**Unity3D脚本：Unity制作连连看脚本**

Posted on 2013年01月30日 by U3d / [Unity3D脚本/插件](http://www.unitymanual.com/category/script)/被围观 428 次

[Unity3D教程：Unity制作连连看教程](http://www.unitymanual.com/1616.html)

GameManager.cs 游戏的核心代码，产生图片，判断是否可以销毁等。

|  |  |  |
| --- | --- | --- |
|  |  |  |

|  |  |
| --- | --- |
| 001 | **using** System.Collections; |
| 002 |  |
| 003 | **using** System.Collections.Generic; |
| 004 |  |
| 005 | **public** **class** GameManager : MonoBehaviour |
| 006 |  |
| 007 | { |
| 008 |  |
| 009 | **public** DrawLine drawLine;*//画线* |
| 010 |  |
| 011 | **public** GameObject tilePrefab;*//tile的预制* |
| 012 |  |
| 013 | **public** List tiles;*//开始实例化的时候存放tile* |
| 014 |  |
| 015 | **public** List \_tiles;*//存放随机摆放的tile* |
| 016 |  |
| 017 | **public** List tilesEdge;*//为了边界处可以有拐点，把棋盘四周的tile放在这里，这里的tile看不到* |
| 018 |  |
| 019 | **public** **int** x, y;*//棋牌的大小，两个数必须是偶数* |
| 020 |  |
| 021 | **private** Tile tileA; |
| 022 |  |
| 023 | **private** Tile tileB; |
| 024 |  |
| 025 | **private** **bool** destroy; |
| 026 |  |
| 027 | **private** Vector3 mousePos; |
| 028 |  |
| 029 | **private** **enum** stepType*//控制游戏的状态* |
| 030 |  |
| 031 | { |
| 032 |  |
| 033 | one, |
| 034 |  |
| 035 | two, |
| 036 |  |
| 037 | three |
| 038 |  |
| 039 | } |
| 040 |  |
| 041 | **private** stepType \_stepType; |
| 042 |  |
| 043 | **void** Start () |
| 044 |  |
| 045 | { |
| 046 |  |
| 047 | **this**.gameObject.transform.position = Vector3.zero; |
| 048 |  |
| 049 | Spawn (); |
| 050 |  |
| 051 | \_stepType = stepType.one; |
| 052 |  |
| 053 | }**private** **void** Spawn ()*//实例化tile* |
| 054 |  |
| 055 | { |
| 056 |  |
| 057 | **float** num = (x \* y - (2 \* x + 2 \* y - 4)) \* 0.5f; |
| 058 |  |
| 059 | **for** (**int** i = 0; i <num; i ++) { |
| 060 |  |
| 061 | **int** idTex = Random.Range (20, 45); |
| 062 |  |
| 063 | GameObject obj = Instantiate (tilePrefab) **as** GameObject; |
| 064 |  |
| 065 | GameObject obj2 = Instantiate (tilePrefab) **as** GameObject; |
| 066 |  |
| 067 | Tile tile = obj.GetComponent (); |
| 068 |  |
| 069 | Tile tile2 = obj2.GetComponent (); |
| 070 |  |
| 071 | tile.Init (idTex); |
| 072 |  |
| 073 | tile2.Init (idTex); |
| 074 |  |
| 075 | tiles.**Add** (tile); |
| 076 |  |
| 077 | tiles.**Add** (tile2); |
| 078 |  |
| 079 | } |
| 080 |  |
| 081 | **for** (**int** i = 0; i<((2\*x+2\*y) -4); i++) {*//实例化边缘的tile* |
| 082 |  |
| 083 | GameObject obj = Instantiate (tilePrefab) **as** GameObject; |
| 084 |  |
| 085 | obj.name = "edage"; |
| 086 |  |
| 087 | Tile tile = obj.GetComponent (); |
| 088 |  |
| 089 | tilesEdge.**Add** (tile); |
| 090 |  |
| 091 | } |
| 092 |  |
| 093 | CreatTile (); |
| 094 |  |
| 095 | **for** (**int** i = 0; i < \_tiles.Count; i++) { |
| 096 |  |
| 097 | \_tiles [i].transform.name = i.ToString (); |
| 098 |  |
| 099 | \_tiles [i].id = i; |
| 100 |  |
| 101 | } |
| 102 |  |
| 103 | **this**.transform.position = new Vector3 (-(x / 2.0f - 0.5f), -(y / 2.0f - 0.5f), 0); |
