Character Selection Test

This document describes the automated test implemented in the CharacterSelectionTest.cs script. This test verifies the boundary and stress behavior of the character selection system. It includes tests for the cycling of characters using the "Next" and "Back" buttons. Plus, stress tests to evaluate the system's robustness under heavy load and edge-case scenarios. The test is implemented as a MonoBehaviour and is executed without the Unity Test Runner framework.

Test Description

The CharacterSelectionTest script performs the following tests:

- **Next/Back Button Cycling:** Verifies that the character selection cycles correctly when the "Next" and "Back" buttons are clicked. It checks if the selection wraps around to the first/last character after reaching the end/beginning of the character list.
- **Empty Character Database:** Verifies the behavior of the character selection system when the CharacterDatabase is empty.
- **Single Character:** Verifies the behavior of the character selection system when the CharacterDatabase contains only one character.
- **Null Character Prefabs:** Verifies the behavior of the character selection system when some CharacterVariable entries in the database have null characterPrefab values.
- **Null Animator Controllers:** Verifies the behavior of the character selection system when some CharacterVariable entries in the database have null animatorController values.
- **Rapid Next/Back Button Input:** Evaluates the system's ability to handle a high volume of rapid "Next" and "Back" button inputs.
- **Large Character Database Test:** Assesses the system's performance and memory usage when dealing with a very large number of characters in the CharacterDatabase.
- **Concurrent Scene Loading:** Evaluates how the character selection system behaves when a scene change is initiated while the user is rapidly selecting characters.
- **Prefab Instantiation Stress:** Determines the system's performance under heavy instantiation and destruction of character prefabs.

Implementation Details

- The test is implemented as a MonoBehaviour (CharacterSelectionTest) in the CharacterSelectionTest.cs script.

- The test is executed in the Start() method by calling the TestCharacterSelection() coroutine.
- The TestCharacterSelection() coroutine iterates through the specified number of tests (numberOfTests).
- The script requires the following to be assigned in the Unity Inspector:
  - characterManager: A reference to the CharacterManager instance in the scene.
  - nextButton: A reference to the "Next" Button in the scene.
  - backButton: A reference to the "Back" Button in the scene.
- The test uses a small delay (delayBetweenTests) between button presses for basic tests. Stress tests have their own delays.
- Test results (pass/fail/warnings) are logged to the Unity console using Debug.Log, Debug.LogError, and Debug.LogWarning.
- The script stores the original state of the CharacterDatabase and restores it after the tests are complete.
- The script uses UnityEngine.TestTools and NUnit.Framework for assertions in stress tests.

Test Cases

The TestCharacterSelection coroutine executes the following test cases:

1. **Empty Character Database Test (RunEmptyDatabaseTest)**
   - Sets the CharacterDatabase to be empty.
   - Calls nextOption() and backOption() on the CharacterManager.
   - Checks that CharacterManager handles the empty database without errors.
   - Restores the original CharacterDatabase state.
2. **Single Character Test (RunSingleCharacterTest)**
   - Sets the CharacterDatabase to contain only one character.
   - Calls nextOption() and backOption() on the CharacterManager.
   - Checks that CharacterManager handles the single character correctly.
   - Restores the original CharacterDatabase state.
3. **Null Character Prefab Test (RunNullPrefabTest)**
   - Sets the characterPrefab of one CharacterVariable in the database to null.
   - Calls nextOption() and backOption() on the CharacterManager.
   - Restores the original characterPrefab value.
   - Logs a message to the console indicating that the CharacterManager's behavior with a null prefab should be checked for errors.
4. **Null Animator Controller Test (RunNullAnimatorTest)**
   - Sets the animatorController of one CharacterVariable in the database to null.

   - Calls nextOption() and backOption() on the CharacterManager.
   - Restores the original animatorController value.
   - Logs a message to the console indicating that the CharacterManager's behavior with a null animator controller should be checked for errors.
5. **Next Button Test (RunNextButtonTest)**
   - Stores the initial selected character option.
   - Simulates pressing the "Next" button multiple times (more than the number of characters in the database).
   - Checks if the final selected option wraps around correctly to the initial option.
6. **Back Button Test (RunBackButtonTest)**
   - Stores the initial selected character option.
   - Simulates pressing the "Back" button multiple times (more than the number of characters in the database).
   - Checks if the final selected option wraps around correctly to the initial option.
7. **Rapid Next/Back Button Input Test (RunRapidInputTest)**
   - Simulates pressing the "Next" and "Back" buttons at a very high frequency.
   - Measures the time it takes for the system to respond.
   - Checks for missed button presses or incorrect character selections.
   - Asserts that the final option wraps around correctly.
8. **Large Character Database Test (RunLargeDatabaseTest)**
   - Populates the CharacterDatabase with a large number of characters.
   - Measures the time it takes to iterate through all characters.
   - Monitors memory usage.
   - Asserts that the iteration time is within acceptable limits.
9. **Concurrent Scene Loading Test (RunConcurrentSceneLoadTest)**
   - Initiates a scene change while rapidly selecting characters.
   - Checks if the scene loads correctly and if the correct character is passed to the new scene.
   - Monitors for race conditions or unexpected behavior.
   - Asserts that the scene loaded.
10. **Prefab Instantiation Stress Test (RunPrefabInstantiationTest)**
   - Rapidly switches between a subset of characters, instantiating and destroying their prefabs.
   - Measures the time taken for instantiation and destruction.
   - Monitors CPU and memory usage.
   - Asserts that the time taken is within acceptable limits.

Usage Instructions

- **Scene Setup**

    - Open the Unity scene where the character selection is implemented.
- **Create Test GameObject**
    - Create an empty GameObject in the scene.
    - Attach the CharacterSelectionTest.cs script to the empty GameObject.
- **Inspector Setup**
    - In the Inspector, assign the following to the CharacterSelectionTest component:
        - characterManager: The CharacterManager instance in the scene.
        - nextButton: The "Next" Button in the scene.
        - backButton: The "Back" Button in the scene.
    - Optionally, adjust the following parameters for the stress tests:
        - rapidInputIterations: The number of iterations for the rapid input test.
        - rapidInputDelay: The delay between button presses in the rapid input test.
        - largeDatabaseSize: The number of characters in the large database test.
        - sceneLoadDelay: The delay before triggering scene load.
        - prefabSwitchCount: The number of prefab switches in the prefab instantiation test.
- **Run the Test**
    - Enter Play Mode.
    - The test will execute automatically.
    - View the test results in the Unity console.