MR Spell C++

killit

Bonjour,

Nous allons voir comment réaliser simplement un Mr Spell en C++. Ce tutoriel est fonctionnel sur la dernière version TC 3.3.5.
Ce PNJ sert simplement a acheter un sort, nous l'apprendre et le vendre.
Commençons par organiser notre structure en énumérant différent action et paramètre.

- Nos différents menu :

```
enum Menu
{
 MAIN_MENU = 0,
 GUERRIER_MENU = 1,
 PALADIN_MENU = 2,
 CHASSEUR_MENU = 3,
 VOLEUR_MENU = 4,
 PRETRE_MENU = 5,
 DK_MENU = 6,
 CHAMAN_MENU = 7,
 MAGE_MENU = 8,
 DEMONISTE_MENU = 9,
 DRUIDE_MENU = 10,
};
```

- Nos différents prix :

```
enum Prix
{
 TOKEN_ID = 40753,// ID de monnaie
 PRIX_SPELL_1 = 15,// Prix pour l'achat d'un spell normal
 PRIX_SPELL_2 = 20,// Prix pour l'achat d'un spell supérieur
 PRIX_SPELL_3 = 7,// Prix pour le remboursement d'un spell normal
 PRIX_SPELL_4 = 10,// Prix pour le remboursement d'un spell supérieur
};
```

-Nos différentes actions :

```
enum Actions
{
 // Spell guerrier
 ACTION_GUERRIER_1 = 1000,
 ACTION_GUERRIER_2 = 1001,
 // Spell paladin
 ACTION_PALADIN_1 = 2000,
 ACTION_PALADIN_2 = 2001,
 // Spell chasseur
 ACTION_CHASSEUR_1 = 3000,
 ACTION_CHASSEUR_2 = 3001,
 // Spell voleur
 ACTION_VOLEUR_1 = 4000,
```

```
    ACTION_VOLEUR_2 = 4001,
    // Spell pretre
    ACTION_PRETRE_1 = 5000,
    ACTION_PRETRE_2 = 5001,
    // Spell chevalier de la mort
    ACTION_DK_1 = 6000,
    ACTION_DK_2 = 6001,
    // Spell chaman
    ACTION_CHAMAN_1 = 7000,
    ACTION_CHAMAN_2 = 7001,
    // Spell mage
    ACTION_MAGE_1 = 8000,
    ACTION_MAGE_2 = 8001,
    // Spell demoniste
    ACTION_DEMONISTE_1 = 9000,
    ACTION_DEMONISTE_2 = 9001,
    // Spell druide
    ACTION_DRUIDE_1 = 10000,
    ACTION_DRUIDE_2 = 10001,
};
```

Ensuite nous allons afficher nos différents menu crée précédemment. Je souhaite afficher tous les menu disponible sauf celui qui me concerne, sauf si vous souhaitez acheter un sort que vous possédez déjà... ici, on filtrera les menus par classe, on utilisara cette fonction :

`GetClass()`

Ce qui reviens à faire par exemple :

