**人物行走脚本**

Posted on 2013年06月08日 by U3d / [Unity3D脚本/插件](http://www.unitymanual.com/category/script)/被围观 86 次

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

|  |  |
| --- | --- |
| 001 | **using** UnityEngine; |
| 002 |  |
| 003 | **using** System.Collections; |
| 004 |  |
| 005 | **public** **class** FPSWalker\_edit : MonoBehaviour |
| 006 |  |
| 007 | { |
| 008 |  |
| 009 | */// 类似于VRML的控制方式* |
| 010 |  |
| 011 | */// ↑前进 ↓后退 →右转 ←左转* |
| 012 |  |
| 013 | */// Ctrl + →右平移 Ctrl + ←左平移* |
| 014 |  |
| 015 | */// 按住鼠标左键，可以通过鼠标上下左右转动视角* |
| 016 |  |
| 017 | **public** **float** speed = f; |
| 018 |  |
| 019 | **public** **float** jumpSpeed = f; |
| 020 |  |
| 021 | **public** **float** gravity = f; |
| 022 |  |
| 023 | **public** **bool** MouseChange = **true**; |
| 024 |  |
| 025 | **private** Vector moveDirection = Vectorzero; |
| 026 |  |
| 027 | **private** **bool** grounded = **false**; |
| 028 |  |
| 029 | **void** FixedUpdate() |
| 030 |  |
| 031 | { |
| 032 |  |
| 033 | **if** (grounded) |
| 034 |  |
| 035 | { |
| 036 |  |
| 037 | **if** (!(Input.GetKey(KeyCode.LeftControl)||Input.GetKey(KeyCode.RightControl))) *//如果没有按下左Ctrl键* |
| 038 |  |
| 039 | { |
| 040 |  |
| 041 | *//只能前后平移* |
| 042 |  |
| 043 | moveDirection = new Vector(, , Input.GetAxis(“Vertical”)); |
| 044 |  |
| 045 | moveDirection = transform.TransformDirection(moveDirection); |
| 046 |  |
| 047 | moveDirection \*= speed; |
| 048 |  |
| 049 | } |
| 050 |  |
| 051 | **else** *//* |
| 052 |  |
| 053 | { |
| 054 |  |
| 055 | *//可前后左右平移* |
| 056 |  |
| 057 | moveDirection = new Vector(Input.GetAxis(“Horizontal”), , Input.GetAxis(“Vertical”)); |
| 058 |  |
| 059 | moveDirection = transform.TransformDirection(moveDirection); |
| 060 |  |
| 061 | moveDirection \*= speed; |
| 062 |  |
| 063 | } |
| 064 |  |
| 065 | **if** (Input.GetButton(“Jump”)) *//跳跃* |
| 066 |  |
| 067 | {&nbsp; Unity3D教程手册 |
| 068 |  |
| 069 | moveDirection.y = jumpSpeed; |
| 070 |  |
| 071 | } |
| 072 |  |
| 073 | } |
| 074 |  |
| 075 | *//重力* |
| 076 |  |
| 077 | moveDirection.y -= gravity \* Time.deltaTime; |
| 078 |  |
| 079 | *//移动controller* |
| 080 |  |
| 081 | CharacterController controller = GetComponent(“CharacterController”) **as** CharacterController; |
| 082 |  |
| 083 | CollisionFlags flags = controller.Move(moveDirection \* Time.deltaTime); |
| 084 |  |
| 085 | grounded = (flags & CollisionFlags.CollidedBelow) != ; *//当controller处在空中间，grounded为false，即跳动和行走都无效* |
| 086 |  |
| 087 | *//鼠标控制视角* |
| 088 |  |
| 089 | *// if (Input.GetMouseButton() && (Input.GetKey(KeyCode.LeftControl) || Input.GetKey(KeyCode.RightControl)) && MouseChange) //如果按下鼠标左键并且鼠标MouseChange为真* |
| 090 |  |
| 091 | **if** (Input.GetMouseButton() && MouseChange) *//如果按下鼠标左键并且鼠标MouseChange为真* |
| 092 |  |
| 093 | { |
| 094 |  |
| 095 | *///鼠标旋转视角部分* |
| 096 |  |
| 097 | *///* |
| 098 |  |
| 099 | **if** (axes == RotationAxes.MouseXAndY) |
| 100 |  |
| 101 | { |
| 102 |  |
| 103 | *// Read the mouse input axis* |
| 104 |  |
| 105 | *//这里，rotationX和rotationY用来保存对象现有的角度，同时还将鼠标的移动中计算出增减的角度并合进来* |
| 106 |  |
| 107 | rotationX += Input.GetAxis(“Mouse X”) \* sensitivityX; |
| 108 |  |
| 109 | rotationY += Input.GetAxis(“Mouse Y”) \* sensitivityY; |
| 110 |  |
| 111 | rotationX = ClampAngle(rotationX, minimumX, maximumX); |
| 112 |  |
| 113 | rotationY = ClampAngle(rotationY, minimumY, maximumY); |
| 114 |  |
| 115 | Quaternion xQuaternion = Quaternion.AngleAxis(rotationX, Vectorup); *//通过左右值和Vectorup（作为以Y为旋转轴的向量值）求出左右旋转度的四元数值* |
| 116 |  |
| 117 | Quaternion yQuaternion = Quaternion.AngleAxis(rotationY, Vectorleft); *//通过上下值和Vectorleft（作为以X为旋转轴的向量值）求出上下旋转度的四元数值* |
| 118 |  |
