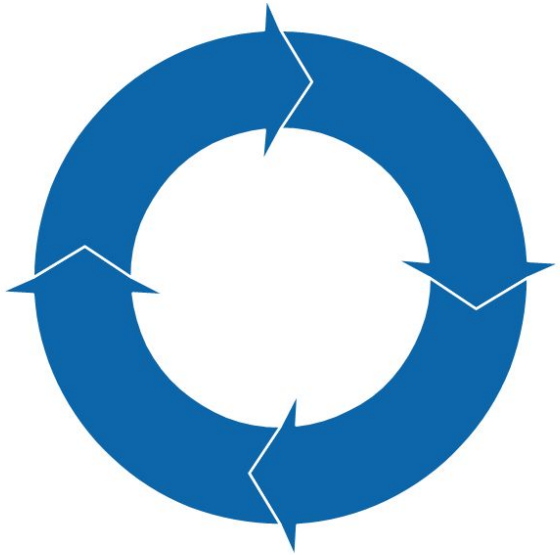
GUI Backend

Code structures and techniques behind the beauty

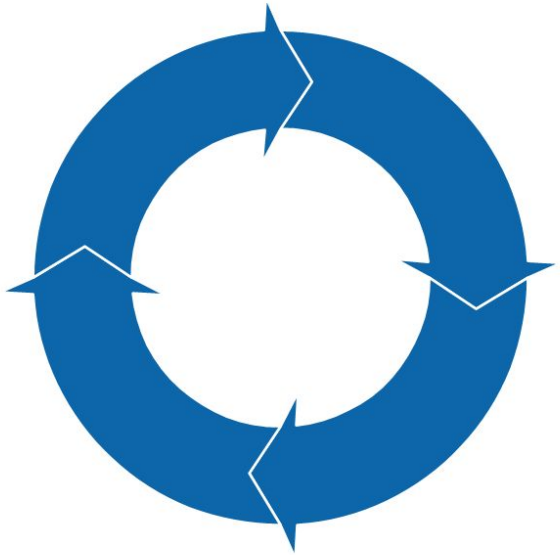
Event Systems

Around and around the event system goes...



Event Systems

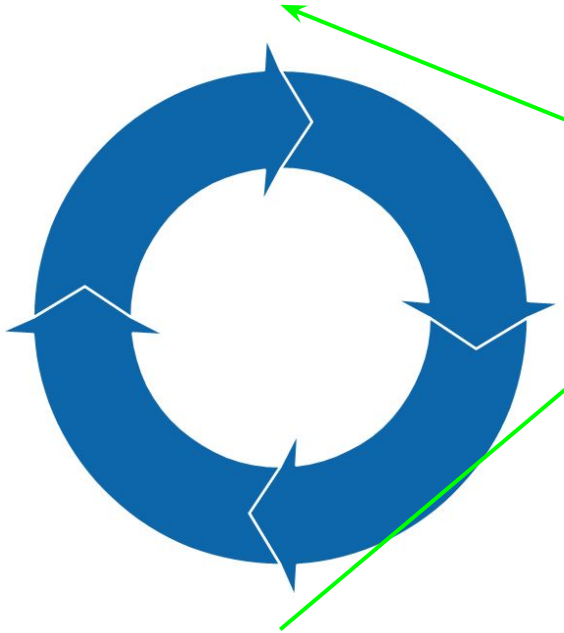
Around and around the event system goes...



Basic concept is a loop

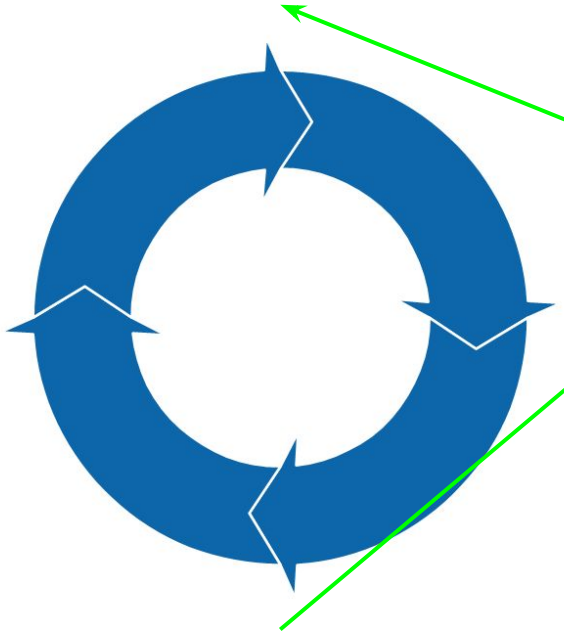
Event Systems

Usually branches from main stack.



```
Main

    initializeStuff();
    ….
    // Create Event Object
    event_object.execute()


    …
    cleanup();
    return;
```

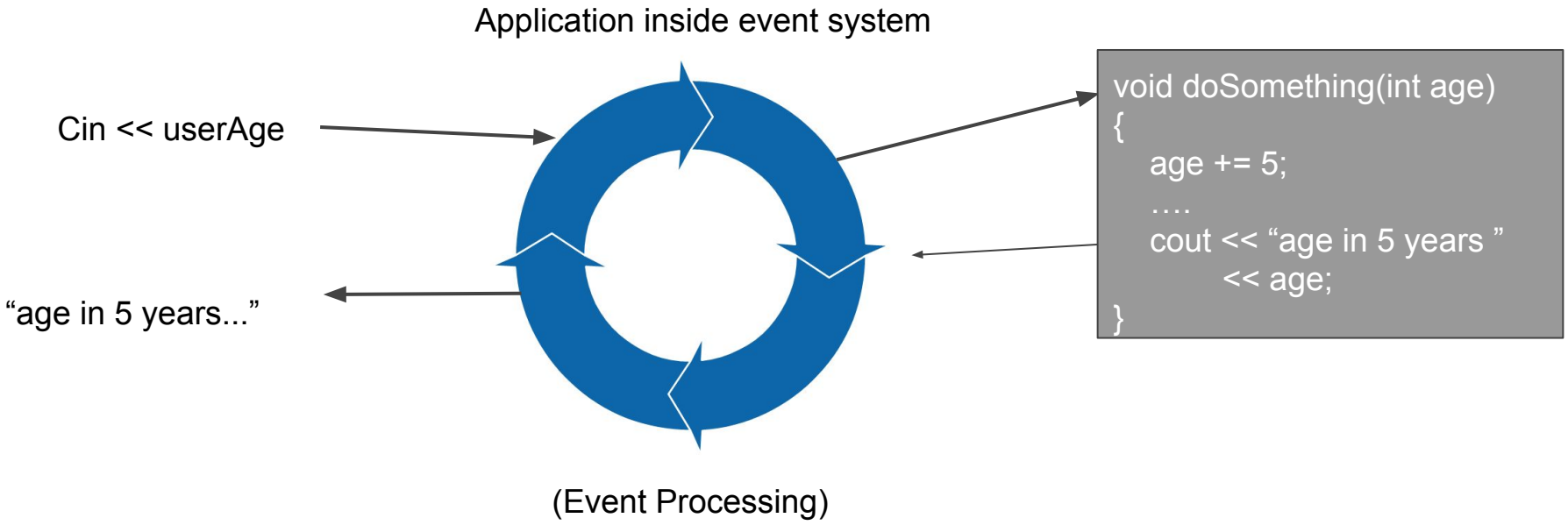
Event Systems (User defined)



