Tutorial de Sonido

Sumario

Detalles

Como es usual, comenzaremos por cargar nuestro fichero de configuración, obteniendo el reloj principal y registrando nuestra función `Update` a este y, por último, por crear nuestro objeto soldado. Por favor, diríjase a los tutoriales previos para más detalles.

Creamos entonces una música y la reproducimos.

```
orxSOUND *pstMusic;
pstMusic = orxSound_CreateFromConfig("Music");

orxSound_Play(pstMusic);
```

Como podemos ver, música y sonido son de tipo `orxSOUND`. La diferencia principal es que la música escuchada mientras el sonido es completamente cargada en memoria. Como veremos más adelante, definiremos esta diferencia en el fichero de configuración.

El último paso de nuestra función `Init`: nos suscribimos a los eventos de sonido.

```
orxEvent_AddHandler(orxEVENT_TYPE_SOUND, EventHandler);
```

Solo registramos cuando los sonidos son iniciados/parados, echemos un vistazo al código correspondiente.

```
orxSOUND_EVENT_PAYLOAD *pstPayload;

pstPayload = (orxSOUND_EVENT_PAYLOAD *)_pstEvent->pstPayload;

switch(_pstEvent->eID)
{
  case orxSOUND_EVENT_START:
    orxLOG("Sound <%s>@<%s> has started!", pstPayload->zSoundName,
orxObject_GetName(orxOBJECT(_pstEvent->hRecipient)));
    break;

  case orxSOUND_EVENT_STOP:
    orxLOG("Sound <%s>@<%s> has stoped!", pstPayload->zSoundName,
orxObject_GetName(orxOBJECT(_pstEvent->hRecipient)));
    break;
}

return orxSTATUS_SUCCESS;
```

Nada realmente nuevo como puedes ver.

Veamos como añadimos sonidos a nuestro soldado.

```
if(orxInput_IsActive("RandomSFX") && orxInput_HasNewStatus("RandomSFX"))
{
  orxObject_AddSound(pstSoldier, "RandomBip");
  orxObject_SetColor(pstSoldier, orxColor_Set(&stColor,
orxConfig_GetVector("RandomColor", &v), orxFLOAT_1));
}

if(orxInput_IsActive("DefaultSFX") && orxInput_HasNewStatus("DefaultSFX"))
{
  orxObject_AddSound(pstSoldier, "DefaultBip");
  orxObject_SetColor(pstSoldier, orxColor_Set(&stColor, &orxVECTOR_WHITE,
orxFLOAT_1));
}
```

Como podemos ver, añadir un sonido a un objeto solo requiere una única linea de código y, más importante, se hace de la misma manera para nuestro sonido aleatorio y nuestra única frecuencia fija. Como veremos después, la diferencia esta solamente expresada en el fichero de configuración. Cuando añadimos sonido RandomBip, solo cambiamos el color de nuestro soldado aleatoriamente, ya que está definido en nuestro fichero de configuración con la llave RandomColor. Cuando reproducimos un DefaultBip' en el, simplemente regresamos su color a blanco.
PD: Un sonido será reproducido en el soldado siempre y cuando la correspondiente entrada sea activada.
Hasta ahora solo importaba si una entrada estaba activa o no, aquí queremos hacer algún tipo de acción en el preciso momento que la entrada es activada. Para hacer eso, usaremos la función orxInput_HasNewStatus() que retornará orxTRUE cuando la entrada vaya de pasiva a activa y, inversamente, cuando vaya de activa a pasiva.
Utilizando el resultado de esta función a lo largo de la que obtenemos de orxInput_IsActive(), se asegura de que sólo se reproducirá el sonido cuando la entrada va de pasivo a activo! Ahora juguemos con la activación de la música. <code c>if(orxInput_IsActive(“ToggleMusic”) && orxInput_HasNewStatus(“ToggleMusic”)) { if(orxSound_GetStatus(pstMusic) != orxSOUND_STATUS_PLAY) { orxSound_Play(pstMusic); orxObject_Enable(pstSoldier, orxTRUE); } else { orxSound_Pause(pstMusic); orxObject_Enable(pstSoldier, orxFALSE); } }</code> Este código simple, cuando activando la entrada ToggleMusic'', o comenzamos la música y activamos nuestro soldado si la música no se estaba reproduciendo o, por el contrario, la pararemos y desactivamos nuestro soldado si la música se estaba reproduciendo.

Recursos