R-Type - Engine

Generated by Doxygen 1.9.1

Chapter 1

Engine

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

Class Documentation

4.1 Archetypes Class Reference

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

- src/Archetype/Archetypes.h

4.2 Audio Class Reference

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

- src/Components/all_components/Audio.h

4.3 Components Class Reference

Inheritance diagram for Components:

Public Member Functions

- virtual bool **init** ()
- virtual void **update** ()
- template<typename T >
  ComponentTypeID **getComponentTypeID** () noexcept

Protected Types

- using **ComponentTypeID** = std::size_t

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

- src/Components/Components.h
- src/Components/Components.cpp

4.4 DrawableComponent Class Reference

Inheritance diagram for DrawableComponent:

Public Member Functions

- virtual void **draw** (sf::RenderWindow &window) const =0

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

- src/Components/DrawableComponent.h

4.5 Entity Class Reference

Entity class: Entity is a class that represents an entity in the game.

```
#include <entity.h>
```

Inheritance diagram for Entity:

Collaboration diagram for Entity:

Public Member Functions

- Entity ()=default

  Default Entity constructor.
- Entity (std::string nameEntity, Archetypes newArchetype=Archetypes())

  Entity constructor.
- ∼Entity () override=default

  Entity destructor.
- bool init () override

  init(): Initialize the entity
- std::string getName () const

  genName(): Get the name of the entity
- void setName (std::string newName)

  setName(): Set the name of the entity
- void **addDrawable** (Components ∗component)
- void **draw** (sf::RenderWindow &window)
- template<typename T , typename... TArgs>
  T & addComponent (TArgs &&... args)

  addComponent(): Add a component to the entity
- template<typename T >
  T & getComponent ()

  getComponent(): Get a component from the entity
- std::bitset< 3 > **getComponentBitset** () const
- std::vector< DrawableComponent ∗ > **getDrawableComponents** () const
- std::array< Components ∗, 3 > **getComponentArrays** () const

Protected Types

- using **ComponentTypeID** = std::size_t

Protected Member Functions

- virtual void **update** ()
- template<typename T >
  ComponentTypeID **getComponentTypeID** () noexcept

4.5.1 Detailed Description

Entity class: Entity is a class that represents an entity in the game.

The Entity class manages components associated with the entity.

4.5.2 Constructor & Destructor Documentation

4.5.2.1 Entity() [1/2]

```
Entity::Entity ( )  [default]
```

Default Entity constructor.

Parameters

void	

Returns

void

4.5.2.2 Entity() [2/2]

```
Entity::Entity (
            std::string nameEntity,
            Archetypes newArchetype = Archetypes() )  [inline], [explicit]
```

Entity constructor.

Parameters

nameEntity	name of the entity
newArchetype	archetype of the entity (optional, default = new archetype)

Returns

void

4.5.2.3 ∼Entity()

```
Entity::∼Entity ( )  [override], [default]
```

Entity destructor.

Parameters

void	

Returns

void

4.5.3 Member Function Documentation

4.5.3.1 addComponent()

```
template<typename T , typename...  TArgs>
T & Entity::addComponent (
            TArgs &&...  args )
```

addComponent(): Add a component to the entity

Template Parameters

T	Type of the component
TArgs	Variadic template for component constructor arguments.

Parameters

args	arguments of the component

Returns

T&: reference of the component

4.5.3.2 getComponent()

```
template<typename T >
T & Entity::getComponent
```

getComponent(): Get a component from the entity

Template Parameters

T	Type of the component

Parameters

void	

Returns

T&: reference of the component

4.5.3.3 getName()

```
std::string Entity::getName ( ) const  [inline]
```

genName(): Get the name of the entity

Parameters

void	

Returns

std::string: name of the entity

4.5.3.4 init()

```
bool Entity::init ( )  [inline], [override], [virtual]
```

init(): Initialize the entity

Parameters

void	

Returns

bool: true if the entity is initialized, false otherwise

Reimplemented from Components.

Reimplemented in World, and EntityManager.

