

Space Object Simulation

Generated by Doxygen 1.13.2

1 Hierarchical Index	1
1.1 Class Hierarchy	1
2 Class Index	3
2.1 Class List	3
3 File Index	5
3.1 File List	5
4 Class Documentation	7
4.1 DynamicMovementStrategy Class Reference	7
4.1.1 Detailed Description	8
4.1.2 Constructor & Destructor Documentation	8
4.1.2.1 DynamicMovementStrategy()	8
4.1.3 Member Function Documentation	8
4.1.3.1 update_position()	8
4.1.3.2 update_velocity()	9
4.2 IMovementStrategy Class Reference	9
4.2.1 Detailed Description	10
4.2.2 Constructor & Destructor Documentation	10
4.2.2.1 IMovementStrategy()	10
4.2.3 Member Function Documentation	10
4.2.3.1 update_position()	10
4.2.3.2 update_velocity()	11
4.3 SpaceObject Class Reference	11
4.3.1 Constructor & Destructor Documentation	12
4.3.1.1 SpaceObject() [1/2]	12
4.3.1.2 SpaceObject() [2/2]	13
4.3.2 Member Function Documentation	13
4.3.2.1 get_id()	13
4.3.2.2 get_mass()	13
4.3.2.3 get_name()	14
4.3.2.4 get_object_count()	14
4.3.2.5 get_position()	14
4.3.2.6 get_radius()	14
4.3.2.7 get_velocity()	14
4.3.2.8 is_movable()	15
4.3.2.9 operator=()	15
4.3.2.10 print_info()	15
4.3.2.11 set_mass()	15
4.3.2.12 set_movability()	15
4.3.2.13 set_name()	16
4.3.2.14 set_position()	16

4.3.2.15 set_radius()	16
4.3.2.16 set_velocity()	16
4.3.2.17 update_position()	17
4.3.2.18 update_velocity()	17
4.3.3 Friends And Related Symbol Documentation	17
4.3.3.1 operator<<	17
4.3.3.2 operator>>	18
4.4 SpaceObjectException Class Reference	18
4.4.1 Detailed Description	19
4.4.2 Constructor & Destructor Documentation	19
4.4.2.1 SpaceObjectException()	19
4.5 SpaceObject::SpaceObjectImpl Class Reference	19
4.6 StaticMovementStrategy Class Reference	20
4.6.1 Detailed Description	21
4.6.2 Constructor & Destructor Documentation	21
4.6.2.1 StaticMovementStrategy()	21
4.6.3 Member Function Documentation	21
4.6.3.1 update_position()	21
4.6.3.2 update_velocity()	21
5 File Documentation	23
5.1 exception.hpp	23
5.2 dynamic.hpp	23
5.3 i_movement_strategy.hpp	23
5.4 static.hpp	24
5.5 space_object.hpp	24

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

IMovementStrategy	9
DynamicMovementStrategy	7
StaticMovementStrategy	20
std::runtime_error	
SpaceObjectException	18
SpaceObject	11
SpaceObject::SpaceObjectImpl	19

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

DynamicMovementStrategy	
Movement strategy for dynamic movement based on gravitational forces	7
IMovementStrategy	
Abstract base class for movement strategies of space objects	9
SpaceObject	11
SpaceObjectException	
Exception class for handling errors related to SpaceObject	18
SpaceObject::SpaceObjectImpl	19
StaticMovementStrategy	
Movement strategy for static objects with fixed position and velocity	20

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

exception.hpp	23
space_object.hpp	24
movement/dynamic.hpp	23
movement/i_movement_strategy.hpp	23
movement/static.hpp	24

Chapter 4

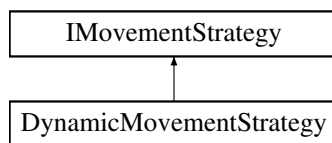
Class Documentation

4.1 DynamicMovementStrategy Class Reference

Movement strategy for dynamic movement based on gravitational forces.

```
#include <dynamic.hpp>
```

Inheritance diagram for DynamicMovementStrategy:



Public Member Functions

- [DynamicMovementStrategy](#) (double *mass_ptr, double *radius_ptr, sf::Vector2f *position_ptr, sf::Vector2f *velocity_ptr)
Constructor that initializes the dynamic movement strategy.
- void [update_velocity](#) (const std::vector< const [SpaceObject](#) * > &others, const float gravitational_constant, const float delta_time) override
Updates the velocity based on gravitational interactions with other objects.
- void [update_position](#) (const float delta_time) override
Updates the position based on the object's velocity.

