Particle System Tutorials

Creating the particle system.

Particle system may be created using the this example code:

```
const char* objectTypeName = "obj_ParticleSystem" ;
const char* objectName     = "fire_cameradrone" ;

GameObject* particleSysObj = srv_CreateGameObject(    objectTypeName, objectName,
                                                      r3dPoint3D(0,0,0) ) ;

if( particleSysObj != NULL )
{
       obj_ParticleSystem* particleSys = static_cast<obj_ParticleSystem*>(particleSysObj);
}
```

In this code, *objectTypeName* is common type name for all particle systems. *objectName* corresponds to .prt particle file with the same name. This file contains particle system library data, which is common among all particle system instances of `"fire_cameradrone"`. It is internaly cached Third parameter is the initial position of the object.

After calling this function and successfully retrieving a non-zero pointer to *GameObject* instance, it is safe to convert it to *obj_ParticleSystem*, as we passed it as *objectTypeName*.

Restarting the particle system.

Particle system may be restarted without having to recreate the particle system object. This can be done like in the following code snippet.

```
const char* objectTypeName = "obj_ParticleSystem" ;
const char* objectName     = "muzzle_asr" ;

obj_ParticleSystem* muzzleParticles =
             static_cast<obj_ParticleSystem*> (
                    srv_CreateGameObject(      objectTypeName, objectName,
                                               r3dPoint3D(0,0,0) ) ) ;

…
…
…

muzzleParticles->Restart(r3dGetTime()) ;
```

As we can see, Restart call is made. Game time, returned by *r3dGetTime()*, is supplied as the only argument.

Moving the particle system.

Particle system can be moved using *GameObject* methods. Consider the following code snippet.

```
obj_ParticleSystem* particleSys = ...;

particleSys->SetPosition( particleSys->GetPosition() + r3dPoint3D(0,1,0) ) ;
```

This code will offset particle system *particleSys* up 1 meter.