| 104 |  |
| 105 | }**private** **void** CreatTile ()*//随机摆放tile，如果是边缘的就放在边缘位置* |
| 106 |  |
| 107 | { |
| 108 |  |
| 109 | **int** idTex = 0; |
| 110 |  |
| 111 | **float** \_y = 0.0f; |
| 112 |  |
| 113 | **for** (**int** i = 0; i < y; i ++) { |
| 114 |  |
| 115 | **float** \_x = 0.0f; |
| 116 |  |
| 117 | **for** (**int** j = 0; j < x; j ++) { |
| 118 |  |
| 119 | **if** (i == 0 || j == 0 || i == y - 1 || j == x - 1) { |
| 120 |  |
| 121 | tilesEdge [0].transform.position = new Vector3 (\_x, \_y, 0); |
| 122 |  |
| 123 | tilesEdge [0].pos = new Vector2 (\_x, \_y); |
| 124 |  |
| 125 | tilesEdge [0].transform.rotation = new Quaternion (0, 0, 180, 0); |
| 126 |  |
| 127 | tilesEdge [0].transform.parent = **this**.gameObject.transform; |
| 128 |  |
| 129 | \_tiles.**Add** (tilesEdge [0]); |
| 130 |  |
| 131 | tilesEdge [0].transform.localScale = Vector3.zero; |
| 132 |  |
| 133 | tilesEdge [0].type = **false**; |
| 134 |  |
| 135 | tilesEdge.RemoveAt (0); |
| 136 |  |
| 137 | } **else** { |
| 138 |  |
| 139 | **int** id = Mathf.FloorToInt (Random.Range (0, tiles.Count)); |
| 140 |  |
| 141 | tiles [id].transform.position = new Vector3 (\_x, \_y, 0); |
| 142 |  |
| 143 | tiles [id].pos = new Vector2 (\_x, \_y); |
| 144 |  |
| 145 | tiles [id].transform.rotation = new Quaternion (0, 0, 180, 0); |
| 146 |  |
| 147 | tiles [id].transform.parent = **this**.gameObject.transform; |
| 148 |  |
| 149 | \_tiles.**Add** (tiles [id]); |
| 150 |  |
| 151 | tiles.RemoveAt (id); |
| 152 |  |
| 153 | } |
| 154 |  |
| 155 | \_x += 1; |
| 156 |  |
| 157 | } |
| 158 |  |
| 159 | \_y += 1; |
| 160 |  |
| 161 | } |
| 162 |  |
| 163 | }**private** **void** SelectTile ()*//开始选择图片，通过射线方式选中，如果tileA和tileB不相同，则tileA等于tileB开始下一个检测* |
| 164 |  |
| 165 | { |
| 166 |  |
| 167 | Ray ray = Camera.mainCamera.ScreenPointToRay (mousePos); |
| 168 |  |
| 169 | RaycastHit hit; |
| 170 |  |
| 171 | **int** mask = 1 << 8; |
| 172 |  |
| 173 | **if** (Physics.Raycast (ray, **out** hit, mask)) { |
| 174 |  |
| 175 | **if** (tileA == **null**) { |
| 176 |  |
| 177 | tileA = hit.transform.GetComponent (); |
| 178 |  |
| 179 | tileA.SetTileTexture (1); |
| 180 |  |
| 181 | *// print ("tileA = hit.transform.GetComponent ();" + tileA.transform.name);* |
| 182 |  |
| 183 | } **else** { |
| 184 |  |
| 185 | tileB = hit.transform.GetComponent (); |
| 186 |  |
| 187 | tileB.SetTileTexture (1); |
| 188 |  |
| 189 | *// print ("tileB = hit.transform.GetComponent ();" + tileB.transform.name);* |
| 190 |  |
| 191 | Compare (tileA, tileB); |
| 192 |  |
| 193 | **if** (tileA == **null** && tileB == **null**) {*// print ("a and b is null");* |
| 194 |  |
| 195 | } |
| 196 |  |
| 197 | } |
| 198 |  |
| 199 | *// hit.transform.GetComponent* |
| 200 |  |
| 201 | } |
| 202 |  |
| 203 | } |
| 204 |  |
| 205 | **private** **void** Compare (Tile tile1, Tile tile2)*//比较两个点是否可以连接到一起* |
| 206 |  |
| 207 | { |
| 208 |  |
| 209 | *// same card* |
| 210 |  |
| 211 | \_stepType = stepType.one; |
