Saving Throws

This discusses how the engine handles saving throws primarily for builders/engine workings interest but useful for players.

Saving Throw Basics

The engine will do saving throws both internally and from scripts - primarily spells and abilities.

These are done by this calculation:

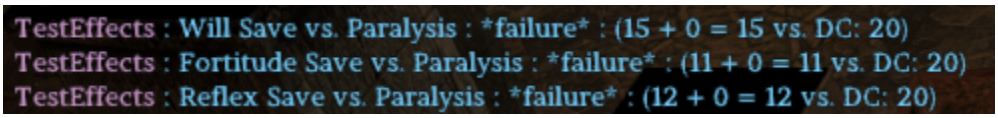
```
d20 + Bonuses/penalties vs. Difficulty Class
```

The bonuses are the base will/fortitude/reflex statistics on the character sheet, plus any specific ones against the saving throw type. For reference these are the games saving throw types, most are available as item properties (and can be added back in see iprp_saveelement.2da).

```
int SAVING_THROW_TYPE_ALL               = 0;
int SAVING_THROW_TYPE_NONE              = 0;
int SAVING_THROW_TYPE_MIND_SPELLS       = 1;
int SAVING_THROW_TYPE_POISON            = 2;
int SAVING_THROW_TYPE_DISEASE           = 3;
int SAVING_THROW_TYPE_FEAR              = 4;
int SAVING_THROW_TYPE_SONIC             = 5;
int SAVING_THROW_TYPE_ACID              = 6;
int SAVING_THROW_TYPE_FIRE              = 7;
int SAVING_THROW_TYPE_ELECTRICITY       = 8;
int SAVING_THROW_TYPE_POSITIVE          = 9;
int SAVING_THROW_TYPE_NEGATIVE          = 10;
int SAVING_THROW_TYPE_DEATH             = 11;
int SAVING_THROW_TYPE_COLD              = 12;
int SAVING_THROW_TYPE_DIVINE            = 13;
int SAVING_THROW_TYPE_TRAP              = 14;
int SAVING_THROW_TYPE_SPELL             = 15;
int SAVING_THROW_TYPE_GOOD              = 16;
int SAVING_THROW_TYPE_EVIL              = 17;
int SAVING_THROW_TYPE_LAW               = 18;
int SAVING_THROW_TYPE_CHAOS             = 19;
```

"Mind Spells" is a bit of misnomer since all kinds of monster abilities and On Hit effects cause "mind spells" saves.

There appears to be a missing one from this list, Paralysis, used only seemingly by Death Attack, it's ID is 20 and can be added to the saving throw functions fine (item properties need to test):



The saving throw type usually is listed on a spells description.

You can with the advent of NWN:EE do most of the calculations for the saving throw in nwscript getting all the item properties and effect properties and gathering what the total modifiers would be. This may be useful as we'll see when it comes to Spellcraft, or any instance where more than one "Type" might be required.

The feedback given to the player (as the person asking for a save - eg the spellcaster, or the target) usually says the type. However there is an exception for vs. Spells when it's an assumed spell - see **Spellcraft** below.

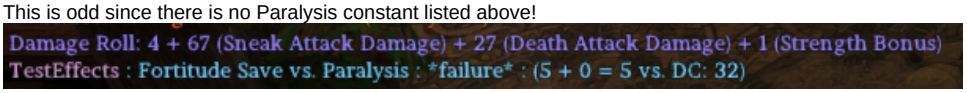
The saving throw string reference is 10473, which is as below:

```
<CUSTOM0><CUSTOM1> : <CUSTOM2><CUSTOM3> : *<CUSTOM4>* : (<CUSTOM5> <CUSTOM6> <CUSTOM7> = <CUSTOM8> vs. DC:
<CUSTOM9>)
```

Hardcoded Saving Throws

As noted above the engine will do some of it's own saving throws, this is a possibly incomplete list as a reference.

Note that several have DCs partially or fully listed in ruleset.2da which means they can be altered a bit.