`player->GetClass() != CLASS_WARRIOR`

Plus grossièrement cela donne ceci :

```
bool OnGossipHello(Player* player)
    {
    ClearGossipMenuFor(player);

    if(player->GetClass() != CLASS_WARRIOR)
    {
     AddGossipItemFor(player, GOSSIP_ICON_INTERACT_2, "|TInterface\\icons\\INV_Sword_27:30|tWarrior", GOSSIP_SENDER_MAIN, GUERRIER
    }
    if(player->GetClass() != CLASS_PALADIN)
    {
        AddGossipItemFor(player, GOSSIP_ICON_INTERACT_2, "|TInterface\\icons\\SPELL_Holy_DivineIntervention__:30|tPaladin", GOSSIP_SENI
    }
    if(player->GetClass() != CLASS_HUNTER)
    {
     AddGossipItemFor(player, GOSSIP_ICON_INTERACT_2, "|TInterface\\icons\\INV_Weapon_Bow_07:30|tHunter", GOSSIP_SENDER_MAIN, CHA
    }
    if(player->GetClass() != CLASS_ROGUE)
    {
     AddGossipItemFor(player, GOSSIP_ICON_INTERACT_2, "|TInterface\\icons\\INV_ThrowingKnife_:30|tRogue", GOSSIP_SENDER_MAIN, VOLEU
    }
    if(player->GetClass() != CLASS_PRIEST)
    {
     AddGossipItemFor(player, GOSSIP_ICON_INTERACT_2, "|TInterface\\icons\\INV_Staff_30:30|tPriest", GOSSIP_SENDER_MAIN, PRETRE_MEN
    }
    if(player->GetClass() != CLASS_DEATH_KNIGHT)
    {
     AddGossipItemFor(player, GOSSIP_ICON_INTERACT_2, "|TInterface\\icons\\Spell_Deathknight_ClassIcon__:30|tDeath Knight", GOSSIP_SEND
    }
    if(player->GetClass() != CLASS_SHAMAN)
```

```
{
 AddGossipItemFor(player, GOSSIP_ICON_INTERACT_2, "|TInterface\\icons\\Spell_Nature\Bloodlust_:30|tShaman", GOSSIP_SENDER_MAIN, C
}
if(player->GetClass() != CLASS_MAGE)
{
 AddGossipItemFor(player, GOSSIP_ICON_INTERACT_2, "|TInterface\\icons\\INV_Staff_13:30|tMage", GOSSIP_SENDER_MAIN, MAGE_MENU)
}
if(player->GetClass() != CLASS_WARLOCK)
{
 AddGossipItemFor(player, GOSSIP_ICON_INTERACT_2, "|TInterface\\icons\\Spell_Nature_Drowsy:30|tWarlock", GOSSIP_SENDER_MAIN, DE
}
if(player->GetClass() != CLASS_DRUID)
{
 AddGossipItemFor(player, GOSSIP_ICON_INTERACT_2, "|TInterface\\icons\\INV_Misc_MonsterClaw_04___30|tDruid", GOSSIP_SENDER_MAI
}
SendGossipMenuFor(player, 1, me->GetGUID());
return true;
    }
```

Ensuite une fois que l'on aura sélectionné notre menu, nous allons afficher des noms de sort que le souhaite mettre à disposition, ici nous utiliserons le sort "Poigne du titan" **https://wotlkdb.com/?spell=46917**.

```
case GUERRIER_MENU:
 ClearGossipMenuFor(player);
 AddGossipItemFor(player, GOSSIP_ICON_MONEY_BAG, "Poigne du Titan", GOSSIP_SENDER_MAIN, ACTION_GUERRIER_1);
 AddGossipItemFor(player, GOSSIP_ICON_MONEY_BAG, "Frénésie sanglante", GOSSIP_SENDER_MAIN, ACTION_GUERRIER_2);
 SendGossipMenuFor(player, 1, me->GetGUID());
 return true;
break;
```

Suite au choix du sort, on doit ensuite proposer quelque solution comme acheter ou vendre le sort.

```
 case ACTION_GUERRIER_1:
  ClearGossipMenuFor(player);
  AddGossipItemFor(player, GOSSIP_ICON_MONEY_BAG, "Achetez le sort", GOSSIP_SENDER_MAIN, ACTION_GUERRIER_1 + 100);
  AddGossipItemFor(player, GOSSIP_ICON_MONEY_BAG, "Rembourser le sort", GOSSIP_SENDER_MAIN, ACTION_GUERRIER_1 + 101);
  AddGossipItemFor(player, GOSSIP_ICON_MONEY_BAG, "Back", GOSSIP_SENDER_MAIN, MAIN_MENU);
  SendGossipMenuFor(player, 1, me->GetGUID());
  return true;
 break;
```

Le coût d'un sort est rarement gratuit, on utilisera la fonction `HasItemCount` pour fixer un prix, en fonction d'un ID objet défini au début. Suivis de cette fonction `LearnSpell`, pour apprendre le sort au joueur, et `DestroyItemCount` pour détruite le nombre d'objet correspondant au prix. Une petite vérification si le joueur n'a pas déja le sort, et le prix.

```
case ACTION_GUERRIER_1 + 100:
 if(player->HasItemCount(TOKEN_ID, PRIX_SPELL_2, true))
 {
  if(player->HasSpell(46917))
  {
   player->GetSession()->SendNotification("Vous possédez déjà le sort.");
   CloseGossipMenuFor(player);
  }
  else
  {
   CloseGossipMenuFor(player);
```

```
    player->LearnSpell(46917, false);
    player->DestroyItemCount(TOKEN_ID, PRIX_SPELL_2, true);I
    player->GetSession()->SendNotification("Félicitation pour votre achat.");
   }
  }
  else
  {
   CloseGossipMenuFor(player);
   player->GetSession()->SendNotification("Vous n'avez pas les objet nécessaire pour cette achat.");
  }
 break;`
```

