System and Unit Tests

Not Zombies

Team: C.C.R.A.M

March 10, 2015

System Test Scenarios

Sprint 1:

Assuming you have already pressed the .exe for our game and pressed the start button in the main menu.

User Story 1: As a gamer, I want to be able to collect and use items that I find in the world.

Scenario:

1. Walk around the map until you find an item in sight
    a. Use WASD to move around
2. Once you find an item, walk over to it and stand over it
3. Once you collide, the game will prompt you to press E to pick up
4. Press the E key
5. Now the item is in your inventory
    a. Press I to open your inventory
    b. Click and drag item to your hand to use them
    c. Right click food to eat

User Story 2: As a gamer, I want to be able to use the keyboard so that my character moves.

Scenario:

1. Use the keys WASD to move up, left, down, and right respectively.
2. Use the mouse to look whatever direction you want while moving

User Story 3: As a gamer, I want a large regional open world so that I may explore it.

Sprint #3 - User Story 3: As an explorer, I want there to be a complete, populated map with transitions between them so I can explore the dynamic game world.

Scenario:

1. Start the executable and press the Start Button.
2. Use any movement key to move around the map
3. When come the edge of a scene, continue running and you will be transitioned to a new scene.
4. This shows our open world.

User Story 4: As a developer, I want to implement an inventory with a weight limit so that the player can only have limited items at the same time.
Scenario:

1. While in game, if you are standing over many items, press E to pick up each item.
2. After exceeding your limit, you can no longer press E to pick up items.
3. Press I to open your inventory.
4. Click and drag items to move them around.
5. Click and drag them away from your inventory to drop items and lessen your overall weight.

Sprint 2:

User Story 1: As a player, I want an inventory so that I can see and select all items I've obtained.

1. See previous sprint for information on this.

User Story 2: As a wounded survivor, I want food that I can eat so that I can restore health and be stronger.
Scenario:

1. Wander until you find a food item.
2. Press 'E' over the food to pick it up.
3. Open inventory with the 'I' key
4. Hover over your food with the mouse
5. Read the restore values of your different kinds of food
6. Right click your desired food and it will restore HP.

User Story 3: As as survivor, I want weapons to attack the enemies so that they die.
User Story 4: As a survivor, I want to be able to use my weapons to attack the enemies so that they die.
Scenario:

1. Wander the world until you find a weapon. (Melee or Ranged)
2. Press E to pick up said item on the ground.
3. Press I to open your inventory
4. Click and drag your weapon to your hand or holster.
5. If the item is a melee item
    a. After you drag the weapon to the hand icon, close the inventory
    b. Left-click to use your item
    c. Melee weapons have infinite uses
    d. Next find an enemy

    e. Click within melee range to kill the enemy
6. If the item is a ranged weapon
    a. Drag the item to your hand icon
    b. Close the inventory
    c. Make sure you have ammo to shoot
    d. If not, wander until you pick up ammo
    e. With ammo, left click to fire your weapon
    f. Move the mouse so you are lined up with an enemy
    g. Left click/shoot until the enemy dies

<u>Sprint 3:</u>

User Story 1: As a player, I want to have enemies that I can fight so that I can obtain better equipment and progress in the game.

Scenario:

1. Wander the level until you find a weapon to use on an enemy
2. Pick up the weapon
3. Equip the weapon
4. Left-click with your item until you kill your desired enemy
5. On death, the enemy will drop a random number of food/weapon items
6. Press E on the items to pick them up

User Story 2: As a survivor, I want to have a day and night cycle to change gameplay and enemy interactions.

Sprint 2 - User Story 5: As a survivor, I want a realistic world that has a day and night cycle.

Scenario:

1. Double click the .exe file for our game
2. Press the start button to begin the game
3. Play the game/wait for time to pass
4. Around afternoon the screen transitions to a slightly darker day
5. Then it goes full dark, then lightens up for dawn, and returns to full light at the start of a new day.

User Story 4: Resolve the end-game.

Scenario:

1. Given you're already playing the game and know how to play, you could be ready for the end game.
2. Play until you've collected enough weapons, food, and ammo to take on a stronghold.

3. Journey to the center of the map to find the stronghold.
4. Enter and kill all enemies in the stronghold
5. After all enemies die, a new door appears
6. Enter through door to start final cutscene
7. Watch and enjoy as you have beaten the game.

Unit Tests

(Alex) Food:

Tests for food include picking the food up and eating the food/restoring the correct amount of health.