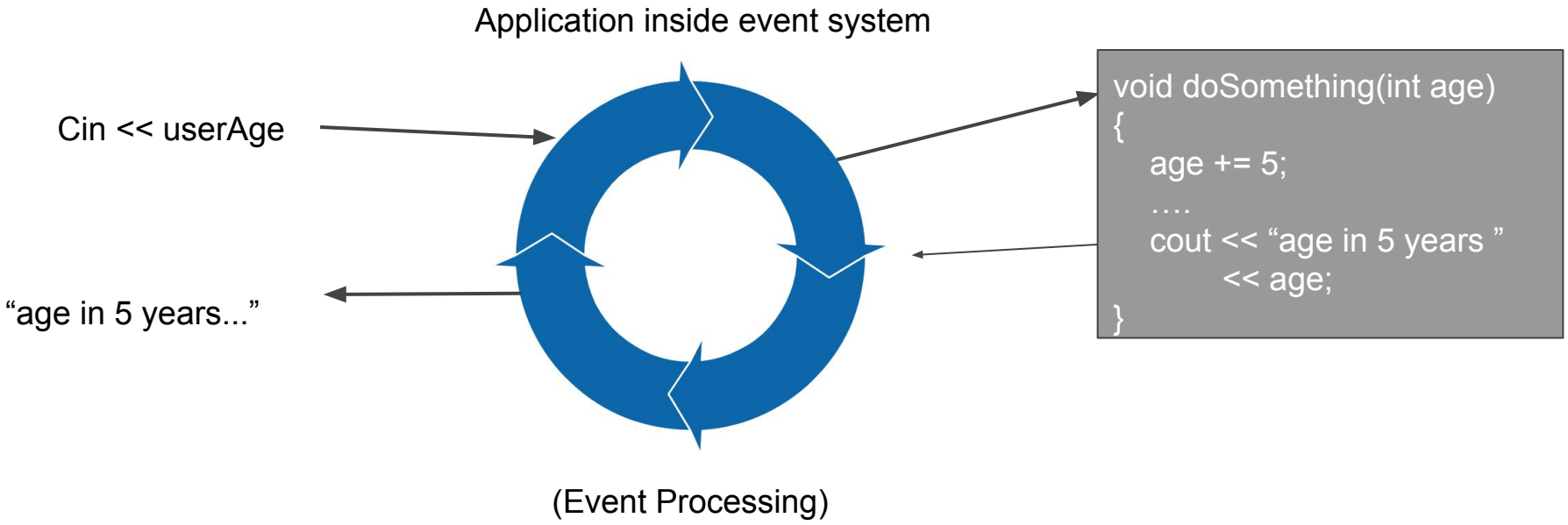
```
Main

    initializeStuff();
    ….
    // Create Event Object
    while(getting_user_input)
        ...

    …
    cleanup();
    return;
```

A fundamental concept for any application with user **I/O.**

Event Systems (Non-Extensible)



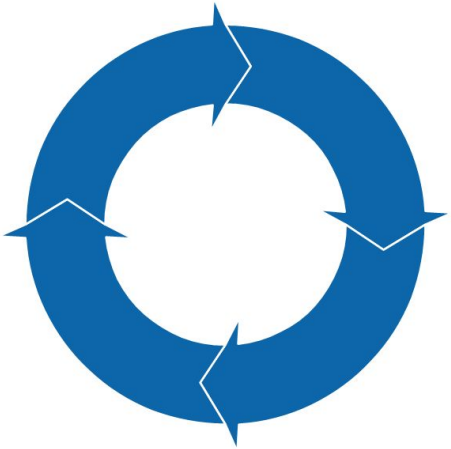
Event Systems (Non-Extensible)



Just a "run" loop wrapped around I/O processing

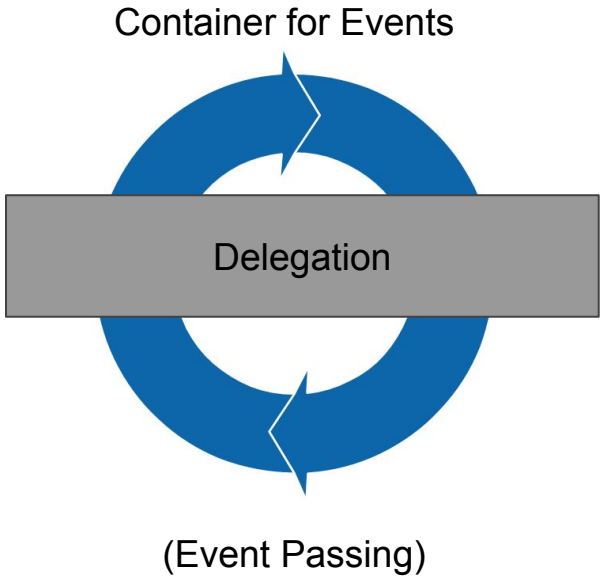
Event Systems (extensible)

Container for Events

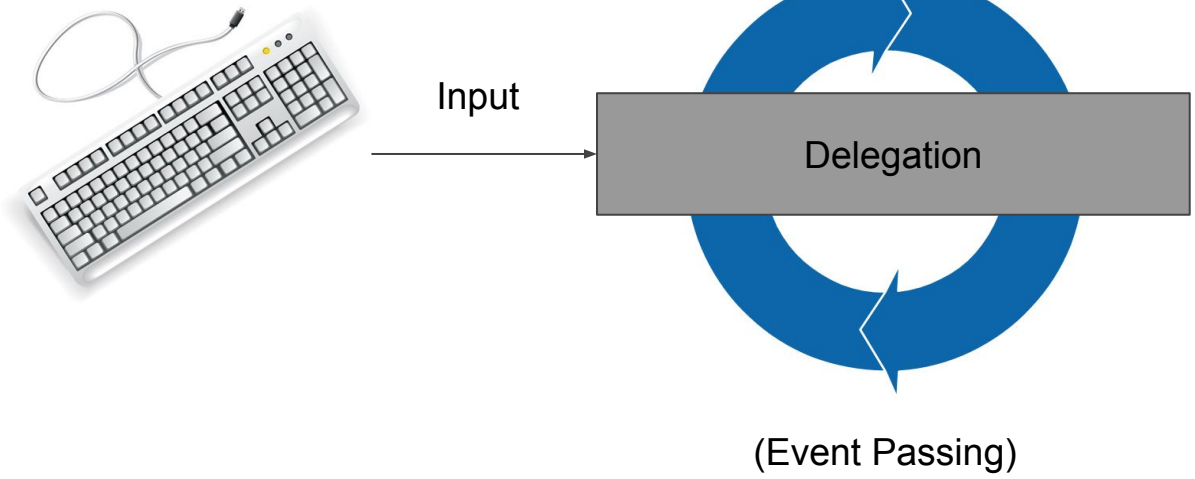


(Event Passing)

