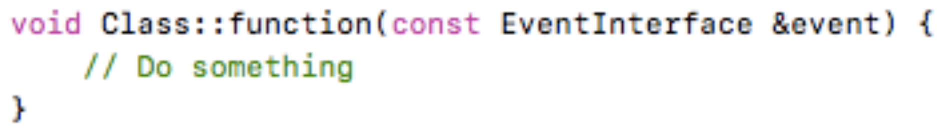
Setting Up an Event Listener for the Event Manager

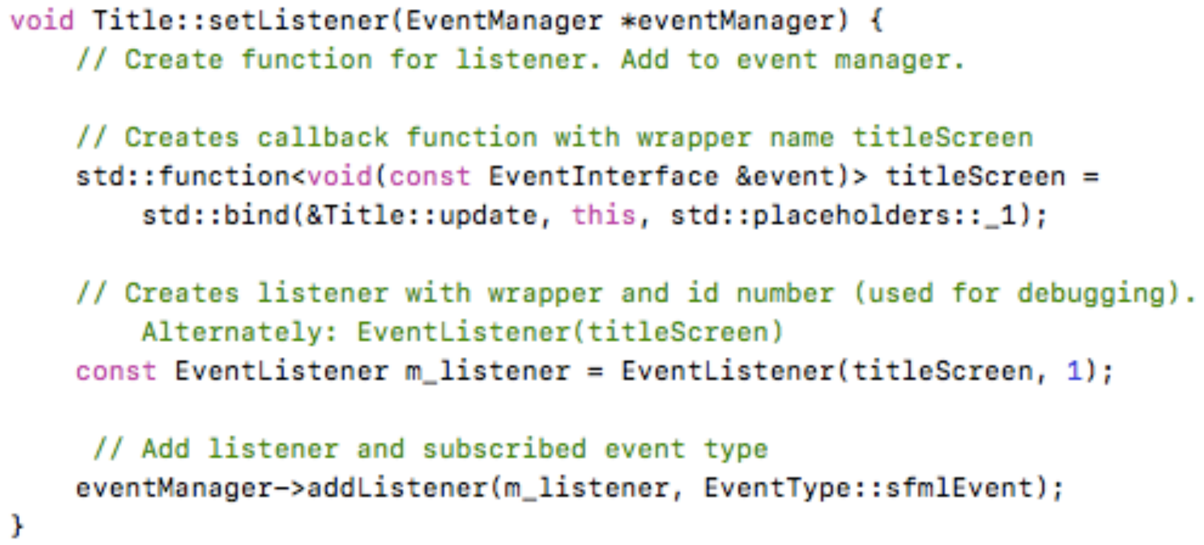
1. Write a function with the prototype `void function(const EventInterface &event)`

   Ex:



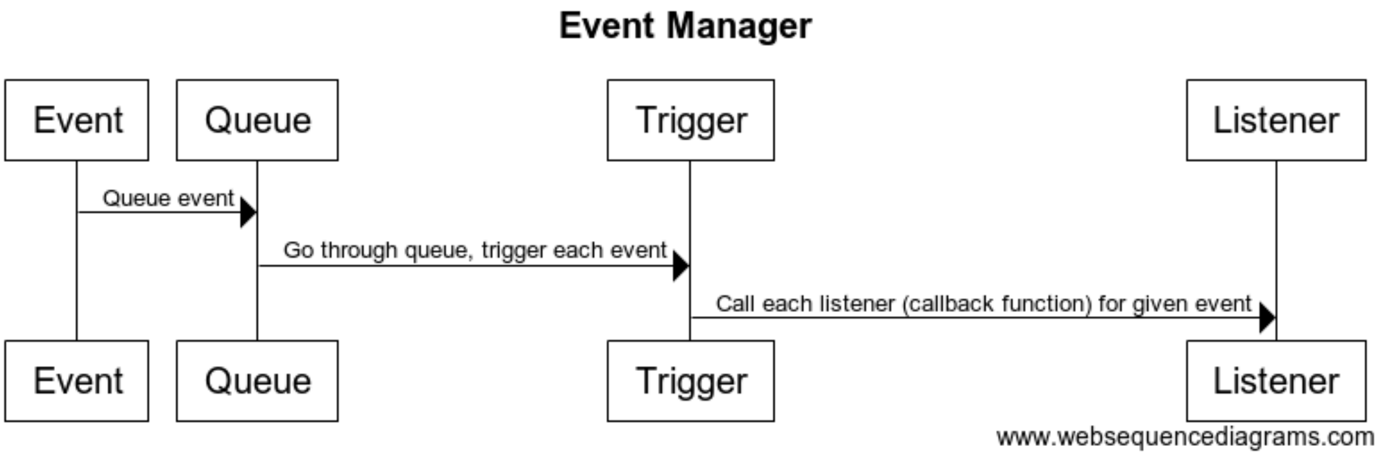
```
void Class::function(const EventInterface &event) {
    // Do something
}
```