Pour rembourser le sort cela se passe de la même façon. On utilise `RemoveSpell` pour supprimer le sort, et `AddItem` , pour redonner un prix.

```
 case ACTION_GUERRIER_1 + 101:
  if(player->HasSpell(46917))
  {
   CloseGossipMenuFor(player);
   player->RemoveSpell(46917);
   player->AddItem(TOKEN_ID, PRIX_SPELL_4);
   player->GetSession()->SendNotification("Remboursement effectué.");
  }
  else
  {
   CloseGossipMenuFor(player);
   player->GetSession()->SendNotification("Vous n'avez pas ce sort.");
  }
 break;`
```

En somme, suivis de quelque détail en bonne et due forme on obtient ceci :

```
#include "ScriptedGossip.h"
#include "WorldSession.h"

enum Menu
{
 MAIN_MENU = 0,
 GUERRIER_MENU = 1,
 PALADIN_MENU = 2,
 CHASSEUR_MENU = 3,
 VOLEUR_MENU = 4,
 PRETRE_MENU = 5,
 DK_MENU = 6,
 CHAMAN_MENU = 7,
 MAGE_MENU = 8,
 DEMONISTE_MENU = 9,
 DRUIDE_MENU = 10,
};

enum Prix
{
 TOKEN_ID = 40753,// ID de monnaie
 PRIX_SPELL_1 = 15,// Prix pour l'achat d'un spell normal
 PRIX_SPELL_2 = 20,// Prix pour l'achat d'un spell supérieur
 PRIX_SPELL_3 = 7,// Prix pour le remboursement d'un spell normal
 PRIX_SPELL_4 = 10,// Prix pour le remboursement d'un spell supérieur
};
```

```
enum Actions
{
 // Spell guerrier
 ACTION_GUERRIER_1 = 1000,
 ACTION_GUERRIER_2 = 1001,
 // Spell paladin
 ACTION_PALADIN_1 = 2000,
 ACTION_PALADIN_2 = 2001,
 // Spell chasseur
 ACTION_CHASSEUR_1 = 3000,
 ACTION_CHASSEUR_2 = 3001,
 // Spell voleur
 ACTION_VOLEUR_1 = 4000,
 ACTION_VOLEUR_2 = 4001,
 // Spell pretre
 ACTION_PRETRE_1 = 5000,
 ACTION_PRETRE_2 = 5001,
 // Spell chevalier de la mort
 ACTION_DK_1 = 6000,
 ACTION_DK_2 = 6001,
 // Spell chaman
 ACTION_CHAMAN_1 = 7000,
 ACTION_CHAMAN_2 = 7001,
 // Spell mage
 ACTION_MAGE_1 = 8000,
 ACTION_MAGE_2 = 8001,
 // Spell demoniste
 ACTION_DEMONISTE_1 = 9000,
 ACTION_DEMONISTE_2 = 9001,
 // Spell druide
 ACTION_DRUIDE_1 = 10000,
 ACTION_DRUIDE_2 = 10001,
};

class PNJ_SPELL : public CreatureScript
{
public:
    PNJ_SPELL() : CreatureScript("PNJ_SPELL") { }

    struct PNJ_SPELLAI : public ScriptedAI
    {
        PNJ_SPELLAI(Creature* creature) : ScriptedAI(creature) { }

        bool OnGossipHello(Player* player)
        {
   ClearGossipMenuFor(player);//On vide le gossip du PNJ
   //Si le joueur n'est pas de la classe Guerrier on affiche le menu de la classe Guerrier
   if(player->GetClass() != CLASS_WARRIOR)//Début du gossip Guerrier
   {
    AddGossipItemFor(player, GOSSIP_ICON_INTERACT_2, "|TInterface\\icons\\INV_Sword_27:30|tWarrior", GOSSIP_SENDER_MAIN, GUERRIER
   }//Fin du gossip Guerrier
   //Si le joueur n'est pas de la classe Paladin on affiche le menu de la classe Paladin
   if(player->GetClass() != CLASS_PALADIN)//Début du gossip Paladin
   {
   AddGossipItemFor(player, GOSSIP_ICON_INTERACT_2, "|TInterface\\icons\\Spell_Holy_DivineIntervention:30|tPaladin", GOSSIP_SENDER_MAI
   }//Fin du gossip Paladin
   //Si le joueur n'est pas de la classe Chasseur on affiche le menu de la classe Chasseur
   if(player->GetClass() != CLASS_HUNTER)//Début du gossip Chasseur
   {
```

```
 AddGossipItemFor(player, GOSSIP_ICON_INTERACT_2, "|TInterface\\icons\\INV_Weapon_Bow_07:30|tHunter", GOSSIP_SENDER_MAIN, CHA
}//Fin du gossip Chasseur
//Si le joueur n'est pas de la classe Voleur on affiche le menu de la classe Voleur
if(player->GetClass() != CLASS_ROGUE)//Début du gossip Voleur
{
 AddGossipItemFor(player, GOSSIP_ICON_INTERACT_2, "|TInterface\\icons\\INV_ThrowingKnife_04:30|tRogue", GOSSIP_SENDER_MAIN, VOL
}//Fin du gossip Voleur
//Si le joueur n'est pas de la classe Prêtre on affiche le menu de la classe Prêtre
if(player->GetClass() != CLASS_PRIEST)//Début du gossip Prêtre
{
 AddGossipItemFor(player, GOSSIP_ICON_INTERACT_2, "|TInterface\\icons\\INV_Staff_30:30|tPriest", GOSSIP_SENDER_MAIN, PRETRE_MEN
}//Fin du gossip Prêtre
//Si le joueur n'est pas de la classe DK on affiche le menu de la classe DK
if(player->GetClass() != CLASS_DEATH_KNIGHT)//Début du gossip DK
{
 AddGossipItemFor(player, GOSSIP_ICON_INTERACT_2, "|TInterface\\icons\\Spell_Deathknight_ClassIcon:30|tDeath Knight", GOSSIP_SENDEF
}//Fin du gossip DK
//Si le joueur n'est pas de la classe Chaman on affiche le menu de la classe Chaman
if(player->GetClass() != CLASS_SHAMAN)//Début du gossip Chaman
{
