← Back (/tutorials?cat=write_plugin)

Programmatic World Control

This plugin example programmatically modifies gravity.

Prerequisites:

- Model Manipulation (http://gazebosim.org/tutorials/?tut=plugins_model)
- Plugin Tutorial (http://gazebosim.org/tutorials/?tut=plugins_hello_world)

Setup:

Source: gazebo/examples/plugins/world_edit (https://bitbucket.org/osrf/gazebo/src/gazebo7/examples/plugins/world_edit)

Use the `gazebo_plugin_tutorial` from the previous plugin tutorials

```
$ mkdir ~/gazebo_plugin_tutorial; cd ~/gazebo_plugin_tutorial
```

Create a file called `~/gazebo_plugin_tutorial/ world_edit .world`

```
$ gedit world_edit.world
```

Add the following contents to it:

```
<?xml version ='1.0'?>
<sdf version ='1.4'>
  <world name='default'>
    <include>
      <uri>model://ground_plane</uri>
    </include>

    <include>
      <uri>model://sun</uri>
    </include>

    <plugin filename="libworld_edit.so" name="world_edit"/>
  </world>
</sdf>
```

Code

Create a file called `~/gazebo_plugin_tutorial/ world_edit .cc` :

```
$ gedit world_edit.cc
```

Add the following content to it:

```
#include <sdf/sdf.hh>
#include <ignition/math/Pose3.hh>
#include "gazebo/gazebo.hh"
#include "gazebo/common/Plugin.hh"
#include "gazebo/msgs/msgs.hh"
#include "gazebo/physics/physics.hh"
#include "gazebo/transport/transport.hh"

/// \example examples/plugins/world_edit.cc
/// This example creates a WorldPlugin, initializes the Transport system by
/// creating a new Node, and publishes messages to alter gravity.
namespace gazebo
{
  class WorldEdit : public WorldPlugin
  {
    public: void Load(physics::WorldPtr _parent, sdf::ElementPtr _sdf)
    {
      // Create a new transport node
      transport::NodePtr node(new transport::Node());

      // Initialize the node with the world name
      node->Init(_parent->GetName());

      // Create a publisher on the ~/physics topic
      transport::PublisherPtr physicsPub =
        node->Advertise<msgs::Physics>("~/physics");

      msgs::Physics physicsMsg;
      physicsMsg.set_type(msgs::Physics::ODE);

      // Set the step time
      physicsMsg.set_max_step_size(0.01);

      // Change gravity
      msgs::Set(physicsMsg.mutable_gravity(),
          ignition::math::Vector3d(0.01, 0, 0.1));
      physicsPub->Publish(physicsMsg);
    }
  };

  // Register this plugin with the simulator
  GZ_REGISTER_WORLD_PLUGIN(WorldEdit)
}
```

The Code Explained

```
      // Create a new transport node
      transport::NodePtr node(new transport::Node());

      // Initialize the node with the world name
      node->Init(_parent->GetName());
```

We create a new node pointer, and initialize it to using the world name. The world name allows the node to communicate with one specific world.

```
      // Create a publisher on the ~/physics topic
      transport::PublisherPtr physicsPub =
        node->Advertise<msgs::Physics>("~/physics");
```

A publisher is created for sending physics messages on the "~/physics" topic.

```
      msgs::Physics physicsMsg;
      physicsMsg.set_type(msgs::Physics::ODE);

      // Set the step time
      physicsMsg.set_max_step_size(0.01);

      // Change gravity
      msgs::Set(physicsMsg.mutable_gravity(),
          ignition::math::Vector3d(0.01, 0, 0.1));
      physicsPub->Publish(physicsMsg);
```

A physics message is created, and the step time and gravity are altered. This message is then published to the "~/physics" topic.

Build

Assuming the reader has gone through the Plugin Overview Tutorial (http://gazebosim.org/tutorials/?tut=plugins_hello_world), all that needs to be done in addition is save the above code as `~/gazebo_plugin_tutorial/ world_edit .cc` and add the following lines to `~/gazebo_plugin_tutorial/` **`CMakeLists`** `.txt`

```
add_library(world_edit SHARED world_edit.cc)
target_link_libraries(world_edit ${GAZEBO_LIBRARIES})
```

Compiling this code will result in a shared library, `~/gazebo_plugin_tutorial/ build/libworld_edit .so` , that can be inserted in a Gazebo simulation.

```
$ mkdir ~/gazebo_plugin_tutorial/build
$ cd ~/gazebo_plugin_tutorial/build
$ cmake ../
$ make
```

Run Tutorial

First you need to add the folder to the `GAZEBO_PLUGIN_PATH` environment variable:

```
export GAZEBO_PLUGIN_PATH=${GAZEBO_PLUGIN_PATH}:~/gazebo_plugin_tutorial/build/
```

Then in a terminal

```
$ cd ~/gazebo_plugin_tutorial
$ gazebo world_edit.world
```

You should see an empty world.

Now add a box to the world using the Box icon located above the render window. The box should float up and away from the camera.

(//plus.google.com/u/0/115981436296571800301?prsrc=3)
(https://www.youtube.com/channel/UCJyqf9XJpDoM9X