BigZaphod / AStar

AStar / AStar.h Branch: master ▼

Copy path Find file



BigZaphod Messed with some data types and added goalNode as a parameter to the ...

128d35b on 20 Apr 2012

1 contributor

72 lines (54 sloc) 3.65 KB

/* 1

Copyright (c) 2012, Sean Heber. All rights reserved.

3 4

2

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

5 6 7

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

8 9 10

11

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

12 14

15

3. Neither the name of Sean Heber nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

17 18

19

21 22

23

24

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL SEAN HEBER BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF

```
LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE
26
     OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF
27
     ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
      */
28
29
     #ifndef AStar h
31
     #define AStar h
     #include <stdlib.h>
34
     typedef struct __ASNeighborList *ASNeighborList;
     typedef struct ASPath *ASPath;
38
     typedef struct {
                                                                                                          // the size of the structure
39
         size t nodeSize;
                                                                                                         // add nodes to the neighbor
         void
                 (*nodeNeighbors)(ASNeighborList neighbors, void *node, void *context);
41
         float
                (*pathCostHeuristic)(void *fromNode, void *toNode, void *context);
                                                                                                         // estimated cost to transit
42
         int
                 (*earlyExit)(size_t visitedCount, void *visitingNode, void *goalNode, void *context); // early termination, return
                 (*nodeComparator)(void *node1, void *node2, void *context);
43
         int
                                                                                                          // must return a sort order
44
     } ASPathNodeSource;
45
46
     // use in the nodeNeighbors callback to return neighbors
    void ASNeighborListAdd(ASNeighborList neighbors, void *node, float edgeCost);
47
48
49
     // if goalNode is NULL, it searches the entire graph and returns the cheapest deepest path
50
    // context is optional and is simply passed through to the callback functions
51
     // startNode and nodeSource is required
     // as a path is created, the relevant nodes are copied into the path
     ASPath ASPathCreate(const ASPathNodeSource *nodeSource, void *context, void *startNode, void *goalNode);
54
     // paths created with ASPathCreate() must be destroyed or else it will leak memory
     void ASPathDestroy(ASPath path);
     // if you want to make a copy of a path result, this function will do the job
59
     // you must call ASPathDestroy() with the resulting path to clean it up or it will cause a leak
```

```
ASPath ASPathCopy(ASPath path);
60
61
    // fetches the total cost of the path
62
    float ASPathGetCost(ASPath path);
63
64
    // fetches the number of nodes in the path
65
    size_t ASPathGetCount(ASPath path);
66
67
    // returns a pointer to the given node in the path
68
    void *ASPathGetNode(ASPath path, size_t index);
69
    #endif
71
```