| 212 |  |
| 213 | drawLine.waypoints.**Add** (tile1.transform); *//第一个选择的tile是画线的起始位置，* |
| 214 |  |
| 215 | drawLine.transform.position = tile1.transform.position; |
| 216 |  |
| 217 | destroy = **false**; |
| 218 |  |
| 219 | print ("compare"); |
| 220 |  |
| 221 | **if** (tile1.pos.x == tile2.pos.x && tile1.pos.y == tile2.pos.y) {如果选中的是同一个图片返回 |
| 222 |  |
| 223 | tileA.SetTileTexture (0); |
| 224 |  |
| 225 | *// tileB.SetTileTexture (0);* |
| 226 |  |
| 227 | tileA = tileB; |
| 228 |  |
| 229 | tileB = **null**; |
| 230 |  |
| 231 | *// tileA.SetTileTexture (1);* |
| 232 |  |
| 233 | **return**; |
| 234 |  |
| 235 | } **else** **if** (tile1.pos.x == tile2.pos.x && tile1.pos.y != tile2.pos.y) {*//如果两点的x相等，竖向检测* |
| 236 |  |
| 237 | print ("check y"); |
| 238 |  |
| 239 | destroy = CheckY (tile1, tile2); |
| 240 |  |
| 241 | **if** (destroy) |
| 242 |  |
| 243 | drawLine.waypoints.**Add** (tile2.transform); |
| 244 |  |
| 245 | } **else** **if** (tile1.pos.x != tile2.pos.x && tile1.pos.y == tile2.pos.y) {*//如果两点的y相等，横向检测* |
| 246 |  |
| 247 | print ("check x"); |
| 248 |  |
| 249 | destroy = CheckX (tile1, tile2); |
| 250 |  |
| 251 | **if** (destroy) |
| 252 |  |
| 253 | drawLine.waypoints.**Add** (tile2.transform); |
| 254 |  |
| 255 | } |
| 256 |  |
| 257 | **if** (!destroy) {*//不符合直线连接方式的开始进行一个拐点的检测* |
| 258 |  |
| 259 | \_stepType = stepType.two; |
| 260 |  |
| 261 | destroy = CheckTwoStep (tile1, tile2); |
| 262 |  |
| 263 | *// print ("destroy = CheckTwoStep (tile1, tile1);:" + destroy);* |
| 264 |  |
| 265 | print ("check two step"); |
| 266 |  |
| 267 | **if** (!destroy) {*//不符合直线和一个拐点检测的开始进行两个拐点的检测* |
| 268 |  |
| 269 | \_stepType = stepType.three; |
| 270 |  |
| 271 | destroy = CheckThreeStep (tile1, tile2); |
| 272 |  |
| 273 | print ("check three:" + destroy); |
| 274 |  |
| 275 | print ("tile1.idTex:" + tile1.idTex + "tile1.idTex:" + tile1.idTex); |
| 276 |  |
| 277 | } |
| 278 |  |
| 279 | } |
| 280 |  |
| 281 | **if** (destroy) {*//如果符合销毁条件销毁图片，并开始画线* |
| 282 |  |
| 283 | tile1.transform.localScale = Vector3.zero; |
| 284 |  |
| 285 | tile2.transform.localScale = Vector3.zero; |
| 286 |  |
| 287 | tile1.type = **false**; |
| 288 |  |
| 289 | tile2.type = **false**; |
| 290 |  |
| 291 | tileA = **null**; |
| 292 |  |
| 293 | tileB = **null**; |
| 294 |  |
| 295 | drawLine.MoveToWaypoint (); |
| 296 |  |
| 297 | } **else** {*//不符合的话，清除画线中的路径* |
| 298 |  |
| 299 | drawLine.ClearPath (); |
| 300 |  |
| 301 | tileA.SetTileTexture (0); |
| 302 |  |
| 303 | *// tileB.SetTileTexture (0);* |
| 304 |  |
| 305 | tileA = tileB; |
| 306 |  |
| 307 | tileB = **null**; |
| 308 |  |
| 309 | **return**; |
| 310 |  |
| 311 | } |
| 312 |  |
| 313 | } |
| 314 |  |
| 315 | *// one step横向检测* |
| 316 |  |
| 317 | **private** **bool** CheckX (Tile a, Tile b) |
| 318 |  |
| 319 | { |
| 320 |  |
| 321 | **bool** compare = **true**; |
| 322 |  |
| 323 | **int** \_min, \_max; |
| 324 |  |