Public Member Functions inherited from [IMovementStrategy](#)

- [IMovementStrategy](#) (double *mass_ptr, double *radius_ptr, sf::Vector2f *position_ptr, sf::Vector2f *velocity_ptr)
Constructor to initialize the movement strategy with required pointers.

Additional Inherited Members

Protected Attributes inherited from [IMovementStrategy](#)

- double * **mass**
Pointer to the object's mass.
- double * **radius**
Pointer to the object's radius.
- sf::Vector2f * **position**
Pointer to the object's position.
- sf::Vector2f * **velocity**
Pointer to the object's velocity.

4.1.1 Detailed Description

Movement strategy for dynamic movement based on gravitational forces.

Updates velocity and position of a space object considering gravitational forces from other objects.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 DynamicMovementStrategy()

```
DynamicMovementStrategy::DynamicMovementStrategy (
    double * mass_ptr,
    double * radius_ptr,
    sf::Vector2f * position_ptr,
    sf::Vector2f * velocity_ptr) [inline]
```

Constructor that initializes the dynamic movement strategy.

Parameters

<i>mass_ptr</i>	Pointer to the object's mass.
<i>radius_ptr</i>	Pointer to the object's radius.
<i>position_ptr</i>	Pointer to the object's position.
<i>velocity_ptr</i>	Pointer to the object's velocity.

4.1.3 Member Function Documentation

4.1.3.1 update_position()

```
void DynamicMovementStrategy::update_position (
    const float delta_time) [override], [virtual]
```

Updates the position based on the object's velocity.

Parameters

<code>delta_time</code>	Time step for the update.
-------------------------	---------------------------

Implements [IMovementStrategy](#).

4.1.3.2 update_velocity()

```
void DynamicMovementStrategy::update_velocity (
    const std::vector< const SpaceObject * > & others,
    const float gravitational_constant,
    const float delta_time) [override], [virtual]
```

Updates the velocity based on gravitational interactions with other objects.

Parameters

<code>others</code>	List of other space objects in the universe for gravitational calculation.
<code>gravitational_constant</code>	The gravitational constant.
<code>delta_time</code>	Time step for the update.

Implements [IMovementStrategy](#).

The documentation for this class was generated from the following files:

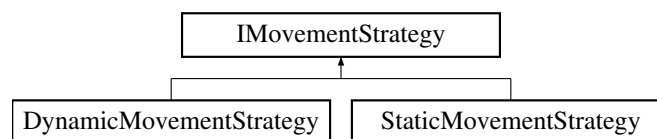
- movement/dynamic.hpp
- C:/Users/ugniu/code/cpp-2025/src/core/movement_dynamic.cpp

4.2 IMovementStrategy Class Reference

Abstract base class for movement strategies of space objects.

```
#include <i_movement_strategy.hpp>
```

Inheritance diagram for IMovementStrategy:



Public Member Functions

- [IMovementStrategy](#) (double *mass_ptr, double *radius_ptr, sf::Vector2f *position_ptr, sf::Vector2f *velocity_ptr)
Constructor to initialize the movement strategy with required pointers.
- virtual void [update_velocity](#) (const std::vector< const [SpaceObject](#) * > &others, const float gravitational_constant, const float delta_time)=0
Updates the velocity based on the movement strategy.
- virtual void [update_position](#) (const float delta_time)=0
Updates the position based on the movement strategy.

Protected Attributes

- double * **mass**
Pointer to the object's mass.
- double * **radius**
Pointer to the object's radius.
- sf::Vector2f * **position**
Pointer to the object's position.
- sf::Vector2f * **velocity**
Pointer to the object's velocity.

4.2.1 Detailed Description

Abstract base class for movement strategies of space objects.

