xp_enhancedFeatures

V1.1.15 – 2026-02-26 by Septirage

Guide for Admin 3

Requirements 3

Installation of xp_enhancedFeatures : 3

List of xp_enhancedFeatures.ini options 3

loglevel: 3

EnhanceStoreRetrieve: 3

PatchDestroy: 4

PatchSetTag: 4

PatchAddNewTag: 4

PatchCloneArea: 4

FixCalcSafeLocation: 5

MaxCalcSafeLocation: 5

FixGetEffectType: 5

EnhanceGetResRef: 6

FixItemPropertyLoad: 6

FixItmPrpDmgBonus: 6

FixSaveThrow: 6

FixImmunityOverflow: 6

FixWildShapeUsage : 7

FixNumberOfFeatUsages : 7

FixDodgeVSDamageTypeRemoval : 7

FixDecreaseSpellOnBonusLoss: 7

FixItemStacking : 8

KeepLocalVarOnSplit : 8

FixSetFirstName : 8

DisableTumbleAC: 8

DisableSpellCraftSave: 8

DisableTrapSynergy: 8

MonkWeaponList: 8

AreaTypeBitX: 8

SpeedFeatsFile: 9

SkillFeatsFile: 9

WeaponFinesseFile: 9

ReduceSpeedFile: 9

OnGoldChangeScript: 9

GlobalOnClientEnterScript: 9

StartTransitionScript: 9

OnItemStackScript: 10

OnSplitStackScript: 10

OnItemBaseCostCalculationScript: 10

FixHasInfiniteSpellClass: 10

DefineInfiniteSpells: 10

AllowSetAreaTime: 10

Rules System Files 11

Common: 11

SpeedFeatEFF: 13

SkillFeatEFF: 14

WeaponFinesseEFF: 15

ReduceSpeedEFF: 15

HitPointFeatsEFF: 16

Specific Scripts 17

OnGoldChange : 17

GlobalOnClientEnter : 17

StartTransition : 17

OnItemStack : 18

OnSplit : 18

OnItemBaseCostCalculation : 19

Changelog 20

Guide for Admin

Requirements

To use the xp_enhancedFeature plugin you will need :

- Nwnx4

Installation of xp_enhancedFeatures :

Put **xp_enhancedFeatures.dll** and **xp_enhancedFeatures.ini** in your nwnx4 folder.

Add the **nwnx_enhancedfeatures.nss** and **nwnx_enhancedfeatures_spell.nss** files in your module (or import it with xp_enhancedfeatures.erf), this files contain all the functions you can use and will be detailed latter in this documentation.

Look at our website (https://septirage.com/nwn2/d/Plugins?id=xpenhancedfeatures) to have an overview.

List of xp_enhancedFeatures.ini options

xp_enhancedFeatures use at least one ini file. This ini file can also define other ini files, as we will see below

loglevel:

Adjust the log verbosity level as needed (from 0 to 5).

EnhanceStoreRetrieve:

Set to 1 to enhance the possibility of **StoreCampaignObject** and **RetrieveCampaignObject**.

By default, these functions are limited to Creatures and Items. However, with this patch, they will also work for Placeables, Doors, Triggers, Waypoints, Lights, and PlacedEffects.

Note: If you use **SQLStore/RetrieveObject**, these functions will also be able to store the additional object types in a database, as they rely on **Store/RetrieveCampaignObject**.

PatchDestroy:

Set to 1 to fix the **DestroyObject** function.

This patch ensures that the function works correctly on Areas and prevents memory leaks when deleting an object with an inventory (such as a Creature or Chest).

Previously, if an object with inventory was deleted, the object itself would be deleted, but not all items in it.

This would leave the items inaccessible and cause a memory leak.

This patch fixes this issue, and the game will now correctly delete all items in the inventory when the object is destroyed.

Note : It's important to note that this bug occurs not only on explicit nwnscript calls to DestroyObject, but also whenever an object is destroyed. One particularly significant example of this is when a player logs out, the PC is destroyed, causing memoryleak each time. Obviously, this patch will fix all occurrence of the bug.

