

Spis treści:

Znikanie postaci po określonej kwestii dialogowej.....	3
Znikanie postaci po określonej kwestii dialogowej, z pojedynczym efektem wizualnym.....	3
Znikanie postaci po określonej kwestii dialogowej, z podwójnym efektem wizualnym.....	3
Inicjacja tekstu dowolnego BN, po wejściu na obszar wyzwalacza.....	4
Inicjacja dialogu przy wejściu na obszar wyzwalacza.....	4
Przejście z jednego modułu do drugiego po kliknięciu na obszar wyzwalacza lub przejściu przez drzwi.....	5
Przejście z jednego modułu do drugiego po wejściu na obszar wyzwalacza.....	5
Przejście z jednego modułu do innego, po określonej wypowiedzi.....	5
Animacja czynności postaci BN przy określonej wypowiedzi.....	6
Chaotyczne chodzenie BN.....	6
Kłęknięcie BN.....	7
BN jako zwłoki.....	7
BN siedzi na krześle (lub robi coś innego z obiektem).....	7
Otwieranie drzwi przez BN przy określonej wypowiedzi.....	8
Obiekt na którym może usiąść postać gracza.....	8
Dźwignia otwierająca drzwi.....	8
Zakończenie gry po określonej wypowiedzi.....	8
Postać BN spoczywa na ziemi w pozycji półsiedzącej.....	9
Rzucanie przez BN zaklęcia na siebie.....	9
Przyłączenie dowolnego BN jako towarzysza po określonej wypowiedzi.....	10
Odlączenie towarzysza po określonej wypowiedzi.....	11
Identyfikacja ekwipunku gracza przez towarzysza.....	12
Uzdrowienie przez BN przy określonej wypowiedzi.....	12
Dialog nawiązywany jednorazowo z graczem przez towarzyszącego mu BN'a.....	13
Pierścień translokacji (teleportacji).....	15
Zmiana obszaru po określonej wypowiedzi.....	16
Teleportacja po interakcji z obiektem.....	17
Rozjaśnienie obrazu po wkroczeniu na obszar.....	17

Śmierć z efektem wizualnym.....	18
Uzdrowienie po interakcji z obiektem.....	18
Zmiana liczby PD posiadanych przez postać gracza na początku modułu, a także zniszczenie dotychczasowego ekwipunku, i wyposażenie postaci w zaprojektowaną przez nas zbroję.....	19
Zmiana ilości PD posiadanych przez postać gracza na początku modułu, ale bez niszczenia dotychczasowego ekwipunku.....	23
Samozamykające się drzwi.....	24
Polecenie utrzymywania przez towarzysza średniej odległości.....	25
Polecenie utrzymywania przez towarzysza bliskiej odległości.....	26
Wyzwalacz wypowiedzi nocnej postaci, z odsyłaczem do punktu nawigacyjnego.....	26
Tekst pojawiający się gdy nasza postać dysponuje określoną ilością złota.....	27
Tekst nocny/dzienny.....	28
Brama otwarta za dnia, i zamknięta nocą.....	28
Tymczasowe ściemnienie ekranu (lub inny efekt wizualny) przy danej wypowiedzi.....	29
Założenie przez towarzysza określonej zbroi.....	29
BN spoczywa na ziemi pozornie martwy, ale istnieje możliwość interakcji i rozmowy.....	32
Modyfikacja charakteru.....	33
Otwarcie/ zamknięcie dowolnych drzwi przy określonej wypowiedzi.....	34
Odlot postaci skrzydlatej.....	35
Anulowanie akcji BN (chodzenia, siedzenia itp.).....	35
Nawiązanie dialogu z martwym przedmiotem.....	35
Stworzenia żyjące tylko nocą.....	36
Ciało (lub obiekt martwy) eteryczne, świecące.....	37
Stworzenie postaci przy określonej wypowiedzi, z efektem wizualnym.....	38
Awans towarzysza wraz z postacią gracza.....	38
Znikanie obiektu martwego kiedy zbliży się bohater.....	38
Zmiana obiektu w potwora, z efektem wizualnym, po zbliżeniu się gracza.....	39
Pojawienie się stworzenia po zamknięciu/otwarcu obiektu, z efektem wizualnym, w danym punkcie nawigacyjnym.....	40

Znikanie postaci po określonej kwestii dialogowej (wstawiane w dialogu, w zakładce „podjęte działania):

```
void main()
{
ClearAllActions();

ActionDoCommand(DestroyObject(OBJECT_SELF));

}
```

Znikanie postaci po określonej kwestii dialogowej, z pojedynczym efektem wizualnym (wstawiane w dialogu, w zakładce „podjęte działania):

```
void main()
{
ClearAllActions();

ActionDoCommand(DestroyObject(OBJECT_SELF));

effect eVis = EffectVisualEffect (nazwa efektu wizualnego, np: VFX_IMP_FLAME_M);

ApplyEffectToObject (DURATION_TYPE_PERMANENT, eVis, OBJECT_SELF);

ActionWait(1.0);

    ActionResumeConversation();

}
```

Znikanie postaci po określonej kwestii dialogowej, z podwójnym efektem wizualnym (wstawiane w dialogu, w zakładce „podjęte działania):

```
void main()
{
ClearAllActions();

ActionDoCommand(DestroyObject(OBJECT_SELF));

effect eDur = EffectVisualEffect(nazwa pierwszego efektu wizualnego);

effect eVis = EffectVisualEffect(nazwa drugiego efektu wizualnego);

ApplyEffectToObject (DURATION_TYPE_PERMANENT, eVis, OBJECT_SELF);

ApplyEffectToObject (DURATION_TYPE_PERMANENT, eDur, OBJECT_SELF);

ActionWait(1.0);
```

```

    ActionResumeConversation();
}

```

Inicjacja tekstu dowolnego BN, po wejściu na obszar wyzwalacza (wstawiane: właściwości – skrypty – on enter):

```

void main()
{
    object oPC = GetEnteringObject();
    if (GetLocalInt(oPC, "x_entering_text1") == FALSE)
    {
        AssignCommand (GetObjectByTag ("etykieta BNktóry ma wypowiedzieć kwestię"),
        SpeakString("tekst wypowiedzi"));
        SetLocalInt(oPC, "x_entering_text1", TRUE);
    }
}

```

Inicjacja dialogu przy każdym wejściu na obszar wyzwalacza; działa również z postaciami pojawiającymi się z wyzwalacza (wstawiane: właściwości – skrypty – on enter):

```

void main()
{
    object oEntered = GetEnteringObject();

    object oInfo = GetObjectByTag("etykieta postaci z którą gracz ma nawiązać dialog");

    if (GetIsPC(oEntered) &&
        GetLocalInt(oEntered,"NW_L_M2IMIEInfo") == 0 &&
        IsInConversation(oInfo) == FALSE)
    {
        AssignCommand(oEntered,ClearAllActions());

        AssignCommand(oInfo,ClearAllActions());

        AssignCommand(oInfo,ActionStartConversation(oEntered));
    }
}