Defines the interface for updating velocity and position of space objects.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 IMovementStrategy()

```
IMovementStrategy::IMovementStrategy (
    double * mass_ptr,
    double * radius_ptr,
    sf::Vector2f * position_ptr,
    sf::Vector2f * velocity_ptr) [inline]
```

Constructor to initialize the movement strategy with required pointers.

Parameters

<i>mass_ptr</i>	Pointer to the object's mass.
<i>radius_ptr</i>	Pointer to the object's radius.
<i>position_ptr</i>	Pointer to the object's position.
<i>velocity_ptr</i>	Pointer to the object's velocity.

4.2.3 Member Function Documentation

4.2.3.1 update_position()

```
virtual void IMovementStrategy::update_position (
    const float delta_time) [pure virtual]
```

Updates the position based on the movement strategy.

Parameters

<i>delta_time</i>	Time step for the update.
-------------------	---------------------------

Implemented in [DynamicMovementStrategy](#), and [StaticMovementStrategy](#).

4.2.3.2 update_velocity()

```
virtual void IMovementStrategy::update_velocity (
    const std::vector< const SpaceObject * > & others,
    const float gravitational_constant,
    const float delta_time) [pure virtual]
```

Updates the velocity based on the movement strategy.

Parameters

<i>others</i>	List of other space objects.
<i>gravitational_constant</i>	Gravitational constant.
<i>delta_time</i>	Time step for the update.

Implemented in [DynamicMovementStrategy](#), and [StaticMovementStrategy](#).

The documentation for this class was generated from the following file:

- movement/i_movement_strategy.hpp

4.3 SpaceObject Class Reference

Classes

- class [SpaceObjectImpl](#)

Public Member Functions

- [SpaceObject](#) (std::string name, double mass, double radius, sf::Vector2f position, sf::Vector2f velocity={0, 0}, bool [is_movable](#)=true)
Constructs a [SpaceObject](#) with specified properties.
- [SpaceObject](#) (const [SpaceObject](#) &other)
Copy constructor for [SpaceObject](#).
- [SpaceObject](#) & [operator=](#) (const [SpaceObject](#) &other)
Assignment operator for [SpaceObject](#).
- void [print_info](#) (std::ostream &output) const
Prints the information of the space object.
- void [update_velocity](#) (std::vector< const [SpaceObject](#) * > &others, const float gravitational_constant, const float delta_time)
Updates the velocity of the space object.
- void [update_position](#) (const float delta_time)
Updates the position of the space object.
- int [get_id](#) () const
Gets the unique ID of the space object.
- std::string [get_name](#) () const
Gets the name of the space object.
- double [get_mass](#) () const
Gets the mass of the space object.

- double [get_radius](#) () const
Gets the radius of the space object.
- sf::Vector2f [get_velocity](#) () const
Gets the current velocity of the space object.
- sf::Vector2f [get_position](#) () const
Gets the current position of the space object.
- bool [is_movable](#) () const
Checks whether the space object is movable.
- void [set_name](#) (std::string name)
Sets the name of the space object.
- void [set_mass](#) (double mass)
Sets the mass of the space object.
- void [set_radius](#) (double radius)
Sets the radius of the space object.
- void [set_position](#) (sf::Vector2f position)
Sets the position of the space object.
- void [set_velocity](#) (sf::Vector2f velocity)
Sets the velocity of the space object.
- void [set_movability](#) (bool movable)
Sets whether the object is movable or static.

Static Public Member Functions

- static int [get_object_count](#) ()
Gets the total number of active [SpaceObject](#) instances.

Friends

- std::ofstream & [operator<<](#) (std::ofstream &out, const [SpaceObject](#) &obj)
Serialization operator (<<) for saving [SpaceObject](#) to a file.
- std::ifstream & [operator>>](#) (std::ifstream &in, [SpaceObject](#) &obj)
Deserialization operator (>>) for reading [SpaceObject](#) from a file.

4.3.1 Constructor & Destructor Documentation

4.3.1.1 SpaceObject() [1/2]

```
SpaceObject::SpaceObject (
    std::string name,
    double mass,
    double radius,
    sf::Vector2f position,
    sf::Vector2f velocity = {0, 0},
    bool is_movable = true)
```

Constructs a [SpaceObject](#) with specified properties.