4.5.3.5 setName()

```
void Entity::setName (
            std::string newName )  [inline]
```

setName(): Set the name of the entity

Parameters

newName	new name of the entity

Returns

void

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

- src/Entity/entity.h
- src/Entity/entity.cpp

4.6 EntityManager Class Reference

Inheritance diagram for EntityManager:

Collaboration diagram for EntityManager:

Public Member Functions

- EntityManager ()=default

  Default EntityManager constructor.
- ∼EntityManager ()=default

  EntityManager destructor.
- Entity & addEntity (std::string nameEntity, Archetypes newArchetype=Archetypes())

  addEntity(): Create and add a new entity to the entity manager.
- Entity & getEntity (std::string nameEntity)

  getEntity(): Get an entity from the entity manager by its name.
- std::map< std::string, Entity ∗ > getEntities () const

  getEntities(): Get the EntityManager's entities.
- std::map< std::string, Entity ∗ > getEntityMap () const

  getEntityMap(): Get the EntityManager's entity map.
- bool init () override

  init(): Initialize the entity

Protected Types

- using **ComponentTypeID** = std::size_t

Protected Member Functions

- std::string getName () const

  genName(): Get the name of the entity
- void setName (std::string newName)

  setName(): Set the name of the entity
- void **addDrawable** (Components ∗component)
- void **draw** (sf::RenderWindow &window)
- template< typename T , typename... TArgs>
  T & addComponent (TArgs &&... args)

  addComponent(): Add a component to the entity
- template< typename T >
  T & getComponent ()

  getComponent(): Get a component from the entity
- std::bitset< 3 > **getComponentBitset** () const
- std::vector< DrawableComponent ∗ > **getDrawableComponents** () const
- std::array< Components ∗, 3 > **getComponentArrays** () const
- virtual void **update** ()
- template< typename T >
  ComponentTypeID **getComponentTypeID** () noexcept

4.6.1 Constructor & Destructor Documentation

4.6.1.1 EntityManager()

```
EntityManager::EntityManager ( )  [default]
```

Default EntityManager constructor.

Parameters

void	

Returns

void

4.6.1.2 ∼EntityManager()

```
EntityManager::∼EntityManager ( )  [default]
```

EntityManager destructor.

Parameters

void	

Returns

void

4.6.2 Member Function Documentation

4.6.2.1 addComponent()

```
template<typename T , typename...  TArgs>
T & Entity::addComponent (
             TArgs &&...  args )  [inherited]
```

addComponent(): Add a component to the entity

Template Parameters

T	Type of the component
TArgs	Variadic template for component constructor arguments.

Parameters

args	arguments of the component

Returns

T&: reference of the component

4.6.2.2 addEntity()

```
Entity & EntityManager::addEntity (
             std::string nameEntity,
             Archetypes newArchetype = Archetypes() )  [inline]
```

addEntity(): Create and add a new entity to the entity manager.

Template Parameters

T	Type of the entity.
TArgs	Type of the arguments.

Parameters

args	Arguments of the entity.

4.6.2.3 getComponent()

```
template<typename T >
T & Entity::getComponent  [inherited]
```

getComponent(): Get a component from the entity

Template Parameters

T	Type of the component

Parameters

void	

Returns

T&: reference of the component

4.6.2.4 getEntities()

```
std::map< std::string, Entity * > EntityManager::getEntities ( ) const  [inline]
```

getEntities(): Get the EntityManager's entities.

Parameters

void	

Returns

std::map<std::string, Entity ∗>: Entities.

4.6.2.5 getEntity()

```
Entity & EntityManager::getEntity (
             std::string nameEntity )  [inline]
```

getEntity(): Get an entity from the entity manager by its name.

Template Parameters

T	Type of the entity.

Parameters

nameEntity	Name of the entity.

Returns

T&: Reference of the entity.

4.6.2.6 getEntityMap()

```
std::map<std::string, Entity*> EntityManager::getEntityMap ( ) const  [inline]
```

getEntityMap(): Get the EntityManager's entity map.

