Mysterious Steam

Train Controller Test Plan

By: Andrew Beers

1. **Create Train**
2. **Create Panel**
3. **Increase Speed**
4. **Decrease Speed**
5. **Light On**
6. **Light Off**
7. **Passenger On**
8. **Passenger Off**
9. **Door Open**
10. **Door Close**
11. **Failure Handling**
12. **Emergency Brake**

**Test Vectors:**

1. Create Train(Input: Train Controller Created; Output: SUCCESS/FAIL)
   1. CTC Office creates instance of Train Controller
   2. Train Controller creates instance of Train Model
   3. Check return value if the train was created or not

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Andrew Beers | Caleb Dusenbery | Pass | *No returns to check* |

1. Create Panel(Input: Button Click; Output: GUI)
   1. CTC Office creates instance of Train Controller
   2. Train controller creates an instance of Panel
   3. Panel creates all buttons for the GUI
   4. GUI displayed

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Andrew Beers | Caleb Dusenbery | Pass | *None* |

1. Increase Speed(Input: Button Click; Output: Power)
   1. User clicks Increase Speed
   2. Train Controller calculates power needed for speed increase
   3. Train Controller sends power to Train model

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Andrew Beers | Caleb Dusenbery | Pass | *Not an increase button but a text box to set a desired speed* |

1. Decrease Speed(Input: Button Click; Output: Brake amount)
   1. User clicks Decrease Speed
   2. Train Controller calculates amount of braking needed for speed decrease
   3. Train Controller sends brake amount to Train Model

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Andrew Beers | Caleb Dusenbery | Pass | *Not a decrease button but a text box to set a desired speed* |

1. Light On(Input: Button Click; Output: Light ON)
   1. User clicks Light on button
   2. Train Controller sets the light to be on

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Andrew Beers | Caleb Dusenbery | Pass | *Not a “light on” button but a switch* |

1. Light Off(Input: Button Click; Output: Light OFF)
   1. User clicks Light off button
   2. Train Controller sets the light to be off

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Andrew Beers | Caleb Dusenbery | Pass | *Not a “light off” button but a switch* |

1. Passenger On(Input: Train is at station; Output: Passenger on value)
   1. Train Model is at a station
   2. Train Controller gets number of passengers on the train
   3. Train Controller gets maximum allowed number of passengers
   4. Train Controller gets a random number from maxPassengers - currPassengers
   5. Train Controller sets passengers on as this random number

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Andrew Beers | Caleb Dusenbery | Pass | *Not distinct from passengers off* |

1. Passenger Off(Input: Train is at station; Output: Passenger off value)
   1. Train enters a station
   2. Train Controller gets current number of passengers on the Train Model
   3. Train Controller gets a random number from the current amount of passengers
   4. Train Controller sets this random number as the passengers off
   5. Train Controller subtracts current passenger count and passengers off
   6. Train Controller sets the new current passenger count to the Train Model

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Andrew Beers | Caleb Dusenbery | Pass | *Not distinct from passengers on* |

1. Door Open(Input: Button Click; Output: Door OPEN)
   1. User Clicks Door open button
   2. Train controller checks if it is safe to open the door
   3. Door on train opens

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Andrew Beers | Caleb Dusenbery | Pass | *Not a “door open” button but a switch* |

1. Door Close(Input: Button Click; Output: Door CLOSE)
   1. User Clicks Door close button
   2. Train controller checks if it is safe to close the door
   3. Door on train closes

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Andrew Beers | Caleb Dusenbery | Pass | *Not a “door close” button but a switch* |

1. Handle Failure(Input: Failure; Output: Stop Train)
   1. Train Model Signals a failure
   2. Alert CTC of the failure to close track that the train is on
   3. Send a brake signal to stop the train

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Andrew Beers | Caleb Dusenbery | Pass | *Failure signaled by ui* |

1. Emergency Brake(Input: Button Click; Output: SUCCESS/FAIL)
   1. User clicks Emergency Brake button
   2. Train Controller sends maximum allowed deceleration value to Train Model
   3. Return a success or failure value

|  |  |  |  |
| --- | --- | --- | --- |
| **Author** | **Tester** | **Pass/Fail** | **Comment** |
| Andrew Beers | Caleb Dusenbery | Pass | *Must reset after calling this* |