```

Przejście z jednego modułu do drugiego po kliknięciu na obszar wyzwalacza lub przejściu przez drzwi (wstawiany: właściwości – skrypty – on click [jeśli wyzwalacz] lub *On Area transition click*, [jeśli drzwi – o ile w zakładce „obszar przejściowy” zaznaczony jest pkt nawigacyjny]):

```
void main()
{
    object oClicked = GetClickingObject();

    if (GetIsPC(oClicked))
    {
        DelayCommand(0.5,StartNewModule("tytuł modułu docelowego"));
    }
}
```

Przejście z jednego modułu do drugiego po wejściu na obszar wyzwalacza (wstawiany: właściwości – skrypty – on enter):

```
void main()
{
    object oEntered = GetEnteringObject();

    {
        DelayCommand(0.5,StartNewModule("tytuł modułu docelowego "));
    }
}
```

Przejście z jednego modułu do innego, po określonej wypowiedzi (wstawiany: zakładka „podjęte działania” przy wybranej kwestii dialogowej):

```
void main()
{
    object oPC = GetPCSpeaker();

    ApplyEffectToObject(DURATION_TYPE_PERMANENT, EffectCutsceneParalyze(), oPC);

    FadeToBlack(oPC);
}
```

```

        DelayCommand(0.5,StartNewModule("nazwa modułu"));
    }

```

Animacja czynności postaci BN przy określonej wypowiedzi (wstawiany: w dialogu, przy kwestii podczas której ma zostać wykonana czynność, w zakładce „podjęte działania”):

```

void main()
{
    object oInfo = GetItemPossessedBy(GetPCSpeaker(),"");
    SetLocalObject(OBJECT_SELF,"NW_L_TakeJournal",oInfo);
    ActionPauseConversation();
    ActionWait(0.5);
    ActionPlayAnimation(ANIMATION_FIREFORGET_SALUTE);
    ActionWait(1.0);
    ActionResumeConversation();
}

```

UWAGA: Jeśli zamiast przykładowego SALUTE wpisujemy inne polecenie, to postać wykona inne działanie. Poniżej rodzaje komend:

Czytanie – READ

Uklon królewski – BOW

Machanie ręką – GREETING

Salutowanie - SALUTE

Te same efekty możemy też uzyskać ustawiając odpowiednie polecenie: najpierw przy wybranej kwestii dialogowej wchodzimy w zakładkę „inne działania” i wybieramy z listy w lewym okienku na dole; prawe okienko służy do wybierania wyedytowanych uprzednio wpisów w dzienniku.

Chaotyczne chodzenie BN (wstawiany: właściwości - skrypty – on spawn / on heartbeat)

```

void main()
{
    ActionRandomWalk();
}

```

Kłęknięcie BN (wstawiany w zakładce „działania” przy określonej kwestii dialogowej, lub w skrypcie „on spawn” postaci, jeśli chcemy aby BN kłęczał przez cały czas):

```
void main()
{
  ActionPlayAnimation(ANIMATION_LOOPING_MEDITATE, 1.0, 780.0);
}
```

Lub:

```
void main()
{
  object oInfo = GetItemPossessedBy(GetPCSpeaker(),"");
  SetLocalObject(OBJECT_SELF,"NW_L_TakeJournal",oInfo);
  ActionPauseConversation();
  ActionPlayAnimation(ANIMATION_LOOPING_MEDITATE, 1.0);
  ActionResumeConversation();
}
```

BN jako zwłoki (wstawiany: właściwości – skrypty – on spawn, lub w zakładce “podjęte działania” przy określonej wypowiedzi, po której postać „umrze”):

```
void main()
{
  SetIsDestroyable (FALSE,FALSE);
  ApplyEffectToObject (DURATION_TYPE_PERMANENT, EffectDeath (TRUE,TRUE),
  OBJECT_SELF);
}
```

BN siedzi na krześle (lub robi coś innego z obiektem). (Wstawiany: właściwości postaci -skrypty – on spawn, lub w zakładce “podjęte działania” przy określonej wypowiedzi, po której postać wykona pożądana czynność):

```
void main()
{
  ActionSit (GetNearestObjectByTag ("ETYKIETA OBIEKTU (KRZESŁA)"));
}
```

(Jeśli zamienimy *Sit* na *Attack*, to BN będzie atakować dany obiekt; *jeśli na PickupItem* – postać pójdzie wziąć dany przedmiot – przydatne do kierowania BN w pożądanym kierunku)

Otwieranie drzwi przez BN przy określonej wypowiedzi (wstawiany: w zakładce „podjęte działania” przy określonej wypowiedzi po której postać otworzy drzwi, pod warunkiem że w jej ekwipunku znajduje się klucz do nich):

```
void main()
{
    ActionPauseConversation();
    ActionOpenDoor (GetNearestObjectByTag ("ETYKIETA OBIEKTU (drzwi)"));
    ActionResumeConversation();
}
```

Obiekt na którym może usiąść postać gracza (wstawiany: właściwości przedmiotu – skrypty – on used):

```
void main()

{

    object oChair = OBJECT_SELF;

    AssignCommand (GetLastUsedBy(),ActionSit(oChair));

}
```

Dźwignia otwierająca drzwi (Wstawiany: właściwości (dźwigni) – skrypty – on used)

```
void main()
{
    string sTag = GetTag (OBJECT_SELF);
    object oDoor = GetNearestObjectByTag ("ETYKIETA DRZWI");
    if (GetLocked (oDoor) == TRUE)
    {
        SetLocked( oDoor, FALSE);
        ActionOpenDoor(oDoor);
    }
    else
    {
        ActionCloseDoor (oDoor);
        SetLocked (oDoor, TRUE);
    }
}
```

Zakończenie gry (z muzyczką i literami) po określonej wypowiedzi (wstawiany: zakładka „podjęte działania” przy danej kwestii w edytorze dialogów):

```
void main()

{

    object oPC = GetPCSpeaker();
```



```

ApplyEffectToObject(DURATION_TYPE_PERMANENT, EffectCutsceneParalyze(), oPC);

FadeToBlack(oPC);

DelayCommand(3.0, EndGame("XP2_Closing"));

}

```

Postać BN spoczywa na ziemi w pozycji pólśiedzającej, ale żyje (wstawiany: właściwości postaci – skrypty – on spawn, lub w zakładkę „podjęte działania” jeśli postać ma upaść po określonej wypowiedzi):

```

void main()
{
ActionPlayAnimation(ANIMATION_LOOPING_SIT_CROSS, 0.5, 99999.);
}

```

Rzucanie przez BN zaklęcia na siebie (wstawiany: zakładka „podjęte działania” przy określonej wypowiedzi):

```

void main()
{
    object oPC = GetPCSpeaker();

    ActionPauseConversation();

    ActionCastFakeSpellAtObject(etykieta czaruj, np: SPELL_IMPLOSION, OBJECT_SELF);

    effect eVis = EffectVisualEffect (etykieta efektu wizualnego np: VFX_DUR_SANCTUARY);

    effect eDur = EffectVisualEffect(etykieta drugiego efektu wizualnego np: VFX_DUR_INVISIBILITY);

    ApplyEffectToObject (DURATION_TYPE_PERMANENT, eVis, OBJECT_SELF);

    ApplyEffectToObject (DURATION_TYPE_PERMANENT, eDur, OBJECT_SELF);

    ActionWait(1.0);

    ActionResumeConversation();
}