Parameters

<i>name</i>	Name of the space object.
<i>mass</i>	Mass of the space object.
<i>radius</i>	Radius of the space object.
<i>position</i>	Position of the space object.
<i>velocity</i>	Velocity of the space object (default is {0, 0}).
<i>is_movable</i>	Whether the space object is movable (default is true).

4.3.1.2 SpaceObject() [2/2]

```
SpaceObject::SpaceObject (  
    const SpaceObject & other)
```

Copy constructor for [SpaceObject](#).

Parameters

<i>other</i>	Another SpaceObject to copy from.
--------------	---

4.3.2 Member Function Documentation

4.3.2.1 get_id()

```
int SpaceObject::get_id () const
```

Gets the unique ID of the space object.

Returns

The ID of the space object.

4.3.2.2 get_mass()

```
double SpaceObject::get_mass () const
```

Gets the mass of the space object.

Returns

The mass as a double.

4.3.2.3 `get_name()`

```
std::string SpaceObject::get_name () const
```

Gets the name of the space object.

Returns

The name as a string.

4.3.2.4 `get_object_count()`

```
int SpaceObject::get_object_count () [static]
```

Gets the total number of active [SpaceObject](#) instances.

Returns

The current count of active SpaceObjects.

4.3.2.5 `get_position()`

```
sf::Vector2f SpaceObject::get_position () const
```

Gets the current position of the space object.

Returns

The position as an `sf::Vector2f`.

4.3.2.6 `get_radius()`

```
double SpaceObject::get_radius () const
```

Gets the radius of the space object.

Returns

The radius as a double.

4.3.2.7 `get_velocity()`

```
sf::Vector2f SpaceObject::get_velocity () const
```

Gets the current velocity of the space object.

Returns

The velocity as an `sf::Vector2f`.

4.3.2.8 is_movable()

```
bool SpaceObject::is_movable () const
```

Checks whether the space object is movable.

Returns

True if movable, false otherwise.

4.3.2.9 operator=()

```
SpaceObject & SpaceObject::operator= (  
    const SpaceObject & other)
```

Assignment operator for [SpaceObject](#).

Parameters

<i>other</i>	Another SpaceObject to assign from.
--------------	---

Returns

A reference to this [SpaceObject](#).

4.3.2.10 print_info()

```
void SpaceObject::print_info (  
    std::ostream & output) const
```

Prints the information of the space object.

Parameters

<i>output</i>	Output stream to print information.
---------------	-------------------------------------

4.3.2.11 set_mass()

```
void SpaceObject::set_mass (  
    double mass)
```

Sets the mass of the space object.

Parameters

<i>mass</i>	New mass value.
-------------	-----------------

4.3.2.12 set_movability()

```
void SpaceObject::set_movability (  
    bool movable)
```

Sets whether the object is movable or static.

Parameters

<i>movable</i>	True to make the object movable, false to make it static.
----------------	---

4.3.2.13 set_name()

```
void SpaceObject::set_name (  
    std::string name)
```

Sets the name of the space object.

Parameters

<i>name</i>	New name as a string.
-------------	-----------------------

4.3.2.14 set_position()

```
void SpaceObject::set_position (  
    sf::Vector2f position)
```

Sets the position of the space object.

Parameters

<i>position</i>	New position as an sf::Vector2f.
-----------------	----------------------------------

4.3.2.15 set_radius()

```
void SpaceObject::set_radius (  
    double radius)
```

Sets the radius of the space object.

Parameters

<i>radius</i>	New radius value.
---------------	-------------------

4.3.2.16 set_velocity()

```
void SpaceObject::set_velocity (  
    sf::Vector2f velocity)
```

Sets the velocity of the space object.

Parameters

<i>velocity</i>	New velocity as an <code>sf::Vector2f</code> .
-----------------	--

4.3.2.17 update_position()

```
void SpaceObject::update_position (
    const float delta_time)
```

Updates the position of the space object.