PatchSetTag:

Set to 1 to fix the **SetTag** function.

Bug Explanation:

Without the patch, due to a bug in the engine, when you change an object tag, this object will be returned by GetObjectByTag(newTag) AND GetObjectByTag(oldTag). The patch fix this behavior and make the object retrievable only by the newTag.

PatchAddNewTag:

Set to 1 to fix tag insertion and adjust how objects with the same tag are ordered.

Bug Explanation:

If you have several objects with the same tag, you can retrieve them by incrementing the second parameter of GetObjectByTag. Transitions use the one with index 0.

Without the patch, when you create a new object with an already existing tag (or use SetTag with an existing one), the “new” object is inserted at the beginning (index 0).
With the patch, such objects are inserted at the end, preserving existing indices and preventing transitions from breaking.

PatchCloneArea:

Set to 1 to patch the **CreateInstancedAreaFromSource** function.

The Patch will allow to use this function with all area, even the one created by a previous call of CreateInstancedAreaFromSource. It will also add the ID of original Area as CreatorID to the new one.

The CreatorID can be retrieved by the nwnscript function GetAreaCreatorID_xpAM. Added by the AspectManager plugin.

FixCalcSafeLocation:

Set to 1 to patch the **CalcSafeLocation** function.

By default, the parameter *location lPosition* is purely a position, so the Area tested will be the oCreature current one. If you set this to 1, CalcSafeLocation will correctly test the full location (so with area) if bWalkStraightLineRequired is to FALSE.

MaxCalcSafeLocation:

By default, the Max value of fSearchRadius is 5.0. With this option you can change that.

FixGetEffectType:

Set to 1 to fix the **GetEffectType** function.

By default, several effects return EFFECT_INVALIDEFFECT. With this fix, they will return the following values: (those values are present in nwnx_enhancedfeatures.nss file)

```
const int EFFECT_TYPE_CUTSCENEDOMINATED = 111;
const int EFFECT_TYPE_DEATH = 112;
const int EFFECT_TYPE_KNOCKDOWN = 113;
const int EFFECT_TYPE_DAMAGE = 114;
const int EFFECT_TYPE_HEAL = 115;
const int EFFECT_TYPE_LINKEFFECT = 116;
const int EFFECT_TYPE_MODIFYATTACK = 117;
const int EFFECT_TYPE_LOWLIGHTVISION = 118;
const int EFFECT_TYPE_DARKVISION = 119;
const int EFFECT_TYPE_DISAPPEAR = 120;
const int EFFECT_TYPE_APPEAR = 121;
const int EFFECT_TYPE_SETSCALE = 122;
const int EFFECT_TYPE_SEETRUEHPS = 123;
const int EFFECT_TYPE_BABMINIMUM = 124;
const int EFFECT_TYPE_SUMMONCOPY = 125;
const int EFFECT_TYPE_SUMMONCREATURE = 126;
```

EnhanceGetResRef:

Set to 1 to enable the fix.

By setting to 1, you will get the AreaResRef with GetResRef(oArea) instead of empty string.

FixItemPropertyLoad:

Set to 1 to enable the fix.

Bug Explanation:

Without the patch, if a player logs in with items that provide bonuses equipped, the bonuses will not be removed as they should be when the item is unequipped. With the patch, this issue is fixed and the bonuses will be properly removed when the item is unequipped.

FixItmPrpDmgBonus:

Set to 1 to enable the fix.

Bug Explanation:

Without the patch, the functions *ItemPropertyDamageBonus*{xxx} (i.e., all functions starting with *ItemPropertyDamageBonus*, such as *ItemPropertyDamageBonus*, *ItemPropertyDamageBonusVsRace*, etc.) could incorrectly create a DamageBonus Burst property.

With the patch, this issue is fixed and the property will now be a correct DamageBonus. To intentionally create a Burst property, add `| 0x80000000` to the last parameter (the *int nDamage* one).

FixSaveThrow:

Set to 1 to enable the fix.