Parameters

void	

Returns

Entity::EntityMap: Entity map.

4.6.2.7 getName()

```
std::string Entity::getName ( ) const  [inline], [inherited]
```

genName(): Get the name of the entity

Parameters

void	

Returns

std::string: name of the entity

4.6.2.8 init()

```
bool EntityManager::init ( )  [inline], [override], [virtual]
```

init(): Initialize the entity

Parameters

void	

Returns

bool: true if the entity is initialized, false otherwise

Reimplemented from Entity.

Reimplemented in World.

4.6.2.9 setName()

```
void Entity::setName (
             std::string newName )  [inline], [inherited]
```

setName(): Set the name of the entity

Parameters

newName	new name of the entity

Returns

void

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

- src/Entity/entityManager.h
- src/Entity/entityManager.cpp

4.7 EntityManagerTest Class Reference

Inheritance diagram for EntityManagerTest:

Collaboration diagram for EntityManagerTest:

Protected Member Functions

- void **SetUp** () override
- void **TearDown** () override

Protected Attributes

- EntityManager **entityManager** {}

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

- src/tests/Entity/TestEntityManager.cpp

4.8 EntityTest Class Reference

Inheritance diagram for EntityTest:

Collaboration diagram for EntityTest:

Protected Attributes

- Entity **entity**

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

- src/tests/Entity/TestEntity.cpp

4.9 EventEngine Class Reference

Inheritance diagram for EventEngine:

Public Member Functions

- bool **init** () const
- sf::Event & **getEvent** ()
- void **addKeyPressed** (sf::Keyboard::Key keyboard, std::function< void()> function)
- std::map< sf::Keyboard::Key, std::function< void()> > & **getKeyPressedMap** ()

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

- src/Event/event.h
- src/Event/event.cpp

4.10 GameEngine Class Reference

Inheritance diagram for GameEngine:

Collaboration diagram for GameEngine:

Public Member Functions

- **GameEngine** (sf::VideoMode mode, std::string type, sf::String title, sf::Uint32 style=sf::Style::Default, const sf::ContextSettings &settings=sf::ContextSettings())
- void **run** (std::map< std::string, std::unique_ptr< World >> mapWorld, std::map< std::string, std::string > pathRessources, std::string firstScene)
- void **run** ()
- void **renderGameEngine** ()
- void **eventGameEngine** ()
- bool **isWindowOpen** ()
- void **updateGameEngine** ()
- void **initialize** (std::map< std::string, std::unique_ptr< World >> mapWorld, std::map< std::string, std↩ ::string > pathRessources, std::string firstScene)
- void **initializeSprite** ()
- void **initializeTexture** (std::string path)
- void **initializeWorldMap** (std::map< std::string, std::unique_ptr< World >> mapWorld)
- const auto & **getWindow** ()
- void **setWindow** ()
- EventEngine & **getEventEngine** ()
- void **setCurrentWorld** (World ∗world)
- World ∗ **getCurrentWorld** ()
- World & **addWorld** (std::string nameWorld, std::unique_ptr< World > world)
- World & **getWorld** (std::string nameWorld)
- std::map< std::string, sf::Texture > **getMapTexture** () const
- std::map< std::string, World ∗ > **getWorldMap** () const

Protected Types

- using **ComponentTypeID** = std::size_t

Protected Member Functions

- void **createEntities** (std::map< std::string, std::pair< std::unique_ptr< EntityManager >, std::vector< std↩ ::string >>> &mapEntityManager, std::string keyEntityManager)
- EntityManager & **addEntityManager** (std::string NameEntityManager)
- EntityManager & **getEntityManager** (std::string NameEntityManager)
- void **setNameWorld** (std::string newName)
- std::string **getNameWorld** () const
- std::map< std::string, EntityManager ∗ > **getEntityManagerMap** () const
- bool init () override

  init(): Initialize the entity
- Entity & addEntity (std::string nameEntity, Archetypes newArchetype=Archetypes())

  addEntity(): Create and add a new entity to the entity manager.
- Entity & getEntity (std::string nameEntity)

  getEntity(): Get an entity from the entity manager by its name.