```

Etykiety niektórych czarów i efektów wizualnych:

SPELL_GREATER RESTORATION

SPELL_IMPLOSION

SPELL_SANCTUARY
 VFX_FNF_FIRESTORM
 VFX_DUR_INVISIBILITY
 VFX_DUR_SANCTUARY
 VFX_IMP_IMPROVE_ABILITY_SCORE
 VFX_IMP_DOOM
 VFX_IMP_BREACH
 VFX_IMP_FLAME_M
 VFX_IMP_CHAOS_HELP
 VFX_IMP_LIGHTNING_S
 VFX_IMP_LIGHTNING_M
 VFX_IMP_CHARM
 VFX_IMP_HEAD_HOLY
 VFX_IMP_RESTORATION_GREATER
 VFX_IMP_REDUCE_ABILITY_SCORE
 VFX_IMP_HOLY_AID

Przyłączenie dowolnego BN jako towarzysza po określonej wypowiedzi (wstawiany: zakładka „podjęte działania):

```

void main()

{
  object oHench = GetObjectByTag("ETYKIETA TOWARZYSZA");
  object oPC = GetPCSpeaker();
  AddHenchman(oPC, oHench);
}
  
```

Lub skrypt bez etykiety (ale przyłączyć można w ten sposób tylko BN’a z którym aktualnie rozmawiamy):

```

//::////////////////////////////////////
//:: X0_D1_HEN_REJOIN
//:: Copyright (c) 2002 Floodgate Entertainment
  
```

```
//::////////////////////////////////////
```

```
/*
```

Actions taken when a henchman rejoins his/her current
master.

```
*/
```

```
//::////////////////////////////////////
```

```
//:: Created By: Naomi Novik
```

```
//:: Created On: 09/13/2002
```

```
//::////////////////////////////////////
```

```
#include "x0_i0_henchman"
```

```
void main()
```

```
{
```

```
    HireHenchman(GetPCSpeaker());
```

```
}
```

UWAGA!

Zdarza się, że po przyłączeniu bohatera ten nie podąża za nami tak jak powinien, i nie reaguje na komendy, nawet te z okrągłego menu; przyczyną takiej sytuacji musi być występowanie w którymkolwiek z obszarów modułu drugiej postaci o tej samej etykiecie. Do sytuacji takiej może dojść np. wskutek scenariusza zakładającego czasowe rozstanie, i ponowne spotkanie po pewnym czasie, w innej lokacji. Gdybyśmy spotykali się z naszym odłączonym towarzyszem w tej samej lokacji w której go pozostawiliśmy, nic by się nie stało, ale jeśli np. po odłączeniu nasz towarzysz znika, a chcemy by podczas kolejnego spotkania miał inne wyposażenie, wygląd lub parametry , najłatwiej jest skopiować tę postać i umieścić w innej wybranej przez nas lokacji wraz ze wszystkimi modyfikacjami ekwipunku, nowym zestawem tekstów itd. W takim jednak przypadku konieczne trzeba dla tego drugiego wariantu towarzysza stworzyć nową etykietę.

Odłączenie towarzysza po określonej wypowiedzi (wstawiany: zakładka „podjęte działania):

```
//::////////////////////////////////////
```

```
//:: X0_D1_HEN_FIRED
```

```
//:: Copyright (c) 2002 Floodgate Entertainment
```

```
//::////////////////////////////////////
```

```
/*
```

```
Fires the current henchman and leaves the player with  
no henchman.
```

```
*/
```

```
//::////////////////////////////////////
```

```
//:: Created By: Naomi Novik
```

```
//:: Created On: 09/13/2002
```

```
//::////////////////////////////////////
```

```
#include "x0_i0_henchman"
```

```
void main()
```

```
{ ClearAllActions();
```

```
    FireHenchman(GetPCSpeaker());
```

```
}
```

Lub wpisując nazwę skryptu (jest w zasobach oficjalnych kampanii) w okienko „podjęte działania”:

x0_d1_hen_quits

x0_d1_hen_fired

Identyfikacja ekwipunku gracza przez towarzysza (jeśli posiada wystarczającą wiedzę) (wstawiany: poniższa nazwa skryptu, wpisana w okienko „podjęte działania” przy określonej wypowiedzi):

x1_hen_identify

Uzdrowienie przez BN przy określonej wypowiedzi (wstawiany: poniższa nazwa skryptu, wpisana w okienko „podjęte działania” przy określonej wypowiedzi):

nw_d1_templeheal

Dialog nawiązywany jednorazowo z graczem przez towarzyszącego mu BN'a (wstawiany: zakładka „tekst pojawia się gdy” przy linii dialogowej którą chcemy zainicjować):

```
//::////////////////////////////////////  
//:: X0_D2_HEN_LINEwartość klucza wyzwacza dla tego dialogu  
//:: Copyright (c) 2002 Floodgate Entertainment  
//::////////////////////////////////////  
/*  
Check to see if henchman should make this one-liner.  
*/  
//::////////////////////////////////////  
//:: Created By: Naomi Novik  
//:: Created On: 09/16/2002  
//::////////////////////////////////////
```

```
#include "x0_i0_henchman"
```

```
int StartingConditional()  
{  
    return (GetOneLiner() == wartość klucza wyzwacza);  
}
```

Drugi skrypt, który należy wstawić przy tej samej linii dialogowej, ale w okienko „podjęte działania”:

```
//::////////////////////////////////////  
//:: X0_D1_HEN_NOLINE  
//:: Copyright (c) 2002 Floodgate Entertainment  
//::////////////////////////////////////
```

```

/*
Clears the henchman's one-liner.

*/

//::////////////////////////////////////
//:: Created By: Naomi Novik
//:: Created On: 09/16/2002
//::////////////////////////////////

```

```
#include "x0_i0_henchman"
```

```

void main()
{
    SetOneLiner(FALSE);
}

```

Aby wszystko działało jak należy, należy także ustawić odpowiednie wyzwalacze na obszarze, na którym chcemy zainicjować dialog. Z nieznanych mi przyczyn, wyzwalacz stworzony specjalnie w tym celu nie działa poprawnie (przynajmniej u mnie), można jednak osiągnąć ten sam efekt używając innych wyzwalaczy: najpierw jako ustalenie zmiennej wyzwalacz typu "XP2 jedna linijka, nielosowa" o wartości klucza zgodnej z ustawioną w skryptach dialogu (podane wyżej) –ustala się to w zakładce „zaawansowane” wyzwalacza.

Następnie wyzwalacz typu "XP2 wtrącenie, nielosowe" który inicjuje właściwy dialog (ten też musi mieć zgodną wartość klucza).

Ważne!