Bug Explanation:

Without the fix, Will and Reflex saving throws were not correctly subject to automatic failure. This meant that if the total roll was higher than the DC and the roll was a natural 1, the save still succeeded instead of automatically failing as it should have.

FixImmunityOverflow:

Set to 1 to enable the fix.

Bug Explanation:

Without the patch, if you have more than 125% immunity stacked for a specific damage type, it becomes vulnerability (and vice versa for vulnerability).

With the patch, this issue is fixed, and immunity/vulnerability is now properly capped at 100%.

FixWildShapeUsage :

Set to 1 to enable the fix.

Bug Explanation:

Without the fix, if the PC has the “Extra Wild Shape” feat, “*Oaken Resilience”* and “*Elephant's Hide”* will restore charges of “Wild Shape” instead of consuming them.

FixNumberOfFeatUsages :

Set to 1 to enable the fix.

Bug Explanation:

Without the fix, if the PC has the “Extra XXX” feat, when he will log in, the number of charges of feat XXX will not be properly restored but increased by the value added by “Extra XXX”. The fix properly retrieve the right amount of feat usage.

Example: If you have “Extra rage”. Each time you will login, you will have “LastNumberOfRageUsage +2” (still correctly maxed though).

FixDodgeVSDamageTypeRemoval :

Set to 1 to enable the fix.

Bug Explanation:

Without the fix, if the PC unequips an item with a property “Dodge AC Bonus vs Damage Type”, the game will remove this specific bonus but also the same amount of Dodge AC. This will repeat each time, causing the Dodge AC value of the creature to be reduced again and again.

FixDecreaseSpellOnBonusLoss:

Explanation:

Removing a bonus spell slot (from item or buff, with spell slot bonus or spellCast Ability bonus), on a Sorceror, Bard, or similar caster penalizes the player a spell slot, even if they had not rested since receiving the bonus spell slot.
Therefore repeatedly gaining and losing bonus spell slots would consume additional spell slots each time.

Example :

You have a Sorcerer with 5 lvl 1 spell per day with a ring that grant you +1 lvl1 spell. If you unequip it, your number of available spell will reduce to 4. It’s normal. But if you equip it and unequip it again, your number of available spell will reduce to 3 etc.

This option allow you to choose how to react for this.

Remove this option or set it to 0 to keep the base game behavior.
Set it to 1 to reduce only if the new maxSpellPerDay is lower than your current available spell.
Set it to 2 to never reduce this number.

FixItemStacking :

Set to 1 to enable the fix.

Bug Explanation:

Without the fix, the game allow item to stack if their number of “passive” properties (not on activation) are the same. Even if the properties differ. So, a player can stack a pack of +1 piercing arrow in a stack of +5 arrow. Activate the fix will properly check the properties instead of their number only.

KeepLocalVarOnSplit :

Set to 1 to enable the fix.

By default, when you split a stack, the resulting stack lost all the localvar that can be present.

This fix change that by copying all the localvar so the new stack will have the same as the original one.

FixSetFirstName :

Set to 1 to enable the fix.

By default, SetFirstName don’t work properly on Trigger, Encounter and Waypoint. This option fix that.

DisableTumbleAC:

Set it to 1 to Disable the bonus AC given by Tumble Skill score.
By default, the game grant +1 AC every 10 rank in Tumble.

DisableSpellCraftSave:

Set it to 1 to Disable the bonus to SpellSave given by SpellCraft skill score.
By default, the game grant +1 Save vs Spell every 5 ranks in SpellCraft.

DisableTrapSynergy:

Set it to 1 to Disable the bonus to SetTrap given by DisableTrap and vice versa
By default, the game grant +2 to SetTrap if the character have at least 5 in DisableTrap
And +2 to DisableTrap if the character have at least 5 in SetTrap

MonkWeaponList:

Here you can list the BaseItem will see as monk weapons.
For info, base game list is : “36 40 50 59 78”

AreaTypeBitX:

Those are specials Option. You will be able to link a bit of Area Type (that can be accessed with xp_aspectManager), with a name.