The position is updated based on the object's velocity and movement strategy (either dynamic or static). If the object is dynamic, its position will be adjusted based on its velocity. If the object is static, its position will remain unchanged.

Parameters

<i>delta_time</i>	Time step for the update.
-------------------	---------------------------

4.3.2.18 update_velocity()

```
void SpaceObject::update_velocity (
    std::vector< const SpaceObject * > & others,
    const float gravitational_constant,
    const float delta_time)
```

Updates the velocity of the space object.

The velocity is updated based on the object's movement strategy (either dynamic or static). If the object is dynamic, its velocity will be influenced by gravitational forces and other space objects. If the object is static, its velocity will be set to zero.

Parameters

<i>others</i>	List of other space objects for gravitational calculation.
<i>gravitational_constant</i>	Gravitational constant for the calculation.
<i>delta_time</i>	Time step for the update.

4.3.3 Friends And Related Symbol Documentation**4.3.3.1 operator<<**

```
std::ofstream & operator<< (
    std::ofstream & out,
    const SpaceObject & obj) [friend]
```

Serialization operator (<<) for saving `SpaceObject` to a file.

Parameters

<i>out</i>	Output file stream.
<i>obj</i>	SpaceObject to write.

Returns

Output file stream.

4.3.3.2 operator>>

```
std::ifstream & operator>> (
    std::ifstream & in,
    SpaceObject & obj) [friend]
```

Deserialization operator (>>) for reading [SpaceObject](#) from a file.

Parameters

<i>in</i>	Input file stream.
<i>obj</i>	SpaceObject to read into.

Returns

Input file stream.

The documentation for this class was generated from the following files:

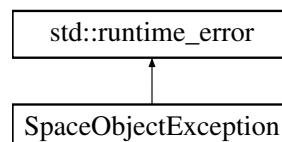
- `space_object.hpp`
- `C:/Users/ugniu/code/cpp-2025/src/core/space_object.cpp`

4.4 SpaceObjectException Class Reference

Exception class for handling errors related to [SpaceObject](#).

```
#include <exception.hpp>
```

Inheritance diagram for SpaceObjectException:

**Public Member Functions**

- [SpaceObjectException](#) (const std::string &message)
Constructs a [SpaceObjectException](#) with a given message.

4.4.1 Detailed Description

Exception class for handling errors related to [SpaceObject](#).

This class extends `std::runtime_error` to provide a custom exception for errors that occur during the processing of space objects.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 SpaceObjectException()

```
SpaceObjectException::SpaceObjectException (
    const std::string & message) [inline], [explicit]
```

Constructs a [SpaceObjectException](#) with a given message.

Parameters

<i>message</i>	The error message to describe the exception.
----------------	--

The documentation for this class was generated from the following file:

- exception.hpp

4.5 SpaceObject::SpaceObjectImpl Class Reference

Public Member Functions

- **SpaceObjectImpl** (std::string name, double mass, double radius, sf::Vector2f position, sf::Vector2f velocity, bool movable)
- **SpaceObjectImpl** (const [SpaceObjectImpl](#) &other)
- void **set_name** (std::string name)
- void **set_mass** (double mass)
- void **set_radius** (double radius)
- void **set_position** (sf::Vector2f position)
- void **set_velocity** (sf::Vector2f velocity)
- void **set_movability** (bool movable)
- void **set_movement_strategy** (bool movable)

Public Attributes

- int **id**
- std::string **name**
- double **mass**
- double **radius**
- sf::Vector2f **position**
- sf::Vector2f **velocity**
- bool **movable**
- [IMovementStrategy](#) * **movement_strategy** = nullptr

Static Public Attributes

- static int **object_count** = 0

The documentation for this class was generated from the following file:

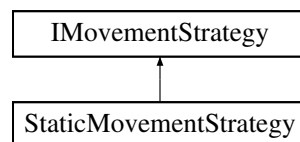
- C:/Users/ugniu/code/cpp-2025/src/core/space_object.cpp

4.6 StaticMovementStrategy Class Reference

Movement strategy for static objects with fixed position and velocity.

```
#include <static.hpp>
```

Inheritance diagram for StaticMovementStrategy:



Public Member Functions

- [StaticMovementStrategy](#) (double *mass_ptr, double *radius_ptr, sf::Vector2f *position_ptr, sf::Vector2f *velocity_ptr)
Constructor that initializes the static movement strategy.
- void [update_velocity](#) (const std::vector< const [SpaceObject](#) * > &others, const float gravitational_constant, const float delta_time) override
Sets the objects velocity to zero.
- void [update_position](#) (const float delta_time) override
Does not update the position for static objects.