1)Jeśli wyzwalacz typu „XP2 wtrącenie, nielosowe” zostanie nadeptnięty wcześniej niż "XP2 jedna linijka, nielosowa" , to uruchomiona zostanie podstawa dialogowa (dialog podstawowy, który uruchamia się gdy zagadniemy towarzysza w dowolnej chwili), przy czym późniejsze nadeptnięcie wyzwalaczy we właściwej kolejności, nie da już żadnego efektu. Aby tego uniknąć, należy sprytnie ustawić wyzwalacze w takich miejscach, aby gracz musiał uruchamiać wyzwalacze we właściwej kolejności, np. zastawiając pierwszym wyzwalaczem całą powierzchnię jedyne przejsia.

2)Postać towarzysząca musi posiadać niestandardowy zestaw skryptów (nie udało mi się ustalić który z nich odpowiada za właściwe reakcje na wyzwalacze), który można skopiować z BN'ów towarzyszących graczowi w oficjalnych kampaniach, np. LinuXP2 (reaguje na skrypty podane wyżej). Aby tego dokonać, należy otworzyć edycję oficjalnych modułów kampanii (najlepiej Hordy Podmroku – korzystają z najnowszych

skryptów), odnaleźć w palecie postaci BN'a który towarzyszy graczowi, wejść we właściwości, otworzyć zakładkę „skrypty” i skopiować je (na dole okienka jest specjalne polecenie: „zapisz ustawienia skryptu”). Następnie otwieramy swój własny moduł, odnajdujemy we własnej palecie postaci tę, którą chcemy jako towarzysza, PPM wchodzimy w jego właściwości, otwieramy zakładkę „skrypty”, na dole okienka naciskamy „wczytaj ustawienia skryptu”, i z listy wybieramy nazwę pod którą zapisaliśmy wcześniej skrypty towarzysza z oficjalnych kampanii.

I jeszcze jedno: dodatkowe dialogi jednorazowe nie stanowią „podstawy dialogowej” z daną postacią; można w nich jednak umieścić zmienne, które sprawią że podstawa się zmieni. Wówczas jednak, każde przypadkowe zainicjowanie wyzwalacza dialogu jednorazowego uruchomi nową podstawę, chyba że będzie to dialog jednorazowy umieszczony wyżej na liście, tzn. późniejszy niż nowa podstawa (najlepiej tak wszystko ustalić, aby wyzwalacze tych dialogów znajdowały się w newralgicznych miejscach, tak aby bohaterowie musieli przez nie przejść).

Pierścień translokacji (pozwala na teleportację w jedną stronę do określonego punktu)

Najpierw należy stworzyć **skrypt** następującej treści:

```
void main()
{
object oPC = GetItemActivator ();
object oTarget = GetObjectByTag ("etykieta punktu nawigacyjnego do którego chcemy się przenosić");
effect eVis = EffectVisualEffect (etykieta efektu wizualnego który ma towarzyszyć teleportacji);
DelayCommand (2.0, AssignCommand (oPC, JumpToObject (oTarget)));
ApplyEffectToObject (DURATION_TYPE_INSTANT, eVis,
```

następnie wchodzimy w menu górne edytora: **Edytuj – właściwości modułu – zdarzenia**, otwieramy skrypt *OnActivateItem* i pomiędzy ostatni a przedostatni nawias typu { wstawiamy to:

```
object oPC = GetItemActivator();
string sltem = GetTag(oltem);
if (sltem == "etykieta pierścienia") ExecuteScript("nazwa skryptu stworzonego na początku", oPC);
```

Zmiana obszaru po określonej wypowiedzi (wstawiany: w zakładkę „podjęte działania” przy wybranej wypowiedzi)

```
void main()
{
    object oPC = GetPCSpeaker();

    object oTarget = GetObjectByTag ("etykieta punktu nawigacyjnego");

    DelayCommand (2.0, AssignCommand (oPC, JumpToObject (oTarget)));
}
```

Ten sam efekt, ze ściemnieniem ekranu (rozjaśnić trzeba osobno, we właściwościach obszaru docelowego)

```
void main()
{
    object oPC = GetPCSpeaker();

    object oTarget = GetObjectByTag ("etykieta punktu nawigacyjnego");

    FadeToBlack(oPC);

    BlackScreen(oPC);

    ActionResumeConversation();

    DelayCommand (1.0, AssignCommand (oPC, JumpToObject (oTarget)));
}
```

Ten sam efekt z dodatkowym efektem wizualnym (np. jako zakłęcie teleportujące)

```
void main()
{
    object oPC = GetPCSpeaker();

    ActionPauseConversation();

    ActionCastFakeSpellAtObject(SPELL_IMPLOSION, OBJECT_SELF);

    effect eVis = EffectVisualEffect (etykieta efektu wizualnego np: VFX_IMP_LIGHTNING_S);
}
```



```

ApplyEffectToObject (DURATION_TYPE_PERMANENT, eVis, OBJECT_SELF);

ActionWait(1.0);

    ActionResumeConversation();

object oTarget = GetObjectByTag ("docelowy pkt nawigacyjny");

    DelayCommand (2.0, AssignCommand (oPC, JumpToObject (oTarget)));

}

```

Teleportacja po interakcji z obiektem (wstawiane: właściwości obiektu – skrypty – on click)

```

void main()
{
object oPC = GetLastUsedBy ();
object oTarget = GetObjectByTag ("ETYKIETA PUNKTU NAWIGACYJNEGO");
effect eVis = EffectVisualEffect (ETYKIETA EFEKTU WIZUALNEGO);
ApplyEffectToObject (DURATION_TYPE_INSTANT, eVis, oPC);
AssignCommand (oPC, JumpToObject (oTarget));
}

```

Rozjaśnienie przy wkroczeniu na obszar (wstawiane: w zakładkę “wydarzenia – on enter”)

```
// * Grovel initiates with the player
```

```

void main()

{

    if (GetLocalInt(OBJECT_SELF, "nGrovelTalk") == 1)

        return;

    object oPC = GetEnteringObject();

    if (GetIsPC(oPC) == TRUE)

    {

        //Fairy Monitor will talk with the first PC down to Undermountain

        BlackScreen(oPC);

        FadeFromBlack(oPC);

        object oMonitor = GetObjectByTag("q2bgrovel");

        DelayCommand(1.0, AssignCommand(oMonitor, ActionStartConversation(oPC)));
    }
}

```

```

        SetLocalInt(OBJECT_SELF, "nGrovelTalk", 1);

    }

}

```

Śmierć z efektem wizualnym (wstawiane: właściwości postaci, skrypty – on death)

```

void main()
{
    SetIsDestroyable (FALSE,FALSE);
    ApplyEffectToObject (DURATION_TYPE_PERMANENT, EffectDeath (TRUE,TRUE),
    OBJECT_SELF);
    effect eVis = EffectVisualEffect (nazwa efektu wizualnego np:VFX_FNF_FIRESTORM);

    ApplyEffectToObject (DURATION_TYPE_PERMANENT, eVis, OBJECT_SELF);

    ActionWait(1.0);

    ActionResumeConversation();
}

