ID Handles

General

Storing

A ID handle is used to store and retrieve a object. Instead of storing the object handle directly the ID of the object should be stored.

```
class cMyClass {
  iPhysicsBody@ mpBody; // unsafe
  tID mBodyID; // safe
}
```

Storing a ID instead of class handle is safer and allows for saving of the handle.

```
void OnStart()
{
 // Retrive the object from name and setup the id
 iPhysicsBody@ pBody = Map_GetEntityFromName("...");
 mBodyID = pBody.GetID();
}
```

The ID of a object is returned from the GetID() function or by calling one of the functions that return the ID of a object directly.

Accessing

After storing the ID it can be used to retrieve the object. Retrieving the object is safe and will return a null pointer if it has been deleted.

```
void OnUpdate()
{
  cLux_ID_Body(mBodyID).SetMass(2.0); // Retrieve object and set mass to 2.0
}
```

Even if the body has been deleted elsewhere the code will not crash, since accessing a null pointer only generates a warning.

Retrieving the object using the ID handle is very fast and can be used directly. When changing multiple properties of a object it is adviced to retrieve it once and then call all the function from the object itself.

```
void OnUpdate()
{
  // Retrieve the object
  iPhysicsBody@ pBody = cLux_ID_Body(mBodyID);
  if(pBody !is null) { // Check if it is valid
    pBody.SetMass(2.0);
    pBody.ApplyForce(cVector3f(1.0, 0.0 ,0.0);
  }
}
```

The retrieval functions perform safe cast. The ID for a Prop can be used to retrieve a Entity, but if you use it to try to retrieve a Area then it will return null.

Default Value

The default value of the tID class is tID_Invalid. Calling any retrieval function with that value will cause the function to return null. When a handle is no longer in use it should have its value set to tID_Invalid.

Saving and loading

A ID handle can be saved to the save file and it will retrieve the same object after load. The ID will also stay the same when working on the map which means that the save file will stay valid even after a patch on a map. This will also work with the realtime reload of the entity maptrack.