- std::map< std::string, Entity ∗ > getEntities () const

  getEntities(): Get the EntityManager's entities.
- std::map< std::string, Entity ∗ > getEntityMap () const

  getEntityMap(): Get the EntityManager's entity map.
- std::string getName () const

  genName(): Get the name of the entity
- void setName (std::string newName)

  setName(): Set the name of the entity
- void **addDrawable** (Components ∗component)
- void **draw** (sf::RenderWindow &window)
- template<typename T , typename... TArgs>
  T & addComponent (TArgs &&... args)

  addComponent(): Add a component to the entity
- template<typename T >
  T & getComponent ()

  getComponent(): Get a component from the entity
- std::bitset< 3 > **getComponentBitset** () const
- std::vector< DrawableComponent ∗ > **getDrawableComponents** () const
- std::array< Components ∗, 3 > **getComponentArrays** () const
- virtual void **update** ()
- template<typename T >
  ComponentTypeID **getComponentTypeID** () noexcept
- bool **init** () const
- sf::Event & **getEvent** ()
- void **addKeyPressed** (sf::Keyboard::Key keyboard, std::function< void()> function)
- std::map< sf::Keyboard::Key, std::function< void()> > & **getKeyPressedMap** ()

4.10.1 Member Function Documentation

4.10.1.1 addComponent()

```
template<typename T , typename...  TArgs>
T & Entity::addComponent (
            TArgs &&...  args )  [inherited]
```

addComponent(): Add a component to the entity

Template Parameters

T	Type of the component
TArgs	Variadic template for component constructor arguments.

Parameters

args	arguments of the component

Returns

T&: reference of the component

4.10.1.2 addEntity()

```
Entity & EntityManager::addEntity (
            std::string nameEntity,
            Archetypes newArchetype = Archetypes() )  [inline], [inherited]
```

addEntity(): Create and add a new entity to the entity manager.

Template Parameters

T	Type of the entity.
TArgs	Type of the arguments.

Parameters

args	Arguments of the entity.

4.10.1.3 getComponent()

```
template<typename T >
T & Entity::getComponent  [inherited]
```

getComponent(): Get a component from the entity

Template Parameters

T	Type of the component

Parameters

void	

Returns

T&: reference of the component

4.10.1.4 getEntities()

```
std::map< std::string, Entity * > EntityManager::getEntities ( ) const  [inline], [inherited]
```

getEntities(): Get the EntityManager's entities.

Parameters

void	

Returns

std::map<std::string, Entity *>: Entities.

4.10.1.5 getEntity()

```
Entity & EntityManager::getEntity (
             std::string nameEntity )  [inline], [inherited]
```

getEntity(): Get an entity from the entity manager by its name.

Template Parameters

T	Type of the entity.

Parameters

nameEntity	Name of the entity.

Returns

T&: Reference of the entity.

4.10.1.6 getEntityMap()

```
std::map<std::string, Entity*> EntityManager::getEntityMap ( ) const  [inline], [inherited]
```

getEntityMap(): Get the EntityManager's entity map.

Parameters

void	

Returns

Entity::EntityMap: Entity map.

4.10.1.7 getName()

```
std::string Entity::getName ( ) const  [inline], [inherited]
```

genName(): Get the name of the entity

Parameters

void	

Returns

std::string: name of the entity

4.10.1.8 init()

```
bool World::init ( )  [inline], [override], [virtual], [inherited]
```

init(): Initialize the entity

Parameters

void	

Returns

bool: true if the entity is initialized, false otherwise

Reimplemented from EntityManager.

4.10.1.9 setName()

```
void Entity::setName (
             std::string newName )  [inline], [inherited]
```

setName(): Set the name of the entity

Parameters

newName	new name of the entity

Returns

void

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

- src/GameEngine/gameEngine.h
- src/GameEngine/gameEngine.cpp

4.11 Sprite Class Reference

Sprite class: Sprite is a class that represents the rendering properties of a Component.