```

Uzdrowienie po interakcji z obiektem (wstawiane: właściwości obiektu - skrypty – on click)

```

void main()
{
    object oPC = GetLastUsedBy ();
    effect eHeal = EffectHeal (ilość PŻ jaką chcemy odzyskać);
    effect eVis = EffectVisualEffect (etykieta efektu wizualnego);
    ApplyEffectToObject (DURATION_TYPE_INSTANT, eVis, oPC);
    ApplyEffectToObject (DURATION_TYPE_INSTANT, eHeal, oPC);
}

```

Zmiana liczby PD posiadanych przez postać gracza na początku modułu, a także zniszczenie dotychczasowego ekwipunku, i wyposażenie postaci w zaprojektowaną przez nas **zbroję**
(wstawiany: edytuj - właściwości modułu – zdarzenia - on client enter)

```
//:////////////////////
```

```
//: Created By:  ?
```

```
//: Created On:  2.05.2013
```

```
//:////////////////////
```

```
// Na początku robimy trzy rzeczy:
```

```
// zabieramy obecne wyposażenie, dajemy minimalne nowe wyposażenie
```

```
// i ustalamy doswiadczenie na poziomie 0
```

```
void Dajemy(object oPC)
```

```
{
```

```
    string sCiuchy = "etykieta zaprojektowanej przez nas zbroi";
```

```
    if(GetLevelByClass(CLASS_TYPE_WIZARD, oPC) > 0 || GetLevelByClass(CLASS_TYPE_SORCERER, oPC) > 0 || GetLevelByClass(CLASS_TYPE_BARD, oPC) > 0)
```

```
    {
```

```
        sCiuchy = "NW_CLOTH002";
```

```
    }
```

```
    else if (GetLevelByClass(CLASS_TYPE_FIGHTER, oPC) > 0)
```

```
    {
```

```
        sCiuchy = "NW_CLOTH025";
```

```
    }
```

```
    AssignCommand(oPC, ActionEquipItem(CreateItemOnObject(sCiuchy, oPC), INVENTORY_SLOT_CHEST));
```

```

CreateltemOnObject("NW_WSWDG001", oPC);

CreateltemOnObject("NW_IT_TORCH001", oPC);
}

void Zabieramy(object oPC)
{
    int nZloto = GetGold(oPC);

    AssignCommand(oPC, TakeGoldFromCreature(nZloto, oPC, TRUE));

    object oPrzedmiot = GetItemInSlot(INVENTORY_SLOT_ARMS, oPC);
    if(GetIsObjectValid(oPrzedmiot))
        DestroyObject(oPrzedmiot);

    oPrzedmiot = GetItemInSlot(INVENTORY_SLOT_BELT, oPC);
    if(GetIsObjectValid(oPrzedmiot))
        DestroyObject(oPrzedmiot);

    oPrzedmiot = GetItemInSlot(INVENTORY_SLOT_BOLTS, oPC);
    if(GetIsObjectValid(oPrzedmiot))
        DestroyObject(oPrzedmiot);

    oPrzedmiot = GetItemInSlot(INVENTORY_SLOT_BOOTS, oPC);
    if(GetIsObjectValid(oPrzedmiot))
        DestroyObject(oPrzedmiot);

    oPrzedmiot = GetItemInSlot(INVENTORY_SLOT_CHEST, oPC);

```

```
if(GetIsObjectValid(oPrzedmiot))
```

```
    DestroyObject(oPrzedmiot);
```

```
oPrzedmiot = GetItemInSlot(INVENTORY_SLOT_CLOAK, oPC);
```

```
if(GetIsObjectValid(oPrzedmiot))
```

```
    DestroyObject(oPrzedmiot);
```

```
oPrzedmiot = GetItemInSlot(INVENTORY_SLOT_HEAD, oPC);
```

```
if(GetIsObjectValid(oPrzedmiot))
```

```
    DestroyObject(oPrzedmiot);
```

```
oPrzedmiot = GetItemInSlot(INVENTORY_SLOT_LEFTHAND, oPC);
```

```
if(GetIsObjectValid(oPrzedmiot))
```

```
    DestroyObject(oPrzedmiot);
```

```
oPrzedmiot = GetItemInSlot(INVENTORY_SLOT_LEFTRING, oPC);
```

```
if(GetIsObjectValid(oPrzedmiot))
```

```
    DestroyObject(oPrzedmiot);
```

```
oPrzedmiot = GetItemInSlot(INVENTORY_SLOT_NECK, oPC);
```

```
if(GetIsObjectValid(oPrzedmiot))
```

```
    DestroyObject(oPrzedmiot);
```

```
oPrzedmiot = GetItemInSlot(INVENTORY_SLOT_RIGHTHAND, oPC);
```

```
if(GetIsObjectValid(oPrzedmiot))
```

```
    DestroyObject(oPrzedmiot);
```

```

oPrzedmiot = GetItemInSlot(INVENTORY_SLOT_RIGHTRING, oPC);

if(GetIsObjectValid(oPrzedmiot))
    DestroyObject(oPrzedmiot);


oPrzedmiot = GetItemInSlot(INVENTORY_SLOT_ARROWS, oPC);

if(GetIsObjectValid(oPrzedmiot))
    DestroyObject(oPrzedmiot);


oPrzedmiot = GetItemInSlot(INVENTORY_SLOT_BOLTS, oPC);

if(GetIsObjectValid(oPrzedmiot))
    DestroyObject(oPrzedmiot);


oPrzedmiot = GetItemInSlot(INVENTORY_SLOT_BULLETS, oPC);

if(GetIsObjectValid(oPrzedmiot))
    DestroyObject(oPrzedmiot);


object oInv = GetFirstItemInInventory(oPC);
while(GetIsObjectValid(oInv))
{
    DestroyObject(oInv);
    oInv = GetNextItemInInventory(oPC);
}

}

void main()
{
    object oPC = GetEnteringObject();

```

```

if(GetIsPC(oPC) &&
    GetLocalInt(oPC, "init") == 0)
{
    SetXP(oPC, 0);

    SetXP(oPC, tu wpisać ilość PD jaka chcemy mieć na starcie);    // resetuje PC i ustawiam XP
na 17 poziom doswiadczenia

    Zabieramy(oPC);          // zabieram cały ekwipunek

    DelayCommand(0.5, Dajemy(oPC)); // daje ekwipunek minimalny

    SetLocalInt(oPC, "init", 1);    // zrobione
}
else
{
}
}

```

Zmiana ilości PD posiadanych przez postać gracza na początku modułu, ale bez niszczenia dotychczasowego ekwipunku (wstawiany: edytuj - właściwości modułu – zdarzenia - on client enter)

```

void Dajemy(object oPC)
{

}

void main()

```

```

{
    object oPC = GetEnteringObject();

    if(GetIsPC(oPC) && GetLocalInt(oPC, "init") == 0)
    {
        SetXP(oPC, tu wpisać ilość PD jaką chcemy dysponować na starcie);

        (oPC);

        DelayCommand(0.5, Dajemy(oPC));

        SetLocalInt(oPC, "init", 1);
    }
}

