R-Type architecture

Generated by Doxygen 1.13.2

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

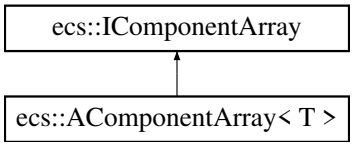
Here is a list of all documented files with brief descriptions:

Chapter 4

Class Documentation

4.1 ecs::AComponentArray< T > Class Template Reference

Inheritance diagram for ecs::AComponentArray< T >:



Public Member Functions

- void **add** (Entity entityId, std::shared_ptr< T > component)
- std::shared_ptr< T > **get** (Entity entityId) const
- std::vector< std::shared_ptr< T > > **getAll** (Entity entityId) const
- void removeComponents (Entity entityId) override
- void removeOneComponent (Entity entityId) override
- bool **has** (Entity entityId) const
- Entity getMaxEntityId () const override
- void clear () override

Private Attributes

- std::vector< std::vector< std::shared_ptr< T > > > **_components**

4.1.1 Member Function Documentation

4.1.1.1 clear()

```
template<typename T>
void ecs::AComponentArray< T >::clear ()  [override], [virtual]
```

Implements ecs::IComponentArray.

4.1.1.2 getMaxEntityId()

```
template<typename T>
Entity ecs::AComponentArray< T >::getMaxEntityId () const  [override], [virtual]
```

Implements ecs::IComponentArray.

4.1.1.3 removeComponents()

```
template<typename T>
void ecs::AComponentArray< T >::removeComponents (
            Entity entityId)  [override], [virtual]
```

Implements ecs::IComponentArray.

4.1.1.4 removeOneComponent()

```
template<typename T>
void ecs::AComponentArray< T >::removeOneComponent (
            Entity entityId)  [override], [virtual]
```

Implements ecs::IComponentArray.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/ECS/entity/componentArray/AComponentArray.hpp

4.2 ActionFactory Class Reference

Public Types

- using **ActionFunction** = std::function<void(std::shared_ptr<ecs::Registry>, std::shared_ptr<ResourceManager>, ecs::Entity, ecs::Entity)>

Public Member Functions

- void **registerAction** (const std::string &actionId, ActionFunction action)
- void **executeAction** (const std::string &actionId, std::shared_ptr< ecs::Registry > registry, std::shared_ptr< ResourceManager > resourceManager, ecs::Entity self, ecs::Entity other) const
- bool **hasAction** (const std::string &actionId) const

Static Public Member Functions

- static const ActionFactory & **getInstance** ()

Private Member Functions

- **ActionFactory** (const ActionFactory &)=delete
- ActionFactory & **operator=** (const ActionFactory &)=delete
- void **initializeConditions** ()

Private Attributes

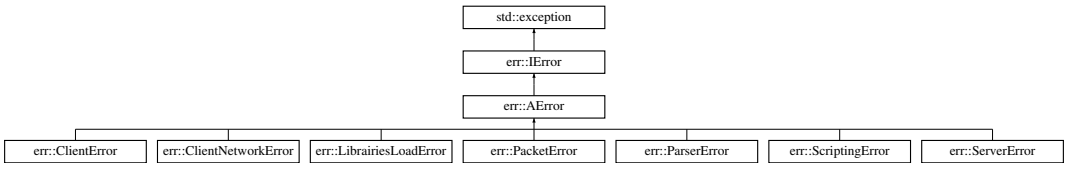
- std::unordered_map< std::string, ActionFunction > **_actions**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/interactions/ActionFactory.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/interactions/ActionFactory.cpp

4.3 err::AError Class Reference

Inheritance diagram for err::AError:



Public Member Functions

- **AError** (const std::string &message, int code=0)
- const char * what () const noexcept override
- int getCode () const noexcept override
- std::string getDetails () const noexcept override
- virtual std::string getType () const noexcept override=0

Protected Attributes

- std::string **m_message**
- int **m_code**

4.3.1 Member Function Documentation

4.3.1.1 getCode()

```
int err::AError::getCode () const  [override], [virtual], [noexcept]
```

Implements err::IError.

4.3.1.2 getDetails()

```
std::string err::AError::getDetails () const  [override], [virtual], [noexcept]
```

Implements err::IError.

4.3.1.3 getType()

```
virtual std::string err::AError::getType () const  [override], [pure virtual], [noexcept]
```

Implements err::IError.

4.3.1.4 what()

```
const char * err::AError::what () const  [override], [virtual], [noexcept]
```

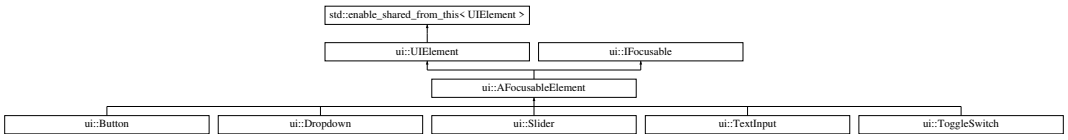
Implements err::IError.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Error/AError.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/Error/AError.cpp

4.4 ui::AFocusableElement Class Reference

Inheritance diagram for ui::AFocusableElement:



Public Member Functions

- **AFocusableElement** (std::shared_ptr< ResourceManager > resourceManager)
- virtual void setFocused (bool focused) override
- virtual bool isFocused () const override
- virtual bool canBeFocused () const override
- virtual void onFocusGained () override
- virtual void onFocusLost () override
- virtual void onActivated () override
- void **setOnFocusGained** (std::function< void()> callback)
- void **setOnFocusLost** (std::function< void()> callback)
- void **setOnActivated** (std::function< void()> callback)
- virtual void handleInput (const math::Vector2f &mousePos, bool mousePressed) override

Public Member Functions inherited from ui::UIElement

- **UIElement** (std::shared_ptr< ResourceManager > resourceManager)
- void **setPosition** (const math::Vector2f &position)
- void **setSize** (const math::Vector2f &size)
- math::Vector2f **getPosition** () const
- math::Vector2f **getSize** () const
- math::Vector2f **getAbsolutePosition** () const
- math::Vector2f **getAbsoluteSize** () const
- void **setVisible** (bool visible)
- bool **isVisible** () const
- void **setScalingEnabled** (bool enabled)
- bool **isScalingEnabled** () const
- void **setFocusEnabled** (bool enabled)
- bool **isFocusEnabled** () const
- void **setState** (UIState state)
- UIState **getState** () const
- virtual void **setScale** (UIScale scale)
- UIScale **getScale** () const
- void **setParent** (std::weak_ptr< UIElement > parent)
- std::shared_ptr< UIElement > **getParent** () const
- void **addChild** (std::shared_ptr< UIElement > child)
- void **removeChild** (std::shared_ptr< UIElement > child)
- const std::vector< std::shared_ptr< UIElement > > & **getChildren** () const
- virtual bool **containsPoint** (const math::Vector2f &point) const
- void **setOnClick** (std::function< void()> callback)
- void **setOnHover** (std::function< void()> callback)
- void **setOnRelease** (std::function< void()> callback)
- virtual void **render** ()
- virtual void **update** (float deltaTime)