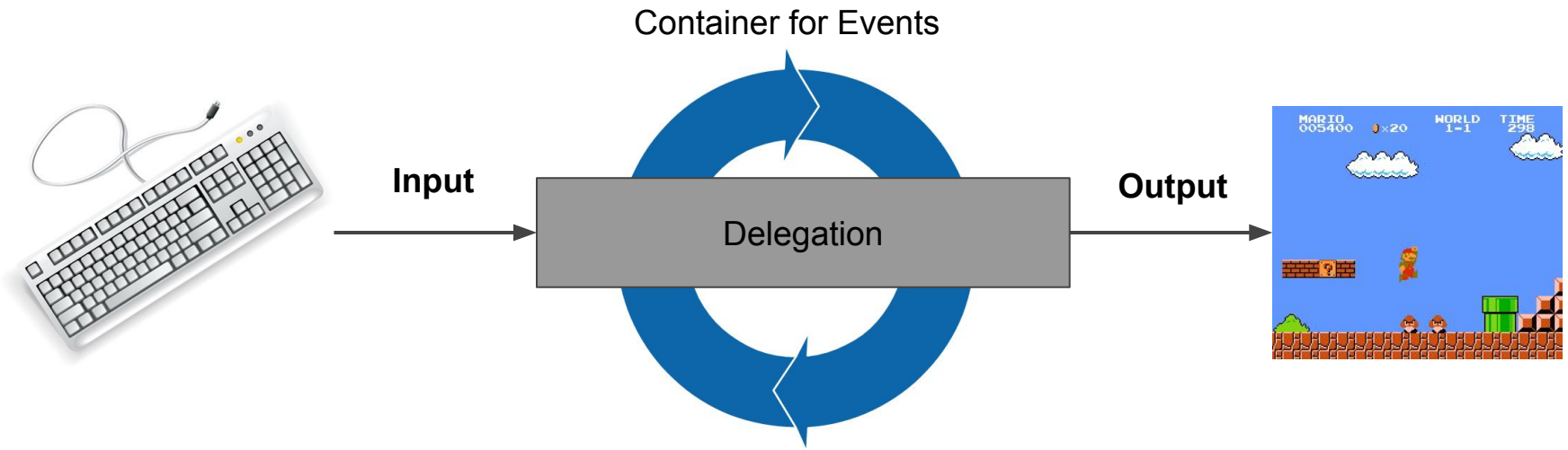
Event Systems (extensible)



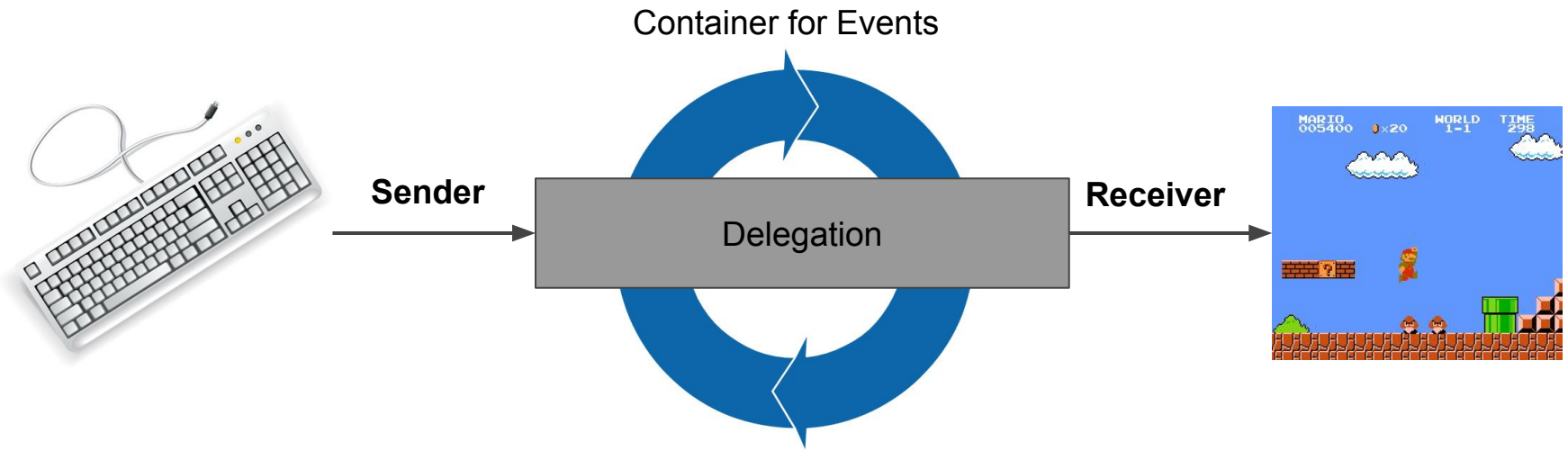
Event Systems (extensible)



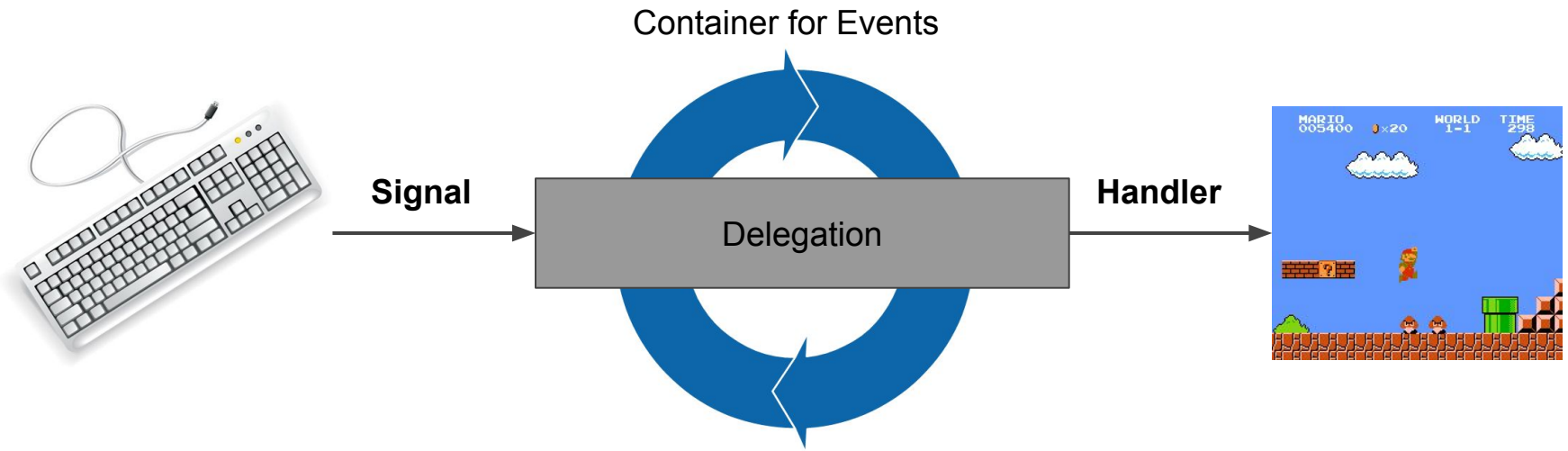
Event Systems (extensible)



