CONVERSATION SCRIPTS FOR EE

v1.00 Proleric 2021

Unlock the power of conversation script parameters in EE.

Until now, NWN conversations typically required many tiny scripts, which were not easy to manage, lacked meaningful names, and duplicated effort.

This script collection might be all you need for single player modules in future.

Conversation lines can be made conditional on

- Quest status
- Module or NPC flags
- Gold
- Quest items
- Speaking a line only once
- Custom code

Action taken can include

- Setting module or NPC flags
- Giving and taking gold, XP or items
- Custom code

Includes demo module and erf with examples.

Simple, easy to use, documented.

Most functions need no scripting. Gold, XP and items are entered as data into a ready-made template script. Custom code requires a basic knowledge of NWScript, obviously.

All scripts and parameters are optional.

Caveat

These scripts are based on an analysis of common conversation cases, as well as experience of module building using pseudo-parameters and EE parameters in NWN conversations. The open-ended customisation features should cater for every situation.

However, I've made trade-offs between legibility and productivity. It would be possible, for example, to reduce the three conditional scripts to one by using more parameters, or simplify the meaning of parameters by having more scripts. I've put this package together fairly quickly, to suit my own requirements for another project.

So, while these scripts illustrate the power of parameters, feel free to rewrite them to your preference.

Definitions

- **NPC** includes any object that can hold a conversation (creatures, doors and placeables)
- **Flag** means a local integer with values TRUE or FALSE (and the string used to name it)
- **Reward** means any transaction in which the PC must possess, gain and/or lose gold, XP or items
- **Possess** means that the item is in the inventory or equipment of a PC or their henchmen

COMMON SCRIPT PARAMETERS		
Name	**Value**	**Description**
t	see scripts	True. For conditional scripts, returns TRUE if the condition implied by Value is true. For action scripts, sets a flag to TRUE (or takes other action specified by Value).
f	see scripts	False. For conditional scripts, returns FALSE if the condition implied by Value is true. For action scripts, clears a flag (or takes other action specified by Value). t and f are mutually exclusive. Omit both if only the DoOnce function is required.
DoOnce CheckOnce	[label]	These parameters are only required for lines which are to be spoken once. [label] is any convenient string that identifies the line, for example: describe pirates An NPC must only use a label on one line, but other NPCs can use the same labels. If t or f are also specified, with a condition that is false, the conversation line will not appear, but remains eligible to be spoken once at a later date. An NPC line with the parameter DoOnce [label] on the conditional script will only appear once. After that, the script will return FALSE. To keep offering a line to the PC until they choose it once, specify CheckOnce [label] on the conditional script and DoOnce (with the same label) on the action script.

CONDITIONAL SCRIPTS			
Script	**Parameter Name**	**Parameter Value**	**Description**
bh_check_quest	t f	[journal_tag] =|<|> [entry]	Examples: jt_pirate=1 jt_treasure>3 The symbols !=, <=, and >= are not supported because f=, f< and f> mean the same thing.
	DoOnce CheckOnce	[label]	
bh_check_flag	t f	[flag]	By default, returns a module flag of that name. You can also add custom code (see script for examples) in which case [flag] identifies a custom condition, evaluated every time. This can be used for module-wide complex conditions, for example, "when jt_pirate is 1 and the PC is a Rogue".

			To avoid clutter, it's recommended to use this only for conditions that arise in many conversations. NPC-specific custom code is better organised in bh__check.
	DoOnce CheckOnce	[label]	
bh__check	t f	[flag]	By default, returns an NPC flag of that name. If [flag] is a Reward (see below), the condition will be true when the PC has any gold or items required for the transaction. You can also add tag-based custom code (see script for examples) in which case [flag] identifies a custom condition which is evaluated every time. This can be used for NPC-specific complex conditions, for example, "at night and jt_treasure is greater than 3". [flag] may specify both a Reward and a custom condition, in which case the script will be true only when both reward and custom conditions are met. If either a reward or custom condition is specified, the NPC flag of that name is ignored.
	DoOnce CheckOnce	[label]	