Public Member Functions inherited from ui::IFocusable

- virtual bool **onNavigateLeft** ()
- virtual bool **onNavigateRight** ()

Protected Member Functions

- virtual void **onFocusStateChanged** (bool focused)

Protected Member Functions inherited from ui::UIElement

- std::pair< int, int > **getWindowSize** () const
- std::pair< int, int > **getLogicalSize** () const
- float **getScaleFactor** () const

Protected Attributes

- bool **_focused** = false
- bool **_pressedInside** = false
- bool **_wasPressed** = false
- std::function< void()> **_onFocusGained**
- std::function< void()> **_onFocusLost**
- std::function< void()> **_onActivated**

Protected Attributes inherited from ui::UIElement

- std::weak_ptr< ResourceManager > **_resourceManager**
- math::Vector2f **_position**
- math::Vector2f **_size**
- bool **_visible** = true
- bool **_scalingEnabled** = true
- bool **_focusEnabled** = true
- UIState **_state** = UIState::Normal
- UIScale **_scale** = UIScale::Normal
- std::weak_ptr< UIElement > **_parent**
- std::vector< std::shared_ptr< UIElement > > **_children**
- bool **_pressedInside** = false
- bool **_wasPressed** = false
- std::function< void()> **_onClick**
- std::function< void()> **_onHover**
- std::function< void()> **_onRelease**

4.4.1 Member Function Documentation

4.4.1.1 canBeFocused()

```
bool ui::AFocusableElement::canBeFocused () const  [override], [virtual]
```

Implements ui::IFocusable.

4.4.1.2 handleInput()

```
void ui::AFocusableElement::handleInput (
            const math::Vector2f & mousePos,
            bool mousePressed)  [override], [virtual]
```

Reimplemented from ui::UIElement.

4.4.1.3 isFocused()

```
bool ui::AFocusableElement::isFocused () const  [override], [virtual]
```

Implements ui::IFocusable.

4.4.1.4 onActivated()

```
void ui::AFocusableElement::onActivated ()  [override], [virtual]
```

Implements ui::IFocusable.

4.4.1.5 onFocusGained()

```
void ui::AFocusableElement::onFocusGained ()  [override], [virtual]
```

Implements ui::IFocusable.

4.4.1.6 onFocusLost()

```
void ui::AFocusableElement::onFocusLost ()  [override], [virtual]
```

Implements ui::IFocusable.

4.4.1.7 setFocused()

```
void ui::AFocusableElement::setFocused (
            bool focused)  [override], [virtual]
```

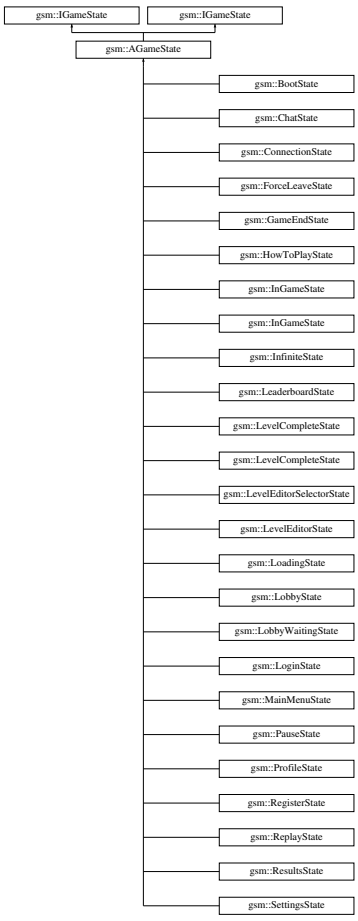
Implements ui::IFocusable.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/core/AFocusableElement.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/core/AFocusableElement.cpp

4.5 gsm::AGameState Class Reference

Inheritance diagram for gsm::AGameState:



Public Member Functions

- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void enter () override
- void update (float deltaTime) override
- void exit () override
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override
- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void enter () override
- void update (float deltaTime) override
- void exit () override
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override
- std::string getStateName () const override=0

Protected Member Functions

- void addSystem (std::shared_ptr< ecs::ISystem > system) override
- void addSystem (std::shared_ptr< ecs::ISystem > system) override

Protected Attributes

- std::weak_ptr< IGameStateMachine > **_gsm**
- std::shared_ptr< ResourceManager > **_resourceManager**
- std::vector< std::shared_ptr< ecs::ISystem > > **_systems**

4.5.1 Member Function Documentation

4.5.1.1 addSystem() [1/2]

```
void gsm::AGameState::addSystem (
            std::shared_ptr< ecs::ISystem > system)  [override], [protected], [virtual]
```

Implements gsm::IGameState.

4.5.1.2 addSystem() [2/2]

```
void gsm::AGameState::addSystem (
            std::shared_ptr< ecs::ISystem > system)  [override], [protected], [virtual]
```

Implements gsm::IGameState.

4.5.1.3 enter() [1/2]

```
void gsm::AGameState::enter ()  [override], [virtual]
```

Implements gsm::IGameState.

4.5.1.4 enter() [2/2]

```
void gsm::AGameState::enter ()  [override], [virtual]
```

Implements gsm::IGameState.

4.5.1.5 exit() [1/2]

```
void gsm::AGameState::exit ()  [override], [virtual]
```

Implements gsm::IGameState.

4.5.1.6 exit() [2/2]

```
void gsm::AGameState::exit ()  [override], [virtual]
```

Implements gsm::IGameState.

4.5.1.7 getStateName()

```
std::string gsm::AGameState::getStateName () const  [override], [pure virtual]
```

Implements gsm::IGameState.

4.5.1.8 getSystems() [1/2]

```
std::vector< std::shared_ptr< ecs::ISystem > > gsm::AGameState::getSystems () const  [override],
[virtual]
```

Implements gsm::IGameState.

4.5.1.9 getSystems() [2/2]

```
std::vector< std::shared_ptr< ecs::ISystem > > gsm::AGameState::getSystems () const  [override],
[virtual]
```

Implements gsm::IGameState.

4.5.1.10 update() [1/2]

```
void gsm::AGameState::update (
            float deltaTime)  [override], [virtual]
```

Implements gsm::IGameState.