Event Systems (extensible)

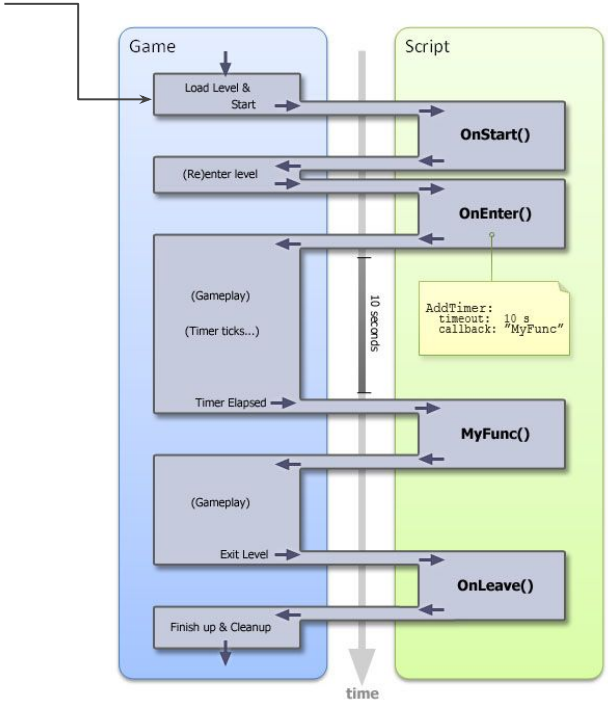


Event Systems (extensible)



Signals and Handlers (Slots)

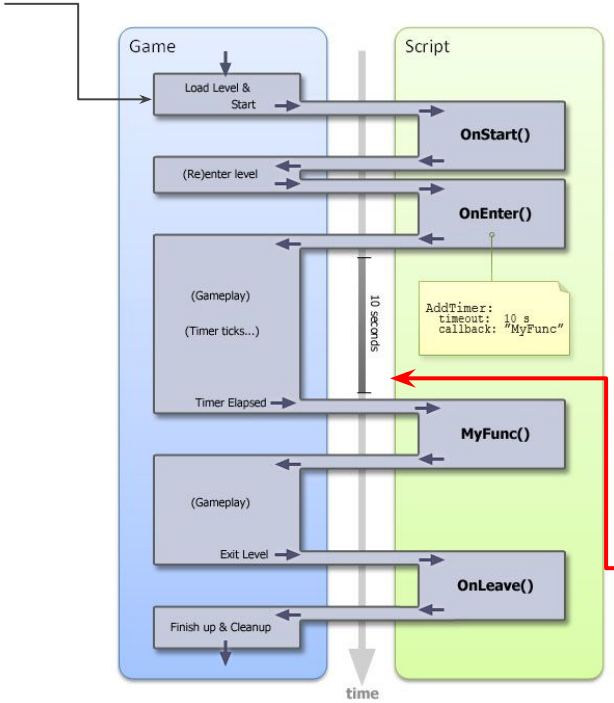
A signal happens



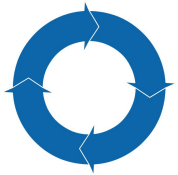
One (to many) function handlers will be called

Signals and Handlers (Slots)

A signal happens



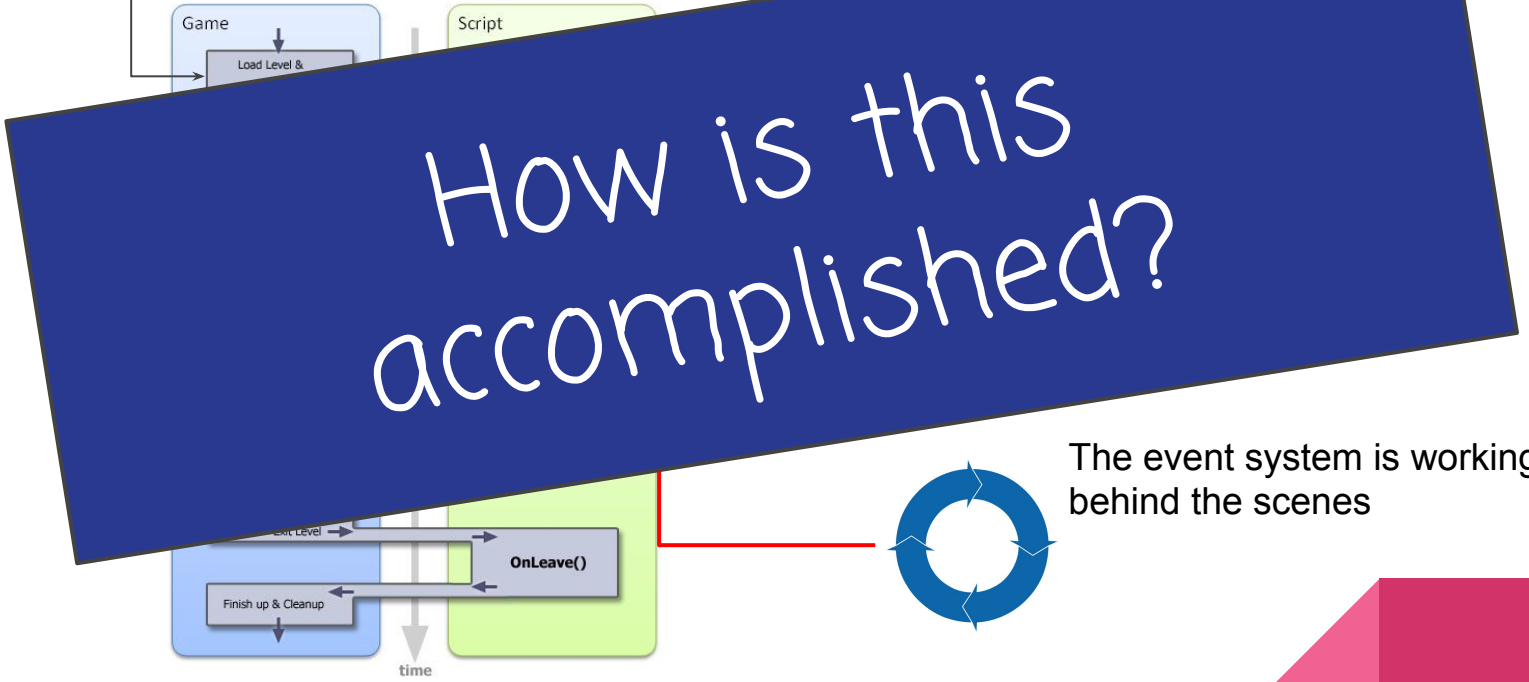
One (to many) function handlers will be called



The event system is working behind the scenes

Signals and Handlers (Slots)

A signal happens

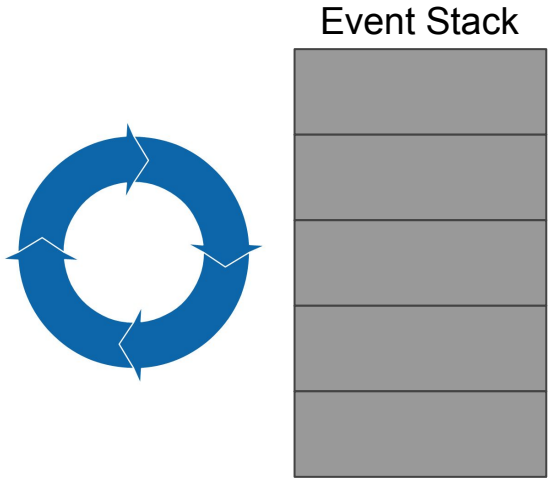


How is this accomplished?