| 119 | *//originalRotation = transform.localRotation;* |
| 120 |  |
| 121 | transform.localRotation = originalRotation \* xQuaternion \* yQuaternion; *//将上面求出来的左右和上下两个四元数值添加入角度中* |
| 122 |  |
| 123 | } |
| 124 |  |
| 125 | **else** **if** (axes == RotationAxes.MouseX) |
| 126 |  |
| 127 | { |
| 128 |  |
| 129 | rotationX += Input.GetAxis(“Mouse X”) \* sensitivityX; |
| 130 |  |
| 131 | rotationX = ClampAngle(rotationX, minimumX, maximumX); |
| 132 |  |
| 133 | Quaternion xQuaternion = Quaternion.AngleAxis(rotationX, Vectorup); |
| 134 |  |
| 135 | transform.localRotation = originalRotation \* xQuaternion; |
| 136 |  |
| 137 | } |
| 138 |  |
| 139 | **else** |
| 140 |  |
| 141 | { |
| 142 |  |
| 143 | rotationY += Input.GetAxis(“Mouse Y”) \* sensitivityY; |
| 144 |  |
| 145 | rotationY = ClampAngle(rotationY, minimumY, maximumY); |
| 146 |  |
| 147 | Quaternion yQuaternion = Quaternion.AngleAxis(rotationY, Vectorleft); |
| 148 |  |
| 149 | transform.localRotation = originalRotation \* yQuaternion; |
| 150 |  |
| 151 | } |
| 152 |  |
| 153 | } |
| 154 |  |
| 155 | **else** |
| 156 |  |
| 157 | { |
| 158 |  |
| 159 | *///左右旋转* |
| 160 |  |
| 161 | *///并且没有按下左或右ctrl键时* |
| 162 |  |
| 163 | **if** (!(Input.GetKey(KeyCode.LeftControl) || Input.GetKey(KeyCode.RightControl))) |
| 164 |  |
| 165 | { |
| 166 |  |
| 167 | Vector angle\_temp = transform.eulerAngles; |
| 168 |  |
| 169 | angle\_temp.y += Input.GetAxis(“Horizontal”) \* sensitivityX \* f; |
| 170 |  |
| 171 | rotationX = ClampAngle(angle\_temp.y, minimumX, maximumX); |
| 172 |  |
| 173 | *//rotationY = ClampAngle(rotationY, minimumY, maximumY);* |
| 174 |  |
| 175 | transform.eulerAngles = angle\_temp; |
| 176 |  |
| 177 | } |
| 178 |  |
| 179 | *//键盘上下键控制俯仰角* |
| 180 |  |
| 181 | *//else* |
| 182 |  |
| 183 | *//{* |
| 184 |  |
| 185 | *// Vector angle\_temp = transform.eulerAngles;* |
| 186 |  |
| 187 | *// angle\_temp.x += Input.GetAxis(“Vertical”) \* - \* sensitivityY \* f;* |
| 188 |  |
| 189 | *// rotationY = ClampAngle(angle\_temp.x , minimumY , maximumY);* |
| 190 |  |
| 191 | *// transform.eulerAngles = angle\_temp;* |
| 192 |  |
| 193 | *//}* |
| 194 |  |
| 195 | } |
| 196 |  |
| 197 | } |
| 198 |  |
| 199 | **public** **enum** RotationAxes { MouseXAndY = , MouseX = , MouseY = } |
| 200 |  |
| 201 | **public** RotationAxes axes = RotationAxes.MouseXAndY; |
| 202 |  |
| 203 | **public** **float** sensitivityX = F; |
| 204 |  |
| 205 | **public** **float** sensitivityY = F; |
| 206 |  |
| 207 | **public** **float** minimumX = -F; |
| 208 |  |
| 209 | **public** **float** maximumX = F; |
| 210 |  |
| 211 | **public** **float** minimumY = -F; |
| 212 |  |
| 213 | **public** **float** maximumY = F; |
| 214 |  |
| 215 | **public** **float** rotationX = F; |
| 216 |  |
| 217 | **public** **float** rotationY = F; |
| 218 |  |
| 219 | Quaternion originalRotation; |
| 220 |  |
| 221 | **void** Start() |
| 222 |  |
| 223 | { |
| 224 |  |
| 225 | *// Make the rigid body not change rotation* |
| 226 |  |
| 227 | *//使刚体不会改变角度* |
| 228 |  |
| 229 | **if** (rigidbody) |
| 230 |  |
| 231 | rigidbody.freezeRotation = **true**; |
| 232 |  |
| 233 | *//originalRotation = transform.localRotation; //BUG处* |
| 234 |  |
| 235 | originalRotation = new Quaternion(f, f, f, f); *//修正代码* |
| 236 |  |
| 237 | } |
| 238 |  |
| 239 | **public** **static** **float** ClampAngle(**float** angle, **float** min, **float** max) |
| 240 |  |
| 241 | { |
| 242 |  |
| 243 | **if** (angle < -F) |
| 244 |  |
| 245 | angle += F; |
| 246 |  |
| 247 | **if** (angle > F) |
| 248 |  |
| 249 | angle -= F; |
| 250 |  |
| 251 | **return** Mathf.Clamp(angle, min, max); |
| 252 |  |
| 253 | } |
| 254 |  |
| 255 | } |
| 256 |  |
| 257 |  |