4.5.1.11 update() [2/2]

```
void gsm::AGameState::update (
            float deltaTime)  [override], [virtual]
```

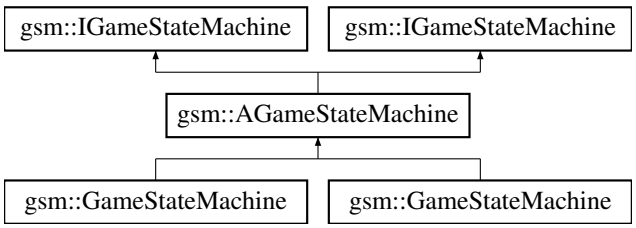
Implements gsm::IGameState.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/base/AGameState.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/gsm/states/AGameState.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/base/AGameState.cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/gsm/states/AGameState.cpp

4.6 gsm::AGameStateMachine Class Reference

Inheritance diagram for gsm::AGameStateMachine:



Public Member Functions

- void changeState (std::shared_ptr< IGameState > newState) override
- void pushState (std::shared_ptr< IGameState > newState) override
- void popState () override
- void requestStateChange (std::shared_ptr< IGameState > newState) override
- void requestStatePush (std::shared_ptr< IGameState > newState) override
- void requestStatePop () override
- void update (float deltaTime) override
- void changeState (std::shared_ptr< IGameState > newState) override
- void pushState (std::shared_ptr< IGameState > newState) override
- void popState () override
- void update (float deltaTime) override
- void requestStateChange (std::shared_ptr< IGameState > newState) override
- std::string **getCurrentStateName** () const

Protected Attributes

- std::stack< std::shared_ptr< IGameState > > **_states**
- std::shared_ptr< IGameState > **_pendingChangeState**
- std::shared_ptr< IGameState > **_pendingPushState**
- bool **_pendingPopState** = false

4.6.1 Member Function Documentation

4.6.1.1 changeState() [1/2]

```
void gsm::AGameStateMachine::changeState (
            std::shared_ptr< IGameState > newState)  [override], [virtual]
```

Implements gsm::IGameStateMachine.

4.6.1.2 changeState() [2/2]

```
void gsm::AGameStateMachine::changeState (
            std::shared_ptr< IGameState > newState)  [override], [virtual]
```

Implements gsm::IGameStateMachine.

4.6.1.3 popState() [1/2]

```
void gsm::AGameStateMachine::popState ()  [override], [virtual]
```

Implements gsm::IGameStateMachine.

4.6.1.4 popState() [2/2]

```
void gsm::AGameStateMachine::popState ()  [override], [virtual]
```

Implements gsm::IGameStateMachine.

4.6.1.5 pushState() [1/2]

```
void gsm::AGameStateMachine::pushState (
            std::shared_ptr< IGameState > newState)  [override], [virtual]
```

Implements gsm::IGameStateMachine.

4.6.1.6 pushState() [2/2]

```
void gsm::AGameStateMachine::pushState (
            std::shared_ptr< IGameState > newState)  [override], [virtual]
```

Implements gsm::IGameStateMachine.

4.6.1.7 requestStateChange() [1/2]

```
void gsm::AGameStateMachine::requestStateChange (
            std::shared_ptr< IGameState > newState)  [override], [virtual]
```

Implements gsm::IGameStateMachine.

4.6.1.8 requestStateChange() [2/2]

```
void gsm::AGameStateMachine::requestStateChange (
            std::shared_ptr< IGameState > newState)  [override], [virtual]
```

Implements gsm::IGameStateMachine.

4.6.1.9 requestStatePop()

```
void gsm::AGameStateMachine::requestStatePop ()  [override], [virtual]
```

Implements gsm::IGameStateMachine.

4.6.1.10 requestStatePush()

```
void gsm::AGameStateMachine::requestStatePush (
            std::shared_ptr< IGameState > newState)  [override], [virtual]
```

Implements gsm::IGameStateMachine.

4.6.1.11 update() [1/2]

```
void gsm::AGameStateMachine::update (
            float deltaTime)  [override], [virtual]
```

Implements gsm::IGameStateMachine.

4.6.1.12 update() [2/2]

```
void gsm::AGameStateMachine::update (
            float deltaTime)  [override], [virtual]
```

Implements gsm::IGameStateMachine.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/machine/AGameStateMachine.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/gsm/machine/AGameStateMachine.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/machine/AGameStateMachine.cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/gsm/machine/AGameStateMachine.cpp

4.7 ecs::AnimationClip Struct Reference

Public Attributes

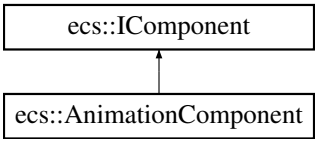
- std::string **texturePath**
- float **frameWidth**
- float **frameHeight**
- int **frameCount**
- float **startWidth**
- float **startHeight**
- float **speed**
- bool **loop**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/components/rendering/AnimationComponent.hpp

4.8 ecs::AnimationComponent Class Reference

Inheritance diagram for ecs::AnimationComponent:



Public Member Functions

- void **addState** (const std::string &name, std::shared_ptr< AnimationClip > clip)
- void **addTransition** (const std::string &from, const std::string &to, const std::vector< AnimationCondition > &conditions, bool playRewind=false)
- void **setCurrentState** (const std::string &state)
- const std::string & **getCurrentState** () const
- float **getTimer** () const
- void **setTimer** (float timer)
- bool **isPlaying** () const
- void **setPlaying** (bool playing)
- bool **isPlayingRewind** () const
- void **setPlayingRewind** (bool rewind)
- int **getRewindStartFrame** () const
- void **setRewindStartFrame** (int frame)
- std::shared_ptr< const AnimationClip > **getCurrentClip** () const
- const std::vector< Transition > & **getTransitions** () const
- int **getCurrentFrame** () const
- void **setCurrentFrame** (int frame)
- const math::FRect & **getFrameRect** () const
- void **setFrameRect** (const math::FRect &rect)
- bool **isValid** () const
- bool **isAnimationFinished** () const
- void **setStateJustChanged** (bool changed)
- bool **getStateJustChanged** () const
- void **setMinAnimationTime** (float time)
- float **getMinAnimationTime** () const
- std::unordered_map< std::string, std::shared_ptr< AnimationClip > > **getStates** () const

Private Attributes

- std::unordered_map< std::string, std::shared_ptr< AnimationClip > > **_states**
- std::vector< Transition > **_transitions**
- std::string **_currentState**
- float **_timer**
- bool **_isPlaying**
- bool **_playRewind**
- int **_currentFrame**
- int **_rewindStartFrame**
- math::FRect **_frameRect**
- bool **_stateJustChanged** = false
- float **_minAnimationTime** = 0.0f

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/components/rendering/AnimationComponent.hpp

4.9 ecs::AnimationCondition Struct Reference

Public Attributes

- std::string **param**
- bool **equals**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/components/rendering/AnimationComponent.hpp

4.10 ecs::AnimationConditionFactory Class Reference

Public Types

- using **ConditionFunction** = std::function<bool(std::shared_ptr<Registry>, Entity)>

Public Member Functions

- void **registerCondition** (const std::string &name, ConditionFunction condition)
- bool **evaluateCondition** (const std::string &name, std::shared_ptr< Registry > registry, Entity entity) const
- bool **hasCondition** (const std::string &name) const
- void **unregisterCondition** (const std::string &name)
- void **clearConditions** ()

Static Public Member Functions

- static const AnimationConditionFactory & **getInstance** ()
- static bool **getConditionValue** (const std::string &param, std::shared_ptr< Registry > registry, Entity entity)

Private Member Functions

- void **initializeConditions** ()
- **AnimationConditionFactory** (const AnimationConditionFactory &)=delete
- AnimationConditionFactory & **operator=** (const AnimationConditionFactory &)=delete

