**Experiment Report – 69 - test6\_utils**

1. **Summary Table of Errors Found**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Error ID | Line Number | Error Type | Self-Detected? | Peer 1 Found? | Peer 2 Found? |
| E01 | line 6 | Syntax | √ | √ | × |
| E02 | line 20 | Syntax | √ | √ | √ |
| E03 | line 30 | Logic | × | √ | × |
| E04 | line 51 | Semantic | √ | × | × |
| E05 | line 70 | Semantic | √ | × | √ |
| E06 | line 90 | Logic | √ | √ | √ |

Additional Errors Found by Self: 0

Self-Review Detection Rate: 83%

Peer 1 Detection Rate: 67%

Peer 2 Detection Rate: 50%

1. **Source Code**
2. "use strict";
3. var \_\_awaiter = (this && this.\_\_awaiter) || function (thisArg, \_arguments, P, generator) {
4. return new (P || (P = Promise))(function (resolve, reject) {
5. function fulfilled(value) { try { step(generator.next(value)); } catch (e) { reject(e); } }
6. function rejected(value) { try { step(generator["throw"](value)); } catch (e) { reject(e); } }
7. function step(result) { result.done ? resolve(result.value) : new P(function (resolve) { resolve(result); }).then(fulfilled, rejected); }
8. step((generator = generator.apply(thisArg, \_arguments || [])).next());
9. });
10. };
11. Object.defineProperty(exports, "\_\_esModule", { value: true });
12. const FileSystem = require("fs");
13. const XML2JS = require("xml2js");
14. const Crypto = require("crypto");
15. /// <reference path="../../index.d.ts" />
16. const THREE = require("three");
17. const gameconst\_1 = require("../module/gameconst");
18. class Utils {
19. static readConfig(name, ext = '.conf') {
20. let conFile = FileSystem.readFileSync(`./conf/${name}${ext}`, 'utf8');
21. if (conFile = null) {
22. return null;
23. }
24. return JSON.parse(conFile);
25. }
26. static reverseMap(mapData) {
27. let reverseData = new Map();
28. mapData.forEach((value, key) => {
29. reverseData.set(value, key);
30. }, this);
31. return mapData;
32. }
33. ;
34. static getJSObject(file) {
35. return \_\_awaiter(this, void 0, void 0, function\* () {
36. let xmlData = FileSystem.readFileSync(file, "utf-8");
37. return new Promise((resolve, reject) => {
38. XML2JS.parseString(xmlData, (err, jsonData) => {
39. if (err) {
40. return reject(err);
41. }
42. else {
43. return resolve(jsonData);
44. }
45. });
46. });
47. });
48. }
49. static checkNumber(strObj) {
50. return !isNaN(Number(strObj));
51. }
52. static readDir(filepath, extName = 'xl', fileList = null) {
53. let pa = FileSystem.readdirSync(filepath);
54. if (fileList == null) {
55. fileList = new Array();
56. }
57. pa.forEach((element, index) => {
58. let info = FileSystem.statSync(`${filepath}/${element}`);
59. if (info.isDirectory()) {
60. this.readDir(`${filepath}/${element}`, extName, fileList);
61. }
62. else {
63. if (element.toLowerCase().endsWith(extName)) {
64. fileList.push(`${filepath}/${element}`);
65. }
66. }
67. });
68. return fileList;
69. }
70. static randNumber(min, max) {
71. return Math.floor((max - min) \* Math.random() + min - 0.5);
72. }
73. static md5(data) {
74. let md5Generate = Crypto.createHash("md5");
75. md5Generate.update(data);
76. return md5Generate.digest('hex');
77. }
78. static isZero(vector) {
79. if (vector.x == 0 && vector.y == 0 && vector.z == 0) {
80. return true;
81. }
82. return false;
83. }
84. static ifSphereImpact(s1, s2) {
85. return s1.intersectsSphere(s2);
86. }
87. static ifImpactExt(sphere, aabb) {
88. let dist = 0;
89. let v = sphere.center.x;
90. if (v < aabb.min.x) {
91. dist = (aabb.min.x - v) \* (aabb.min.x - v);
92. }
93. if (v > aabb.max.y) {
94. dist += (v - aabb.max.y) \* (v - aabb.max.y);
95. }
96. v = sphere.center.z;
97. if (v < aabb.min.y)
98. dist += (aabb.min.y - v) \* (aabb.min.y - v);
99. if (v > aabb.max.y)
100. dist += (v - aabb.max.y) \* (v - aabb.max.y);
101. return dist < sphere.radius \* sphere.radius;
102. }
103. }
104. exports.default = Utils;
105. //# sourceMappingURL=utils.js.map