```

Samozamykające się drzwi (wstawianie: właściwości obiektu – skrypty - on open)

```

void main()
{
    ActionWait(5.0);

    if (GetIsOpen(OBJECT_SELF))
    {
        ActionCloseDoor(OBJECT_SELF);
    }
}

```


Polecenie utrzymywania przez towarzysza średniej odległości (wstawiane: w zakładkę „podjęte działania” przy wybranej wypowiedzi)

```
//::////////////////////////////////////
```

```
//:: Set Distance to 12ft
```

```
//:: NW_CH_DIST_12
```

```
//:: Copyright (c) 2001 Bioware Corp.
```

```
//::////////////////////////////////////
```

```
/*
```

```
    Set Follow Distance to 12ft (4m)
```

```
*/
```

```
//::////////////////////////////////////
```

```
//:: Created By: Preston Watamaniuk
```

```
//:: Created On: Nov 19, 2001
```

```
//::////////////////////////////////////
```

```
#include "NW_IO_GENERIC"
```

```
void main()
```

```
{
```

```
    SetAssociateState(NW_ASC_DISTANCE_2_METERS, FALSE);
```

```
    SetAssociateState(NW_ASC_DISTANCE_4_METERS);
```

```
    SetAssociateState(NW_ASC_DISTANCE_6_METERS, FALSE);
```

```
}
```

Polecenie utrzymywania przez towarzysza bliskiej odległości:

```
//://////////////////////////  
//: Set Distance to 6ft  
//: NW_CH_DIST_6  
//: Copyright (c) 2001 Bioware Corp.  
//://////////////////////////  
/*  
    Set Follow Distance to 6ft (2m)  
*/  
//://////////////////////////  
//: Created By: Preston Watamaniuk  
//: Created On: Nov 19, 2001  
//://////////////////////////  
#include "NW_IO_GENERIC"  
  
void main()  
{  
    SetAssociateState(NW_ASC_DISTANCE_2_METERS);  
    SetAssociateState(NW_ASC_DISTANCE_4_METERS, FALSE);  
    SetAssociateState(NW_ASC_DISTANCE_6_METERS, FALSE);  
}
```

Wyzwalacz wypowiedzi nocnej postaci, z odsyłaczem do punktu nawigacyjnego (uwaga! Tekst nocny jest w zasobach, ale nie u postaci): (wstawiane: właściwości – skrypty – on enter)

```
void main()  
{  
    object oPC = GetEnteringObject();
```

```

if (GetIsPC(oPC) && GetIsNight())
{
    object oNPC = GetObjectByTag("etykieta postaci");

    AssignCommand(oPC, ClearAllActions());

    ActionDoCommand(SetCommandable(FALSE, oPC));

    AssignCommand(oPC, JumpToLocation(GetLocation(GetWaypointByTag("etykieta punktu
nawigacyjnego"))));

    AssignCommand(oNPC, ClearAllActions());

    AssignCommand(oNPC, ActionStartConversation(oPC, "nazwa dialogu"));

    DelayCommand(1.0, SetCommandable(TRUE, oPC));
}
}

```

Tekst pojawiający się gdy nasza postać dysponuje określoną ilością złota (wstawiane: w zakładkę „tekst pojawia się gdy” przy wybranej wypowiedzi)

```

int StartingConditional()
{
    int iResult;

    iResult = GetGold(GetPCSpeaker()) >= 200;

    return iResult;
}

```

Tekst nocny/dzienny (w razie potrzeby zmienić Night na Day) (wstawiane: w zakładkę „tekst pojawia się gdy” przy wybranej wypowiedzi)

```
int StartingConditional()
{
    int iResult;

    iResult = GetIsNight();

    return iResult;
}
```

Brama otwarta za dnia, i zamknięta nocą (wstawianie: właściwości – skrypty – on heartbeat):

```
void main()
{
    if (GetIsNight())
    {
        if (GetIsOpen(OBJECT_SELF))
        {
            ActionCloseDoor(OBJECT_SELF);
        }

        if (!GetLocked(OBJECT_SELF))
        {
            SetLocked(OBJECT_SELF, TRUE);
        }
    }

    else if (GetIsDay())
    {
        if (!GetIsOpen(OBJECT_SELF))
```

```

{
    SetLocked(OBJECT_SELF, FALSE);
//    ActionOpenDoor(OBJECT_SELF);
}
}
}

```

Tymczasowe ściemnienie ekranu (lub inny efekt wizualny) przy danej wypowiedzi (wstawiany: zakładka „podjęte działania” przy danej kwestii w edytorze dialogów):

```

void main()
{
    object oPC = GetPCSpeaker();

    FadeToBlack(oPC);

    BlackScreen(oPC);

    effect eCiemnosc = EffectBlindness();

    ApplyEffectToObject(DURATION_TYPE_TEMPORARY, eCiemnosc, oPC, 3.0);

    ActionResumeConversation();

    FadeFromBlack(oPC);
}

```

Założenie przez towarzysza określonej zbroi (istniejącej w zasobach i zidentyfikowanej). W nawiasie nie etykieta, ale res ref schematu (najlepiej jeśli są takie same, wtedy uniknie się pomyłki). (wstawiany: zakładka „podjęte działania” przy danej kwestii w edytorze dialogów):

```

void main()
{
    object oHench = GetObjectByTag("Etykieta towarzysza");
}

```

```

{
    string sCiuchy = "resref schematu zbroi";

    if(GetLevelByClass(CLASS_TYPE_WIZARD, oHench) > 0 || GetLevelByClass(CLASS_TYPE_SORCERER,
oHench) > 0 || GetLevelByClass(CLASS_TYPE_BARD, oHench) > 0)
    {
        sCiuchy = " resref schematu zbroi ";
    }
    else if (GetLevelByClass(CLASS_TYPE_FIGHTER, oHench) > 0)
    {
        sCiuchy = " resref schematu zbroi ";
    }

    AssignCommand(oHench, ActionEquipItem(CreateItemOnObject(sCiuchy, oHench),
INVENTORY_SLOT_CHEST));
}
{
    int nZloto = GetGold(oHench);
    AssignCommand(oHench, TakeGoldFromCreature(nZloto, oHench,TRUE));

    object oPrzedmiot = GetItemInSlot(INVENTORY_SLOT_ARMS, oHench);
    if(GetIsObjectValid(oPrzedmiot))
        DestroyObject(oPrzedmiot);

    oPrzedmiot = GetItemInSlot(INVENTORY_SLOT_BELT, oHench);
    if(GetIsObjectValid(oPrzedmiot))
        DestroyObject(oPrzedmiot);
}