Private Attributes

- std::unordered_map< std::string, ConditionFunction > **_conditions**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/Animation/AnimationConditionFactory.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/Animation/AnimationConditionFactory.cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/Animation/AnimationConditionsRegistry.cpp

4.11 ui::SpritePreview::AnimationData Struct Reference

Public Attributes

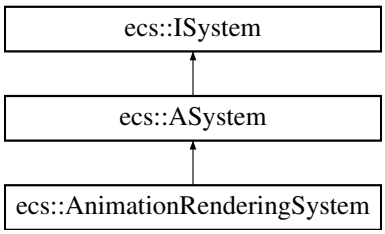
- std::string **texturePath**
- math::Vector2f **frameSize**
- float **frameCount** = 0.0f
- float **startWidth** = 0.0f
- float **startHeight** = 0.0f
- float **speed** = 0.1f
- bool **loop** = true

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/elements/SpritePreview.hpp

4.12 ecs::AnimationRenderingSystem Class Reference

Inheritance diagram for ecs::AnimationRenderingSystem:



Protected Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Private Attributes

- std::unordered_map< Entity, float > **_waitTimers**

Additional Inherited Members

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

4.12.1 Member Function Documentation

4.12.1.1 update()

```
void ecs::AnimationRenderingSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [protected], [virtual]
```

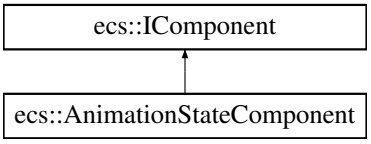
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/rendering/AnimationRenderingSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/rendering/AnimationRenderingSystem.cpp

4.13 ecs::AnimationStateComponent Class Reference

Inheritance diagram for ecs::AnimationStateComponent:



Public Member Functions

- **AnimationStateComponent** (const std::string &initialState="")
- void **setCurrentState** (const std::string &state)
- std::string **getCurrentState** () const

Private Attributes

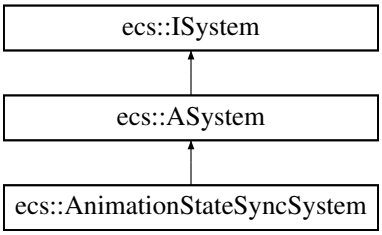
- std::string **_currentState**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/permanent/AnimationStateComponent.↩ hpp

4.14 ecs::AnimationStateSyncSystem Class Reference

Inheritance diagram for ecs::AnimationStateSyncSystem:



Protected Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Additional Inherited Members

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

4.14.1 Member Function Documentation

4.14.1.1 update()

```
void ecs::AnimationStateSyncSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [protected], [virtual]
```

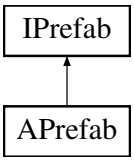
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/animationState/AnimationStateSyncSystem.↩ hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/animationState/AnimationStateSyncSystem.↩ cpp

4.15 APrefab Class Reference

Inheritance diagram for APrefab:



Public Member Functions

- ecs::Entity instantiate (const std::shared_ptr< ecs::Registry > &registry, const std::shared_ptr< ecs::IEntityFactory > &factory, const ecs::EntityCreationContext &context=ecs::EntityCreationContext↩ ::forLocalClient()) override
- ecs::Entity instantiate (const std::shared_ptr< ecs::Registry > &registry) override

4.15.1 Member Function Documentation

4.15.1.1 instantiate() [1/2]

```
ecs::Entity APrefab::instantiate (
            const std::shared_ptr< ecs::Registry > & registry)  [override], [virtual]
```

Implements IPrefab.

4.15.1.2 instantiate() [2/2]

```
ecs::Entity APrefab::instantiate (
            const std::shared_ptr< ecs::Registry > & registry,
            const std::shared_ptr< ecs::IEntityFactory > & factory,
            const ecs::EntityCreationContext & context = ecs::EntityCreationContext::forLocalClient())
[override], [virtual]
```

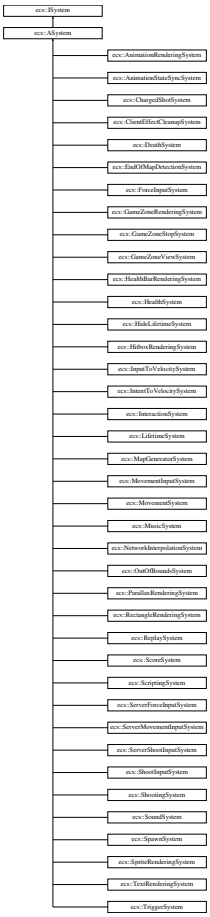
Implements IPrefab.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Prefab/APrefab.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/Prefab/APrefab.cpp

4.16 ecs::ASystem Class Reference

Inheritance diagram for ecs::ASystem:



Public Member Functions

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Protected Member Functions

- virtual void **update** (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime)=0

4.16.1 Member Function Documentation

4.16.1.1 updateSystem()

```
void ecs::ASystem::updateSystem (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

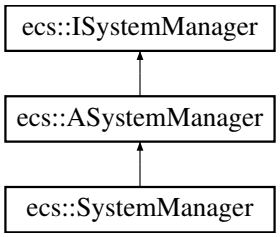
Implements ecs::ISystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/base/ASystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/base/ASystem.cpp

4.17 ecs::ASystemManager Class Reference

Inheritance diagram for ecs::ASystemManager:



Public Member Functions

- void updateAllSystems (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override
- void addSystem (std::shared_ptr< ISystem > system) override
- void removeSystem (std::shared_ptr< ISystem > system) override
- void clearAllSystems () override

Private Attributes

- std::vector< std::shared_ptr< ISystem > > **_systems**

4.17.1 Member Function Documentation

4.17.1.1 addSystem()

```
void ecs::ASystemManager::addSystem (
            std::shared_ptr< ISystem > system)  [override], [virtual]
```

Implements ecs::ISystemManager.

4.17.1.2 clearAllSystems()

```
void ecs::ASystemManager::clearAllSystems ()  [override], [virtual]
```

Implements ecs::ISystemManager.

4.17.1.3 removeSystem()

```
void ecs::ASystemManager::removeSystem (
            std::shared_ptr< ISystem > system)  [override], [virtual]
```

Implements ecs::ISystemManager.

4.17.1.4 updateAllSystems()

```
void ecs::ASystemManager::updateAllSystems (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

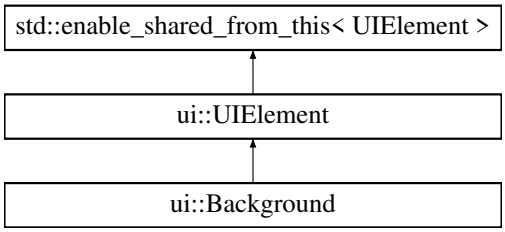
Implements ecs::ISystemManager.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/systemManager/ASystemManager.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/systemManager/ASystemManager.cpp

4.18 ui::Background Class Reference

Inheritance diagram for ui::Background:



Classes

- struct Layer

Public Member Functions

- **Background** (std::shared_ptr< ResourceManager > resourceManager)
- void render () override
- void update (float deltaTime) override
- void **addLayer** (const std::string &texturePath, float speedX, float speedY=0.0f, const math::Vector2f &sourceSize=math::Vector2f(constants::MAX_WIDTH, constants::MAX_HEIGHT))

Public Member Functions inherited from ui::UIElement

- **UIElement** (std::shared_ptr< ResourceManager > resourceManager)
- void **setPosition** (const math::Vector2f &position)
- void **setSize** (const math::Vector2f &size)
- math::Vector2f **getPosition** () const
- math::Vector2f **getSize** () const
- math::Vector2f **getAbsolutePosition** () const
- math::Vector2f **getAbsoluteSize** () const
- void **setVisible** (bool visible)
- bool **isVisible** () const
- void **setScalingEnabled** (bool enabled)
- bool **isScalingEnabled** () const
- void **setFocusEnabled** (bool enabled)
- bool **isFocusEnabled** () const
- void **setState** (UIState state)
- UIState **getState** () const
- virtual void **setScale** (UIScale scale)
- UIScale **getScale** () const
- void **setParent** (std::weak_ptr< UIElement > parent)
- std::shared_ptr< UIElement > **getParent** () const
- void **addChild** (std::shared_ptr< UIElement > child)
- void **removeChild** (std::shared_ptr< UIElement > child)
- const std::vector< std::shared_ptr< UIElement > > & **getChildren** () const
- virtual void **handleInput** (const math::Vector2f &mousePos, bool mousePressed)
- virtual bool **containsPoint** (const math::Vector2f &point) const
- void **setOnClick** (std::function< void()> callback)
- void **setOnHover** (std::function< void()> callback)
- void **setOnRelease** (std::function< void()> callback)

Private Member Functions

- float **calculateScale** (const Layer &layer, float screenWidth)

Private Attributes

- std::vector< Layer > **_layers**

Additional Inherited Members

Protected Member Functions inherited from ui::UIElement

- std::pair< int, int > **getWindowSize** () const
- std::pair< int, int > **getLogicalSize** () const
- float **getScaleFactor** () const

Protected Attributes inherited from ui::UIElement

- std::weak_ptr< ResourceManager > **_resourceManager**
- math::Vector2f **_position**
- math::Vector2f **_size**
- bool **_visible** = true
- bool **_scalingEnabled** = true
- bool **_focusEnabled** = true
- UIState **_state** = UIState::Normal
- UIScale **_scale** = UIScale::Normal
- std::weak_ptr< UIElement > **_parent**
- std::vector< std::shared_ptr< UIElement > > **_children**
- bool **_pressedInside** = false
- bool **_wasPressed** = false
- std::function< void()> **_onClick**
- std::function< void()> **_onHover**
- std::function< void()> **_onRelease**

4.18.1 Member Function Documentation

4.18.1.1 render()

```
void ui::Background::render ()  [override], [virtual]
```

Reimplemented from ui::UIElement.

4.18.1.2 update()

```
void ui::Background::update (
            float deltaTime)  [override], [virtual]
```

Reimplemented from ui::UIElement.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/elements/Background.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/elements/Background.cpp

4.19 ui::LayoutConfig::BackgroundConfig Struct Reference

Public Attributes

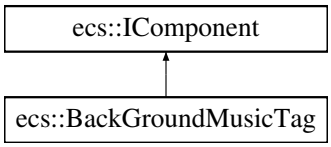
- bool **enabled** = false
- color_t **fillColor** = {0, 0, 0, 0}
- color_t **outlineColor** = {0, 0, 0, 0}
- float **cornerRadius** = 0.0f

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/core/UILayout.hpp

4.20 ecs::BackGroundMusicTag Class Reference

Inheritance diagram for ecs::BackGroundMusicTag:

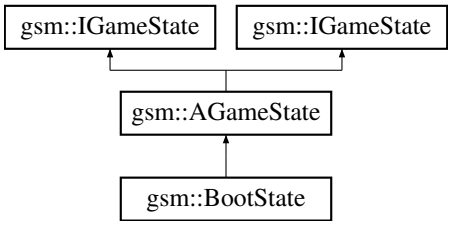


The documentation for this class was generated from the following file:

- /home/albane/epitech/tech3/r-type/ryanR-type/client/components/tags/BackGroundMusicTag.hpp

4.21 gsm::BootState Class Reference

Inheritance diagram for gsm::BootState:



Public Member Functions

- **BootState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void enter () override
- std::string getStateName () const override

Public Member Functions inherited from gsm::AGameState

- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void update (float deltaTime) override
- void exit () override
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override
- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void update (float deltaTime) override
- void exit () override
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override

Additional Inherited Members

Protected Member Functions inherited from gsm::AGameState

- void addSystem (std::shared_ptr< ecs::ISystem > system) override
- void addSystem (std::shared_ptr< ecs::ISystem > system) override

Protected Attributes inherited from gsm::AGameState

- std::weak_ptr< IGameStateMachine > **_gsm**
- std::shared_ptr< ResourceManager > **_resourceManager**
- std::vector< std::shared_ptr< ecs::ISystem > > **_systems**

4.21.1 Member Function Documentation

4.21.1.1 enter()

```
void gsm::BootState::enter ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.21.1.2 getStateName()

