Project 3

TriCode

Members:
Jessica Haines
K'Lani McClean
Johnathon Moore

Objective

- Work as a group to create a fifteen puzzle game using html, css, and javascript
- There should also be at least four extra features

Roles/Responsibilities

- Everyone was a programmer and a tester
- Jessica
  - The base game (everything except the extra features)
- K'Lani
  - Extra features
- Johnathon
  - Extra features, compile code

Key Features

- Location
- Movable
- Is_movable
- Move
- Shuffle
- Apply_hover
- Set_default

Location

- Made a function that creates objects with an x and y coordinate
- This way the locations can be hard coded and reused elsewhere

- function tlocation(x,y)
- {
  - this.x = x;
  - this.y = y;
- }
- var r1_c1 = new tlocation(0,0);

Movable

- Made a css class to handle the hover functionality
- When a movable tile is hovered over:
  - The border changes to red
  - The number is changed to green and is underlined in green, and
  - The cursor is changed to a hand

- .movable:hover{
  - border: 2px solid red;
  - color: #006600;
  - text-decoration: underline;
  - text-decoration-color: #006600;
  - cursor: grab;
- }

Is_movable

- This function checks to see if the tile is a neighbor of the empty space and if so returns true
- This is only a portion of the function
- This is the logic for checking if the tile is to the left or above the empty space

- if(//check left
  - empty.x - 100 == parseInt(tile.style.left) &&
  - empty.y == parseInt(tile.style.top))
- {
  - return true;
- }
-

Move

- This function first runs the is_movable function
- If the tile can be moved the locations of the tile and the empty space are swapped
- Afterwards is runs the apply_hover function

- function move(tile)
- {
  - if(is_movable(tile)){
    - let temp = new tlocation
    - temp.x = parseInt(tile.style.left);
    - temp.y = parseInt(tile.style.top);
    -
    - tile.style.left = empty.x + 'px';

Shuffle

- This function goes through each tile and if it is a neighbor to the empty space it is added to an array
- Afterwards a random tile in that array is chosen and moved
- This is done 1000 times

- function shuffle(){
  - for(let i = 0; i < 1000; i++){
    - //find neighbors of empty space
    - let movable_tile_list = [];
    - let tile_list = document.querySelectorAll(" .tile")
    -
    - for (let key in tile_list){
    - let tile = tile_list[key];
    - if(is_movable(tile)){

Apply_hover

- This function goes through every tile and:
- If it is movable it adds the movable class to it
- If it is not movable and has a class the movable class is removed from it

- function apply_hover(){
  - let movable_tile_list = [];
  - let tile_list = document.querySelectorAll(".tile")
  - for (let key in tile_list){
    - let tile = tile_list[key];
    - if(is_movable(tile)){
      - // attach move styles
      - tile.classList.add("movable");

Set_default

- This function:
  - Sets the initial locations and image positions for all of the tiles
  - Sets the location of the empty space, and
  - Runs the apply_hover function
- Putting this into a function instead of having it in main allows for it to be called

- function set_default()
- {
  - tile1.style.left = r1_c1.x + 'px';
  - tile1.style.top = r1_c1.y + 'px';
  - tile1.style.backgroundPositi on = "0px 0px";
  - …
  - tile15.style.left = r4 c3.x +

Extra Features

- Timer
- Move Counter
- Win Check

Timer

- This is how we implemented the timer
- The timer is started after the game starts

- function startTimer(duration, display){
  - var timer = duration, mins, secs;
  - setInterval(function(){
    - mins = parseInt(timer / 60, 10);
    - secs = parseInt(timer % 60, 10);
    -

Move Counter

- A movecounts variable was made and is incremented every time the player makes a move
- The new count is then displayed on the page

- movecounts++;
- Document.getElementById (“movecount”).innerHTML = movecounts;

Win Check

- This function checks to see how many of the tiles are in the proper location
- If they are all in the right places it gives a congrats message
- If not it gives a keep trying message

- function checkWin(){
  - Var final=0;
  - if(r1_c1.x == 0 && r1_c1.y == 0){
    - Final++;
  - }
  - …
  - if(final == 16){
    - document.getElementByID(“gamewin”).innerHTML =

Scrum Framework

- Our approach to this project was about the same as the previous two
- We each worked separately but communicated frequently, shared our code, and asked questions or for help when needed

Test Case

- When testing something:
  - Make sure it runs
  - Make sure it looks correct
    - Check that it performs correctly given expected input both correct and incorrect
      - ie. Clicking on a tile that is not a neighbor of the empty space should not cause it to move, and vice versa.
  - Check how it behaves for potentially unexpected input
    - ie. When is_movable() was called from inside the shuffle function some unexpected values were passed in and this had to be worked around.
  - If a bug is found

Up Next

- Code and Demonstration