Subject: Re: FDS Crash
Posted by Neijwiert on Mon, 18 Nov 2013 22:52:38 GMT
View Forum Message <> Reply to Message

As soon as I wrote this part "I think somewhere in Scripts.dll there is code that assumes that all SSGM scripts are still attached." and ate a cookie. I realised that it must have been in the SSGM code.

The one who is responsible of SSGM should change these 2 functions:

Toggle Spoiler

```
bool SSGMGameManager::RefillHook(GameObject *purchaser)
{
 SSGM_Soldier *script = (SSGM_Soldier *)Find_Script_On_Object(purchaser,"SSGM_Soldier");
 if (RefillLimit)
 {
  if (The_Game()->Get_Game_Duration_S() - script->RefillTime < RefillLimit)
  {
   return false;
  }
 }
 script->RefillTime = The_Game()->Get_Game_Duration_S();
 for (int i = 0;i < RegisteredEvents[EVENT_REFILL_HOOK].Count();i++)
 {
  if (!RegisteredEvents[EVENT_REFILL_HOOK][i]->OnRefill(purchaser))
  {
   return false;
  }
 }
 return true;
}

int SSGMGameManager::CharacterPurchaseHook(BaseControllerClass *base,GameObject
*purchaser,unsigned int cost,unsigned int preset,const char *data)
{
 SSGM_Soldier *script = (SSGM_Soldier *)Find_Script_On_Object(purchaser,"SSGM_Soldier");
 if (RefillLimit)
 {
  if (The_Game()->Get_Game_Duration_S() - script->RefillTime < RefillLimit)
  {
   return 4;
  }
 }
 if (IsPresetDisabled(preset))
 {
  return 4;
 }
```

```
 StringClass str;
 const char *str2 = Get_Translated_Definition_Name_Ini(Get_Definition_Name(preset));
 const char *str3 = Get_Player_Name(purchaser);
 str.Format("Purchase: %s - %s",str3,str2);
 delete[] str3;
 delete[] str2;
 SSGMGameLog::Log_Message(str,"_PURCHASE");
 bool fp = FreePurchases;
 for (int i = 0;i < RegisteredEvents[EVENT_CHARACTER_PURCHASE_HOOK].Count();i++)
 {
  int ret =
RegisteredEvents[EVENT_CHARACTER_PURCHASE_HOOK][i]->OnCharacterPurchase(base,p
urchaser,cost,preset,data);
  if (ret == -2)
  {
   fp = true;
  }
  else if (ret != -1)
  {
   return ret;
  }
 }
 if (fp)
 {
  return -2;
 }
 return -1;
}
```

to:

Toggle Spoiler

```
bool SSGMGameManager::RefillHook(GameObject *purchaser)
{
 SSGM_Soldier *script = (SSGM_Soldier *)Find_Script_On_Object(purchaser,"SSGM_Soldier");
 if(script)
 {
  if (RefillLimit)
  {
   if (The_Game()->Get_Game_Duration_S() - script->RefillTime < RefillLimit)
   {
    return false;
   }
  }
  script->RefillTime = The_Game()->Get_Game_Duration_S();
```

```
  for (int i = 0;i < RegisteredEvents[EVENT_REFILL_HOOK].Count();i++)
  {
   if (!RegisteredEvents[EVENT_REFILL_HOOK][i]->OnRefill(purchaser))
   {
    return false;
   }
  }
 }
 return true;
}

int SSGMGameManager::CharacterPurchaseHook(BaseControllerClass *base,GameObject
*purchaser,unsigned int cost,unsigned int preset,const char *data)
{
 SSGM_Soldier *script = (SSGM_Soldier *)Find_Script_On_Object(purchaser,"SSGM_Soldier");
 if(script)
 {
  if (RefillLimit)
  {
   if (The_Game()->Get_Game_Duration_S() - script->RefillTime < RefillLimit)
   {
    return 4;
   }
  }
  if (IsPresetDisabled(preset))
  {
   return 4;
  }
  StringClass str;
  const char *str2 = Get_Translated_Definition_Name_Ini(Get_Definition_Name(preset));
  const char *str3 = Get_Player_Name(purchaser);
  str.Format("Purchase: %s - %s",str3,str2);
  delete[] str3;
  delete[] str2;
  SSGMGameLog::Log_Message(str,"_PURCHASE");
  bool fp = FreePurchases;
  for (int i = 0;i < RegisteredEvents[EVENT_CHARACTER_PURCHASE_HOOK].Count();i++)
  {
   int ret =
RegisteredEvents[EVENT_CHARACTER_PURCHASE_HOOK][i]->OnCharacterPurchase(base,p
urchaser,cost,preset,data);
   if (ret == -2)
   {
    fp = true;
   }
   else if (ret != -1)
   {
    return ret;
```

```
   }
  }
  if (fp)
  {
   return -2;
  }
 }
 return -1;
}
```