*Florida International University*

*School of Computing and Information Sciences*

Software Engineering Focus

Feature Document

User Story ID #722 Input Device Events - Mouse

**Name: Bernardo Pla**

**Team Member(s): Pachev Joseph, Daniel Rivero, Daniel Khawand,**

**Project: WebVR 1.0**

**Product Owner(s)**:

**Mentor(s)**: Francisco Ortega

**Instructor**: Masoud Sadjadi

**User Story Name: #722 Input Device Events - Mouse**

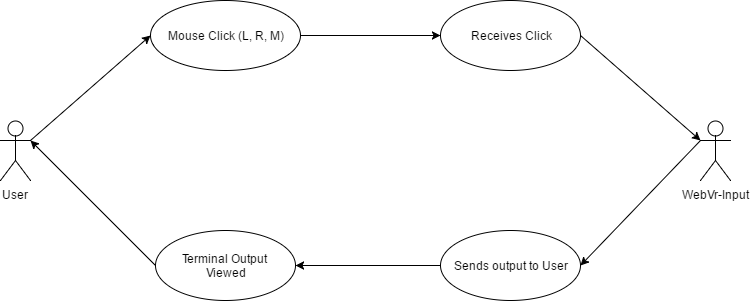
* As a developer, I want to create mouse input device events, so that I can handle states and update device tuple accordingly.

Acceptance Criteria

* Handle Mouse Events.

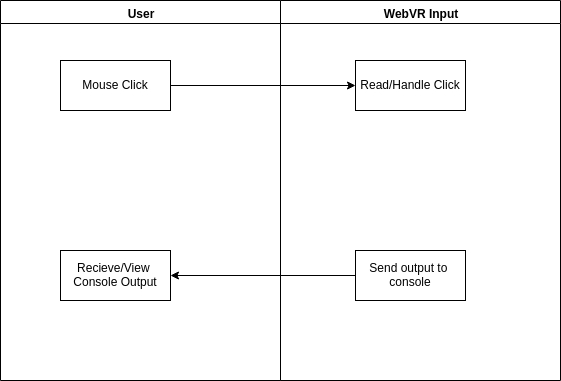
**Use Case**

* Name: Mouse device events
* Actor: User, Device, webvr-input window
* Preconditions: WebVR-input window is running and listening for events from device. Assume device is already connected to the machine.
* Description:
  + Mouse is connected
    - Connection is handled by and device is instantiated
  + User performs clicks on different parts of generated window.
  + Properties of event function calls are recorded on terminal.

**Use Case Diagram** 

**Fig. Use Case Diagram:** The diagram tracks the flow between the user and the WebVR-Input window handling the click

**Sequence Diagram**

  
**Fig. Sequence Diagram:** This is the sequence of events between the user and the input window.

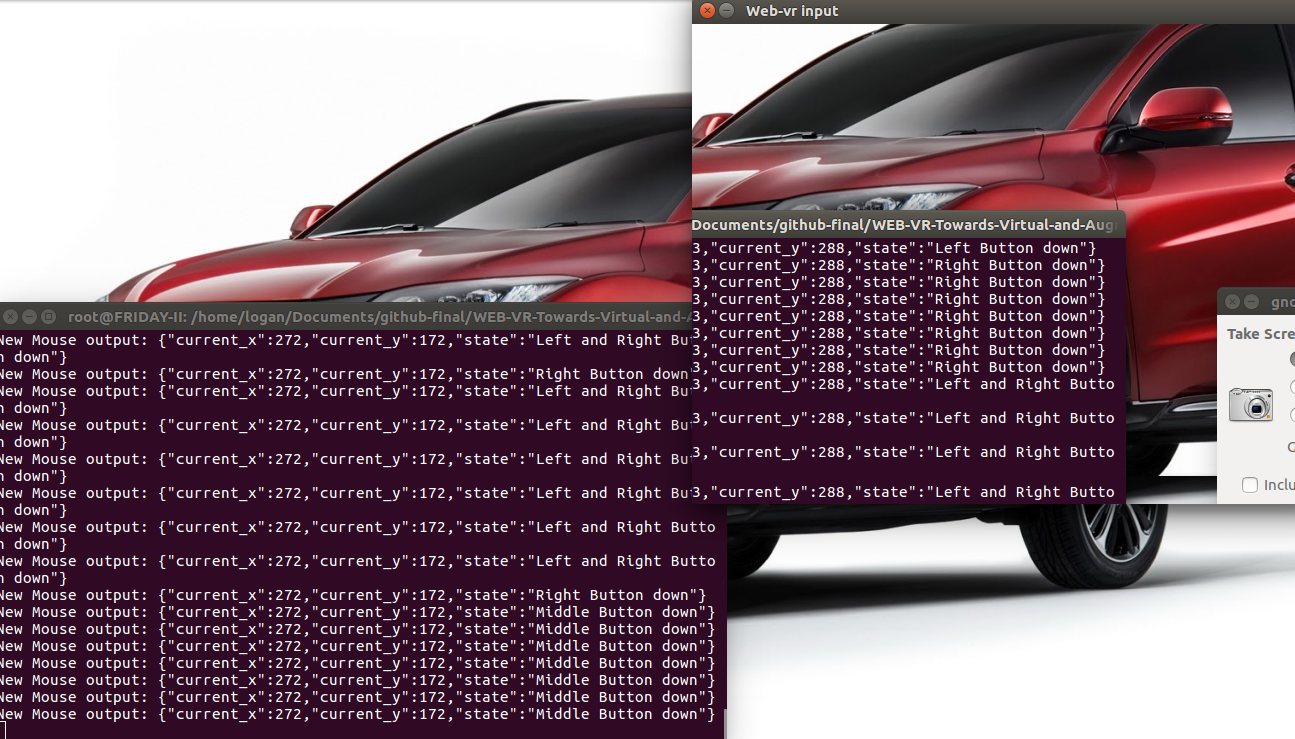
**Class Diagram**

**Unit Test**

* Test case ID: Unit001 - Record states of mouse input device - Sunny001
* Description/Summary of Test: The developer will launch the demo class and record a left click, a right click, and a center click on different points of the screen.
* Pre-condition: The main class in the rust library is running. A mouse must be connected to the machine.
* Expected Results: When the mouse is connected, a left click will print “Left button down.” A right click will print “Right button down”. A center click will print “Middle button down”. The output of these will be recorded on a terminal.
* Actual Result: The terminal read “Left button down” on a left click event. It read “Right button down” on a right click event. It read “Middle button down” on a center click event. The output was recorded on a terminal thus proving the events were being handled as expected.
* Status (Fail/Pass): Pass

**Integration Test**

**Visual User Guide**



**Figure 1**: Output from mouse. For this demo, a window is generated on the top right corner. When the user clicks through this window, the coordinates and the state of the device are printed to the terminal, shown on the bottom left corner.