Pick-up items:

- [pick up Banana] : pass
    - [eat banana]: pass
- [pick up Pear] : pass
    - [eat pear] : pass
- [pick up Beef Jerky] : pass
    - [eat Beef Jerky]: pass
- [pick up Burger] : pass
    - [eat Burger] : pass
- [pick up Candy Bar] : pass
    - [eat Candy] : pass
- [pick up Canned Beans] : pass
    - [eat beans] : pass
- [pick up Canned Fruit] : pass
    - [eat fruit] : pass
- [pick up Cereal] : pass
    - [eat cereal] : pass
- [pick up Chicken] : pass
    - [eat Chicken] : pass
- [pick up Chips] : pass
    - [eat chips] : pass
- [pick up Lime Pop] : pass
    - [eat Lime Pop] : pass
- [pick up Monster Pop] : pass
    - [eat Monster] : pass
- [pick up Protein Bar] : pass
    - [eat protein] : pass
- [pick up Steak] : pass
    - [eat steak] : pass

(Mat) Map Bounds:

Tests for map bounds includes ensuring scene transitions move to the correct scenes, and invisible walls correctly seal in all maps:

[Aridae]:
- [transition to Savann]: pass
- [transition to Coastalis]: pass
- [invisible walls seal map]: pass

[Coastalis]:
- [transition to Aridae]: pass
- [transition to Old Coastalis]: pass
- [invisible walls seal map]: pass

[Old Coastalis]:
- [transition to Coastalis]: pass
- [transition to Northern Wastes]: pass
- [invisible walls seal map]: pass

[Northern Wastes]:
- [transition to Old Coastalis]: pass
- [transition to Boreal]: pass
- [invisible walls seal map]: pass

[Boreal]:
- [transition to Northern Wastes]: pass
- [transition to Laketon]: pass
- [invisible walls seal map]: pass

[Laketon]:
- [transition to Boreal]: pass
- [transition to Savann]: pass
- [invisible walls seal map]: pass

[Savann]:
- [transition to Laketon]: pass
- [transition to Aridae]: pass
- [invisible walls seal map]: pass

(Cameron) Weapons:

Tests for weapons include ensuring each gun fires the correct type of projectile, each melee weapon swings with a different sprite, the projectiles deal correct damage, and the melee weapons deal the correct damage.

[Guns fire correct projectile]:
- [pistol]: pass

- [laser pistol]: pass
- [sub-machine gun]: pass
- [rocket launcher]: pass
- [shotgun]: pass

[melee weapons show correct sprite]:

- [sword]: pass
- [mace]: pass
- [crowbar]: pass
- [lightsaber]: pass

[Guns deal correct damage]:

- [pistol]: pass
- [laser pistol]: pass
- [sub-machine gun]: pass
- [rocket launcher]: pass
- [shotgun]: pass

[melee weapons deal correct damage]:

- [sword]: pass
- [mace]: pass
- [crowbar]: pass
- [lightsaber]: pass

(Cameron) Player Movement:

Tests for player movement include ensuring the player moves in the right directions, the player can sprint, and the player always looks at the current mouse position.

[player can sprint]: pass

[player looks at mouse position]: pass

[player moves on 8 axes (2D)]:

- [←]: pass
- [↑]: pass
- [→]: pass
- [↓]: pass
- [↖]: pass
- [↗]: pass
- [↘]: pass
- [↙]: pass

(Ryan) Roof/Tree Fade:

- The roof will fade to transparent when the player enters a building, allowing the player to see what is inside. When the player exits the building, the roof will fade back to solid.

  [Hut Building]
  - [player enters building and roof fades to transparent] : PASS
  - [player exits building and roof fades to solid] : PASS

  [L Building]
  - [player enters building and roof fades to transparent] : PASS
  - [player exits building and roof fades to solid] : PASS

  [Divider Building]
  - [player enters building and roof fades to transparent] : PASS
  - [player exits building and roof fades to solid] : PASS

  [Stable Building]
  - [player enters building and roof fades to transparent] : PASS
  - [player exits building and roof fades to solid] : PASS

  [Igloo Building]
  - [player enters building and roof fades to transparent] : PASS
  - [player exits building and roof fades to solid] : PASS

- Trees fade to 50% opacity when the player walks under them, and fade back to solid when the player is no longer under them.

  [Evergreen Tree]
  - [player goes under tree and tree fades to 50% opacity] : PASS
  - [player goes away from tree and it fades back to solid] : PASS

  [Apple Tree]
  - [player goes under tree and tree fades to 50% opacity] : PASS
  - [player goes away from tree and it fades back to solid] : PASS

(Alex) Day Night Cycle:

Tests for the Day Night Cycle include cycling from light, to afternoon, to evening, to dawn, and back to light. The day night cycle also needs to cycle after a full day.

[Day Night Cycle]

- [Cycle from full light to day fog] : pass
- [Cycle from day fog to evening] : pass
- [Cycle from evening to dawn] : pass
- [Cycle from dawn back to light] : pass

[Day Night Cycle Loop]

- [Initial Day Loop] : pass
- [Any day after first day] : pass
    - [Cycle from dawn to light] : fail

<u>(Mat) Menu Transitions:</u>

Tests for menu transitions include ensuring the New Game spawns player exactly once, that Test Suite button navigates to acceptance test scenes, that quit button actually quits game, that test menu “Back to Main Menu” button navigates back to Main Menu, and that each scene in Test Suite is loaded properly.