 AddGossipItemFor(player, GOSSIP_ICON_INTERACT_2, "|TInterface\\icons\\Spell_Nature_BloodLust:30|tShaman", GOSSIP_SENDER_MAIN, C
}//Fin du gossip Chaman
//Si le joueur n'est pas de la classe Mage on affiche le menu de la classe Mage
if(player->GetClass() != CLASS_MAGE)//Début du gossip Mage
{
 AddGossipItemFor(player, GOSSIP_ICON_INTERACT_2, "|TInterface\\icons\\INV_Staff_13:30|tMage", GOSSIP_SENDER_MAIN, MAGE_MENU)
}//Fin du gossip Mage
//Si le joueur n'est pas de la classe Démoniste on affiche le menu de la classe Démoniste
if(player->GetClass() != CLASS_WARLOCK)//Début du gossip Démoniste
{
 AddGossipItemFor(player, GOSSIP_ICON_INTERACT_2, "|TInterface\\icons\\Spell_Nature_Drowsy:30|tWarlock", GOSSIP_SENDER_MAIN, DEM
}//Fin du gossip Démoniste
//Si le joueur n'est pas de la classe Druide on affiche le menu de la classe Druide
if(player->GetClass() != CLASS_DRUID)//Début du gossip Druide
{
 AddGossipItemFor(player, GOSSIP_ICON_INTERACT_2, "|TInterface\\icons\\INV_Misc_MonterClaw:30|tDruid", GOSSIP_SENDER_MAIN, DRUI
}//Fin du gossip Druide
SendGossipMenuFor(player, 1, me->GetGUID());
return true;
    }

    bool OnGossipSelect(Player* player, uint32 menuId, uint32 gossipListId)
    {
uint32 const sender = player->PlayerTalkClass->GetGossipOptionSender(gossipListId);
        uint32 const action = player->PlayerTalkClass->GetGossipOptionAction(gossipListId);

ClearGossipMenuFor(player);

if (sender != GOSSIP_SENDER_MAIN)
            return false;

switch (action)
{
 case MAIN_MENU:
  OnGossipHello(player);
  return true;
 break;
 case GUERRIER_MENU:
  ClearGossipMenuFor(player);//On vide le gossip du PNJ
  AddGossipItemFor(player, GOSSIP_ICON_MONEY_BAG, "Poigne du Titan", GOSSIP_SENDER_MAIN, ACTION_GUERRIER_1);
```

```
    AddGossipItemFor(player, GOSSIP_ICON_MONEY_BAG, "Poigne du titan", GOSSIP_SENDER_MAIN, ACTION_GUERRIER_1);
    AddGossipItemFor(player, GOSSIP_ICON_MONEY_BAG, "Frénésie sanglante", GOSSIP_SENDER_MAIN, ACTION_GUERRIER_2);
    SendGossipMenuFor(player, 1, me->GetGUID());
    return true;
   break;
   case ACTION_GUERRIER_1:
    ClearGossipMenuFor(player);//On vide le gossip du PNJ
    AddGossipItemFor(player, GOSSIP_ICON_MONEY_BAG, "Achetez le sort", GOSSIP_SENDER_MAIN, ACTION_GUERRIER_1 + 100);
    AddGossipItemFor(player, GOSSIP_ICON_MONEY_BAG, "Rembourser le sort", GOSSIP_SENDER_MAIN, ACTION_GUERRIER_1 + 101);
    AddGossipItemFor(player, GOSSIP_ICON_MONEY_BAG, "Back", GOSSIP_SENDER_MAIN, MAIN_MENU);
    SendGossipMenuFor(player, 1, me->GetGUID());
    return true;
   break;
   case ACTION_GUERRIER_1 + 100://Achat du sort Poigne du titan du guerrer
    if(player->HasItemCount(TOKEN_ID, PRIX_SPELL_2, true))//si le joeur possède la quantité d'objet nécessaire
    {
     if(player->HasSpell(46917))//Si le joueur possède déjà le sort
     {
      player->GetSession()->SendNotification("Vous possédez déjà le sort.");
      CloseGossipMenuFor(player);//Fermeture du gossip
     }
     else//Si le joueur n'a pas le sort
     {
      CloseGossipMenuFor(player);//Fermture du gossip
      player->LearnSpell(46917, false);//Aprentissage du sort
      player->DestroyItemCount(TOKEN_ID, PRIX_SPELL_2, true);//Détruire l'objet de la l'inventaire enfonction du prix
      player->GetSession()->SendNotification("Félicitation pour votre achat.");
     }
    }
    else
    {
     CloseGossipMenuFor(player);//Fermeture du gossip
     player->GetSession()->SendNotification("Vous n'avez pas les objet nécessaire pour cette achat.");
    }
   break;
   case ACTION_GUERRIER_1 + 101://Remboursement du sort Poigne du titan de la classe Guerrier
    if(player->HasSpell(46917))//Si le joueur possède le sort
    {
     CloseGossipMenuFor(player);//Fermeture du gossip
     player->RemoveSpell(46917);//Supression du sort au joueur
     player->AddItem(TOKEN_ID, PRIX_SPELL_4);//ajouter l'objet de la l'inventaire enfonction du prix
     player->GetSession()->SendNotification("Remboursement effectué.");
    }
    else//Si le joueur n'a pas le sort
    {
     CloseGossipMenuFor(player);//Fermeture du gossip
     player->GetSession()->SendNotification("Vous n'avez pas ce sort.");
    }
   break;
  }
  return true;
      }
  };

  CreatureAI* GetAI(Creature* creature) const override
  {
      return new PNJ_SPELLAI(creature);
  }
};
```

```
void AddSC_PNJ_SPELL()
{
    new PNJ_SPELL();
}
```

