A2

Authors: Kanu Kanu kjk225

Summary

Challenges. Understanding the assignment was the hardest part. Once I understood how to start I think I got the picture. Also implementing the data structure to represent the json object was also very challenging.

Major issues.

There are no major issues in my implementation.

Design issues. At first I felt there were many design choices, however when I got to actually designing I realized that I did not have as many options if I wanted to keep my implementation efficient.

Known problems. After I raise Illegal, my code exits.

Specification

We were not asked to produce any specifications for this assignment.

Design and Implementation

Modules. We did not need to break any programming tasks down into
modules for this assignment.

Architecture. Because we did not need to define modules, there is no
architecture to be described here.

Code design. There were not many code design issues.

Programming. My code was written by me

Testing

Test plan.

```
let tests =
[
  "max" >:: (fun _ -> assert_equal 11111 (j |> init_state |>
max_score));
  "score" >:: (fun _ -> assert_equal 10001 (j |> init_state |>
score));
  "turns" >:: (fun _ -> assert_equal 0 (j |> init_state |>
turns));
  "room?" >:: (fun _ -> assert_equal "room1" (j |> init_state |>
current_room_id));
  "inventory" >:: (fun _ -> assert_equal ["white hat"] (j |>
init_state |> inv));
  "visited" >:: (fun _ -> assert_equal ["room1"] (j |>
init_state |> visited));
  "locations" >:: (fun _ -> assert_equal [("black hat",
"room1"); ("red hat", "room1")] (j |> init_state |> locations));
]

let state_tests =
[

let roomslist = j |> member "rooms" |> to_list in
let allrooms = buildrooms roomslist [] in
let roomtree = make_roomtree allrooms Empty in

let itemslist = j |> member "items" |> to_list in
let allitems = builditems itemslist [] in
let itemtree = make_itemtree allitems Empty in

let initialstate = {inventory = ["white hat"]; room = "room1" ;
                         points = 10001; turns =0; max_score=
11111; visited = ["room1"];
                         locations = [("black hat",
"room1");("red hat", "room1")];
                         roomtree = roomtree;
                         itemtree = itemtree;
                         } in

  "initial_state"  >:: (fun _ -> assert_equal initialstate
(init_state j));
]

**Test results.** My code passed all of our test cases. For
brevity of
this example document, these test cases are omitted here.
However, note that most of my testing was done through play
testing

Known problems
--------------

My code exits after I raise Illegal

Comments
--------
```