2. Create a callback function in the class that subscribes to an event.

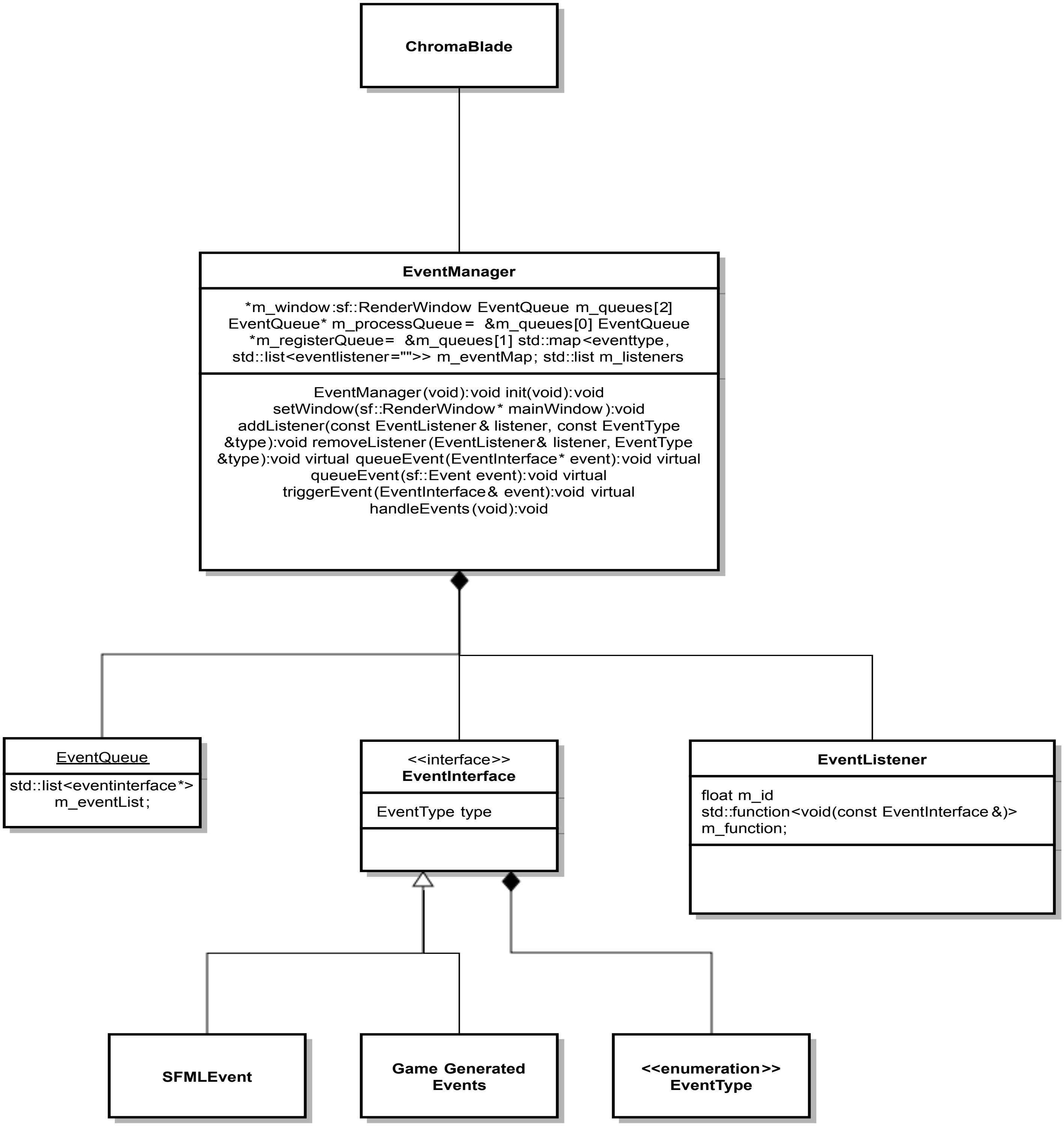


```
void Title::setListener(EventManager *eventManager) {
    // Create function for listener. Add to event manager.

    // Creates callback function with wrapper name titleScreen
    std::function<void(const EventInterface &event)> titleScreen =
        std::bind(&Title::update, this, std::placeholders::_1);

    // Creates listener with wrapper and id number (used for debugging).
        Alternately: EventListener(titleScreen)
    const EventListener m_listener = EventListener(titleScreen, 1);

     // Add listener and subscribed event type
    eventManager->addListener(m_listener, EventType::sfmlEvent);
}
```

Event Manager Sequence Diagram


Event Manager
Event
Queue
Trigger
Listener
Queue event
Go through queue, trigger each event
Call each listener (callback function) for given event
Event
Queue
Trigger
Listener
www.websequencediagrams.com

Event Manager Class Diagram


ChromaBlade
EventManager
m_window:sf::RenderWindow EventQueue m_queues[2] EventQueue m_processQueue = &m_queues[0] EventQueue *m_registerQueue = &m_queues[1] std::map<eventtype, std::list<eventlistener="">> m_eventMap; std::list m_listeners
EventManager(void):void init(void):void setWindow(sf::RenderWindow* mainWindow):void addListener(const EventListener& listener, const EventType &type):void removeListener(EventListener& listener, EventType &type):void virtual queueEvent(EventInterface* event):void virtual queueEvent(sf::Event event):void virtual triggerEvent(EventInterface& event):void virtual handleEvents(void):void
EventQueue
std::list<eventinterface*> m_eventList;
<<interface>>
EventInterface
EventType type
EventListener
float m_id
std::function<void(const EventInterface&)> m_function;
SFMLEvent
Game Generated Events
<<enumeration>>
EventType