From the "Neverwinter Nights Enhanced Edition (v74).txt" notes that didn't make into the release notes. (patch 8149)

New Script Commands

object CopyArea(object oArea);
Creates a copy of a existing area, including everything inside of it (except players).
Returns the new area, or OBJECT_INVALID on error.
Note: You will have to manually adjust all transitions (doors, triggers) with the relevant script commands, or players might end up in the wrong area.

* object GetFirstArea();
Returns the first area in the module.

* object GetNextArea();
Returns the next area in the module (after GetFirstArea), or OBJECT_INVALID if no more
areas are loaded.

* void SetTransitionTarget(object oTransition, object oTarget);
Sets the transition target for oTransition.
- oTransition can be any valid game object, except areas.
- oTarget can be any valid game object with a location, or OBJECT_INVALID (to unlink).
- Rebinding a transition will NOT change the other end of the transition; for example,
with normal doors you will have to do either end separately.
- Any valid game object can hold a transition target, but only some are used by the game engine
(doors and triggers). This might change in the future. You can still set and query them for
other game objects from nwscript.
- Transition target objects are cached: The toolset-configured destination tag is used for a lookup only once, at first use. Thus, attempting to use SetTag() to

change the
destination for a transition will not work in a predictable fashion.

* void SetHiddenWhenEquipped(object oItem, int nValue);
Sets whether the provided item should be hidden when equipped.
- The intended usage of this function is to provide an easy way to hide helmets, but it
can be used equally for any slot which has creature mesh visibility when equipped,
e.g.: armour, helm, cloak, left hand, and right hand.
- nValue should be TRUE or FALSE.

* int GetHiddenWhenEquipped(object oItem);
Returns whether the provided item is hidden when equipped.

* object CopyItemAndModify(object oItem, int nType, int nIndex, int nNewValue, int bCopyVars=FALSE);
Creates a new copy of an item, while making a single change to the appearance of the item.
Helmet models and simple items ignore iIndex.
iType iIndex iNewValue
ITEM_APPR_TYPE_SIMPLE_MODEL [Ignored] Model #
ITEM_APPR_TYPE_WEAPON_COLOR ITEM_APPR_WEAPON_COLOR_* 1-4
ITEM_APPR_TYPE_WEAPON_MODEL ITEM_APPR_WEAPON_MODEL_* Model #
ITEM_APPR_TYPE_ARMOR_MODEL ITEM_APPR_ARMOR_MODEL_* Model #
ITEM_APPR_TYPE_ARMOR_COLOR ITEM_APPR_ARMOR_COLOR_* [0] 0-175 [1]
[0] Alternatively, where ITEM_APPR_TYPE_ARMOR_COLOR is specified, if per-part coloring is
desired, the following equation can be used for nIndex to achieve that:
ITEM_APPR_ARMOR_NUM_COLORS + (ITEM_APPR_ARMOR_MODEL_ * ITEM_APPR_ARMOR_NUM_COLORS) + ITEM_APPR_ARMOR_COLOR_
For example, to change the CLOTH1 channel of the torso, nIndex would be:
6 + (7 * 6) + 2 = 50
[1] When specifying per-part coloring, the value 255 is allowed and corresponds

with the logical
function 'clear colour override', which clears the per-part override for that part.

Modified Existing Scripting Instructions (For Area Instancing)

-=-
=-=-=

* object GetTransitionTarget(object oTransition);
Get the destination object for the given object.
All objects can hold a transition target, but only Doors and Triggers
will be made clickable by the game engine (This may change in the
future). You can set and query transition targets on other objects for
your own scripted purposes.
Returns OBJECT_INVALID if oTransition does not hold a target.

* void SetName(object oObject, string sNewName="");
Set the name of oObject.
- oObject: the object for which you are changing the name (area, creature, placeable, item, or door).
- sNewName: the new name that the object will use.
SetName() does not work on player objects.
Setting an object's name to "" will make the object
revert to using the name it had originally before any
SetName() calls were made on the object.

RELEASE NOTES

New Scripting Commands

• SetTag(object, string), void SetTag(object,string). Specifies a new tag for the given object. Note that this call will not update any of the references to this tag (such as area transitions).
• string GetEffectTag(effect eEffect). Returns the string tag set for the provided effect. If no tag has been set, returns an empty string.
• effect TagEffect(effect eEffect, string sNewTag). Tags the effect with the provided string. Any other tags in the link will be overwritten.
• int GetEffectCasterLevel(effect eEffect). Returns the caster level of the creature