Base game options are :

AreaTypeBit0 = INTERIOR
AreaTypeBit1 = UNDERGROUND
AreaTypeBit2 = NATURAL

Those names will be used in the rule system, detailed bellow. You are free to change those as you want (just don't forget to update the different configs files). You can also add new one if you manage more.

SpeedFeatsFile:

Set it to the name of the ini file where you will list the Rules for SpeedFeat.
Rule system will be detailed bellow. Plugin came with an example file : SpeedFeatEFF.ini

SkillFeatsFile:

Set it to the name of the ini file where you will list the Rules for SkillFeat and Ability-Skill link.
Rule system will be detailed bellow. Plugin came with an example file : SkillFeatEFF.ini

WeaponFinesseFile:

Set it to the name of the ini file where you will list the Rules for WeaponFinesse.
Rule system will be detailed bellow. Plugin came with an example file : WeaponFinesseEFF.ini

ReduceSpeedFile:

Set it to the name of the ini file where you will list the Rules for ReduceSpeed.
Rule system will be detailed bellow. Plugin came with an example file : ReduceSpeedEFF.ini

OnGoldChangeScript:

Set a script name that will be called on a new event added by this plugin : OnCreatureGoldChange. It will be called each time the gold owned by a creature will change. It will also allow you to modify this change if you want.
More detail about this specific script bellow.

GlobalOnClientEnterScript:

Set a script name that will be called on the OnClientEnter event for every area. It will happens **after** the area-specific OnClientEnter.
This can be used if you want to execute a specific script each time a PC load any area of you module.

StartTransitionScript:

Set a script name that will be called on the new event added by this plugin : OnCreatureStartTransition.
It will be called when a creature will try a transition (by using a door or a transition trigger). It will also allow you to cancel this transition if you want.

More detail about this specific script bellow.

OnItemStackScript:

Set a script name that will be called on the new event added by this plugin: OnItemStack.
It will be called when a player try to stack two items, if all the previous check where valid (see on script explanation for more details). It will also allow you to stop forbid the stack (if you want to run more validations for example).
More detail about this specific script bellow.

OnSplitStackScript:

Set a script name that will be called on the new event added by this plugin: OnSplit.
It will be called when a player split a stack of item.
More detail about this specific script bellow.

OnItemBaseCostCalculationScript:

Set a script name that will be called on the new event added by this plugin: OnItemBaseCostCalculation.
This is called every time the value of an item can change (creation, add or remove properties, activations,..). It will let you adapt the base value of this item.
More detail about this specific script bellow.

FixHasInfiniteSpellClass:

Set to 1 to enable the fix.

Bug Explanation:
Without the fix, the game hardcode which class have infinite spell and which one not without worrying about the value in 2DA (Warlock will always have infinite spells for example). With it, the game will properly manage the 2DA value.

DefineInfiniteSpells:

Allow you to define specifics rules for infinite spells. The option FixHasInfiniteSpellClass must be enabled to use this.
Option must be in this format: idClass:SpellLvl;idClass:SpellLvl;….
Example :
1:0;9:0 will give infinite cantrip to bard and sorcerer.

AllowSetAreaTime:

Set this option to 1 to enable the possibility to use new Get/SetHourForArea_EFF.

Rules System Files

The rule system will allow you to easily add new feat or capability. Changing hardcoded part of the game. Currently, four part of the game work with the Rule system. More will be added with time.

Lot is common between all of those files, but some stuff are specific.

Common:

Each config file can have a **[General]** section that may include the option 'DisableHook = 1'. This is intended to be used during configuration/debugging, as it will be read during the calls to the various XPEnhancedFeatures_Reload****file() functions.
Each rule start with a **[SpecificNameRuleX]**. X being a number. The plugin will start by parsing the rule 1 and continue as long as it can find the next. So please, be sure to have contiguous number in your rules.

Some field will understand math operation, other will understand Boolean operation. Allowing you to really customize it to your need.

Allowed symbol for Boolean operation : | (or) , & (and) , ^ (XOR) , ! (not) and Parenthesis ().

Allowed symbol for Math operation : + (sum) , - (substraction), * (multiplication), / (division) , // (integer division) , % (modulus) and parenthesis ().