`#include <Sprite.h>`

Inheritance diagram for Sprite:

4.12 Transform Class Reference

Transform class: Transform is a class that represents the transform of a Component.

`#include <Transform.h>`

Inheritance diagram for Transform:

Collaboration diagram for Transform:

Public Member Functions

- Transform ()=default

    Default Transform constructor.
- bool **init** () const
- Transform (const std::map< std::string, std::vector< float >> &mapTransform)

    Transform constructor.
- ∼Transform () override=default

    Transform destructor.
- int getBit () const

    getBit(): Get the bitmask of the component
- std::vector< float > getPositionVector () const

    getPositionVector(): Get the position vector of the component;
- std::vector< float > getRotationVector () const

    getRotationVector(): Get the rotation vector of the component;
- std::vector< float > getScaleVector () const

    getScaleVector(): Get the scale vector of the component;
- void setTransform (const std::map< std::string, std::vector< float >> &mapTransform)

    setTransform(): Set the transformation properties of the component
- virtual bool **init** ()
- virtual void **update** ()
- template<typename T >
  ComponentTypeID **getComponentTypeID** () noexcept

Protected Types

- using **ComponentTypeID** = std::size_t

4.12.1 Detailed Description

Transform class: Transform is a class that represents the transform of a Component.

The Transform class manages the position, rotation and scale of a Component.

4.12.2 Constructor & Destructor Documentation

4.12.2.1 Transform() [1/2]

```
Transform::Transform ( )  [default]
```

Default Transform constructor.

Parameters

void	

Returns

void

4.12.2.2 Transform() [2/2]

```
Transform::Transform (
            const std::map< std::string, std::vector< float >> & mapTransform )  [inline],
[explicit]
```

Transform constructor.

Parameters

mapTransform	Map containing transformation properties (std::string, std::vector<float>).

Returns

void

4.12.2.3 ∼Transform()

```
Transform::∼Transform ( )  [override], [default]
```

Transform destructor.

Parameters

void	

Returns

void

4.12.3 Member Function Documentation

4.12.3.1 getBit()

```
int Transform::getBit ( ) const
```

getBit(): Get the bitmask of the component

Parameters

void	

Returns

int: bitmask of the component

4.12.3.2 getPositionVector()

```
std::vector< float > Transform::getPositionVector ( ) const
```

getPositionVector(): Get the position vector of the component;

Parameters

void	

Returns

std::vector<float>: position vector of the component

4.12.3.3 getRotationVector()

```
std::vector< float > Transform::getRotationVector ( ) const
```

getRotationVector(): Get the rotation vector of the component;

Parameters

void	

Returns

std::vector<float>: rotation vector of the component

4.12.3.4 getScaleVector()

```
std::vector< float > Transform::getScaleVector ( ) const
```

getScaleVector(): Get the scale vector of the component;

Parameters

void	

Returns

std::vector<float>: scale vector of the component

4.12.3.5 setTransform()

```
void Transform::setTransform (
             const std::map< std::string, std::vector< float >> & mapTransform )
```

setTransform(): Set the transformation properties of the component

Parameters

mapTransform	Map containing transformation properties (std::string, std::vector<float>).

Returns

void

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

- src/Components/all_components/Transform.h
- src/Components/all_components/Transform.cpp

4.13 TransformTest Class Reference

Inheritance diagram for TransformTest:

Collaboration diagram for TransformTest:

Protected Attributes

- Transform **transform**

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

- src/tests/Components/all_components/TestTransform.cpp

4.14 World Class Reference

Inheritance diagram for World:

Collaboration diagram for World:

