```
// Puts a motor in an object


// Retrieved from Free SL Scripts on  www.gendersquare.org/sl


// by Ben
// Some help from Casval and Dave
rotation rot;
key owner;



reset()
{
    vector pos = llGetPos();
    pos.z = pos.z + 2.0;
    llMoveToTarget(pos, 0.3);
    llRotLookAt(rot, 0.1, 1.0);
    llSleep(1.0);
    llStopLookAt();
    llStopMoveToTarget();
}



default
{



    state_entry()
    {
        llSetSoundQueueing(FALSE);
        llPassCollisions(TRUE);



        llSetSitText("Ride");
```

```
        llSitTarget(<-0.0, 0.0, 0.2>, <0.00000, -0.25882, 0.00000, 0.96593>);
        llSetCameraEyeOffset(<-4.0, 0.0, 3.00>);
        llSetCameraAtOffset(<0.0, 0.0, 2.0>);

        llSetVehicleType(VEHICLE_TYPE_CAR);
        llRemoveVehicleFlags(-1);

        llSetVehicleFlags(VEHICLE_FLAG_HOVER_WATER_ONLY |
VEHICLE_FLAG_HOVER_UP_ONLY | VEHICLE_FLAG_NO_DEFLECTION_UP);

        llSetVehicleFloatParam(VEHICLE_ANGULAR_DEFLECTION_EFFICIENCY, 0.2);
        llSetVehicleFloatParam(VEHICLE_LINEAR_DEFLECTION_EFFICIENCY, 0.15);
        llSetVehicleFloatParam(VEHICLE_ANGULAR_DEFLECTION_TIMESCALE, 1.0);
        llSetVehicleFloatParam(VEHICLE_LINEAR_DEFLECTION_TIMESCALE, 1000.0);

        llSetVehicleFloatParam(VEHICLE_LINEAR_MOTOR_TIMESCALE, 1.0);
        llSetVehicleFloatParam(VEHICLE_LINEAR_MOTOR_DECAY_TIMESCALE, 0.1);
        llSetVehicleFloatParam(VEHICLE_ANGULAR_MOTOR_TIMESCALE, 0.1);
        llSetVehicleFloatParam(VEHICLE_ANGULAR_MOTOR_DECAY_TIMESCALE, 0.1);

        llSetVehicleVectorParam(VEHICLE_LINEAR_FRICTION_TIMESCALE, <10, 10, 10.0>);
        llSetVehicleVectorParam(VEHICLE_ANGULAR_FRICTION_TIMESCALE, <10.0, 10.0,
10>);

        llSetVehicleFloatParam(VEHICLE_VERTICAL_ATTRACTION_EFFICIENCY, 0.2);
        llSetVehicleFloatParam(VEHICLE_VERTICAL_ATTRACTION_TIMESCALE, 0.5);

        llSetVehicleFloatParam(VEHICLE_BUOYANCY, 0);
        llSetVehicleFloatParam(VEHICLE_HOVER_HEIGHT, 0.01);
        llSetVehicleFloatParam(VEHICLE_HOVER_EFFICIENCY, 0.5);

        llSetVehicleFloatParam(VEHICLE_HOVER_TIMESCALE, 0.1);
        llSetVehicleFloatParam(VEHICLE_BANKING_EFFICIENCY, 0.1);
        llSetVehicleFloatParam(VEHICLE_BANKING_TIMESCALE, 0.01);
        llSetVehicleFloatParam(VEHICLE_BANKING_MIX, 1.0);

        llCollisionSound("", 0.0);
    }

    on_rez(integer param)
    {
        if(owner != llGetOwner())
        {
            owner = llGetOwner();
            llGiveInventory(owner, "Innertube Instructions");
        }
```

```
    }

    changed(integer change)
    {
        if (change & CHANGED_LINK)
        {
            key agent = llAvatarOnSitTarget();

            if (agent)
            {
                if (agent != llGetOwner())
                {
                    llSay(0, "You aren't the owner");
                    llUnSit(agent);
                    llPushObject(agent, <0,0,100>, ZERO_VECTOR, FALSE);
                }
                else
                {
                    llSetStatus(STATUS_PHYSICS, TRUE);



                    llRequestPermissions(agent, PERMISSION_TRIGGER_ANIMATION |
PERMISSION_TAKE_CONTROLS);

                    llMessageLinked(LINK_SET, 0, "on", "");
                    llSetTimerEvent(1.0);
                }
        }
            else
            {
                llSetStatus(STATUS_PHYSICS, FALSE);
                llReleaseControls();

                //llStopAnimation("crouch");
                llMessageLinked(LINK_SET, 0, "off", "");
                llSetTimerEvent(0.0);
                llStopSound();
            }
        }

    }

    run_time_permissions(integer perm)
    {
        if (perm)
```

```
        {
            //llStopAnimation("sit");
            //llStartAnimation("crouch");
            llTakeControls(CONTROL_FWD | CONTROL_BACK | CONTROL_RIGHT |
CONTROL_LEFT | CONTROL_ROT_RIGHT | CONTROL_ROT_LEFT | CONTROL_UP |
CONTROL_DOWN, TRUE, FALSE);
        }
    }

    control(key id, integer level, integer edge)
    {
        vector angular_motor;

        if(level & CONTROL_FWD)
        {
            llSetVehicleVectorParam(VEHICLE_LINEAR_MOTOR_DIRECTION, <4, 0, 0>);
            llStartAnimation("falldown");
        }
        if(level & CONTROL_BACK)
        {

            llSetVehicleVectorParam(VEHICLE_LINEAR_MOTOR_DIRECTION, <-4,0,0>);
            llStartAnimation("falldown");
        }

        if(level & (CONTROL_RIGHT|CONTROL_ROT_RIGHT))
        {

            angular_motor.z = -PI * 0.25;
            llStartAnimation("falldown");

        }
        if(level & (CONTROL_LEFT|CONTROL_ROT_LEFT))
        {

            angular_motor.z = PI * 0.25;
            llStartAnimation("falldown");

        }
        if(level & (CONTROL_DOWN))
        {
            //reset();
        }
        else if(level == FALSE)
        {
            llStopAnimation("falldown");
```

```
        }
        llSetVehicleVectorParam(VEHICLE_ANGULAR_MOTOR_DIRECTION, angular_motor);
    }

    timer()
    {
        vector pos = llGetPos();
        if(( pos.z - llGround(ZERO_VECTOR) ) < 0.8)
        {
            rot = llGetRot();
        }
    }


}
```