Public Member Functions inherited from [IMovementStrategy](#)

- [IMovementStrategy](#) (double *mass_ptr, double *radius_ptr, sf::Vector2f *position_ptr, sf::Vector2f *velocity_ptr)
Constructor to initialize the movement strategy with required pointers.

Additional Inherited Members

Protected Attributes inherited from [IMovementStrategy](#)

- double * **mass**
Pointer to the object's mass.
- double * **radius**
Pointer to the object's radius.
- sf::Vector2f * **position**
Pointer to the object's position.
- sf::Vector2f * **velocity**
Pointer to the object's velocity.

4.6.1 Detailed Description

Movement strategy for static objects with fixed position and velocity.

Does not change the position or velocity of the space object.

4.6.2 Constructor & Destructor Documentation

4.6.2.1 StaticMovementStrategy()

```
StaticMovementStrategy::StaticMovementStrategy (
    double * mass_ptr,
    double * radius_ptr,
    sf::Vector2f * position_ptr,
    sf::Vector2f * velocity_ptr) [inline]
```

Constructor that initializes the static movement strategy.

Parameters

<i>mass_ptr</i>	Pointer to the object's mass.
<i>radius_ptr</i>	Pointer to the object's radius.
<i>position_ptr</i>	Pointer to the object's position.
<i>velocity_ptr</i>	Pointer to the object's velocity.

4.6.3 Member Function Documentation

4.6.3.1 update_position()

```
void StaticMovementStrategy::update_position (
    const float delta_time) [override], [virtual]
```

Does not update the position for static objects.

Parameters

<i>delta_time</i>	Time step for the update.
-------------------	---------------------------

Implements [IMovementStrategy](#).

4.6.3.2 update_velocity()

```
void StaticMovementStrategy::update_velocity (
    const std::vector< const SpaceObject * > & others,
    const float gravitational_constant,
    const float delta_time) [override], [virtual]
```

Sets the objects velocity to zero.

Parameters

<i>others</i>	List of other space objects.
<i>gravitational_constant</i>	Gravitational constant.
<i>delta_time</i>	Time step for the update.

Implements [IMovementStrategy](#).

The documentation for this class was generated from the following files:

- movement/static.hpp
- C:/Users/ugniu/code/cpp-2025/src/core/movement_static.cpp

Chapter 5

File Documentation

5.1 exception.hpp

```
00001 #ifndef EXCEPTION_HPP
00002 #define EXCEPTION_HPP
00003
00004 #include <stdexcept>
00005 #include <string>
00006
00013 class SpaceObjectException : public std::runtime_error
00014 {
00015 public:
00021     explicit SpaceObjectException(const std::string &message)
00022         : std::runtime_error(message) {}
00023 };
00024
00025 #endif // EXCEPTION_HPP
```

5.2 dynamic.hpp

```
00001 #ifndef DYNAMIC_HPP
00002 #define DYNAMIC_HPP
00003
00004 #include "i_movement_strategy.hpp"
00005
00011 class DynamicMovementStrategy : public IMovementStrategy
00012 {
00013 public:
00022     DynamicMovementStrategy(double *mass_ptr, double *radius_ptr, sf::Vector2f *position_ptr,
00023                             sf::Vector2f *velocity_ptr)
00024         : IMovementStrategy(mass_ptr, radius_ptr, position_ptr, velocity_ptr) {}
00025
00032     void update_velocity(const std::vector<const SpaceObject *> &others,
00033                         const float gravitational_constant, const float delta_time) override;
00034
00040     void update_position(const float delta_time) override;
00041 };
00042
00043 #endif // DYNAMIC_HPP
```