- EffectSanctuary will roll a Will save with no saving type to the given DC (note this is not vs. Spells)
- Diseases (such as applied by EffectDisease or on hit item properties) roll a Fortitude save vs. Disease
- Poison (such as applied by EffectPoison or on hit item properties) roll a Fortitude save vs. Poison
- Stunning Fist rolls a Fortitude save with no type
- Defensive Roll rolls a Reflex save with no type
- Death Attack rolls a Fortitude save vs. Paralysis
    - This is odd since there is no Paralysis constant listed above!
    - 

- Devastating Critical rolls a Fortitude save with no type
- Quivering Palm rolls a Fortitude save with no type
- Deflect Arrows rolls a Reflex save with no type

On Hit item property effects that have saving throw types seem to be as per below, which *mostly* match spells and is a good basis for how to keep these consistent:

- On Hit: Ability Damage - Fortitude save vs. Negative
- On Hit: Blindness - Fortitude save with no type
- On Hit: Confusion - Will save vs. Mind Spells
- On Hit: Daze - Will save vs. Mind Spells
- On Hit: Deafness - Fortitude save with no type
- On Hit: Disease - as per above, applies the Disease effect that handles it
- On Hit: Doom - Will save with no type
- On Hit: Fear - Will save vs. Fear
- On Hit: Hold (paralysis) - Will save with no type
- On Hit: Level Drain - Fortitude save vs. Negative
- On Hit: Poison - as per above, applies the Poison effect that handles it
- On Hit: Silence - Will save with no type
- On Hit: Slay Alignment/Alignment Group/Racial Type: Fortitude save vs. Death
- On Hit: Sleep - Will save vs. Mind Spells
- On Hit: Slow - Will save with no type
- On Hit: Stun - Will save vs. Mind Spells
- On Hit: Wounding - Fortitude save with no type
- On Hit: Vorpal - Reflex save with no type
    - Not sure why this isn't the same as Slay and vs. Death, except that's D&D or Bioware for you

Scripted Saving Throws

The functions that cause the internal saving throw function to fire are:

- WillSave
- FortitudeSave
- ReflexSave
- GetReflexAdjustedDamage

All of them accept a single Saving Throw Type constant as per above, and a DC/source of the save. If the source is a creature various bonuses to the save may be applied (vs. Alignment for instance).

Apart from this there isn't much more to these saves, except saves executed in spells, as per below!

Saving Throw Bonuses vs Spells

There are four main ways to gain a saving throw bonus vs. spells - the spell **Protection from Spells** which adds +8 against spells only (items may also provide similar bonuses), Dwarf's feat **Hardiness vs. Spells**, the **Arcane Defence** line of feats, and **Spellcraft**.

Arcane Defence

Bug note: This is bugged and only applies when the saving throw vs. spells is implicit.

Spellcraft

Spellcraft has a built in bonus to saving throws; SPELLCRAFT_NUM_RANKS_PER_SAVE_BONUS is usually 5 which means that you get +1 vs. Spells for each 5 ranks, making Sorcerers and Wizards (and usually Clerics, Druids) inherently quite defensively minded when it comes to enemy spells.

If you have 0 ranks in Spellcraft from levelling (ie untrained) you cannot gain a bonus or penalty from this - even if you apply a +50 bonus SKILL_SPELLCRAFT as an effect. Once you have 1 rank of spellcraft you can apply bonuses to it and they'll apply, so any temporary bonuses from items or effects also apply to the generated complete value (as long as you have 1 rank to begin with).

Spellcraft has a notable bug: *any* spell script will trigger the bonus from spellcraft OR when the saving throw type is SAVING_THROW_TYPE_VS_SPELLS. This means it applies to items like, say, Wine/Beer/Sprits, or Garlic as well as all monster and feat abilities such as Harper: Sleep and so on, ie anything with impact scripts.

SAVING_THROW_TYPE_SPELL

Things that explicitly specify **SAVING_THROW_TYPE_SPELL** - this is only used really to be explicit for Ruin and Hellball - which since they are UserType of 2 already apply the bonus! Odd eh?