```
std::string gsm::BootState::getStateName () const  [inline], [override], [virtual]
```

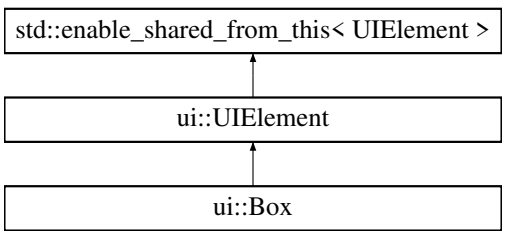
Implements gsm::AGameState.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/server/gsm/states/scenes/Boot/BootState.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/gsm/states/scenes/Boot/BootState.cpp

4.22 ui::Box Class Reference

Inheritance diagram for ui::Box:



Public Member Functions

- **Box** (std::shared_ptr< ResourceManager > resourceManager)
- void **setBackgroundColor** (const gfx::color_t &color)
- void **setBorderColor** (const gfx::color_t &color)
- void **setBorderThickness** (float thickness)
- void **setCornerRadius** (float radius)
- void render () override

Public Member Functions inherited from ui::UIElement

- **UIElement** (std::shared_ptr< ResourceManager > resourceManager)
- void **setPosition** (const math::Vector2f &position)
- void **setSize** (const math::Vector2f &size)
- math::Vector2f **getPosition** () const
- math::Vector2f **getSize** () const
- math::Vector2f **getAbsolutePosition** () const
- math::Vector2f **getAbsoluteSize** () const
- void **setVisible** (bool visible)
- bool **isVisible** () const
- void **setScalingEnabled** (bool enabled)
- bool **isScalingEnabled** () const
- void **setFocusEnabled** (bool enabled)
- bool **isFocusEnabled** () const
- void **setState** (UIState state)
- UIState **getState** () const
- virtual void **setScale** (UIScale scale)
- UIScale **getScale** () const
- void **setParent** (std::weak_ptr< UIElement > parent)
- std::shared_ptr< UIElement > **getParent** () const
- void **addChild** (std::shared_ptr< UIElement > child)
- void **removeChild** (std::shared_ptr< UIElement > child)
- const std::vector< std::shared_ptr< UIElement > > & **getChildren** () const
- virtual void **handleInput** (const math::Vector2f &mousePos, bool mousePressed)
- virtual bool **containsPoint** (const math::Vector2f &point) const
- void **setOnClick** (std::function< void()> callback)
- void **setOnHover** (std::function< void()> callback)
- void **setOnRelease** (std::function< void()> callback)
- virtual void **update** (float deltaTime)

Private Attributes

- gfx::color_t **_backgroundColor** = gfx::color_t{40, 40, 60, 200}
- gfx::color_t **_borderColor** = gfx::color_t{80, 80, 120, 255}
- float **_borderThickness** = 2.0f
- float **_cornerRadius** = 8.0f

Additional Inherited Members

Protected Member Functions inherited from ui::UIElement

- std::pair< int, int > **getWindowSize** () const
- std::pair< int, int > **getLogicalSize** () const
- float **getScaleFactor** () const

Protected Attributes inherited from ui::UIElement

- std::weak_ptr< ResourceManager > **_resourceManager**
- math::Vector2f **_position**
- math::Vector2f **_size**
- bool **_visible** = true
- bool **_scalingEnabled** = true
- bool **_focusEnabled** = true
- UIState **_state** = UIState::Normal
- UIScale **_scale** = UIScale::Normal
- std::weak_ptr< UIElement > **_parent**
- std::vector< std::shared_ptr< UIElement > > **_children**
- bool **_pressedInside** = false
- bool **_wasPressed** = false
- std::function< void()> **_onClick**
- std::function< void()> **_onHover**
- std::function< void()> **_onRelease**

4.22.1 Member Function Documentation

4.22.1.1 render()

```
void ui::Box::render () [override], [virtual]
```

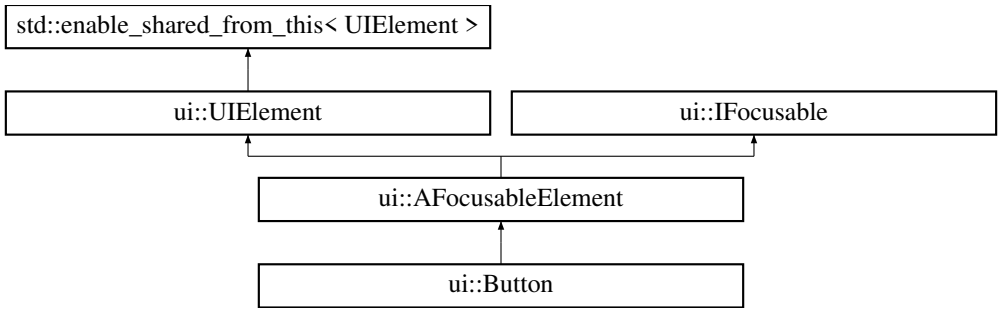
Reimplemented from ui::UIElement.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/elements/Box.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/elements/Box.cpp

4.23 ui::Button Class Reference

Inheritance diagram for ui::Button:



Public Member Functions

- **Button** (std::shared_ptr< ResourceManager > resourceManager)
- void **setText** (const std::string &text)
- const std::string & **getText** () const
- void **setTextColor** (const gfx::color_t &color)
- void **setFontPath** (const std::string &fontPath)
- void **setNormalColor** (const gfx::color_t &color)
- void **setHoveredColor** (const gfx::color_t &color)
- void **setPressedColor** (const gfx::color_t &color)
- void **setDisabledColor** (const gfx::color_t &color)
- void **setFocusedColor** (const gfx::color_t &color)
- void **setBaseFontSize** (size_t fontSize)
- size_t **getBaseFontSize** () const
- void **setIconPath** (const std::string &iconPath)
- void **setIconSize** (const math::Vector2f &size)
- virtual void render () override

Public Member Functions inherited from ui::AFocusableElement

- **AFocusableElement** (std::shared_ptr< ResourceManager > resourceManager)
- virtual void setFocused (bool focused) override
- virtual bool isFocused () const override
- virtual bool canBeFocused () const override
- virtual void onFocusGained () override
- virtual void onFocusLost () override
- virtual void onActivated () override
- void **setOnFocusGained** (std::function< void()> callback)
- void **setOnFocusLost** (std::function< void()> callback)
- void **setOnActivated** (std::function< void()> callback)
- virtual void handleInput (const math::Vector2f &mousePos, bool mousePressed) override

Public Member Functions inherited from ui::UIElement

- **UIElement** (std::shared_ptr< ResourceManager > resourceManager)
- void **setPosition** (const math::Vector2f &position)
- void **setSize** (const math::Vector2f &size)
- math::Vector2f **getPosition** () const
- math::Vector2f **getSize** () const
- math::Vector2f **getAbsolutePosition** () const
- math::Vector2f **getAbsoluteSize** () const
- void **setVisible** (bool visible)
- bool **isVisible** () const
- void **setScalingEnabled** (bool enabled)
- bool **isScalingEnabled** () const
- void **setFocusEnabled** (bool enabled)
- bool **isFocusEnabled** () const
- void **setState** (UIState state)
- UIState **getState** () const
- virtual void **setScale** (UIScale scale)
- UIScale **getScale** () const
- void **setParent** (std::weak_ptr< UIElement > parent)
- std::shared_ptr< UIElement > **getParent** () const

- void **addChild** (std::shared_ptr< UIElement > child)
- void **removeChild** (std::shared_ptr< UIElement > child)
- const std::vector< std::shared_ptr< UIElement > > & **getChildren** () const
- virtual bool **containsPoint** (const math::Vector2f &point) const
- void **setOnClick** (std::function< void()> callback)
- void **setOnHover** (std::function< void()> callback)
- void **setOnRelease** (std::function< void()> callback)
- virtual void **update** (float deltaTime)

Public Member Functions inherited from ui::IFocusable

- virtual bool **onNavigateLeft** ()
- virtual bool **onNavigateRight** ()

Private Member Functions

- gfx::color_t **getCurrentColor** () const
- size_t **getFontSize** () const

Private Attributes

- std::string **_text**
- gfx::color_t **_textColor** = colors::UI_TEXT
- std::string **_fontPath** = constants::MAIN_FONT
- gfx::color_t **_normalColor** = colors::BUTTON_PRIMARY
- gfx::color_t **_hoveredColor** = colors::BUTTON_PRIMARY_HOVER
- gfx::color_t **_pressedColor** = colors::BUTTON_PRIMARY_PRESSED
- gfx::color_t **_disabledColor** = colors::UI_DISABLED
- gfx::color_t **_focusedColor** = colors::UI_FOCUSED
- size_t **_baseFontSize** = constants::BUTTON_FONT_SIZE_BASE
- std::string **_iconPath**
- math::Vector2f **_iconSize** = math::Vector2f(0.f, 0.f)

Additional Inherited Members

Protected Member Functions inherited from ui::AFocusableElement

- virtual void **onFocusStateChanged** (bool focused)

Protected Member Functions inherited from ui::UIElement

- std::pair< int, int > **getWindowSize** () const
- std::pair< int, int > **getLogicalSize** () const
- float **getScaleFactor** () const

Protected Attributes inherited from ui::AFocusableElement

- bool **_focused** = false
- bool **_pressedInside** = false
- bool **_wasPressed** = false
- std::function< void()> **_onFocusGained**
- std::function< void()> **_onFocusLost**
- std::function< void()> **_onActivated**

Protected Attributes inherited from ui::UIElement

- std::weak_ptr< ResourceManager > **_resourceManager**
- math::Vector2f **_position**
- math::Vector2f **_size**
- bool **_visible** = true
- bool **_scalingEnabled** = true
- bool **_focusEnabled** = true
- UIState **_state** = UIState::Normal
- UIScale **_scale** = UIScale::Normal
- std::weak_ptr< UIElement > **_parent**
- std::vector< std::shared_ptr< UIElement > > **_children**
- bool **_pressedInside** = false
- bool **_wasPressed** = false
- std::function< void()> **_onClick**
- std::function< void()> **_onHover**
- std::function< void()> **_onRelease**

4.23.1 Member Function Documentation

4.23.1.1 render()