Math operation also allow four specific functions : ClassLevelSum, ClassLevelMax, Skill and Ability

ClassLevelSum : Will sum the pc level of each of listed class. You can put 1 or as many class as you want as parameters. Usage example : ClassLevelSum(5, 45) will sum the monks and sacred fist level of the creature.

ClassLevelMax : Same format as level sum but will only return the greatest level of listed class.

Skill : Will return the base value of the skill in parameter (without any bonuses). Usage example : Skill(5) will return the number of skill points put in Hide skill.

Ability: Same format as Skill, but will return the base ability score (with racial modificatory). (0=STR, 1=DEX, 2=CON, 3=INT, 4=WIS, 5=CHA)

Each rules can contain the following fields :

Feat : Boolean field. Here you can choose which feat will activate the rules. You can put the feat number. The field will understand Boolean operation so, if you want to activate the rule if the PC have feat 1 or feat 2 and no feat 3, you can write something like that : *Feat = (1 | 2) & !3*.

Area : Boolean field. Here you can choose in which kind of Area the rule will apply. You will need to use the AreaType name listed in the main ini file. For example, if you want that the rule will apply only on a exterior and natural area you can write something like that : *Area = !INTERIOR & !SUBTERRAN & NATURAL*

Bonus : Mathematical field. More detail on specific part.

Extra : Boolean field. Little more specific field. Here you will be able to make your rule depend of the activation (or non-activation) of previous rules. It will also provide you the possibility to use Special Check. See below for their list and explanation. To check if a previous rule is activated, just write RuleX , with x= the number of your specific rule.
/!\ this will always be RuleX, regardless of the SpecificName of the rule.
For example, if your rule must activate only if the rule 1 is ok, the 2 not ok and your pc have MONKPOWER available you can write something like that : Extra : Rule1 & !Rule2 & MONKPOWER.

Extra fiel also add numerical comparator : < , <= , >, >= , == , != that can be used with constant value or functions
Exemple : Skill(5) > 10
/!\ Numerical comparator can only be used with Function or constant value. And those can only be used with numerical comparator /!\ (still a Boolean field)

Special :

- MonkPower : Will be true if you play a monk that don't "loose" his power by wearing an armor for example.
- EncumbranceNone : Will be true if the character is not encumbered
- EncumbranceNormal : Will be true if the character have the first level of encumbrance.
- EncumbranceHeavy : Will be true if the character have the worst level of encumbrance.
- TrackMode : Will be true if the TrackMode is activated
- StealthMode : Will be true if the stealthMode is activated
- DetectMode : Will be true if the DetectMode is activated

Functions :

- The same as in the Mathematical fields (ClassLevelSum, ClassLevelMax, Skill and Ability)

SpeedFeatEFF:

Note : Using this file will remove all base game feats. The example file reimplement them in the RuleSystem so you can easily just add news one.

The speedFeat ini file can start with a **[General]** part with the field **CalculationType**.
Set the CalculationType value to Sum or Factor to choose how the different speed bonus will be calculated.

The base game use “factor mode”. That mean that, if you have two feats that give you a bonus of 10%, you will end up with a speed “factor” of 1.21. (1 * 1.1 * 1.1 = 1.21), like if you have a single bonus of 21%. If you use some, you will have a result of 1.20 (1+ 0.1 + 0.1).

Example :

```
[General]
CalculationType = Factor
```

SpecificName : Your rules name must be in the [SpeedRuleX] format.

Bonus : Will be used to determine the % of bonus you want to apply. In a factor form (0.2 for 20%). A negative value will reduce the speed. A positive one will increase it.

Specific Rules : There is two specific kind of rule in this file :

[MinSpeedRuleX] : Allow you to set the Minimal speed factor possible. Order is important as only the first valid one will be used.

The Bonus field is renamed Min here.

[MaxSpeedRuleX] : Allow you to set the Maximal speed factor possible. Order is important as only the first valid one will be used.

The Bonus field is renamed Max here.