The event system is working behind the scenes

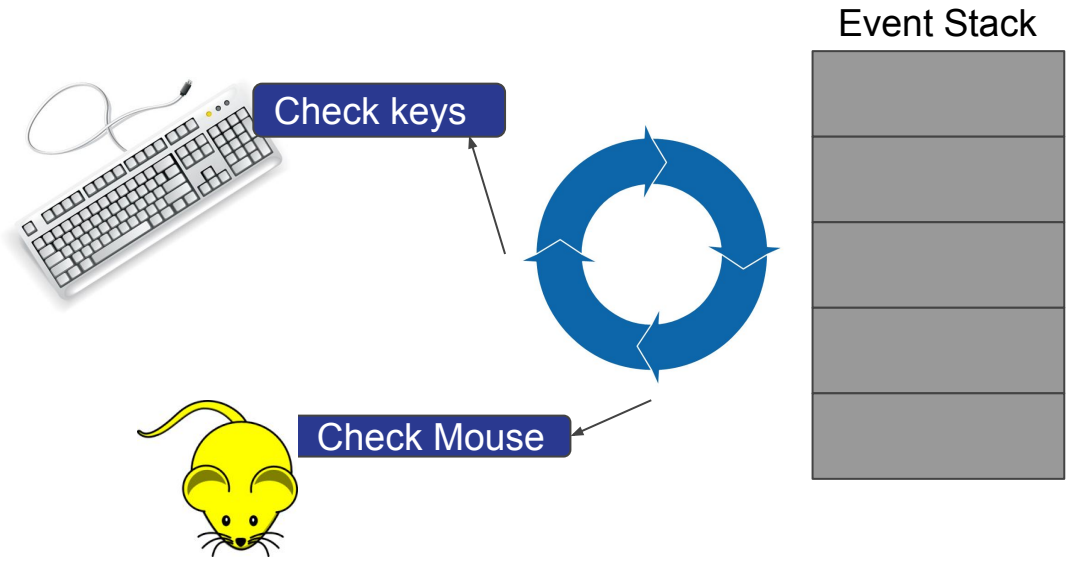
Event Queueing

While the event system runs...



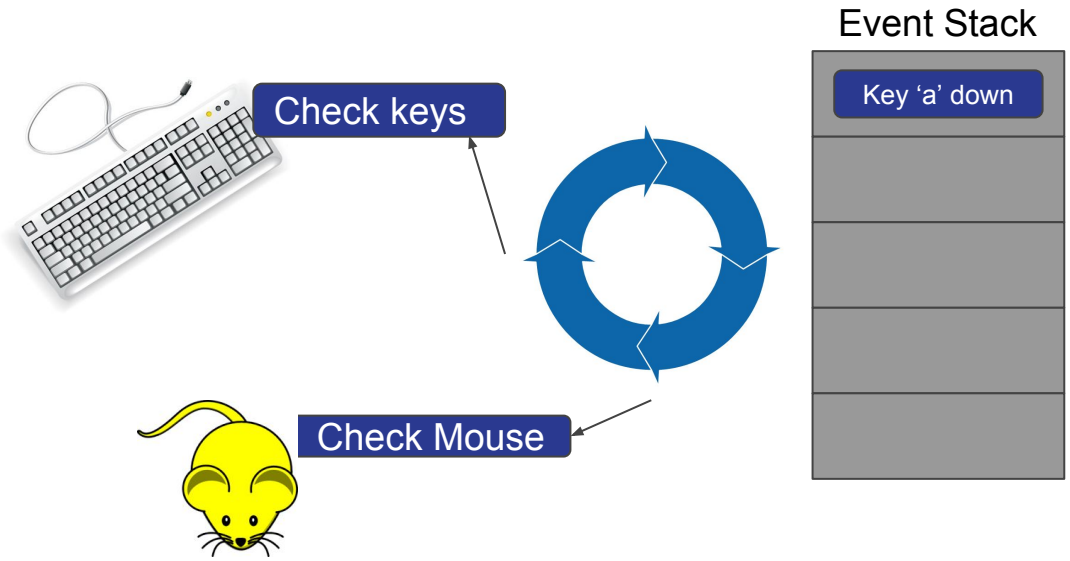
Event Queueing (Input Phase)

I/O processing functions monitor Input



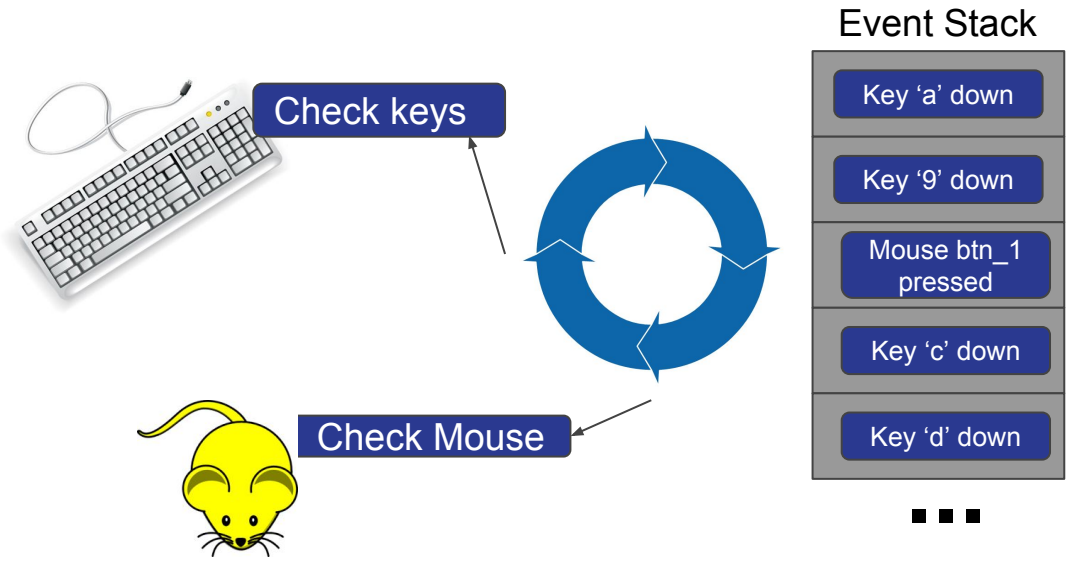
Event Queueing (Input Phase)