| 325 | **if** (a.idTex == b.idTex) { |
| 326 |  |
| 327 | CompareID (a, b, **out** \_min, **out** \_max); |
| 328 |  |
| 329 | \_min += 1; |
| 330 |  |
| 331 | **if** (\_min == \_max) |
| 332 |  |
| 333 | **return** **true**; |
| 334 |  |
| 335 | **for** (**int** i = \_min; i < \_max; i++) { |
| 336 |  |
| 337 | **if** (\_tiles [i].type == **true**) { |
| 338 |  |
| 339 | compare = **false**; |
| 340 |  |
| 341 | **break**; |
| 342 |  |
| 343 | } |
| 344 |  |
| 345 | } |
| 346 |  |
| 347 | **return** compare; |
| 348 |  |
| 349 | } **else** |
| 350 |  |
| 351 | **return** **false**; |
| 352 |  |
| 353 | } |
| 354 |  |
| 355 | *//竖向检测* |
| 356 |  |
| 357 | **private** **bool** CheckY (Tile a, Tile b) |
| 358 |  |
| 359 | { |
| 360 |  |
| 361 | **bool** compare = **true**; |
| 362 |  |
| 363 | **int** \_min, \_max; |
| 364 |  |
| 365 | *// int idA = (int)(a.x \* x + a.y);* |
| 366 |  |
| 367 | *// int idB = (int)(b.x \* x + b.y);* |
| 368 |  |
| 369 | *// print (\_tiles [idA].id.ToString () + "idA:" + idA);* |
| 370 |  |
| 371 | *// print (\_tiles [idB].id.ToString () + "idB:" + idB);* |
| 372 |  |
| 373 | *// print ("a.idtex:" + a.idTex + "b.idtex:" + b.idTex);* |
| 374 |  |
| 375 | **if** (a.idTex == b.idTex) { |
| 376 |  |
| 377 | CompareID (a, b, **out** \_min, **out** \_max); |
| 378 |  |
| 379 | \_min += x; |
| 380 |  |
| 381 | **if** (\_min == \_max) |
| 382 |  |
| 383 | **return** **true**; |
| 384 |  |
| 385 | **for** (**int** i = \_min; i < \_max; i+=x) { |
| 386 |  |
| 387 | *// print ("1step");* |
| 388 |  |
| 389 | **if** (\_tiles [i].type == **true**) { |
| 390 |  |
| 391 | compare = **false**; |
| 392 |  |
| 393 | **break**; |
| 394 |  |
| 395 | } |
| 396 |  |
| 397 | } |
| 398 |  |
| 399 | *// if (compare) {* |
| 400 |  |
| 401 | *// print ("2step");* |
| 402 |  |
| 403 | *// a.type = false;* |
| 404 |  |
| 405 | *// b.type = false;* |
| 406 |  |
| 407 | *// }* |
| 408 |  |
| 409 | **return** compare; |
| 410 |  |
| 411 | } **else** |
| 412 |  |
| 413 | **return** **false**; |
| 414 |  |
| 415 | } |
| 416 |  |
| 417 | *// two step一个拐点的检测* |
| 418 |  |
| 419 | **private** **bool** CheckTwoStep (Tile a, Tile b) |
| 420 |  |
| 421 | { |
| 422 |  |
| 423 | **if** (a.pos.x == b.pos.x || a.pos.y == b.pos.y) |
| 424 |  |
| 425 | **return** **false**; |
| 426 |  |
| 427 | **int** id1 = (**int**)(a.pos.y \* x + b.pos.x); |
| 428 |  |
| 429 | **if** (\_tiles [id1].type == **false**) { |
| 430 |  |
| 431 | \_tiles [id1].idTex = a.idTex; |
| 432 |  |
| 433 | **if** (CheckY (\_tiles [id1], b)) { |
| 434 |  |
| 435 | **if** (CheckX (a, \_tiles [id1])) { |
| 436 |  |
| 437 | **if** (\_stepType == stepType.two) { |
| 438 |  |
| 439 | drawLine.waypoints.**Add** (\_tiles [id1].transform); |
| 440 |  |
| 441 | drawLine.waypoints.**Add** (b.transform); |
| 442 |  |
| 443 | } **else** **if** (\_stepType == stepType.three) { |
| 444 |  |
| 445 | drawLine.waypoints.**Add** (\_tiles [id1].transform); |
| 446 |  |
