Cuerpos físicos:

PhysicBody

Physic body es una clase abstracta, que contiene todos los atributos necesarios para la creación de btCollisionShape de cualquier tipo, adicionalmente introduce clases útiles para los modelos como LockRotation, DisableGravity, DisableModel, AddhidgeConection... incorpora ademas la función virtual ModelAction que permite a sus hijos ejecutar codigo propio cuando se lee el modelo en el render.

```
    /// @brief Add a hinge connection
    /// @param target The target physic body
    /// @param hidge_origin The hinge origin
    /// @param localAorigin The local origin A
    /// @param localBorigin The local origin B
    void AddhidgeConection(shared_ptr< PhysicBody> target, btVector3 hidge_origin, btVector3 localAorigin, btVector3 localBorigin);

    /// @brief Apply motor force
    /// @param velocity The velocity
    /// @param force The force
    void ApplyMotorForce(float velocity, float force)
    {
        for (const auto& hindge : hindgeConection)
        {
            hindge->enableAngularMotor(true, velocity, force); // Enable motor with target velocity of 1.0 rad/sec and max impulse of 10.0
        }
    }

    /// @brief Handle collision
    /// @param object_a The first collision object
    /// @param object_b The second collision object
    virtual void HandleColision(btCollisionObject* object_a, btCollisionObject* object_b) { };

    /// @brief Model action
    virtual void ModelAction() { };

    /// @brief Destructor only exist to make polymorfic
    virtual ~PhysicBody() {}
};
```

PhysicPlane

Clase hija de PhysicBody, instancia una btBoxShape con un cube material de opengl toolkit asociado a el.

PhysicSphere

Clase hija de PhysicBody instancia una btSphereShape con un modelo sphere cargado de la carpeta assets, debido a errores de sincronía de rigid body/ modelo la escala esta siempre fija independientemente de lo que se pongan en el constructor

Tank

clase hija de PhysicPlane, usando el anteriormente mencionado ModelAction añade fuerza al motor que une las hidges de la rueda pudiendo desplazarse en el eje x y z en base a los valores establecidos en la clase escena.

DoorModel

clase hija de PhysicPlane, similar a tank aplica fuerzas a su hidge solo cuando recibe un mensaje de trigger, mandando por key.

Key

clase hija de PhysicPlane, usando la función HandleColision de su padre comprueba si la colisión de un rigidbody es entre el y su target en este caso el tanque, de momento solo reconoce como player a la rueda delantera del tanque.

MovingPlataform
clase hija de PhysicPlane, usando ModelAction aplica velocidades para desplazarse de izquierda a derecha esperando un rato al llegar al final.

Partículas:

Particle
clase base del sistema de particulas, contiene un PhysicSphere con las colisiones desactivadas actuando como partícula. Debido a problemas con el set transform de bullet siempre estan activas, activar o desactivar solo cambia el sentido de moviemiento.

Particle pool
clase que contiene Lista de particles activas e inactivas y gestiona su vida.

Particle Updater
Itera el movimiento de las partículas y cuando su lifetime se acaba los manda a la lista de inactivos, como se menciono anteriormente realmente no se desactivan solo se les invierte el movimiento.

Particle System
Clase maestra del sistema, gestiona la función run y render si lo necesitase.

Mensajeria:

Message
Estructura de datos que contiene un id indicativo del mensaje y dos mapas de datos, uno que permite mandar mensajes con un solo parámetro y otro que admite dos.

Message lisetener
Clase abstracta que recibe mensajes de un message dispacher y gestiona los datos, de dicho mensaje

Message dispacher
Clase que almacena una lista de listener con el mismo id y manda mensajes a aquellos listeners intersados

Kernel:

Colision Checker
Clase hija de message dispacher, usando el manifold_count llama a la función HandleColision de los modelos que colisionan.

InputManager

Clase que lee los inputs del jugador y manda mensajes en base a la tecla presionada

Escena

Kernel general del juego. Originalmente eran 3 clases Windows world y Scene pero debido a problemas con punteros y bullets se tuvo que unir la funcionalidad en una unica clase. Cada tick gestiona colisiones, movimiento de partículas, inputs de jugador y simulaciones de física.