For those two specifics rules type, if you choose to use Extra with RuleX test on it, those will refer to SpeedRuleX.

In this file, for each rules, the only mandatory ones are the Bonus one. (or Min/Max for MinSpeedRuleX/MaxSpeedRulesX types).

SkillFeatEFF:

The skillFeat ini file can start with a **[General]** part with the field **RemoveBaseSkillFeatRules**.

Set the RemoveBaseSkillFeatRules value to 1 in order to remove all base game skill feat bonus. Else the base game feat bonus will apply normally. (the skill synergy —for settrap/disabletrap— is not removed by this one, use the general option to remove it)

Remove the whole part or set the value to 0 to keep the game feats.

Example :

```
[General]
RemoveBaseSkillFeatRules = 1
```

SpecificName : Your rules name must be in the [SkillRuleX] format.

Bonus : Will be used to determine the value of bonus you want to apply. A positive value will increment the skill, a negative one will reduce it. Note that you can only use whole number and no number with decimal point for this one.

Skill : A specific field. Only one number allowed : the skill that will be concerned by the rule.

For each **[SkillRuleX]** mandatory fields are **Bonus** and **Skill**. Other are optional.

This file also allow you to set a second set of rules. The SkillAbility one. Those rules will allow you to dynamically change the ability used for a skill. For example.. Allow to use Strength for intimidate when you have the right feat.

SpecificName: Your rules name must be in the [SkillAbilityX] format.

Skill : A specific field. Only one number allowed : the skill that will be concerned by the rule.

Bonus : Not allowed for those rules.

Extra : Here, this field don't allow the usage of RuleX

Ability : The ability that must be used. Use only one of those possible values : STR, DEX, CON, INT, WIS, CHA.

For each **[SkillAbilityX]** rules, **Skill** and **Ability** fields are mandatory. **Feat** and **Area** fields are optionals

The order of the **[SkillAbilityX]** rules is important as only the first valid one will be used.

WeaponFinesseEFF:

The weaponFinesse ini file can start with a **[General]** part with the field **RemoveBaseWpnFinesseRules**.

Set the RemoveBaseWpnFinesseRules value to 1 in order to remove all base game weapon finesse rules. Else the base game weapon finesse rules will apply normally.

Remove the whole part or set the value to 0 to keep the game behavior.

Example :

```
[General]
RemoveBaseWpnFinesseRules = 1
```

SpecificName : Your rules name must be in the [WpnFinesseRuleX] format.

ItemType : A specific field. Set a list of number, separated by space. List the ItemType allowed to work with Weapon Finesse by this rule.

CheckSize : Set to 1 if this rule need to check the size of the weapon, compared to the size of creature. If set to 0 or absent, the rule will not check the size.

Bonus : Not allowed for those rules.

Extra : Here, this field don't allow the usage of RuleX

For each **[WpnFinesseRuleX]** rules, **ItemType** is mandatory. **Feat**, **Area**, **Extra** and **CheckSize** fields are optional.

If at least one rule is ok, the usage of WeaponFinesse will be allowed for the item. If you don't remove the Base Weapon Finesse Rules, they will be tested if none of your rules allow this item.

ReduceSpeedEFF:

SpecificName : Your rules name must be in the [ReduceSpeedRuleX] format.

Impact : Mathematical rule. The number of "ReduceSpeed" charge given by this rule. Base game use only 1 or 2. You can also put negative value.

For each **[ReduceSpeedRuleX]** rules, **Impact** is mandatory. **Feat**, **Area**, and **Extra** fields are optional.

For each rules that are ok, their **Impact** will be added. If the final value is equal or bellow to 0, no reduction will be applied. The plugin keep the base game behavior for impact. One impact will apply a 50% speed reduction, two 75%, etc.

HitPointFeatsEFF:

SpecificName : Your rules name must be in the [HitPointRuleX] format.

Bonus : Will define the number of HP added to the max if positive value, and removed if negative one.

For each **[HitPointRuleX]** rules, **Bonus** is mandatory. **Feat**, **Area**, and **Extra** fields are optional.

