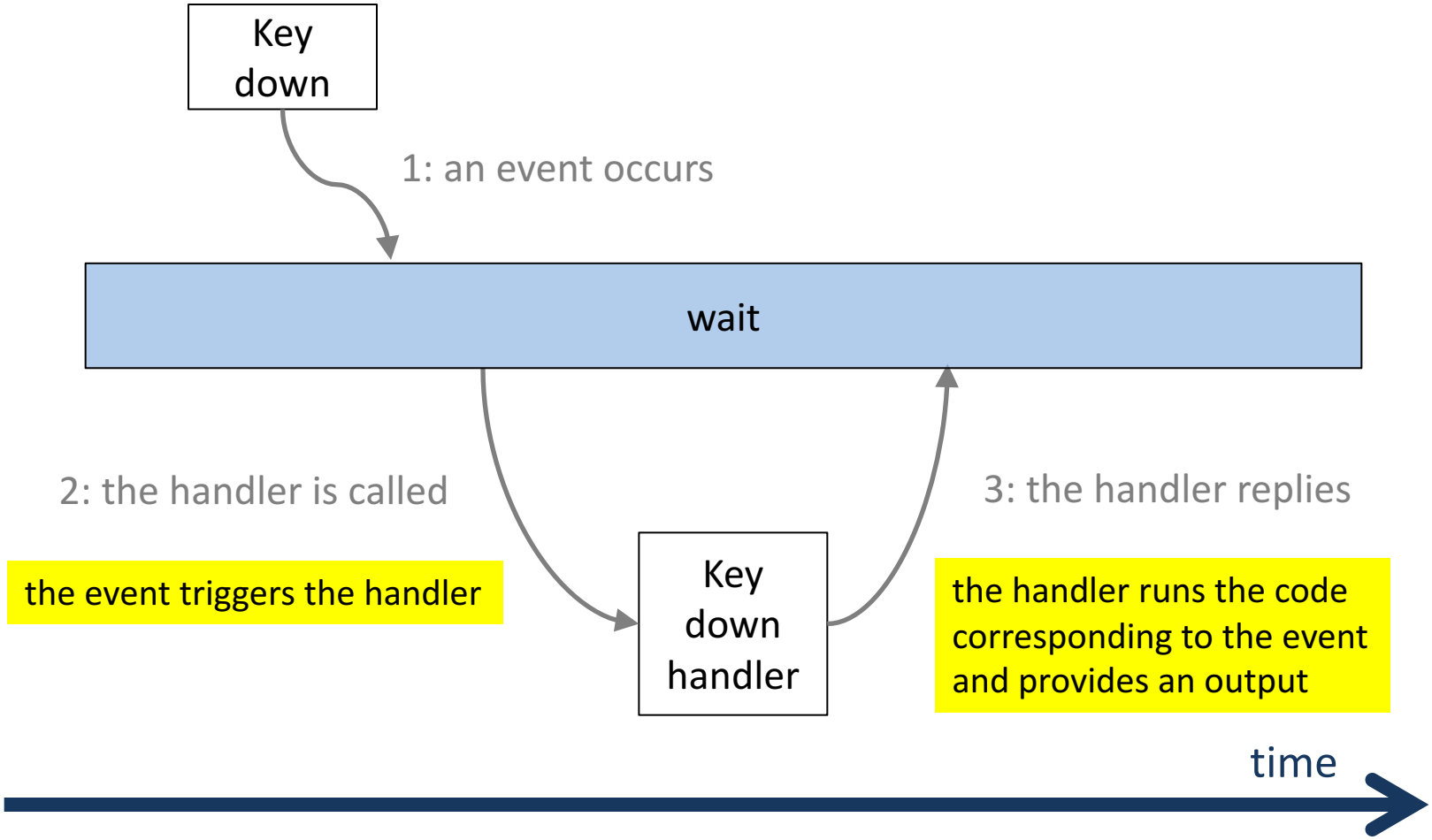
Sequential programming



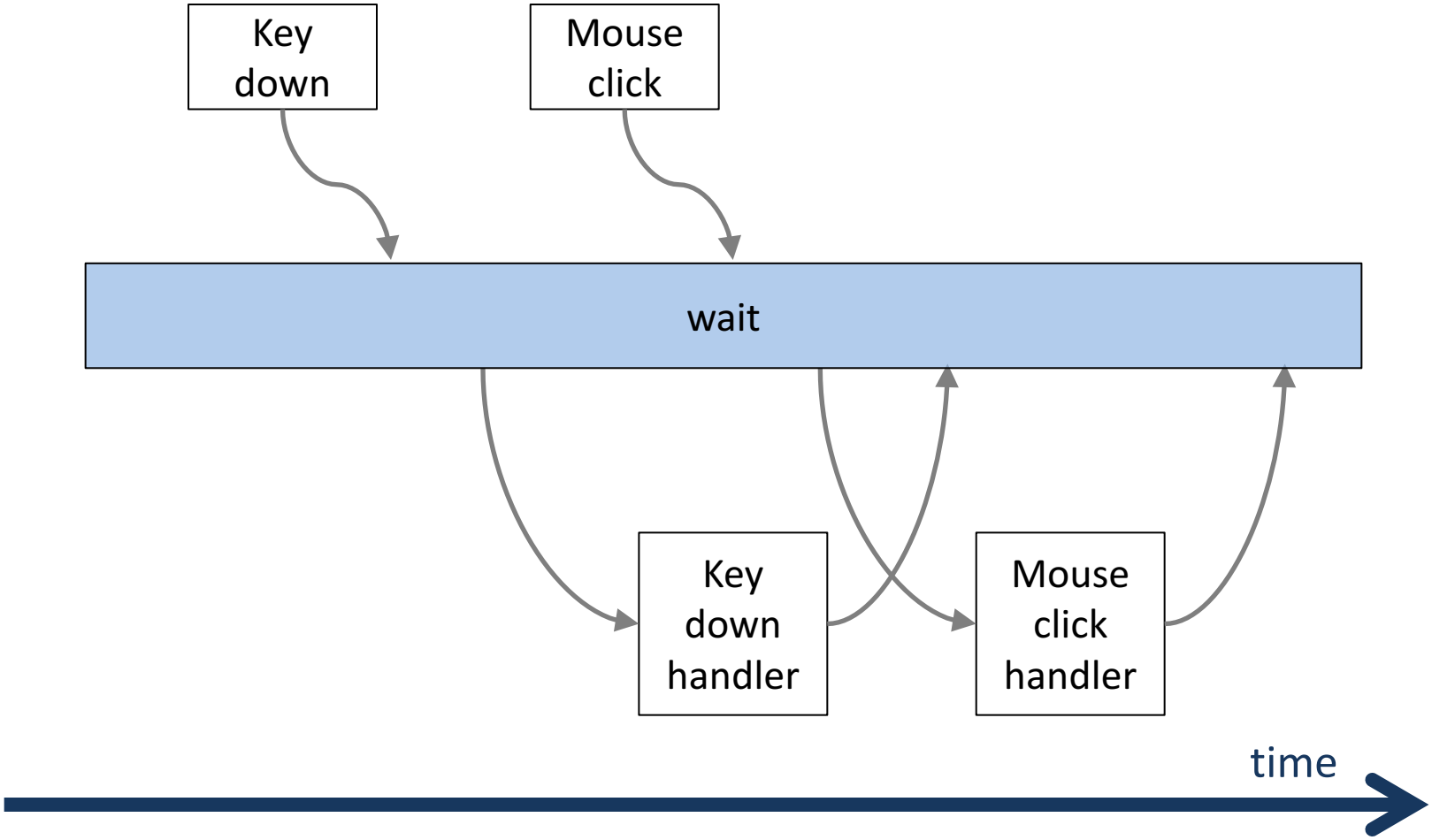
Event-driven programming

If there are no events, the program doesn't do anything!

