Selenium Crash Testing Suggestions

Updated Documentation Fall 2024

Potential Ways to Use Selenium to Simulate a crash

1. Simulate High User Load:
    - Use Selenium to simulate multiple simultaneous users interacting with the game.
    - Stress-test specific game actions, like rapidly clicking buttons, submitting forms, or making requests.
    - This helps identify issues like resource exhaustion or race conditions.
2. Input Edge Cases:
    - Use Selenium to input extremely long strings, special characters, or even invalid input where text is expected (e.g., letters in number fields).
    - Attempt SQL injection or HTML/JavaScript injection (in a controlled, safe environment) to see if the input sanitization is robust.
3. Simulate Network Instability:
    - Manually introduce latency or packet loss using tools like network throttling (available in Chrome Developer Tools).
    - Use Selenium to simulate scenarios where the connection drops or becomes unstable during critical operations (e.g., saving a game state).
4. Interruptions in API Calls:
    - Use Selenium to trigger actions that rely on API calls and then simulate a server error (like a timeout or a 500 Internal Server Error).
    - See how the game behaves when the API call fails, ensuring that error handling works as expected.
5. Randomized Input or Actions:
    - Create a script that uses Selenium to perform random actions within the game repeatedly.
    - This can help uncover unexpected behaviors that might not occur during normal testing.
6. Force Memory Leaks or Resource Consumption:
    - Perform actions in a loop that are known to consume memory, such as loading large assets repeatedly.
    - Monitor the application's memory usage to see if it leads to an out-of-memory error or a crash.