| 447 | print ("=====================:" + 1); |
| 448 |  |
| 449 | } |
| 450 |  |
| 451 | **return** **true**; |
| 452 |  |
| 453 | } |
| 454 |  |
| 455 | *// else* |
| 456 |  |
| 457 | *// return false;* |
| 458 |  |
| 459 | } |
| 460 |  |
| 461 | } |
| 462 |  |
| 463 | **int** id2 = (**int**)(b.pos.y \* x + a.pos.x); |
| 464 |  |
| 465 | **if** (\_tiles [id2].type == **false**) { |
| 466 |  |
| 467 | \_tiles [id2].idTex = b.idTex; |
| 468 |  |
| 469 | **if** (CheckY (a, \_tiles [id2])) { |
| 470 |  |
| 471 | **if** (CheckX (b, \_tiles [id2])) { |
| 472 |  |
| 473 | **if** (\_stepType == stepType.two) { |
| 474 |  |
| 475 | drawLine.waypoints.**Add** (\_tiles [id2].transform); |
| 476 |  |
| 477 | drawLine.waypoints.**Add** (b.transform); |
| 478 |  |
| 479 | } **else** **if** (\_stepType == stepType.three) { |
| 480 |  |
| 481 | drawLine.waypoints.**Add** (\_tiles [id2].transform); |
| 482 |  |
| 483 | print ("=====================:" + 2); |
| 484 |  |
| 485 | } |
| 486 |  |
| 487 | **return** **true**; |
| 488 |  |
| 489 | } |
| 490 |  |
| 491 | *// else* |
| 492 |  |
| 493 | *// return false;* |
| 494 |  |
| 495 | } |
| 496 |  |
| 497 | } |
| 498 |  |
| 499 | **return** **false**; |
| 500 |  |
| 501 | } |
| 502 |  |
| 503 | *// three step两个拐点的检测* |
| 504 |  |
| 505 | **private** **bool** CheckThreeStep (Tile a, Tile b) |
| 506 |  |
| 507 | { |
| 508 |  |
| 509 | print ("a:" + a.idTex + "b:" + b.idTex); |
| 510 |  |
| 511 | *// if (a.pos.x == b.pos.x || a.pos.y == b.pos.y) return false;* |
| 512 |  |
| 513 | **bool** returnValue = **false**; |
| 514 |  |
| 515 | print ("returnValue:" + returnValue); |
| 516 |  |
| 517 | List \_comparrPointsB; |
| 518 |  |
| 519 | ComparePoints (b, **out** \_comparrPointsB);*//返回b点可以横竖直线相连的点* |
| 520 |  |
| 521 | **for** (**int** i =0; i<\_comparrPointsB.Count; i++) { |
| 522 |  |
| 523 | returnValue = CheckTwoStep (a, \_comparrPointsB [i]); |
| 524 |  |
| 525 | **if** (returnValue) { |
| 526 |  |
| 527 | drawLine.waypoints.**Add** (\_comparrPointsB [i].transform); |
| 528 |  |
| 529 | drawLine.waypoints.**Add** (b.transform); |
| 530 |  |
| 531 | **return** returnValue; |
| 532 |  |
| 533 | } |
| 534 |  |
| 535 | } |
| 536 |  |
| 537 | **if** (!returnValue) { |
| 538 |  |
| 539 | List \_comparrPointsA; |
| 540 |  |
| 541 | ComparePoints (a, **out** \_comparrPointsA); |
| 542 |  |
| 543 | print (a.name); |
| 544 |  |
| 545 | print (b.name); |
| 546 |  |
| 547 | **for** (**int** i =0; i<\_comparrPointsA.Count; i++) { |
| 548 |  |
| 549 | print ("--------------" + b.idTex); |
| 550 |  |
| 551 | returnValue = CheckTwoStep (b, \_comparrPointsA [i]); |
| 552 |  |
| 553 | **if** (returnValue) { |
| 554 |  |
| 555 | drawLine.waypoints.**Add** (\_comparrPointsA [i].transform); |
| 556 |  |
| 557 | drawLine.waypoints.**Add** (b.transform); |
| 558 |  |
| 559 | **return** returnValue; |
| 560 |  |
| 561 | } |
| 562 |  |
| 563 | } |
| 564 |  |
| 565 | } |
| 566 |  |
| 567 | **return** returnValue; |
