Particle effect (pfx)

Properties

Appearance

- **FlowType**
  - `Continuous`: A number of particles are generated every second
  - `Explosion`: All particles are generated only once
- **Texture**: Texture file name, with its extension
- **TextureType**:
  - `2D texture`: Static texture
  - `3D texture`: Animated texture (some DDS textures are animated)
  - `Cubic texture`: ?? Not used in NWN2 game files
  - `Null texture`: ?? Not used in NWN2 game files
- **TextureAnimationSpeed**: Animation speed of the texture. Can be negative.
- **FrameBufferEffect**: True to create a distortion effect, False to render textures as you would expect
- **FrameBufferPixelDisplacement**: Amount of distortion if *FrameBufferEffect* is True

Behavior

- **SpawnRate**: Number of particles spawned every second (or at once if *FlowType* is `Explosion`)
- **BeginSize**: Particle size when created
- **EndSize**: Particle size when its life ends
- **SizeVariance**: Percent of *BeginSize* added or subtracted to *BeginSize*
- **OrientationType**:
  - `Camera oriented`: The particle texture will always be perpendicular to the camera.
  - `User-defined explicit orientation`: The particle texture will be facing the same direction than the object owning the effect (creature, placeable, ...), while remaining remain vertical.
  - `Vertically oriented (around Z)`: The particle texture will remain vertical while facing toward the camera.
- **ParticleLifetime**: Duration of the particle until it is destroyed (seconds)
- **LifetimeVariance**: Percent of *ParticleLifetime* variation. `0.5` will make particles last between `0.5 * ParticleLifetime` and `1.5 * ParticleLifetime`, `2` between `0` and `3 * ParticleLifetime`

- **ControlType**: Defines how to control the particle positions.
  - `Cartesian`: Standard control: `X ; Y ; Z`
  - `Polar`: Polar controls (coordinates on a vertical cylinder using one angle and two distances): `  ; r ; Z`. is the angle, `r` the distance from the center and `Z` the height.
    - Limitation: The `r` value can only be set using *EmissionRadius* and cannot be changed with *Acceleration* or *Velocity*.

    `Spherical`: Spherical controls (coordinates on a sphere using two angles and one distance): `r ;  ; `. `r` is the distance from the center, is the angle that governs the height, and is the angle around the `Z` axis.
    - Limitation: The `r` value can only be set using *EmissionRadius* and cannot be changed with *Acceleration* or *Velocity*.
    - Limitation: The value can only be set using *StartTheta*

- **EmissionRadius**: Distance from the center where the particles will be created.
  - If *ControlType* is `Cartesian`: The starting point will be randomly set to a point on a sphere. Note: the particles will bug/blink/disappear if *Velocity* is `0;0;0`, you need to set at least a tiny *Velocity* value.
  - If *ControlType* is `Polar` or `Spherical`: This is the `r` value
- **EmissionRadiusVariance**: Random radius value added or subtracted to *EmissionRadius*
- **Disk**: If true, the *EmissionRadius* will define a circle perpendicular to the direction defined by *Velocity*, instead of a sphere.
- **GravitateTowardCenter**:
  - If *ControlType* is `Cartesian` and *EmissionRadius* > 0 and *Disk* is False: The particles will always go toward the center of the sphere (or if the *EmissionRadius* is small, to the opposite of the center) no matter what direction *Velocity* is pointing to.
  - Doesn't seem to work in other cases
- **StartPhi**: If the *ControlType* is `Spherical`, this sets the value when the particle is created. Set to -1 to generate a random value.
- **StartTheta**: If the *ControlType* is `Polar` or `Spherical`, this sets the value when the particle is created. Set to -1 to generate a random value.
- **Velocity**: Defines the initial direction and speed of the particles when created. Each value correspond to an axis according to *ControlType*.
- **VelocityVariance**: Value randomly added or subtracted to the particle speed. This will not change the direction defined by the *Velocity* value, but will only change the particle movement speed. You can use this value to make particles goes toward two opposite directions using `Velocity = 0.001` and `VelocityVariance = 2`.
- **ConeHalfAngle**:
  - If *ControlType* is `Cartesian`: Random cone angle added to the direction defined by *Velocity*
- **Acceleration**: Particle acceleration. Each value correspond to an axis according to *ControlType*.
- **AngularVelocity**: Rotation speed (degree per second) of the particle themselves around the center of their texture
- **StartAngleVariance**: Random value added or subtracted to the *AngularVelocity* value

- **SystemRelative**: ??

Blend

- **DestinationBlend**: `See Blending modes`
- **SourceBlend**: `See Blending modes`

Color

- **StartColor**: Particle color when created, in `A;R;G;B` or `R;G;B` format.
- **EndColor**: Particle color when its life ends, in `A;R;G;B` or `R;G;B` format.

Misc

- **Position**: ?? - does nothing?
- **Scale**: ?? - does nothing?
- **Selectable**: ??
- **Tag**: Unusable
- **Visible**: Set to False to make the particle effect invisible in the PFX editor, but not in SEF (what's the point?)