ACTION TAKEN SCRIPTS			
Script	**Parameter Name**	**Parameter Value**	**Description**
bh_set_flag	t f	[flag]	By default, sets or clears a module flag of that name. You can also add custom code (see script for examples) in which case [flag] identifies a custom action, which can be any necessary lines of code, including calls to other scripts. In this case, no flag is set or cleared automatically. To avoid clutter, it's recommended to use this only for actions that arise in many conversations. NPC-specific custom code is better organised in bh__action.
	DoOnce CheckOnce	[label]	
bh__action	t f	[flag]	By default, sets or clears an NPC flag of that name. If [flag] is a Reward (see below), the PC will receive or lose any specified gold, XP and items.

			You can also add tag-based custom code (see script for examples) in which case [flag] identifies a custom action, which can be any necessary lines of code, including calls to other scripts. Special flag values ConversationEnd and ConversationAbort can be used to specify custom code for those events. [flag] may specify both a Reward and a custom action, in which case both will be executed. If either a Reward or custom condition is specified, the NPC flag of that name is ignored. Best practice is to organise all code for an NPC in one place, as shown in the example.
	DoOnce CheckOnce	[label]	

Rewards

Optional.

The script bh__rewards can be used to specify a custom reward table for the entire module.

This can be very powerful, as the management of gold, XP and items is entirely data-driven, to eliminate scripting errors and reduce testing time, and is all visible in one place.

A reward transaction needn't necessarily give the PC anything. It also covers the case where the PC must have certain items or gold before a conversation line becomes available, and the removal of gold, XP or items.

Comments and examples in the script should help. As an overview, the information required for each Reward is

- NPC
- Flag - a string identifying the conversation line in question
- Gold - the amount of gold to be given (or taken, if the value is negative)
- XP - the experience points to be given (or taken, if the value is negative)
- CheckItem - items the PC must possess which they will still keep after the transaction
- TakeItem - items that that PC must possess which will be destroyed by the transaction
- GiveItem - items the NPC will give the PC (which must be in the NPC's inventory)
- CreateItem - items that will be created in the PC's inventory (N.B. resref)

Each of the four item lists is comma-delimited, without blanks. For example,

Apple

Apple,Pear,Banana

The first three lists are item tags, but CreateItem is necessarily a list of items resrefs.

In the sample module, notice how the merchant conversation checks that the PC has the necessary gold and items on the PC line "Yes", then repeats the checks on the line that delivers the rewards. This prevents an exploit, where the PC has the necessary, but drops them before clicking "Yes". This is also documented in bh__rewards.

The erf contains a version of x3_mod_pre_enter that executes bh__rewards. If you already have that script, don't allow the erf to overwrite it - just ensure bh__rewards is executed OnClientEnter.

Conversation End

At the time of writing, the EE toolset does not allow script parameters to be specified for conversation normal end or abort.

Hopefully, this was an oversight - a fix has been requested.

Meanwhile, this package includes optional scripts

bh_conv_end

bh_conv_abort

which pass the flags ConversationEnd and ConversationAbort to bh__action.

Custom code can be specified for those conditions in the bh__action block for that NPC, which keeps all the code for the NPC in one place.

By default, the standard NWN walk waypoints will be executed as usual.

Library

The library bh__lib is required by all of these scripts.

Functions which may be useful in custom code include GetJournalQuestEntry(), which returns the current value of a quest, and GetDoneOnce(), which determines whether the PC has already seen a conversation line (and therefore might know the NPC, or some information they divulged).

Multiplayer

By design, these scripts won't work for multiplayer out of the box.

I don't have the resources to test this, but they might well work if you

- Replace GetFirstPC() with GetPCSpeaker()
- Add the PC as a parameter to GetJournalQuestEntry()

throughout.