```

```
oPrzedmiot = GetItemInSlot(INVENTORY_SLOT_BOLTS, oHench);
```

```
if(GetIsObjectValid(oPrzedmiot))
```

```
    DestroyObject(oPrzedmiot);
```

```
oPrzedmiot = GetItemInSlot(INVENTORY_SLOT_BOOTS, oHench);
```

```
if(GetIsObjectValid(oPrzedmiot))
```

```
    DestroyObject(oPrzedmiot);
```

```
oPrzedmiot = GetItemInSlot(INVENTORY_SLOT_CHEST, oHench);
```

```
if(GetIsObjectValid(oPrzedmiot))
```

```
    DestroyObject(oPrzedmiot);
```

```
oPrzedmiot = GetItemInSlot(INVENTORY_SLOT_CLOAK, oHench);
```

```
if(GetIsObjectValid(oPrzedmiot))
```

```
    DestroyObject(oPrzedmiot);
```

```
oPrzedmiot = GetItemInSlot(INVENTORY_SLOT_HEAD, oHench);
```

```
if(GetIsObjectValid(oPrzedmiot))
```

```
    DestroyObject(oPrzedmiot);
```

```
oPrzedmiot = GetItemInSlot(INVENTORY_SLOT_LEFTHAND, oHench);
```

```
if(GetIsObjectValid(oPrzedmiot))
```

```
    DestroyObject(oPrzedmiot);
```

```
oPrzedmiot = GetItemInSlot(INVENTORY_SLOT_LEFTRING, oHench);
```

```
if(GetIsObjectValid(oPrzedmiot))
```

```
    DestroyObject(oPrzedmiot);
```

```
oPrzedmiot = GetItemInSlot(INVENTORY_SLOT_NECK, oHench);
```

```

if(GetIsObjectValid(oPrzedmiot))

    DestroyObject(oPrzedmiot);

oPrzedmiot = GetItemInSlot(INVENTORY_SLOT_RIGHTHAND, oHench);

if(GetIsObjectValid(oPrzedmiot))

    DestroyObject(oPrzedmiot);

}

}

```

BN spoczywa na ziemi pozornie martwy, ale istnieje możliwość interakcji i rozmowy (aby postać nie wstała od razu ale pozostała w pozycji półleżącej, należy zastosować osobny skrypt, umieszczony w edytorze dialogów przy pierwszej wypowiedzi). (wstawianie: właściwości – skrypty – on spawn)

```

#include "NW_O2_CONINCLUDE"

#include "NW_IO_GENERIC"

void main()

{

    int nObra = (GetCurrentHitPoints() - 2);

    ApplyEffectToObject(DURATION_TYPE_INSTANT,EffectDamage(nObra),OBJECT_SELF);

    DelayCommand(1.0,ActionPlayAnimation(ANIMATION_LOOPING_DEAD_FRONT,1.0,999999.0));

    //SetIsDestroyable(FALSE,FALSE);


    SetSpawnInCondition(NW_FLAG_HEARTBEAT_EVENT);    //OPTIONAL BEHAVIOR - Fire User
    Defined Event 1001

    SetSpawnInCondition(NW_FLAG_PERCIEVE_EVENT);    //OPTIONAL BEHAVIOR - Fire User
    Defined Event 1002

    SetSpawnInCondition(NW_FLAG_ATTACK_EVENT);    //OPTIONAL BEHAVIOR - Fire User
    Defined Event 1005

    SetSpawnInCondition(NW_FLAG_DAMAGED_EVENT);    //OPTIONAL BEHAVIOR - Fire User
    Defined Event 1006

```



```

    SetSpawnInCondition(NW_FLAG_DISTURBED_EVENT);    //OPTIONAL BEHAVIOR - Fire User
Defined Event 1008

    SetSpawnInCondition(NW_FLAG_END_COMBAT_ROUND_EVENT); //OPTIONAL BEHAVIOR - Fire
User Defined Event 1003

    SetSpawnInCondition(NW_FLAG_ON_DIALOGUE_EVENT);    //OPTIONAL BEHAVIOR - Fire User
Defined Event 1004

    SetSpawnInCondition(NW_FLAG_DEATH_EVENT);          //OPTIONAL BEHAVIOR - Fire User Defined
Event 1007

    SetSpawnInCondition(NW_FLAG_SPELL_CAST_AT_EVENT);

// DEFAULT GENERIC BEHAVIOR (DO NOT TOUCH)
*****
*****

    SetListeningPatterns(); // Goes through and sets up which shouts the NPC will listen to.
}

```

Modyfikacja charakteru (+3 -5. W zależności czy będzie **G czy **E**)**

```
#include "nw_i0_plot"
```

```

void main()

{

    AutoAlignE(DC_EASY, GetPCSpeaker());

}

```

Można także wpisać w okienko „podjęte działania” przy wybranej wypowiedzi nazwy gotowych skryptów:

```

nw_d1_small_evil (obniża charakter o 5 pkt)

nw_d1_mid_evil (obniża charakter o 7 pkt)

nw_d1_small_good (podwyższa charakter o 3 pkt)

nw_d1_mid_good (podwyższa charakter o 7 pkt)

nw_d1_high_good (podwyższa charakter o 10 pkt)

nw_d1_high_evil (obniża charakter o 10 pkt)

```

Otwarcie dowolnych drzwi przy określonej wypowiedzi. **Dodatkowy efekt – zabranie złota. Można to wyciąć, jeśli nie chcemy tego efektu** . (wstawianie: zakładka „podjęte działania” przy wybranej wypowiedzi w edytorze dialogów)

```
void main()
{
    object oDrzwi = GetObjectByTag("etykieta drzwi");

    if (GetLocked(oDrzwi))
    {
        SetLocked(oDrzwi, FALSE);
    }

    TakeGoldFromCreature(ilość złota, GetPCSpeaker());
}
```

Analogicznie, żeby zamknąć drzwi przy wypowiedzi:

```
void main()
{
    object oDrzwi = GetObjectByTag("etykieta drzwi");

    {
        SetLocked(oDrzwi, TRUE);
    }

    TakeGoldFromCreature(200, GetPCSpeaker());
}
```

WAŻNE! Powyższe skrypty wymagają, aby postać z którą nawiązaliśmy dialog podczas którego ma nastąpić otwarcie, posiadała w ekwipunku klucz do drzwi które mają zostać otwarte. Z nieznanych przyczyn, skrypty nie działają w przypadku bram miejskich i bram w ogrodzeniach zewnętrznych (kamiennych i drewnianych), nadają się natomiast do wszystkich drzwi w budynkach (także do tych wielkich, np. zamkowych)

odlot postaci skrzydlatej (wstawianie: zakładka „podjęte działania” przy wybranej wypowiedzi w edytorze dialogów)

```
void main()
{
    object oSleep = GetObjectByTag("etykieta postaci");

    SetLocalInt(GetModule(), "HX_SLEEPING_INVIS_SILENT", TRUE);

    ApplyEffectToObject(DURATION_TYPE_INSTANT, EffectDisappear(), oSleep);
}
```

Anulowanie akcji BN (chodzenia, siedzenia itp.) (wstawianie: zakładka „podjęte działania” przy wybranej wypowiedzi w edytorze dialogów)

```
void main()
{
    ClearAllActions();
}
```

Nawiązanie dialogu z martwym przedmiotem (np. stołem alchemicznym, w celu przygotowania napoju) (wstawiany: właściwości - skrypty – on Click)

```
void main()
{
    object oPC = GetLastUsedBy ();

    {
        object oNPC = GetObjectByTag("etykieta przedmiotu przeznaczonego do interakcji");

        AssignCommand(oPC, ClearAllActions());
    }
}
```

```

    ActionDoCommand(SetCommandable(FALSE, oPC));

    AssignCommand(oNPC, ClearAllActions());

    AssignCommand(oNPC, ActionStartConversation(oPC, "nazwa dialogu"));

    DelayCommand(1.0, SetCommandable(TRUE, oPC));

}

}