```
void ui::Button::render () [override], [virtual]
```

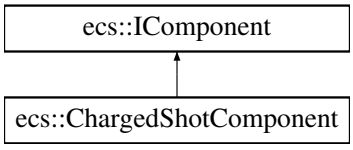
Reimplemented from ui::UIElement.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/elements/focusable/Button.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/elements/focusable/Button.cpp

4.24 ecs::ChargedShotComponent Class Reference

Inheritance diagram for ecs::ChargedShotComponent:



Public Member Functions

- **ChargedShotComponent** (const float &charge=0, const float &maxCharge=0, const float &chargeReload↩ Time=0)
- float **getCharge** () const
- void **setCharge** (const float &charge)
- float **getMaxCharge** () const
- void **setMaxCharge** (const float &maxCharge)
- float **getReloadTime** () const
- void **setReloadTime** (const float &reloadTime)

Private Attributes

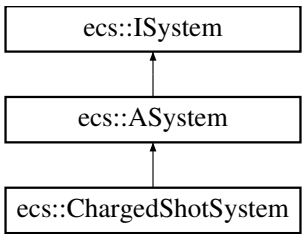
- float **_charge**
- float **_maxCharge**
- float **_chargeReloadTime**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/permanent/ChargedShotComponent.↩ hpp

4.25 ecs::ChargedShotSystem Class Reference

Inheritance diagram for ecs::ChargedShotSystem:



Public Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

4.25.1 Member Function Documentation

4.25.1.1 update()

```
void ecs::ChargedShotSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

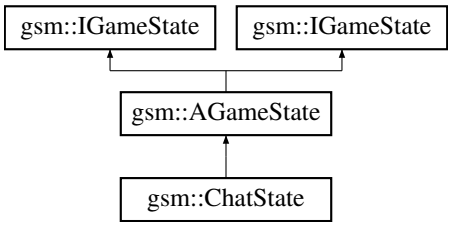
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/shooting/ChargedShotSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/shooting/ChargedShotSystem.cpp

4.26 gsm::ChatState Class Reference

Inheritance diagram for gsm::ChatState:



Public Member Functions

- **ChatState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void enter () override
- void update (float deltaTime) override
- void exit () override
- std::string getStateName () const override

Public Member Functions inherited from gsm::AGameState

- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override
- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override

Private Member Functions

- void **renderUI** ()
- void **onBackButtonClicked** ()
- void **onSendMessage** (const std::string &text)

Private Attributes

- std::unique_ptr< MouseInputHandler > **_mouseHandler**
- std::unique_ptr< ui::UIManager > **_uiManager**
- std::shared_ptr< ui::Background > **_background**
- std::shared_ptr< ui::UILayout > **_mainLayout**
- std::shared_ptr< ui::Button > **_backButton**
- std::shared_ptr< ui::Text > **_titleText**
- std::shared_ptr< ui::TextInput > **_messageInput**
- std::shared_ptr< ui::Button > **_sendButton**
- std::shared_ptr< ui::UILayout > **_messagesContainer**

Additional Inherited Members

Protected Member Functions inherited from gsm::AGameState

- void addSystem (std::shared_ptr< ecs::ISystem > system) override
- void addSystem (std::shared_ptr< ecs::ISystem > system) override

Protected Attributes inherited from gsm::AGameState

- std::weak_ptr< IGameStateMachine > **_gsm**
- std::shared_ptr< ResourceManager > **_resourceManager**
- std::vector< std::shared_ptr< ecs::ISystem > > **_systems**

4.26.1 Member Function Documentation

4.26.1.1 enter()

```
void gsm::ChatState::enter ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.26.1.2 exit()

```
void gsm::ChatState::exit ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.26.1.3 getStateName()

```
std::string gsm::ChatState::getStateName () const  [inline], [override], [virtual]
```

Implements gsm::AGameState.

4.26.1.4 update()

```
void gsm::ChatState::update (
            float deltaTime)  [override], [virtual]
```

Reimplemented from gsm::AGameState.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/Chat/ChatState.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/Chat/ChatState.cpp

4.27 math::Chrono Class Reference

Public Member Functions

- void **start** ()
- void **stop** ()
- void **reset** ()
- float **getElapsedSeconds** () const
- float **getElapsedMilliseconds** () const
- bool **isRunning** () const

Private Attributes

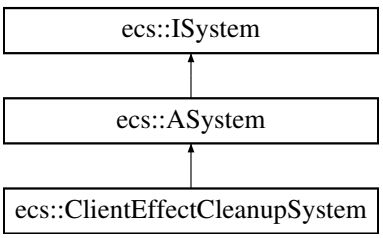
- std::chrono::high_resolution_clock::time_point **_startTime**
- std::chrono::high_resolution_clock::time_point **_stopTime**
- bool **_isRunning**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/types/Chrono.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/types/Chrono.cpp

4.28 ecs::ClientEffectCleanupSystem Class Reference

Inheritance diagram for ecs::ClientEffectCleanupSystem:



