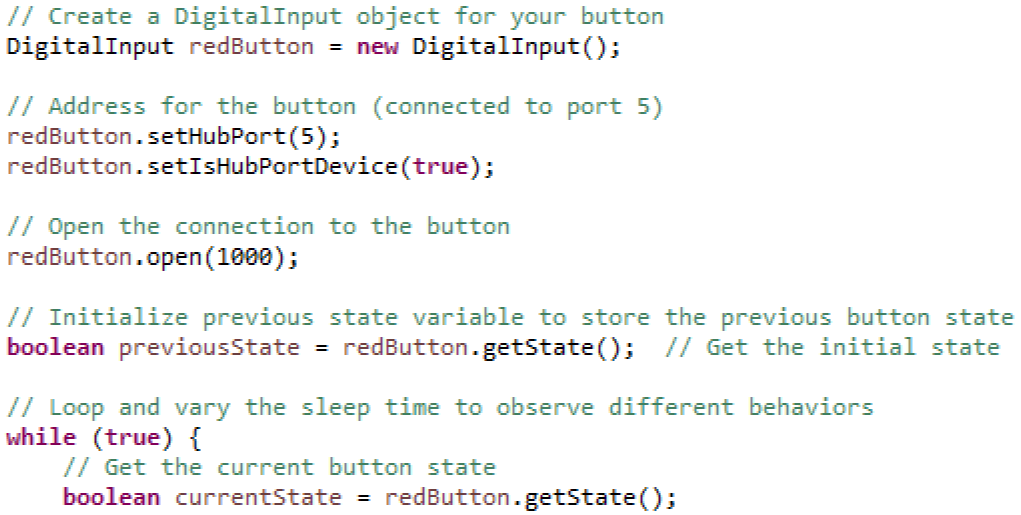
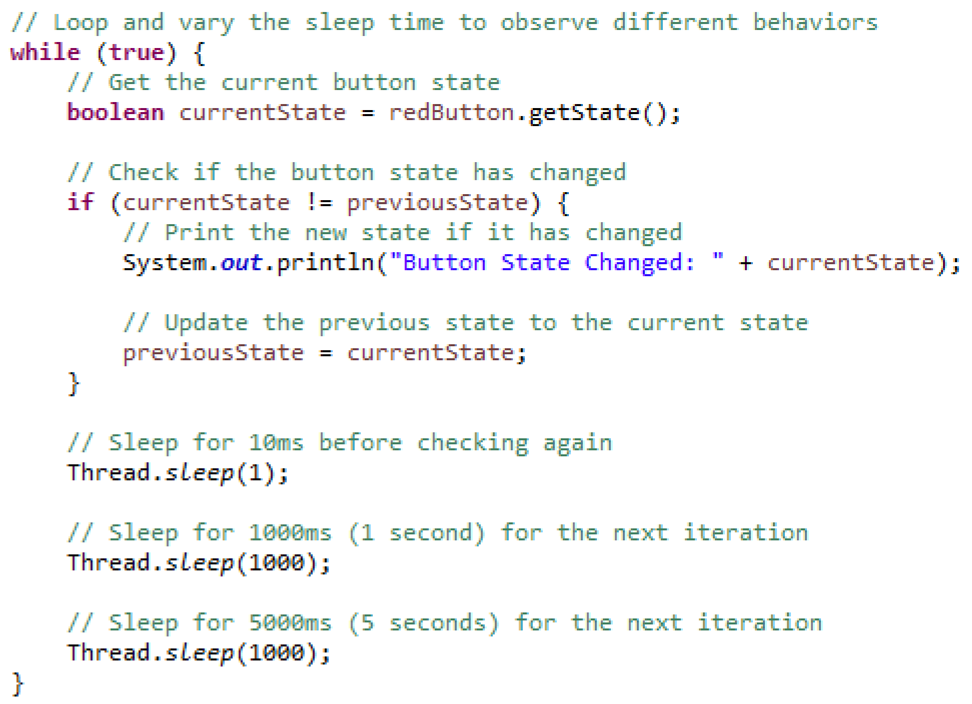
ReflectionLog ReadButton



```java
// Create a DigitalInput object for your button
DigitalInput redButton = new DigitalInput();

// Address for the button (connected to port 5)
redButton.setHubPort(5);
redButton.setIsHubPortDevice(true);

// Open the connection to the button
redButton.open(1000);

// Initialize previous state variable to store the previous button state
boolean previousState = redButton.getState();  // Get the initial state

// Loop and vary the sleep time to observe different behaviors
while (true) {
    // Get the current button state
    boolean currentState = redButton.getState();
```

This code creates a button object and connects it to port 5. It then opens the connection to the button. Finally, it stores the initial state of the button (pressed or not pressed) in the previousState variable.



```
// Loop and vary the sleep time to observe different behaviors
while (true) {
    // Get the current button state
    boolean currentState = redButton.getState();

    // Check if the button state has changed
    if (currentState != previousState) {
        // Print the new state if it has changed
        System.out.println("Button State Changed: " + currentState);

        // Update the previous state to the current state
        previousState = currentState;
    }

    // Sleep for 10ms before checking again
    Thread.sleep(1);

    // Sleep for 1000ms (1 second) for the next iteration
    Thread.sleep(1000);

    // Sleep for 5000ms (5 seconds) for the next iteration
    Thread.sleep(1000);
}
```

This code repeatedly checks the state of the button in a loop. It first checks if the button's state has changed since the last check. If the state has changed, it prints the new state and updates the previousState variable. After each check, the code pauses for a short period, first for 1 millisecond (Thread.sleep(1)), then for 1 second, and then for another 1000 milliseconds (5 seconds in total). The varying sleep times allow for different observation behaviors during each loop iteration.