This file will allow you to create/adjust feat that have an impact of hitpoints like Toughness or Mind over Body.

Specific Scripts

OnGoldChange :

By enabling this option in the .ini file, the specified script will be triggered every time a creature's gold amount changes.

A base script is included in the plugin pack, named SEPT_ONGOLDCHANGE_EFF.NSS, which you can use as a template and customize as needed.

The script should have the following structure:

```
int StartingConditional(object oCreature, int iChangeOfGold)
```

- oCreature : The creature whose gold amount has changed.
- iChangeOfGold : The quantity of gold added (if positive value) or taken (if negative value).

Return value: The actual amounts of gold to add (if positive) or to take (if negative) to/from oCreature.

GlobalOnClientEnter :

By enabling this option in the .ini file, the specified script will be triggered like the OnClientEnter hook specified on an Area but for all Area. This script will be called just after the specific OnClientEnter.

A base script is included in the plugin pack, named SEPT_GLOBALONCLIENTENTER_EFF.NSS, which you can use as a template and customize as needed.

The script it's a basic "void main()" script with the exact same behavior as a normal OnClientEnter.

So, you can use the following nwnscript functions :

- **GetFirstEnteringPC()** : to get the first entering PC.
- **GetNextEnteringPC()** : returns the next entering player object, intended to be used along with GetFirstEnteringPC() in order to iterate over the list of entering players.
- **FiredFromPartyTransition()**
- **OBJECT_SELF :** The Destination Area. Note that you will also get it by calling GetArea(oPC) as they are already in the new area.

StartTransition :

By enabling this option in the ini file, the specified script will be triggered when a creature will try a transition by using a door or a transition trigger. Note that this script will not be called in other case (like use of script).

A base script is included in the plugin pack, named SEPT_GLOBALONCLIENTENTER_EFF.NSS, which you can use as a template and customize as needed.

The script should have the following structure:

```
int StartingConditional(object oCreature, object  oTransitionObject)
```

- oCreature : The creature who try to use the transition
- oTransitionObject : The object used for the transition (door or transition trigger).

Return value: TRUE to allow the transition to continue normally, FALSE to stop it.

OnItemStack :

By enabling this option in the ini file, the specified script will be triggered when two item are about to stack.

A base script is included in the plugin pack, named SEPT_ONITEMSTACK_EFF.NSS, which you can use as a template and customize as needed.

The script should have the following structure:

```
int StartingConditional(object oDestinationStack, object  oSourceStack)
```

- oDestinationStack : The item / stack that will receive the items from oSourceStack
- oSourceStack : The item/stack that will be stacked in oDestinationStack used for the transition (door or transition trigger).

Return value: TRUE to allow the stack, FALSE to forbid it.

Note:

This script will be called if all "base" check are ok (so, if the items can be stacked together).

To know if two items can be stacked together, the game will check if they are/have :

- The same BaseItemType
- The same isIdentified, isPlot, isStolen and isCursed status
- The same Tag
- The same LocalizedName
- The same ModelPart0/1/2 (if applicable)
- The same ArmorRuleType (if applicable)
- The same "On Activation" Properties
- The same number of "passive" Properties. Or, if you use FixItemStacking : The same passive Properties.

When two items stack, the number of items in oDestination (GetNumStackedItem) increases by the number of items in oSourceStack. Any differences between the two will be lost, as only oDestinationStack will remain.

This script allows you to add additional checks or perform operations before allowing the stack.

OnSplit :

By enabling this option in the ini file, the specified script will be triggered when a stack is split.

A base script is included in the plugin pack, named SEPT_ONSPLIT_EFF.NSS, which you can use as a template and customize as needed.

The script should have the following structure:

```
int StartingConditional(object oSourceStack, object  oNewStack)
```

- oSourceStack : The stack that has just been split.
- oNewStack : The newly created Stack.

Return value: The return value doesn't matter.

OnItemBaseCostCalculation :

By enabling this option in the ini file, the specified script will be triggered when the base cost of an item is calculated. It happens during creation, adding or removing item properties, activation, store opening/loading.