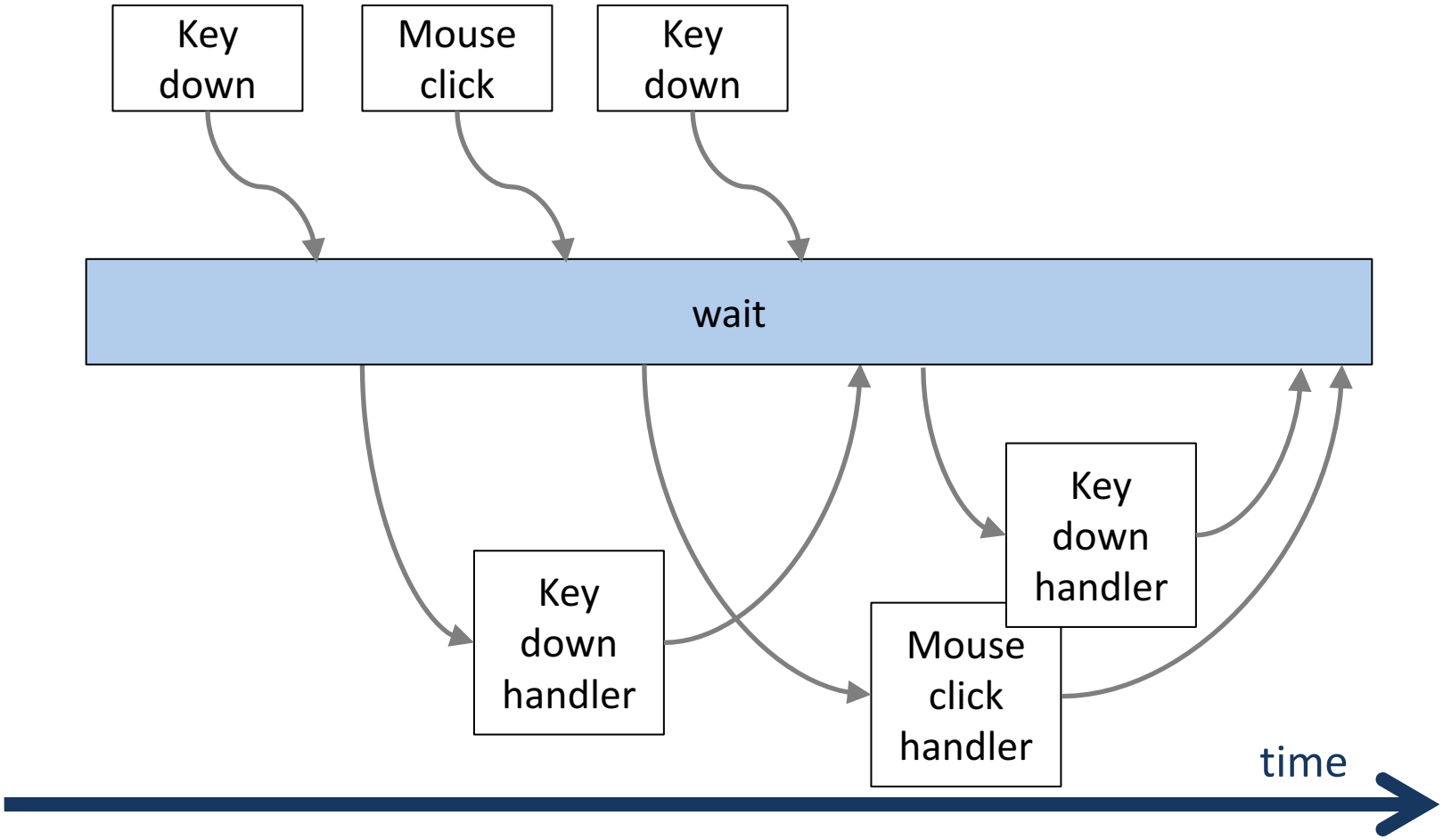
Flow of one event



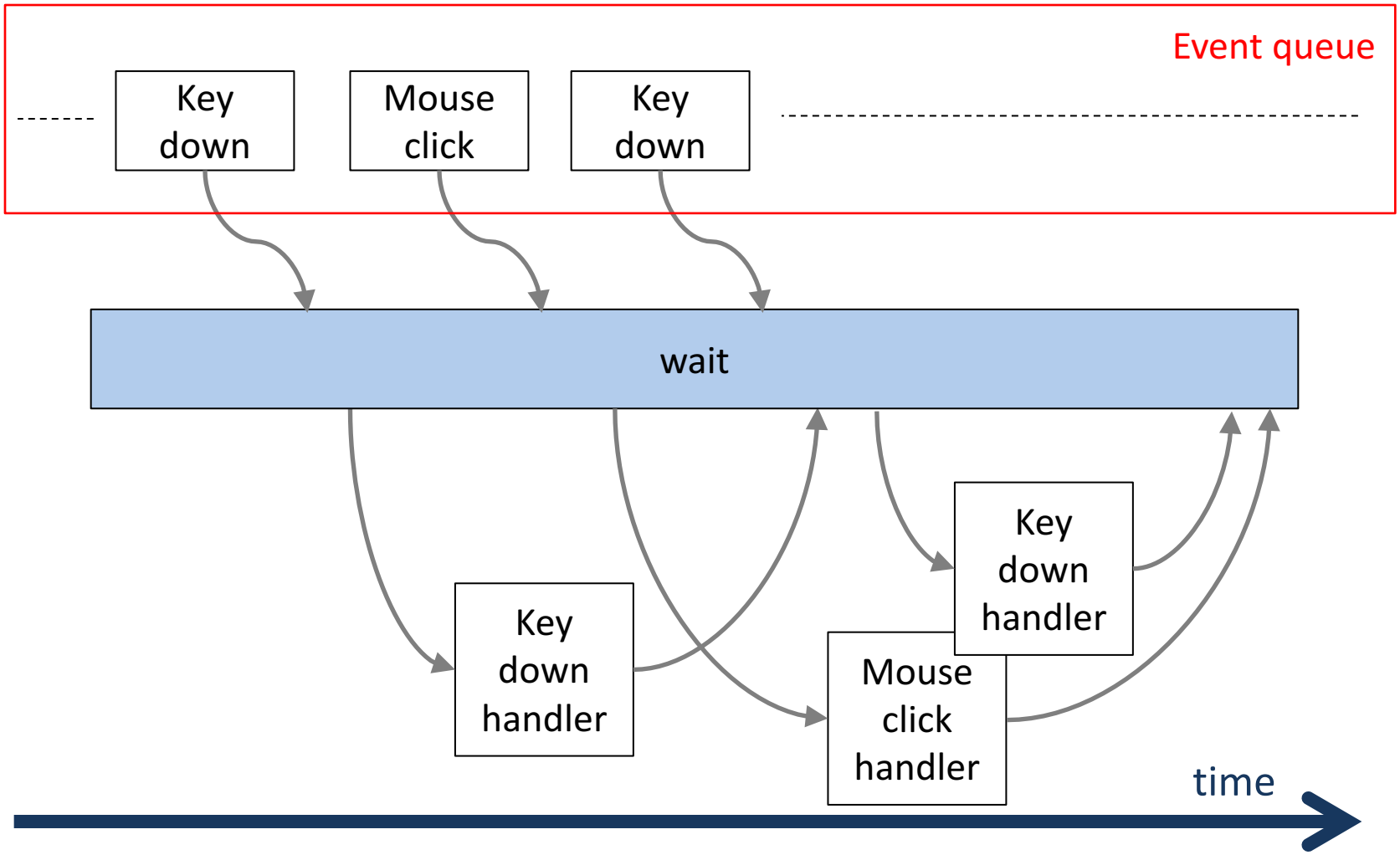
More events


Key down
Mouse click
wait
Key down handler
Mouse click handler
time

Even more events



Lots of events



System-triggered events

- Timer
- Draw

Both are triggered at a regular pace, and have priority over the others

This means, in particular, that if there is not enough time to perform all the computations between to drawing updates, the image will stutter.

Event

- Something that happens “outside” the normal control flow of our program
- Usually it happens as the consequence of some user action
- Typical events are mouse operations and key presses
- A timer can also create an event

Event-driven howto

1. Writing the *event handler*
2. *Binding* the handler to the event
3. The *handler execution* occurs each time the event is triggered

Event handler

- The function that is called in response to an event (it handles the event)
- The handler performs a sequence of actions
- The name of the handler is up to you
- The number of arguments is provided by the library for each type of event:
  - The draw handler requires the canvas as an argument
  - The mouse handler requires the position of the mouse
  - Check the documentation to know what arguments are needed

Binding the handler

- Bind a handler to an event: when the event occurs, the handler is called
- When binding, the name you gave to the event handler is relevant

Handler execution

- The handlers are called in the same order as the corresponding events
- However, during one game loop iteration *several handlers may be called*
- This gives the illusion of *simultaneity* (e.g. pressing two keyboard keys at the same time)

Example – timer

```
import simplegui

# Event handler
def myTimerHandler()
    print(“tick”)

# Register handler (binding the event)
myTimer = simplegui.create_timer(1000, myTimerHandler)

# Start timer
myTimer.start()
```

http://www.codeskulptor.org/#user39_rRvNHb4NmjLmPVD.py

Example – mouse click

```
import simplegui

# Event handler
def myClickHandler(position):
    print(position)

# Create frame (mouse events need a frame)
frame = simplegui.create_frame('Test click', 300, 300)

# Register event
frame.set_mouseclick_handler(myClickHandler)

# Start frame
frame.start()
```

http://www.codeskulptor.org/#user39_xy6aQyHrkMazHlo.py