When input occurs the Event System creates an event on the stack



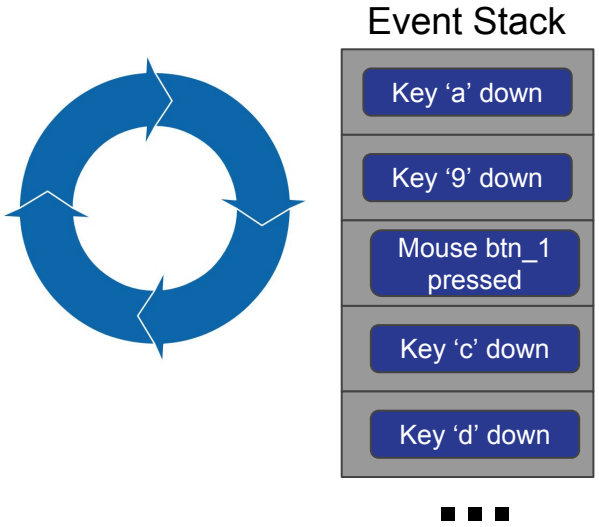
Event Queueing (Input Phase)

Many events can be queued on the stack



Event Queueing (Delegation Phase)

Once current input data is queued, the event system begins **delegating**



To Delegate:

- entrust (a task or responsibility) to another person, typically one who is less senior than oneself

https://www.google.com/?ion=1&espv=2#q=definition+delegate

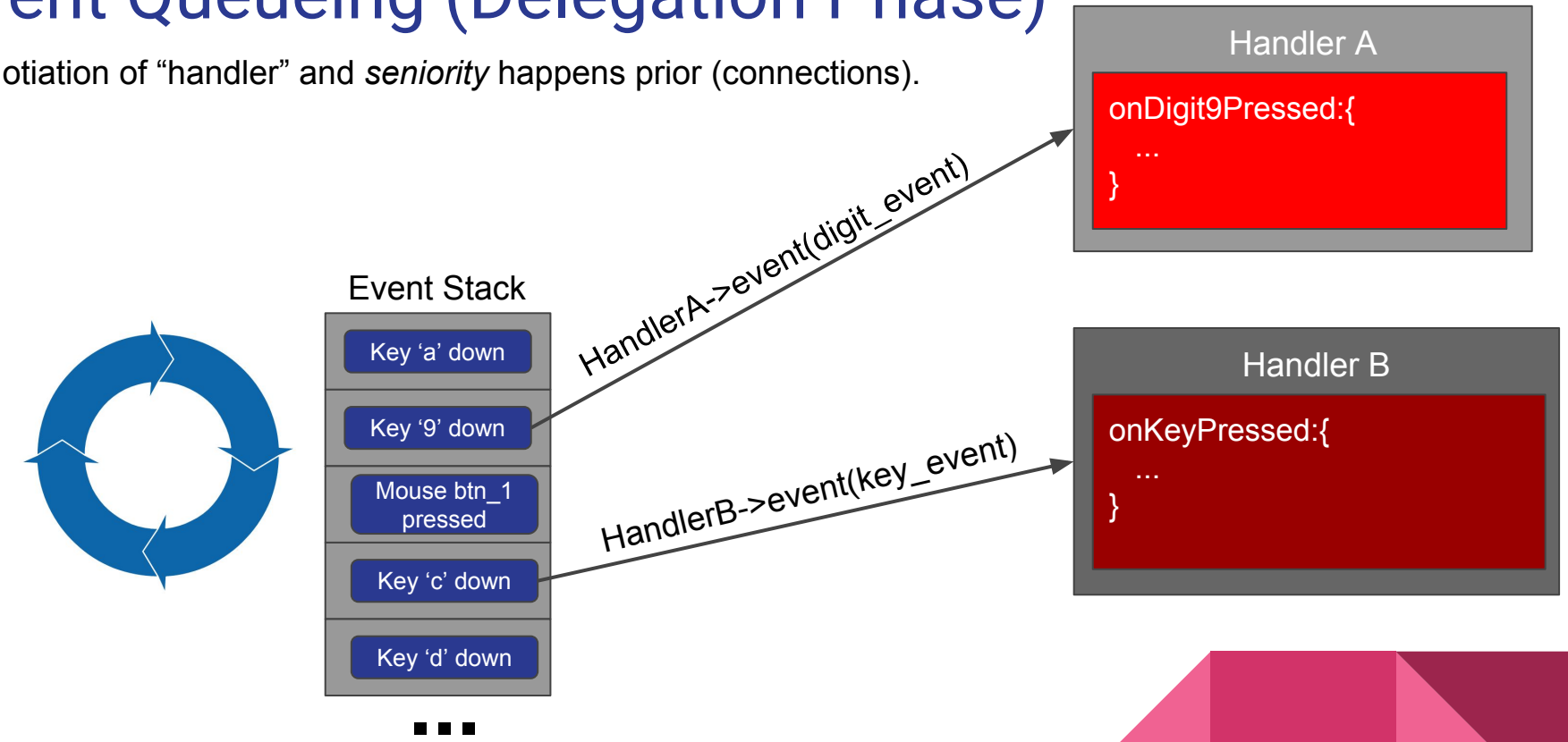
To Wrangle:

- round up, herd, or take charge of (~~livestock~~ events).

https://www.google.com/?ion=1&espv=2#q=definition+wrangle

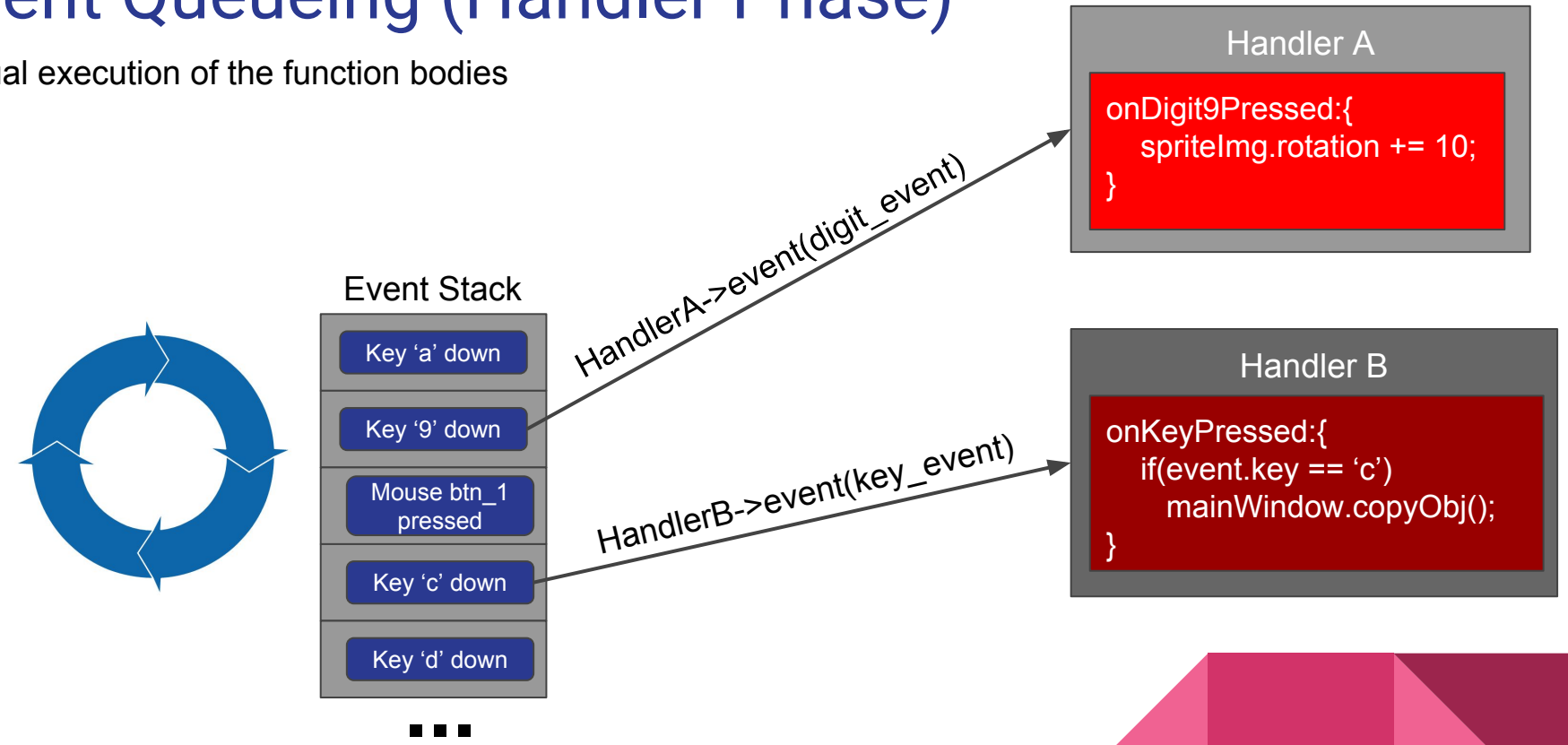
Event Queueing (Delegation Phase)