A base script is included in the plugin pack, named SEPT_ONITEMBASECOST_EFF.NSS, which you can use as a template and customize as needed.

The script should have the following structure:

```
int StartingConditional(object oItem, int  iBaseCost)
```

- `oItem` : The related Item
- iBaseCost : The baseCost that the game just calculated.

Return value: The baseCost you want to apply on oItem.

Note: The baseCost is not the sell price or Buy price. This “final” price depend of Merchant configuration, appraise Skill, BaseCost+ModifierCost (or NonIdentifiedCost).

Changelog

Version 1.1.15 released 2026/02/26

- Add FixImmunityOverflow: Option to fix the bug that happens if you stack more than 100% immunity/vulnerability for a specific damage type.

Version 1.1.14 released 2026/02/02

- Add FixHasInfiniteSpellClass: Option to force the game to use 2DA value instead of hardcoded stuff.
- Add DefineInfiniteSpells: Option to allow to define specifics rules for infinite spells
- Add Get/SetChatDistance_EFF nwnscript functions
- Add SendChatMessageToPC_EFF nwnscript function
- Add SetTimeForPC_EFF nwnscript function
- Add Get/SetHourForArea_EFF nwnscript functions

Version 1.1.12 released 2025/09/23

- Add FixItmPrpDmgBonus: Option to fix functions ItemPropertyDamageBonus* to prevent accidental creation of Burst instead of normal DamageBonus property.

Version 1.1.11 released 2025/08/28

- Add PatchAddNewTag: Option to preserve existing object indices when creating objects with duplicate tags

Version 1.1.6 to 1.1.10, 1.1.10 released 2025/08/03

- Add FixNumberOfFeatUsages : Option to properly retrieve the feat usages left when a PC login
- Add FixGetEffectType: Option to fix the GetEffectType function who can improperly return EFFECT_INVALID
- Add FixCalcSafeLocation: Option to allow CalcSafeLocation to really test the area of the location parameter
- MaxCalcSafeLocation: Option to change the max radius (5.0 by default) of CalcSafeLocation
- EnhanceGetResRef : Option to allow GetResRef to work for Area object
- FixSetFirstName: Option to fix SetFirstName who don't work for Trigger, Encounter and Waypoint object
- Add the management of HitPoint in ruleEngine

Version 1.1.5, released 2024/11/26

- Add DisableTrapSynergy : Option to disable the SetTrap/DisableTrap synergy
- Add two new function in the RuleEngine : Skill and Ability functions can now be used in the rules
- Improve RuleEngine to allow numerical comparator in Extra field : « <, <=, >, >=, !=, == » can now be used in extra fields with constant value and call to the functions. Allowing you to create/manage skills synergy.

Version 1.1.4, released 2024/10/31

- Add KeepLocalVarOnSplit : Option to enable a fix that properly copies the LocalVar on the resulting stack after a split.
- Add OnSplitStackScript : Hook to call a script that allows additional operations after stack split.

Version 1.1.3, released 2024/10/30

- Fix an issue that can cause a server crash in certain circumstance after a destroying an Instanced Area

Version 1.1.2, released 2024/10/28

- Add FixItemStacking : option to enable a fix that properly checks item properties before allowing stacking.
- Add OnItemStackScript : Hook to call a script that allow additional checks or operation before allowing two items to stack.
- Add OnItemBaseCostCalculation: Hook to call a script that gives a more direct control on items price.

Version 1.1.1, released 2024/10/26

- Change a little GlobalOnClientEnter to exactly match the OnClientEnter.

Version 1.1.0, released 2024/10/25

- Added ReduceSpeed management with RuleEngine
- Added the ini option “ReduceSpeedFile” that allows pointing to the new RuleEngine configuration file.
- Added new nwnscript function : XPEnhancedFeatures_ReloadReduceSpeedFile
- Added [General]/DisableHook option possibility for every RuleEngine config files.
- Added 5 new “Special” keyword for the **Extra** fields : TrackMode, DetectMode, StealthMode, EncumbranceNone, EncumbranceNormal and EncumbranceHeavy.