5.3 i_movement_strategy.hpp

```
00001 #ifndef I_MOVEMENT_STRATEGY_HPP
00002 #define I_MOVEMENT_STRATEGY_HPP
00003
00004 #include "space_object.hpp"
00005
00006 class SpaceObject;
00007
00013 class IMovementStrategy
00014 {
```

```

00015 protected:
00016     double *mass;
00017     double *radius;
00018     sf::Vector2f *position;
00019     sf::Vector2f *velocity;
00020
00021 public:
00022     IMovementStrategy(double *mass_ptr, double *radius_ptr, sf::Vector2f *position_ptr, sf::Vector2f
00023 *velocity_ptr)
00024         : mass(mass_ptr), radius(radius_ptr), position(position_ptr), velocity(velocity_ptr) {}
00025
00026     virtual ~IMovementStrategy() = default;
00027
00028     virtual void update_velocity(const std::vector<const SpaceObject *> &others,
00029                                 const float gravitational_constant, const float delta_time) = 0;
00030
00031     virtual void update_position(const float delta_time) = 0;
00032 };
00033
00034 #endif // I_MOVEMENT_STRATEGY_HPP

```

5.4 static.hpp

```

00001 #ifndef STATIC_HPP
00002 #define STATIC_HPP
00003
00004 #include "i_movement_strategy.hpp"
00005
00006 class StaticMovementStrategy : public IMovementStrategy
00007 {
00008 public:
00009     StaticMovementStrategy(double *mass_ptr, double *radius_ptr, sf::Vector2f *position_ptr,
00010 sf::Vector2f *velocity_ptr)
00011         : IMovementStrategy(mass_ptr, radius_ptr, position_ptr, velocity_ptr) {}
00012
00013     void update_velocity(const std::vector<const SpaceObject *> &others,
00014                         const float gravitational_constant, const float delta_time) override;
00015
00016     void update_position(const float delta_time) override;
00017 };
00018
00019 #endif // STATIC_HPP

```

5.5 space_object.hpp

```

00001 #ifndef SPACE_OBJECT_HPP
00002 #define SPACE_OBJECT_HPP
00003
00004 #include <string>
00005 #include <iostream>
00006 #include <fstream>
00007 #include <vector>
00008 #include <SFML/System.hpp>
00009
00010 #include "exception.hpp"
00011
00012 class SpaceObject
00013 {
00014 private:
00015     // Pointer to an implementation of SpaceObject
00016     class SpaceObjectImpl;
00017     SpaceObjectImpl *impl;
00018
00019 public:
00020     // Constructors and Deconstructor
00021
00022     SpaceObject();
00023
00024     SpaceObject(std::string name, double mass, double radius, sf::Vector2f position, sf::Vector2f
00025 velocity = {0, 0}, bool is_movable = true);
00026
00027     SpaceObject(const SpaceObject &other);
00028
00029     ~SpaceObject();
00030
00031     // Operators
00032
00033     SpaceObject &operator=(const SpaceObject &other);

```

```
00054
00062     friend std::ofstream &operator<<(std::ofstream &out, const SpaceObject &obj);
00063
00071     friend std::ifstream &operator>>(std::ifstream &in, SpaceObject &obj);
00072
00073     // Methods
00074
00080     void print_info(std::ostream &output) const;
00081
00093     void update_velocity(std::vector<const SpaceObject *> &others, const float gravitational_constant,
00094                          const float delta_time);
00094
00104     void update_position(const float delta_time);
00105
00106     // Getters
00107
00113     static int get_object_count();
00114
00120     int get_id() const;
00121
00127     std::string get_name() const;
00128
00134     double get_mass() const;
00135
00141     double get_radius() const;
00142
00148     sf::Vector2f get_velocity() const;
00149
00155     sf::Vector2f get_position() const;
00156
00162     bool is_movable() const;
00163
00164     // Setters
00165
00171     void set_name(std::string name);
00172
00178     void set_mass(double mass);
00179
00185     void set_radius(double radius);
00186
00192     void set_position(sf::Vector2f position);
00193
00199     void set_velocity(sf::Vector2f velocity);
00200
00206     void set_movability(bool movable);
00207 };
00208
00209 #endif // SPACE_OBJECT_HPP
```