```

WAŻNE! Każdy taki obiekt musi posiadać inną etykietę, a zatem i skrypty muszą być różne. Trzeba też pamiętać aby we właściwościach obiektu zaznaczone było pole „można używać”

Stworzenie żyjące tylko nocą, a w dzień spalające się i znikające (można je również umieścić w wyzwalaczu o odległym punkcie przywoływania, jeśli chcemy aby gracz spotykał je wyłącznie w określonej porze dnia) (umieszczane: właściwości stworzenia – skrypty – on spawn)

```

void main()

{

    if (GetIsDay())

    {

        SetIsDestroyable (TRUE,TRUE);

        ApplyEffectToObject (DURATION_TYPE_PERMANENT, EffectDeath (TRUE,TRUE),
        OBJECT_SELF);

        effect eVis = EffectVisualEffect (VFX_IMP_FLAME_M);

        ApplyEffectToObject (DURATION_TYPE_PERMANENT, eVis, OBJECT_SELF);

        ActionWait(1.0);

        ActionResumeConversation();

    }

}

```

To samo, ale z efektem poświaty i półprzezroczystości stworzenia

```
void main()
{
    if (GetIsDay())

    {

        SetIsDestroyable (TRUE,TRUE);

        ApplyEffectToObject (DURATION_TYPE_PERMANENT, EffectDeath (TRUE,TRUE),
        OBJECT_SELF);

        effect eVis = EffectVisualEffect (VFX_IMP_FLAME_M);

        ApplyEffectToObject (DURATION_TYPE_PERMANENT, eVis, OBJECT_SELF);

        ActionWait(1.0);

        ActionResumeConversation();

    }

    {

        ApplyEffectToObject(DURATION_TYPE_PERMANENT,
        EffectVisualEffect(VFX_DUR_ETHEREAL_VISAGE), OBJECT_SELF);

    }

}
```

Ciało (lub obiekt martwy) eteryczne, świecące (wstawiane: właściwości – skrypty – on spawn)

```
void main()
{

    ApplyEffectToObject(DURATION_TYPE_PERMANENT,
    EffectVisualEffect(VFX_DUR_ETHEREAL_VISAGE), OBJECT_SELF);

}
```

Stworzenie postaci przy danej wypowiedzi, w **punkcie nawigacyjnym**, z **efektem wizualnym w dowolnym punkcie nawigacyjnym** (wstawiane: zakładka „podjęte działania” przy wybranej wypowiedzi w edytorze dialogów):

```
void main()
{
    ClearAllActions();

    CreateObject(OBJECT_TYPE_CREATURE,"etykieta
    stworzenia",GetLocation(GetObjectByTag("etykieta pkt nawigacyjnego w którym stwór ma się
    pojawić")));

    ApplyEffectAtLocation(DURATION_TYPE_TEMPORARY, EffectVisualEffect(etykieta efektu wizualnego
    towarzyszącego przywołaniu
    np:VFX_FNF_SUMMON_UNDEAD),GetLocation(GetObjectByTag("etykieta pkt nawigacyjnego dla
    efektu wizualnego"))));
}
```

Uwaga! Powyższy skrypt działa wyłącznie z istotami z oficjalnej palety (nie wiem dlaczego)

Awans towarzysza wraz z postacią gracza (wstawiane: właściwości modułu – zdarzenia – okienko *on player level up*) :

```
x1_playerlevelup
```

Znikanie obiektu martwego kiedy zbliży się bohater (wstawiane: właściwości - skrypty – on heartbeat):

```
void main()
{
    object oCreature = GetNearestCreature(CREATURE_TYPE_PLAYER_CHAR, PLAYER_CHAR_IS_PC);
    if (GetIsObjectValid(oCreature) == TRUE && GetDistanceToObject(oCreature) < 10.0)
    {
        DestroyObject(OBJECT_SELF, 0.5);
    }
}
```

Zmiana obiektu w **potwora (tylko z oficjalnej palety) z **efektem wizualnym**, po zbliżeniu się gracza**
(wstawiane: właściwości – skrypty – on heartbeat)

```
//:://////////////////////////////////////

//:: NW_O2_SKELETON.nss

//:: Copyright (c) 2001 Bioware Corp.

//:://////////////////////////////////////

/*

    Turns the placeable into a skeleton

    if a player comes near enough.

*/

//:://////////////////////////////////////

//:: Created By:  Brent

//:: Created On:  January 17, 2002

//:://////////////////////////////////////

void ActionCreate(string sCreature, location lLoc)

{

    CreateObject(OBJECT_TYPE_CREATURE, sCreature, lLoc);

}

void main()

{

    object oCreature = GetNearestCreature(CREATURE_TYPE_PLAYER_CHAR, PLAYER_CHAR_IS_PC);

    if (GetIsObjectValid(oCreature) == TRUE && GetDistanceToObject(oCreature) < 10.0)

    {

        effect eMind = EffectVisualEffect(VFX_FNF_SUMMON_UNDEAD);

        string sCreature = "NW_MUMMY";

        // * 10% chance of a skeleton chief instead

        if (Random(100) > 90)

        {
```

```

    sCreature = "NW_MUMFIGHT";
}

location lLoc = GetLocation(OBJECT_SELF);

DelayCommand(0.3, ActionCreate(sCreature, lLoc));

ApplyEffectAtLocation(DURATION_TYPE_INSTANT, eMind, GetLocation(OBJECT_SELF));

SetPlotFlag(OBJECT_SELF, FALSE);

DestroyObject(OBJECT_SELF, 0.5);

}
}

```

Pojawienie się **stworzenia** (z oficjalnej palety) po zamknięciu/otwarcu obiektu, z **efektem wizualnym**, w danym punkcie nawigacyjnym (wstawiane: właściwości – skrypty – on open/ on close)

```

void main()
{
    ClearAllActions();

    CreateObject(OBJECT_TYPE_CREATURE, "NW_MUMFIGHT", GetLocation(GetObjectByTag("etykieta
pkt nawigacyjnego stworzenia")));

    ApplyEffectAtLocation(DURATION_TYPE_TEMPORARY,
    EffectVisualEffect(VFX_FNF_SUMMON_UNDEAD), GetLocation(GetObjectByTag("etykieta pkt
nawigacyjnego efektu wizualnego"))));
}

```

Polecenia do wpisania w okienko „txt pojawia się gdy” przy wybranej warunkowej wypowiedzi:

nw_d2_gwnd01	(01 - 03) gdy postać gracza jest ranna (nie działa poprawnie)
nw_d2_gwpm02	gdy postać gracza trzyma broń
nw_d2_gnaked	gdy postać gracza nie ma na sobie ubrania
x0_d2_hen_died	gdy BN został właśnie wskrzeszony (nie działa poprawnie)