Negotiation of “handler” and *seniority* happens prior (connections).



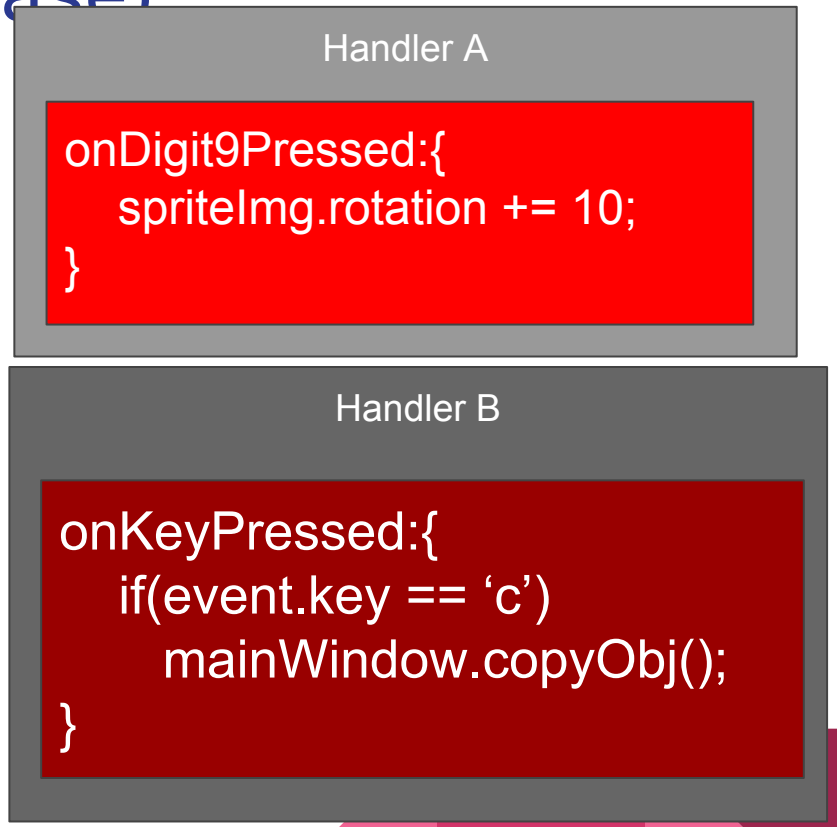
Event Queueing (Handler Phase)

Actual execution of the function bodies



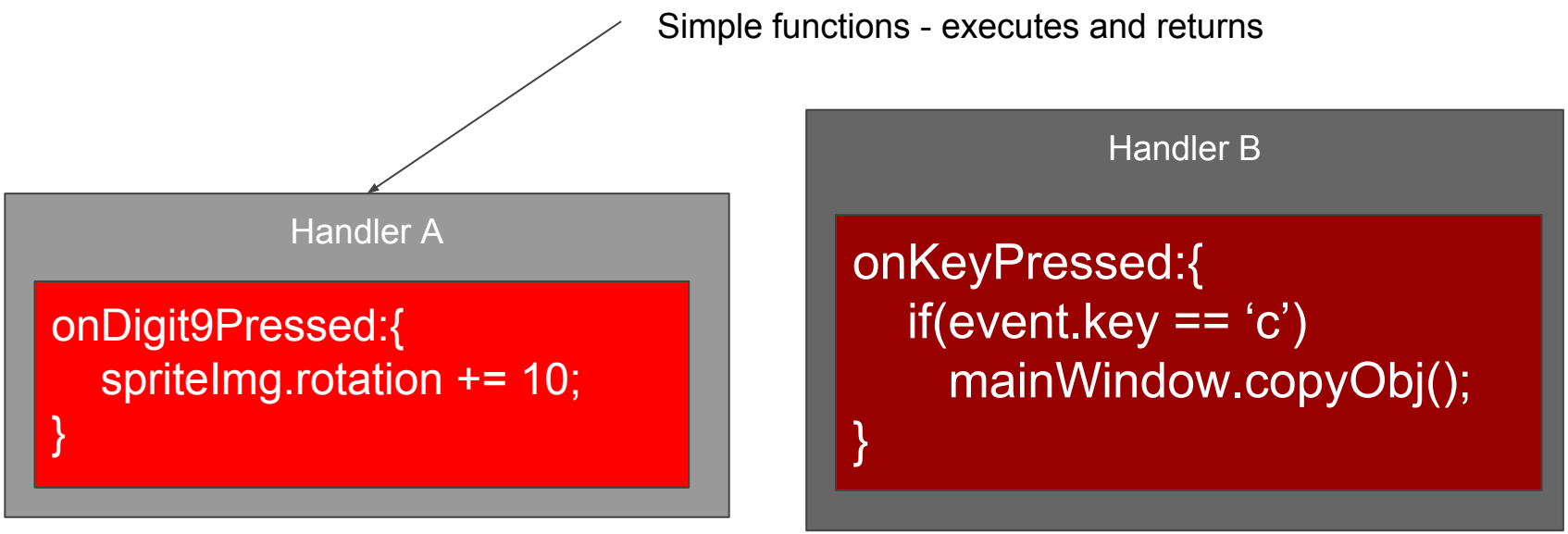
Event Queueing (Handler Phase)

