Estructura de orxBODY

Sumario

Body (Cuerpo)

```
[BodyTemplate]
PartList           = BodyPartTemplate1#BodyPartTemplate2#...
AllowGroundSliding = <bool>
AngularDamping     = <float>
CustomGravity      = <vector>
Dynamic            = <bool>
FixedRotation      = <bool>
HighSpeed          = <bool>
Inertia            = <float>
LinearDamping      = <float>
Mass               = <float>
```

BodyPart (Partes del Cuerpo)

Sphere (Esfera)

```
[BodyPartSphereTemplate]
Type        = sphere
Center      = <vector> | full
Radius      = <float> | full
CheckMask   = <16b flags>
SelfFlags   = <16b flags>
Density     = <float>
Friction    = <float>
Restitution = <float>
Solid       = <bool>
```

Box (Caja)

```
[BodyPartBoxTemplate]
Type        = box
TopLeft     = <vector> | full
BottomRight = <vector> | full
CheckMask   = <16b flags>
SelfFlags   = <16b flags>
Density     = <float>
Friction    = <float>
Restitution = <float>
```

```
Solid        = <bool>
```

Mesh (polygon) (Malla - Polígono)

```
[BodyPartMeshTemplate]
Type        = mesh
VertexList  = <vector>#<vector>#...
CheckMask   = <16b flags>
SelfFlags   = <16b flags>
Density     = <float>
Friction    = <float>
Restitution = <float>
Solid       = <bool>
```

Detalles

Body (Cuerpo)

Vemos aquí una lista de propiedades disponibles para una estructura `orxBODY`:

- `PartList`: Lista de todas las partes de las cuales está compuesta el cuerpo(Body). No hay un limite en el número de partes que pueden ser definidas para un solo cuerpo. Esta propiedad *necesita* ser definida si quieres que colisione con otros cuerpos.
- `AllowGroundSliding`: Si se marca como falso en un objeto dinámico, se evitará que se deslice en pendientes estáticas de mas de 45°. Esto solo funciona con una gravedad vertical arriba-abajo. Por defecto su valor es `true`.
- `AngularDamping`: Amortiguación de la velocidad angular de un cuerpo. Por defecto su valor es 0.0, que significa que no hay amortiguacion.
- `CustomGravity`: Define un vector de gravedad para usar en el cuerpo en vez de la global. Por defecto no existe, lo que nos dice que la gravedad global será la usada en este cuerpo.
- `Dynamic`: Define si el cuerpo debe ser dinámico o estático. Si se espera que el objeto se mueva, esta propiedad debe estar marcada como `true`. Cuerpos estáticos no pueden colisionar con otros cuerpos estáticos. Por defecto, su valor es `false` (ej. statico).
- `FixedRotation`: Define si a nuestro objeto dinámico se le permite girar como resultado de una fuerza de colision. Por defecto su valor es `false` lo que significa que puede rotar libremente.
- `HighSpeed`: Para objetos de alta velocidad (ej. balas), esta propiedad debe ser marcada como `true` para evitar errores de colisión. Sin embargo, cada objeto marcado con `HighSpeed` será mas costoso procesarlo por el motor de física. Por defecto su valor es `false`.
- `Inertia`: Define un valor de inercia para este objeto. Por defecto su valor es 0.0.
- `LinearDamping`: Amortiguacion de velocidad(velocidad linear) para este cuerpo. Por defecto su valor es 0.0, que significa que no tiene amortiguamiento.
- `Mass`: Define una masa, en kg, para este cuerpo. Si las partes están definidas, la masa será reemplazada por un valor calculado automáticamente en función de los tamaños de piezas y sus posiciones.

BodyPart (Parte del Cuerpo)

Común

Tenemos aquí una lista de propiedades disponibles para todos los tipos de partes de un cuerpo:

- `Type`: Define el tipo de una parte del cuerpo. Los tipos disponibles son `esferea`(`sphere`), `caja`(`box`) and `malla`(`mesh` – ej. poligono convexo). Esta propiedad *necesita* ser definida.
- `CheckMask/SelfFlags`: Ambas propiedades son marcas expresadas en 16 bits. El `SelfFlags` define su identidad, mientras que, parte de la Máscara de entrada(`CheckMask`) define qué partes se les permite colisionar con él. Para una colision que ocurre entre dos partes la expresion (Part1.CheckMask & Part2.SelfFlags) y (Part2.CheckMask & Part1.SelfFlags) tienen ambas que estar marcadas como `true`. PD: Dos partes del mismo cuerpo no pueden colisionar entre ellas[1)]
- `Density`: Define la densidad de esta parte. Su valor por defecto es 0.0, que dice que no tiene ninguna influencia en la masa del cuerpo.
- `Friction/Restitution`: Define la friccion y restitucion de esta parte, usualmente entre 0.0 y 1.0. [2)] Por defecto, ambos valores son 0.0.
- `Solid`: Define si esta parte es sólida o no. Solo las partes sólidas desencadenarán una reacción en su cuerpo cuando colisionen con otros. Por defecto su valor es `false` lo que significa que la información de la colisión será señalada a través de eventos, pero la simulación física de este cuerpo no será afectada automáticamente por esto.

Sphere (Esfera)

Tenemos aquí una lista de propiedades disponibles solo para las partes `esfera`:

- `Center`: Define el centro de la esfera(en 2D es un circulo, por supuesto) en el espacio padre(ej. Objetos en el espacio). Por defecto su valor es `full` que significa que el centro coincidirá con el del objeto(ej. el centro de su correspondiente gráfico).
- `Radius`: Define el radio de la esfera(o circulo 2D). Por defecto su valor es `full` lo que significa que el radio de la esfera coincidirá con la mayor dimensión del objeto padre. Puedes encontrar un ejemplo en el tutorial de reproductor [3)].

Box (Caja)

Aquí una lista de propiedades disponibles solo para las partes `caja`:

- `TopLeft/BottomRight`: Define los extremos de la caja (en 2D es un rectangulo, por supuesto) en el espacio padre (ej. objeto en el espacio). Por defecto sus valores son `full`, que significa que `ArribaIzquierda`(`TopLeft`) y `AbajoDerecha`(`BottomRight`) coincidirá con el rectángulo definido por completo, del gráfico actual, del objeto principal. Puedes encontrar un ejemplo en el tutorial de física.

Mesh (polygon) (Malla - polígono)

Tenemos aquí una lista de propiedades disponibles solo para la `malla` [4]:

- `VertexList` (`Lista de Vértices`): Proporciona una lista de los vértices de coordenadas en el espacio del objeto padre. El poligono resultante *necesita* ser convexo. Hasta 8 vertices pueden ser definidos y ellos ***tienen que ser añadidos en función de las manecillas del reloj***. Puedes encontrar un ejemplo en el tutorial de reproductor [5].

1)

Chequear documentación de Box2D(EN) para mas información.

2)

Chequear la documentación de Box2D para más información sobre friction/restitution

3) , 5)

o mirando directamente en los archivos de configuración, ya que no está cubierto en el wiki

4)

polígono convexo

From:
https://www.orx-project.org/wiki/ - **Orx Learning**

Permanent link:
https://www.orx-project.org/wiki/es/orx/config/settings_structure/orxbody?rev=1331645304

Last update: **2025/09/30 17:26 (14 months ago)**