[Main Menu]:

- [New Game spawns player]: pass
- [Test Suite nagivates correctly]: pass
- [Quit Game actually quits]: pass

[Test Suite]:

- [Load Health Runner]: fail
- [Day Night Magic]: pass
- [Mapper Quest]: fail
- [Back To Main Menu]: pass

<u>(Craig) Enemy Spawner:</u>

Tests for the Enemy Spawner include making sure they enemies spawn correctly in the map at the correct locations.

[All Maps]

- [Spawn enemy in the map bounds] : pass
- [Spawn enemy with the correct stats] : pass
- [Stops spawning when enemy limit is reached] : pass
- [Spawn new enemies after their death] : pass
- [Spawn enemies NOT in an object and NOT on screen] : failure

<u>(Alex) Chest Spawning Different Items:</u>

Tests for the Chest spawning different items includes opening the chest and spawning x number of items in a circle around the chest.

[Chest]

- [Open chest with the ‘E’ key] : pass
- [Items spawn from chest]: pass
- [Items spawn in a circle]: pass

(Craig) Pathfinding:

This Test was meant to ensure the Enemies had a way to compute their own paths from one point in the game world to another. Then the AI had to move along the path and they had to not get stuck.

- [Limit the pathfinding to our maps] : pass
- [Find a path between points A and B] : pass
- [If no path exists return an error to Debug.Log()] : pass
- [Move the enemy along the path to the end] : pass
- [Ensure the path between 2 points is walkable] : pass
- [Enemy stops when it gets to the end of the path] : pass

(Ryan) Inventory:

- Pressing the ‘i’ key in-game pauses the game and brings up the inventory view, showing what items are in which slots, ammo count, current weight and max weight. Hovering the mouse over items in the inventory view shows item attributes.

  [Inventory view]
  - [press ‘i’ to pause & open inventory view] : PASS
  - [show inventory items in inventory view] : PASS
  - [show current ammo count] : PASS
  - [show current weight / max weight] : PASS
  - [show item attributes on mouse hover] : PASS

- Picking up items on the ground and placing them in the player inventory in the appropriate slots. All items are placed in the first available slot in the inventory, with the exception of Consumable items not being placed in hand slot (inventory[0]) or holster slot (inventory[1]):

  [Consumable items]
  - [consumable 1 placed in backpack slot 1 (inventory[2])] : PASS
  - [consumable 2 placed in backpack slot 2 (inventory[3])] : PASS

  [Equippable items]
  - [equippable 1 placed in hand slot (inventory[0])] : PASS
  - [equippable 2 placed in holster slot (inventory[1])] : PASS

- [equippable 3 placed in backpack slot 1 (inventory[2]) : PASS

- Items can be dragged and dropped into different inventory slots in the inventory view. Items that are dragged onto an inventory slot that already contains an item swaps the two items. Consumable items cannot be dragged into the hand or holster slots. All items can be dragged out of the inventory view and onto the ground of the game world.

  [All items]
  - [can be dragged out of the inventory onto the ground] : PASS
  - [can be dragged into empty inventory slots] : PASS
  - [can be dragged onto an occupied slot & switch items] : PASS

  [Consumable items]
  - [cannot be dragged onto hand or holster slots] : PASS

- Pressing ‘q’ with an item in the hand slot and an item in the holster slot will swap the two items in the inventory.

  [Equippable items (items that can be in these slots)]
  - [pressing ‘q’ swaps the hand (inventory[0]) and holster (inventory[1]) items] : PASS

<u>(Mat) UI Bar Limits/Bounds:</u>
Tests for the UI Bars ensure that health bars are properly aligned, show a correctly scaled amount, and never surpass their maximal or minimal bounds.

[Health Bar]:
- [Correctly aligned]: pass
- [Correctly scaled]: pass
- [currentHealth ≤ 1000]: pass
- [currentHealth ≥ 0]: pass

[Energy Bar]:
- [Correctly aligned]: pass
- [Correctly scaled]: pass
- [currentEnergy ≤ 100]: pass
- [currentEnergy ≥ 0]: pass

(Craig) AI:

These tests were to prove that the AI worked as intended. There are lots of small rules that they need to follow and I had to test all of them.

- [Finite State Machine Works as a backbone] : pass
- [State class is ready to be overrided] : pass
- [State_Guard has the AI stationary and alert] : pass
- [State_GoToLocationAware uses movement] : pass
- [State_Attack shoots the player and follows] : pass
- [State_Wander randomly moves until it sees player] : pass
- [Place AI in the scene and have them navigate through buildings] : pass
- [When enemies sees the player they switch to the attack state] : pass
- [When enemies lose sight of the player they go to last known location] : pass