kazuma

killit super tutoriel, merci beaucoup pour ce partage □

noc

Magnifique, le Mr Spell le plus aboutie que j'ai jamais vue, bravo

EIntemporel

Hello, je pense à ça : (Pour une amélioration du script)

Pourquoi ne pas créer une table pour chaque classe du genre

```
//initialisation des tableaux dynamique pour le guerrier
vector<string> NOM_SORT_GUERRIER;
vector<int> ID_SORT_GUERRIER;

//Ce premier spell aura pour ID dans les tableaux : 0
NOM_SORT_GUERRIER.push_back("Poigne du titan");
ID_SORT_GUERRIER.push_back(1001);

//Ce second spell aura pour ID dans les tableaux : 1
NOM_SORT_GUERRIER.push_back("Frénésie sanglante");
ID_SORT_GUERRIER.push_back(1002);

//Tu récupères tes valeurs avec une boucle for
//Tu peut te permettre de faire qu'une seule boucle puisque les deux tableaux auront les même tailles
//vu que tu vas initialiser les sorts dans les deux tableaux
ClearGossipMenuFor(player);
    for (int i(0); i<NOM_SORT_GUERRIER.size(); i++) {
        AddGossipItemFor(player, GOSSIP_ICON_MONEY_BAG, NOM_SORT_GUERRIER[i], GOSSIP_SENDER_MAIN, ID_SORT_GUERRIER[i]);
    }
SendGossipMenuFor(player, 1, me->GetGUID());
return true;
```

Au moins dans le futur tu n'aurais que les tableaux de chaque classes à éditer si tu souhaites rajouter ou enlever un sort, ainsi ça sera plus pratique pour toi à l'avenir :)

En tout cas merci quand même pour tout ça déjà !