| 568 |  |
| 569 | } |
| 570 |  |
| 571 | *//两个拐点的时候返回可以与a横竖直线相连的点* |
| 572 |  |
| 573 | **private** **void** ComparePoints (Tile a, **out** List comparePoints) |
| 574 |  |
| 575 | { |
| 576 |  |
| 577 | print ("a.idtex" + a.idTex); |
| 578 |  |
| 579 | comparePoints = new List (); |
| 580 |  |
| 581 | comparePoints.Clear (); |
| 582 |  |
| 583 | *// for (int i = 0; i < y; i ++) { // if (i != a.y) { // int id = (int)(i \* x + a.pos.x); // if (\_tiles [id].type == false) { // comparePoints.Add (\_tiles [id]); // \_tiles [id].idTex = a.idTex; // } // } // } for (int i = (int)a.pos.y - 1; i >-1; i--) {* |
| 584 |  |
| 585 | **int** id = (**int**)(i \* x + a.pos.x); |
| 586 |  |
| 587 | *// print ("three step :" + id);* |
| 588 |  |
| 589 | **if** (\_tiles [id].type == **false**) { |
| 590 |  |
| 591 | comparePoints.**Add** (\_tiles [id]); |
| 592 |  |
| 593 | \_tiles [id].idTex = a.idTex; |
| 594 |  |
| 595 | print ("\_tiles [id].idTex = a.idTex; " + \_tiles [id].idTex); |
| 596 |  |
| 597 | } **else** |
| 598 |  |
| 599 | **break**; |
| 600 |  |
| 601 | } |
| 602 |  |
| 603 | **for** (**int** i = (**int**)a.pos.y + 1; i < y; i++) { **int** id = (**int**)(i \* x + a.pos.x); *// print ("three step :" + id); if (\_tiles [id].type == false) { comparePoints.Add (\_tiles [id]); \_tiles [id].idTex = a.idTex; print ("\_tiles [id].idTex = a.idTex; " + \_tiles [id].idTex); } else break; } for (int i = (int)a.pos.x -1; i >-1; i --) {* |
| 604 |  |
| 605 | **int** id = (**int**)(a.pos.y \* x + i); |
| 606 |  |
| 607 | **if** (\_tiles [id].type == **false**) { |
| 608 |  |
| 609 | comparePoints.**Add** (\_tiles [id]); |
| 610 |  |
| 611 | \_tiles [id].idTex = a.idTex; |
| 612 |  |
| 613 | print ("\_tiles [id].idTex = a.idTex; " + \_tiles [id].idTex); |
| 614 |  |
| 615 | } **else** |
| 616 |  |
| 617 | **break**; |
| 618 |  |
| 619 | } |
| 620 |  |
| 621 | **for** (**int** i = (**int**)a.pos.x +1; i < x; i ++) { |
| 622 |  |
| 623 | **int** id = (**int**)(a.pos.y \* x + i); |
| 624 |  |
| 625 | **if** (\_tiles [id].type == **false**) { |
| 626 |  |
| 627 | comparePoints.**Add** (\_tiles [id]); |
| 628 |  |
| 629 | \_tiles [id].idTex = a.idTex; |
| 630 |  |
| 631 | print ("\_tiles [id].idTex = a.idTex; " + \_tiles [id].idTex); |
| 632 |  |
| 633 | } **else** |
| 634 |  |
| 635 | **break**; |
| 636 |  |
| 637 | } |
| 638 |  |
| 639 | *// for (int i = 0; i < x; i ++) {* |
| 640 |  |
| 641 | *// if (i != a.x) {* |
| 642 |  |
| 643 | *// int id = (int)(a.pos.y \* x + i);* |
| 644 |  |
| 645 | *// if (\_tiles [id].type == false) {* |
| 646 |  |
| 647 | *// comparePoints.Add (\_tiles [id]);* |
| 648 |  |
| 649 | *// \_tiles [id].idTex = a.idTex;* |
| 650 |  |
| 651 | *// }* |
| 652 |  |
| 653 | *// }* |
| 654 |  |
| 655 | *// }* |
| 656 |  |
| 657 | } |
| 658 |  |
| 659 | **private** **void** CompareID (Tile a, Tile b, **out** **int** min, **out** **int** max) |
| 660 |  |
| 661 | { |
| 662 |  |
| 663 | **if** (a.id < b.id) { |
| 664 |  |