Public Member Functions

- void **createEntities** (std::map< std::string, std::pair< std::unique_ptr< EntityManager >, std::vector< std↵ ::string >>> &mapEntityManager, std::string keyEntityManager)
- EntityManager & **addEntityManager** (std::string NameEntityManager)
- EntityManager & **getEntityManager** (std::string NameEntityManager)
- void **setNameWorld** (std::string newName)
- std::string **getNameWorld** () const
- std::map< std::string, EntityManager ∗ > **getEntityManagerMap** () const
- bool init () override

  init(): Initialize the entity

Protected Types

- using **ComponentTypeID** = std::size_t

Protected Member Functions

- Entity & addEntity (std::string nameEntity, Archetypes newArchetype=Archetypes())

    addEntity(): Create and add a new entity to the entity manager.
- Entity & getEntity (std::string nameEntity)

    getEntity(): Get an entity from the entity manager by its name.
- std::map< std::string, Entity ∗ > getEntities () const

    getEntities(): Get the EntityManager's entities.
- std::map< std::string, Entity ∗ > getEntityMap () const

    getEntityMap(): Get the EntityManager's entity map.
- std::string getName () const

    genName(): Get the name of the entity
- void setName (std::string newName)

    setName(): Set the name of the entity
- void **addDrawable** (Components ∗component)
- void **draw** (sf::RenderWindow &window)
- template<typename T , typename... TArgs>
  T & addComponent (TArgs &&... args)

    addComponent(): Add a component to the entity
- template<typename T >
  T & getComponent ()

    getComponent(): Get a component from the entity
- std::bitset< 3 > **getComponentBitset** () const
- std::vector< DrawableComponent ∗ > **getDrawableComponents** () const
- std::array< Components ∗, 3 > **getComponentArrays** () const
- virtual void **update** ()
- template<typename T >
  ComponentTypeID **getComponentTypeID** () noexcept

4.14.1 Member Function Documentation

4.14.1.1 addComponent()

```
template<typename T , typename...  TArgs>
T & Entity::addComponent (
            TArgs &&...  args )  [inherited]
```

addComponent(): Add a component to the entity

Template Parameters

T	Type of the component
TArgs	Variadic template for component constructor arguments.

Parameters

args	arguments of the component

Returns

T&: reference of the component

4.14.1.2 addEntity()

```
Entity & EntityManager::addEntity (
             std::string nameEntity,
             Archetypes newArchetype = Archetypes() )  [inline], [inherited]
```

addEntity(): Create and add a new entity to the entity manager.

Template Parameters

T	Type of the entity.
TArgs	Type of the arguments.

Parameters

args	Arguments of the entity.

4.14.1.3 getComponent()

```
template<typename T >
T & Entity::getComponent  [inherited]
```

getComponent(): Get a component from the entity

Template Parameters

T	Type of the component

Parameters

void	

Returns

T&: reference of the component

4.14.1.4 getEntities()

```
std::map< std::string, Entity * > EntityManager::getEntities ( ) const  [inline], [inherited]
```

getEntities(): Get the EntityManager's entities.

Parameters

void	

Returns

std::map<std::string, Entity *>: Entities.

4.14.1.5 getEntity()

```
Entity & EntityManager::getEntity (
            std::string nameEntity )  [inline], [inherited]
```

getEntity(): Get an entity from the entity manager by its name.

Template Parameters

T	Type of the entity.

Parameters

nameEntity	Name of the entity.

Returns

T&: Reference of the entity.

4.14.1.6 getEntityMap()

```
std::map<std::string, Entity*> EntityManager::getEntityMap ( ) const  [inline], [inherited]
```

getEntityMap(): Get the EntityManager's entity map.

Parameters

void	

Returns

Entity::EntityMap: Entity map.

4.14.1.7 getName()

```
std::string Entity::getName ( ) const  [inline], [inherited]
```

genName(): Get the name of the entity

Parameters

void	

Returns

std::string: name of the entity

4.14.1.8 init()

```
bool World::init ( )  [inline], [override], [virtual]
```

init(): Initialize the entity

Parameters

void	

Returns

bool: true if the entity is initialized, false otherwise

Reimplemented from EntityManager.

4.14.1.9 setName()

```
void Entity::setName (
             std::string newName )  [inline], [inherited]
```

setName(): Set the name of the entity

Parameters

newName	new name of the entity

Returns

void

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

- src/World/world.h
- src/World/world.cpp

Index