who created the effect. If not created by a creature, returns 0. If created by a spell-like ability, returns 0.
• int GetEffectDuration(effect eEffect). Returns the total duration of the effect in seconds. Returns 0 if the duration type of the effect is not DURATION_ TYPE_TEMPORARY.
• int GetEffectDurationRemaining(effect eEffect). Returns the remaining duration of the effect in seconds. Returns 0 if the duration type of the effect is not DURATION_TYPE_TEMPORARY.
• string GetItemPropertyTag(itemproperty nProperty). Returns the string tag set for the provided item property. If no tag has been set, returns an empty string.
• itemproperty TagItemProperty(itemproperty nProperty, string sNewTag). Tags the item property with the provided string. Any tags currently set on the item property will be overwritten.
• int GetItemPropertyDuration(itemproperty nProperty). Returns the total duration of the item property in seconds. Returns 0 if the duration type of the item property is not DURATION_TYPE_ TEMPORARY.
• int GetItemPropertyDurationRemaining(ite mproperty nProperty). Returns the remaining duration of the item property in seconds. Returns 0 if the duration type of the item property is not DURATION_TYPE_TEMPORARY

NWN:EE Head Start Patch Notes v74.8154

New Script Commands

These new scripting commands allow more options for player minimap management. They go hand in hand with area instancing, as they will allow save/load of exploration states on a per-player-and-area basis (with some scripting).

```
// Sets if the given creature has explored tile at x, y of the given area.
// Note that creature needs to be a player- or player-possessed creature.
//
// Return values:
// -1: Area or creature invalid.
// 0: Tile was not explored before setting newState.
// 1: Tile was explored before setting newState.
```

```
int SetTileExplored(object creature, object area, int x, int y, int newState);

// Returns whether the given tile at x, y, for the given creature in the stated area is visible on
// the map.
// Note that creature needs to be a player- or player-possessed creature.
//
// Return values:
// -1: Area or creature invalid.
// 0: Tile is not explored yet.
// 1: Tile is explored.
int GetTileExplored(object creature, object area, int x, int y);

// Sets the creature to auto-explore the map as it walks around.
// Valid arguments: TRUE and FALSE.
// Does nothing for non-creatures.
// Returns the previous state (or -1 if non-creature).
int SetCreatureExploresMinimap(object creature, int newState);

// Returns TRUE if the creature is set to auto-explore the map as it walks around (on by default).
// Returns FALSE if creature is not actually a creature.
int GetCreatureExploresMinimap(object creature);
```

NWN:EE Head Start Patch Notes v74.8156

New Script Commands

```
int GetSurfaceMaterial(location at);
// Get the surface material at the given location. (This is equivalent to the walkmesh type).
// Returns 0 if the location is invalid or has no surface type.
float GetGroundHeight(location at);
// Returns the z-offset at which the walkmesh is at the given location.
// Returns -6.0 for invalid locations.
```

NWN:EE 8157 & 8158 Patch Notes

NWScript

BootPC(object oPC, string sReason = “”) now takes an optional string that is displayed to the client being kicked off.

New Script Commands:

```
// Gets the attack bonus limit.
// - The default value is 20.
int GetAttackBonusLimit();

// Gets the damage bonus limit.
// - The default value is 100.
int GetDamageBonusLimit();

// Gets the saving throw bonus limit.
// - The default value is 20.
int GetSavingThrowBonusLimit();

// Gets the ability bonus limit.
// - The default value is 12.
int GetAbilityBonusLimit();

// Gets the ability penalty limit.
// - The default value is 30.
int GetAbilityPenaltyLimit();

// Gets the skill bonus limit.
// - The default value is 50.
int GetSkillBonusLimit();

// Sets the attack bonus limit.
// - The minimum value is 0.
void SetAttackBonusLimit(int nNewLimit);

// Sets the damage bonus limit.
```

```
// - The minimum value is 0.
void SetDamageBonusLimit(int nNewLimit);

// Sets the saving throw bonus limit.
// - The minimum value is 0.
void SetSavingThrowBonusLimit(int nNewLimit);

// Sets the ability bonus limit.
// - The minimum value is 0.
void SetAbilityBonusLimit(int nNewLimit);

// Sets the ability penalty limit.
// - The minimum value is 0.
void SetAbilityPenaltyLimit(int nNewLimit);

// Sets the skill bonus limit.
// - The minimum value is 0.
void SetSkillBonusLimit(int nNewLimit);
```

Neverwinter Nights: Enhanced Edition Patch 8162

NWScript
New script commands:

```
// Get if oPlayer is currently connected over a relay (instead of directly).
// Returns FALSE for any other object, including OBJECT_INVALID.
int GetIsPlayerConnectionRelayed(object oPlayer);
```

Neverwinter Nights: Enhanced Edition Patch 8164

NWScript

New functions: GetEventScript, SetEventScript to set NWScript event handlers on any supported object (including PCs).