Actual execution of the function bodies



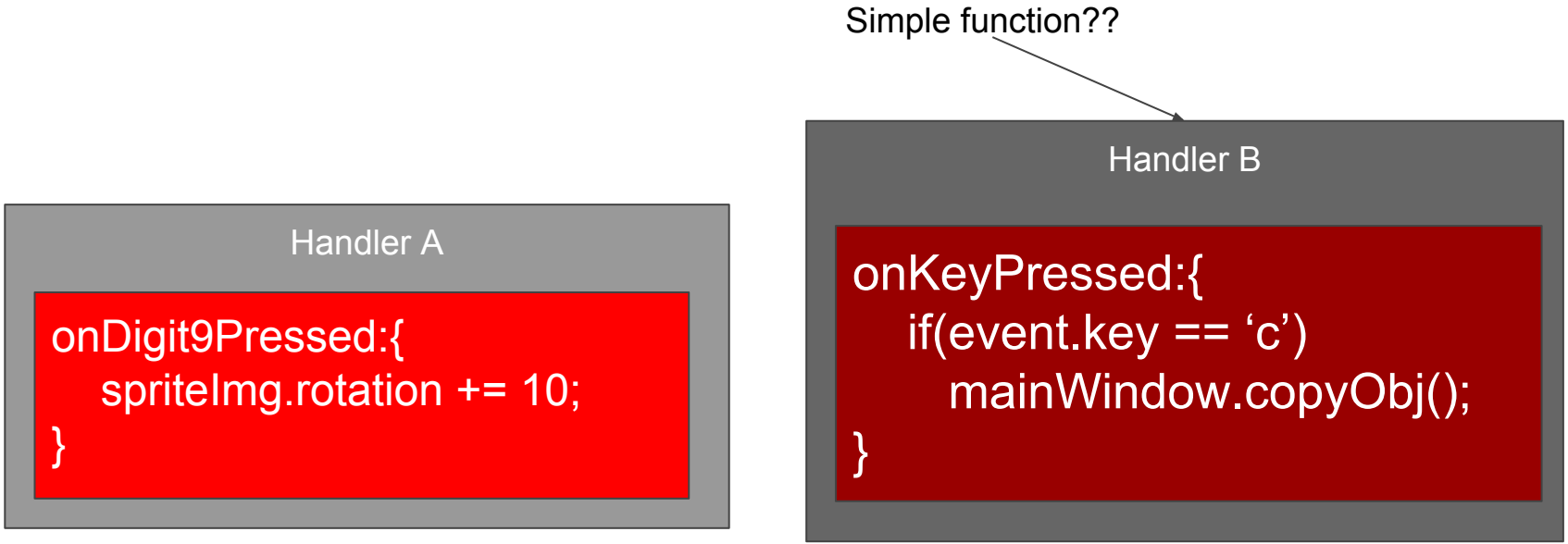
Event Queueing (Handler Phase)

Actual execution of the function bodies



Event Queueing (Handler Phase)

Actual execution of the function bodies



Event Queueing (Handler Phase)

Actual execution of the function bodies

Simple function??

mainWindow.copyObj()

```
copyObj()
{
    QList<QObject*> kids = this->children();
    foreach( QObject *kid in kids)
    {
        QObject * new_kid = new QObject();
        kid->copyObj(new_kid);
        ...
    }
}
```

Handler B

```
onKeyPressed:{
    if(event.key == 'c')
        mainWindow.copyObj();
}
```

Event Queueing (Handler Phase)

Actual execution of the function bodies

Simple function??

mainWindow.copyObj()

```
copyObj()
{
    QList<QObject*> kids = this->children();
    foreach( QObject *kid in kids)
    {
        QObject * new_kid = new QObject();
        kid->copyObj(new_kid);
        ...
    }
}
```

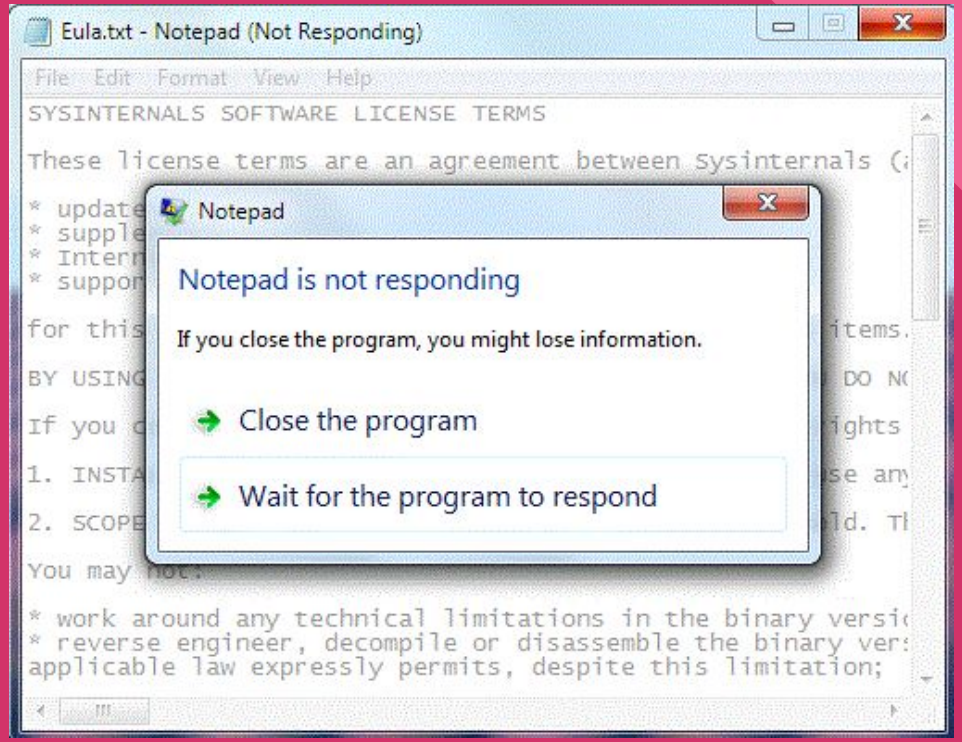
Handler B

```
onKeyPressed:{
    if(event.key == 'c')
        mainWindow.copyObj();
}
```

Long Blocking Process - - prevents input phase

GUI thread stops handling events

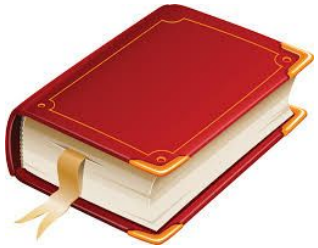
The kernel attempts to deliver the input messages but they are *dropped* instead of queued. After the application fails to respond for long enough the window manager steps in.



The Solution - WORKER THREADS

Monday Week 7 - Advanced C++ integration and worker threads!

Readings This Weekend (week 6)



Dynamic Views (Chapter 6.3)
http://qmlbook.github.io/en/ch06/index.html#dynamic-views

Dynamic QML(Chapter 13)
http://qmlbook.github.io/en/ch13/index.html

Qt and C++ (Chapter 15)
http://qmlbook.github.io/en/ch15/index.html

Events and filters
http://doc.qt.io/qt-5/eventsandfilters.html

Qt/Qml Signals
http://doc.qt.io/qt-5/qtqml-syntax-signals.html

Dynamic Object Creation JS
http://doc.qt.io/qt-5/qtqml-javascript-dynamicobjectcreation.html

Javascript Memory Management

Live code demo.