Some other spells do specify it (such as Sunburst) but these don't need to because of the second, normal, method.

This could be used when an otherwise non-spell script wants to act like a spell and there is no better type to use.

Normal Spell Cast

Spells can come in 4 types, see Spells and Abilities which lays out the different types. The UserType being 1 (Spell) or 2 (Spell-like ability, ie SPELLABILITY_*) and it being cast means it'll be picked up as a "Proper spell" even if it's a creature ability, used by a feat or cheat-cast (in fact the source is irrelevant it seems to just check the 2da value).

This allows the bonus to be applied *in addition* to the usual saving throw bonus type. For instance Fireball has Reflex Save vs. Fire, but on top of it will add the Saving Throw vs. Spells as well.

EG: A Wizard casting Daze causes it to occur, but Harper casting the feat Sleep would not. Spellabilities also have this bonus applied, such as Gaze: Confusion.

NOTE: as mentioned above Spellcraft applies to *all* sources of spells.2da script calls due to bugs. This is only relevant for the feat and effect bonuses.

However this will not apply outside of the base original spell script, meaning:

- If saves are done in a DelayCommand() will make it lose the bonus
- ExecuteScript *does* keep the context of the spell script - even if executed on another object - which may work well. If it isn't needed run it in a DelayCommand instead
- Area of Effects do not appear to apply the bonus (eg; Acid Fog was tested) it's unknown if this is a bug or intentional, but is probably a bug since these should be spells (if coming from the right spell script)!
- EffectRunScript effects are not counted as coming from spells except for the OnApplied part of it if it is applied immediately without delay (in the same context of the currently running spell script). The interval and removed ones do not act like spells.

Saving Throw vs. Traps

This bonus is usually from Uncanny Dodge that Rogues get. It's quite common so worth noting.

This is similar to spells - there is the specific **SAVING_THROW_TYPE_TRAP** of course but it isn't used in all the trap scripts, eg X2_T1_ColdEpicC uses **SAVING_THROW_TYPE_COLD** instead.

The game seems to know when it is a "trap script" solely by testing if the calling object is a object that is trapped; setting the trap to deactivated during the trap script doesn't affect this it seems. This means that both Cold and Traps saving throw bonuses are applied to the example above. This also means *any* save caused by an object that is trapped - even if a trap specifically isn't triggered - causes the bonus to be applied. See below for oddities of this.

However any DelayCommand will again lose this if the trap is one shot, thus the object at that point in time is no longer trapped. DelayCommands in general in trap scripts seem to be used only for applying effects (even then usually at 0.0 delay for damage to work properly) so probably not worth doing that ever - floor trigger traps after all may disappear entirely once triggered so delays may not fire correctly.

Spell-Traps with Correct Bonuses

Cheat-cast ActionCastSpellAtObject from a **currently trapped** object, which remains trapped, will grant the bonus - on top of the "is a spell" bonus so everything is correct! Since actions are delayed this won't work on the trapped object itself which won't be trapped in 0.1 seconds when the spell script might activate.

This means you can get this bonus to be applied properly for "spell-like-traps" if you had a (hidden) trapped object of any kind cast the spell via. ExecuteScript (since AssignCommand can be a bit delayed/buggy with placeables) for instance:

```
void main()
{
    // Tagged object "trap_caster" is a placeable which has a trap on it but otherwise is inaccessible/unusable
    object oCaster = GetObjectByTag("trap_caster");
    object oTarget = GetEnteringObject();
    SetScriptParam("SPELL_ID", IntToString(SPELL_FIREBALL));
    SetScriptParam("SPELL_TARGET", ObjectToString(oTarget));
    ExecuteScript("cast_trap_spell", oCaster);
}

// "cast_trap_spell" script, this object should be trapped with a fake trap and be unusable
void main()
{
    int nSpell = StringToInt(GetScripParam("SPELL_ID"));
    object oTarget = StringToObject(GetScriptParam("SPELL_TARGET"));
    ActionCastSpellAtObject(nSpell, oTarget, METAMAGIC_ANY, TRUE);
}
```