| 665 | min = a.id; |
| 666 |  |
| 667 | max = b.id; |
| 668 |  |
| 669 | } **else** { |
| 670 |  |
| 671 | min = b.id; |
| 672 |  |
| 673 | max = a.id; |
| 674 |  |
| 675 | } |
| 676 |  |
| 677 | } |
| 678 |  |
| 679 | Vector2 TexSize () |
| 680 |  |
| 681 | { |
| 682 |  |
| 683 | Vector2 size = new Vector2 (1 / x, 1 / y); |
| 684 |  |
| 685 | **return** size; |
| 686 |  |
| 687 | } |
| 688 |  |
| 689 | Vector2 TexOffset (**int** \_idTex) |
| 690 |  |
| 691 | { |
| 692 |  |
| 693 | **int** a = (**int**)(\_idTex / x); |
| 694 |  |
| 695 | *// print (a + "a:" + \_idTex);* |
| 696 |  |
| 697 | **int** b = (**int**)(\_idTex % x); |
| 698 |  |
| 699 | *// print (b + "b:" + \_idTex);* |
| 700 |  |
| 701 | Vector2 offset = new Vector2 (b / x, (y - 1 - a) / y); |
| 702 |  |
| 703 | **return** offset; |
| 704 |  |
| 705 | } |
| 706 |  |
| 707 | **void** Update () |
| 708 |  |
| 709 | { |
| 710 |  |
| 711 | **if** (Input.GetMouseButtonUp (0)) { |
| 712 |  |
| 713 | mousePos = Input.mousePosition; |
| 714 |  |
| 715 | SelectTile (); |
| 716 |  |
| 717 | } |
| 718 |  |
| 719 | } |
| 720 |  |
| 721 | **private** **void** ClearTiles (List tiles) |
| 722 |  |
| 723 | { |
| 724 |  |
| 725 | tiles.Clear (); |
| 726 |  |
| 727 | *// this.gameObject.transform.DetachChildren();* |
| 728 |  |
| 729 | } |
| 730 |  |
| 731 | } |
| 732 |  |
| 733 | *// ari* |

DrawLine.cs，画线脚本，用的itween。

|  |  |  |
| --- | --- | --- |
|  |  |  |

|  |  |
| --- | --- |
| 01 | **using** UnityEngine; |
| 02 |  |
| 03 | **using** System.Collections; |
| 04 |  |
| 05 | **using** System.Collections.Generic; |
| 06 |  |
| 07 | **public** **class** DrawLine : MonoBehaviour |
| 08 |  |
| 09 | { |
| 10 |  |
| 11 | **public** List waypoints = new List (); |
| 12 |  |
| 13 | **public** **float** rate = 1; |
| 14 |  |
| 15 | **private** **int** currentWaypoint = 1; |
| 16 |  |
| 17 | **public** **void** MoveToWaypoint () |
| 18 |  |
| 19 | { |
| 20 |  |
| 21 | print ("public void MoveToWaypoint (): " + waypoints.Count); |
| 22 |  |
| 23 | StartCoroutine ("move"); |
| 24 |  |
| 25 | } |
| 26 |  |
| 27 | **public** **void** ClearPath () |
| 28 |  |
| 29 | { |
| 30 |  |
| 31 | waypoints.Clear (); |
| 32 |  |
| 33 | print ("path.Clear ();"); |
| 34 |  |
| 35 | } |
| 36 |  |
| 37 | IEnumerator move () |
| 38 |  |
| 39 | { |
| 40 |  |
| 41 | **for** (**int** i = 0; i < waypoints.Count; i++) { |
| 42 |  |
| 43 | iTween.MoveTo (**this**.gameObject, waypoints [i].position, rate); |
| 44 |  |
| 45 | print ("now id:" + i); |
| 46 |  |
| 47 | **yield** **return** new WaitForSeconds(rate); |
| 48 |  |
| 49 | } |
| 50 |  |
| 51 | waypoints.Clear (); |
| 52 |  |
| 53 | } |
| 54 |  |
| 55 | } |

Tile.cs

|  |  |  |
| --- | --- | --- |
|  |  |  |

|  |  |
| --- | --- |
| 01 | **using** UnityEngine; |
| 02 |  |
| 03 | **using** System.Collections; |
| 04 |  |
| 05 | **public** **class** Tile : MonoBehaviour |
| 06 |  |
| 07 | { |
| 08 |  |
| 09 | **public** **int** id; |
| 10 |  |
| 11 | **public** **int** idTex; *//通过这个判断两个图片是否相同* |
| 12 |  |
| 13 | **public** Vector2 pos ; |
| 14 |  |
| 15 | **public** **bool** type = **true**;*//控制图片的状态，当销毁的时候为false，其他判断的时候可以通过该点* |
| 16 |  |
| 17 | **public** **float** x, y; |
| 18 |  |
| 19 | **public** Texture texA, texB;*//鼠标选中的时候可以换贴图* |
| 20 |  |
| 21 | **public** GameObject mask;*//鼠标选中的时候上边显示的框框* |
| 22 |  |
| 23 | **public** **void** Init (**int** \_idTex) |
| 24 |  |
| 25 | { |
| 26 |  |
| 27 | idTex = \_idTex; |
| 28 |  |
| 29 | Vector2 offset = TexOffset (\_idTex); |
| 30 |  |
| 31 | **this**.renderer.material.SetTextureOffset ("\_MainTex", offset); |
| 32 |  |
| 33 | **this**.renderer.material.SetTextureScale ("\_MainTex", new Vector2 (0.2f, 0.1f));} |
| 34 |  |
| 35 | *//设置tile显示的贴图和大小* |
| 36 |  |
| 37 | **public** **void** SetTileTexture (**int** i) |
| 38 |  |
| 39 | { |
| 40 |  |
| 41 | **if** (i == 0) { |
| 42 |  |
| 43 | **this**.renderer.material.mainTexture = texA; |
| 44 |  |
| 45 | mask.transform.localScale = Vector3.zero; |
| 46 |  |
| 47 | }**if** (i == 1) { |
| 48 |  |
| 49 | **this**.renderer.material.mainTexture = texB; |
| 50 |  |
| 51 | mask.transform.localScale = new Vector3 (0.1380835f, 0.1380835f, 0.1380835f); |
| 52 |  |
| 53 | } |
| 54 |  |
| 55 | } |
| 56 |  |
| 57 | *//这个就是裁剪一张大图，变成一个个小的，贴到tile上* |
| 58 |  |
| 59 | Vector2 TexOffset (**int** \_idTex) |
| 60 |  |
| 61 | { |
| 62 |  |
| 63 | **int** a = (**int**)(\_idTex / x); |
| 64 |  |
| 65 | **int** b = (**int**)(\_idTex % x); |
| 66 |  |
| 67 | Vector2 offset = new Vector2 (b / x, (y - 1 - a) / y); |
| 68 |  |
| 69 | **return** offset; |
| 70 |  |
| 71 | } |
| 72 |  |
| 73 | Vector2 TexSize () |
| 74 |  |
| 75 | { |
| 76 |  |
| 77 | Vector2 size = new Vector2 (1 / x, 1 / y); |
| 78 |  |
| 79 | **return** size; |
| 80 |  |
| 81 | } |
| 82 |  |
| 83 | } |

Menu.cs，添加两个按钮。

|  |  |  |
| --- | --- | --- |
|  |  |  |

|  |  |
| --- | --- |
| 01 | **using** UnityEngine; |
| 02 |  |
| 03 | **using** System.Collections; |
| 04 |  |
| 05 | **public** **class** Menu : MonoBehaviour |
| 06 |  |
| 07 | { |
| 08 |  |
| 09 | **public** GameManager gameManager; |
| 10 |  |
| 11 | **private** GameManager \_gameManger; |
| 12 |  |
| 13 | **private** **bool** start = **true**;void OnGUI () |
| 14 |  |
| 15 | { |
| 16 |  |
| 17 | **if** (start) { |
| 18 |  |
| 19 | **if** (GUI.Button (new Rect (10, 10, 100, 50), "start")) { |
| 20 |  |
| 21 | start = **false**; |
| 22 |  |
| 23 | \_gameManger = Instantiate (gameManager) **as** GameManager; |
| 24 |  |
| 25 | } |
| 26 |  |
| 27 | } |
| 28 |  |
| 29 | **if** (GUI.Button (new Rect (10, 70, 100, 50), "restart")) { |
| 30 |  |
| 31 | **if** (\_gameManger != **null**) { |
| 32 |  |
| 33 | Destroy (\_gameManger.gameObject); |
| 34 |  |
| 35 | print ("Destroy(\_gameManger.gameObject);"); |
| 36 |  |
| 37 | } |
| 38 |  |
| 39 | \_gameManger = Instantiate (gameManager) **as** GameManager; |
| 40 |  |
| 41 | }} |
| 42 |  |
| 43 | } |