Public Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

4.28.1 Member Function Documentation

4.28.1.1 update()

```
void ecs::ClientEffectCleanupSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

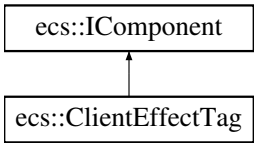
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/effects/ClientEffectCleanupSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/effects/ClientEffectCleanupSystem.cpp

4.29 ecs::ClientEffectTag Class Reference

Inheritance diagram for ecs::ClientEffectTag:

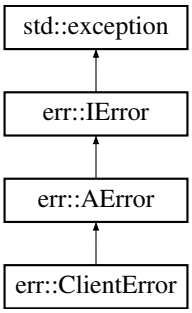


The documentation for this class was generated from the following file:

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/tags/ClientEffectTag.hpp

4.30 err::ClientError Class Reference

Inheritance diagram for err::ClientError:



Public Types

- enum **ErrorCode** {
  UNKNOWN = 2000 , **CONNECTION_FAILED** = 2001 , **DISCONNECTED** = 2002 , **TIMEOUT** = 2003 ,
  NOT_INITIALIZED = 2004 , **CAN_NOT_OPEN_FILE** = 2005 }

Public Member Functions

- **ClientError** (const std::string &message, ErrorCode code=UNKNOWN)
- std::string getType () const noexcept override

Public Member Functions inherited from err::AError

- **AError** (const std::string &message, int code=0)
- const char * what () const noexcept override
- int getCode () const noexcept override
- std::string getDetails () const noexcept override

Additional Inherited Members

Protected Attributes inherited from err::AError

- std::string **m_message**
- int **m_code**

4.30.1 Member Function Documentation

4.30.1.1 getType()

```
std::string err::ClientError::getType () const  [override], [virtual], [noexcept]
```

Implements err::AError.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Error/ClientError.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/Error/ClientError.cpp

4.31 ClientNetwork Class Reference

Public Member Functions

- void **init** ()
- void **start** ()
- void **stop** ()
- void **connect** ()
- uint16_t **getPort** () const
- void **setPort** (int port)
- std::string **getIp** () const
- void **setIp** (const std::string &ip)
- std::shared_ptr< net::INetwork > **getNetwork** () const
- void **setDebugMode** (bool isDebug)
- bool **isDebugMode** () const
- void **loadNetworkLibrary** ()
- void **loadBufferLibrary** ()
- void **loadPacketLibrary** ()
- void **sendConnectionData** (std::vector< uint8_t > packet)
- std::string **getName** () const
- void **setName** (const std::string &name)
- uint8_t **getIdClient** () const
- void **setIdClient** (uint8_t idClient)
- std::string **getLobbyCode** () const
- void **setLobbyCode** (std::string lobbyCode)
- net::ConnectionState **getConnectionState** () const
- void **eventPacket** (const constants::EventType &eventType, double depth)
- void **disconnectionPacket** ()
- void **connectionPacket** ()
- void **sendWhoAmI** ()
- void **requestCode** ()
- void **sendLobbyConnection** (std::string lobbyCode)
- void **sendMasterStartGame** ()
- void **sendRegisterPacket** (const std::string &username, const std::string &password)
- void **sendLoginPacket** (const std::string &username, const std::string &password)
- void **sendRequestLeaderboardPacket** ()
- void **sendRequestProfilePacket** ()
- void **sendMessageToServer** (const std::string &message)
- void **sendRequestGameRulesUpdate** (uint8_t ruleType, uint8_t value)
- void **sendDisconnectFromLobby** ()
- const std::vector< std::pair< std::string, std::string > > & **getLeaderboardData** () const
- bool **isLeaderboardDataUpdated** () const
- void **clearLeaderboardDataUpdateFlag** ()
- const std::vector< std::string > & **getProfileData** () const
- bool **isProfileDataUpdated** () const
- void **clearProfileDataUpdateFlag** ()
- const std::vector< std::pair< std::string, std::string > > & **getLastMessages** () const
- void **addToEventQueue** (const NetworkEvent &event)
- void **clearEntitiesAndMappings** ()
- bool **isConnected** () const
- bool **isReady** () const
- size_t **getConnectedClients** () const
- size_t **getReadyClients** () const
- uint8_t **getClientId** () const

- bool **getClientReadyStatus** () const
- bool **isConnectedToLobby** () const
- bool **isLobbyMaster** () const
- void **setResourceManager** (std::shared_ptr< ResourceManager > resourceManager)
- void **setGameStateMachine** (std::shared_ptr< gsm::IGameStateMachine > gsm)
- std::shared_ptr< gsm::IGameStateMachine > **getGameStateMachine** () const
- void **redoServerEndpoint** ()

Public Attributes

- std::atomic< bool > **_isConnected**
- std::atomic< bool > **_ready**
- std::atomic< bool > **_isConnectedToLobby**
- std::atomic< bool > **_isLobbyMaster**
- std::atomic< size_t > **_connectedClients**
- std::atomic< size_t > **_readyClients**
- std::atomic< uint8_t > **_clientId**
- std::atomic< bool > **_clientReadyStatus**
- std::atomic< bool > **_shouldDisconnect**
- std::chrono::steady_clock::time_point **_lastLeaveLobbyTime**

Protected Member Functions

- std::pair< int, std::chrono::steady_clock::time_point > **tryConnection** (const int maxRetries, std::chrono::steady_clock::time_point lastRetryTime)
- void **handlePacketType** (uint8_t type)

Private Types

- typedef void(ClientNetwork::* **PacketHandler**) ()
- typedef size_t(ClientNetwork::* **ComponentParser**) (const std::vector< uint64_t > &, size_t, ecs::Entity)

Private Member Functions

- void **handleNoOp** ()
- void **handleConnectionAcceptation** ()
- void **handleBatchedGameState** ()
- void **handleEndGame** ()
- void **handleCanStart** ()
- void **handleEntitySpawn** ()
- void **handleEntityDeath** ()
- void **handleWhoAmI** ()
- void **handleServerStatus** ()
- void **handleCode** ()
- void **handleLevelComplete** ()
- void **handleNextLevel** ()
- void **handleLobbyConnectValue** ()
- void **handleConnectUser** ()
- void **handleLeaderboard** ()
- void **handleProfile** ()
- void **handleRegisterFail** ()

- void **handleBroadcastedChat** ()
- void **handleGameRules** ()
- void **handleForceLeave** ()
- void **handleAckLeaveLobby** ()
- size_t **parseTransformComponent** (const std::vector< uint64_t > &payload, size_t index, ecs::Entity entityId)
- size_t **parseHealthComponent** (const std::vector< uint64_t > &payload, size_t index, ecs::Entity entityId)
- size_t **parseScoreComponent** (const std::vector< uint64_t > &payload, size_t index, ecs::Entity entityId)
- size_t **parseChargedShotComponent** (const std::vector< uint64_t > &payload, size_t index, ecs::Entity entityId)
- size_t **parseAnimationStateComponent** (const std::vector< uint64_t > &payload, size_t index, ecs::Entity entityId)

Private Attributes

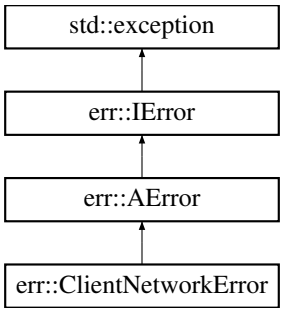
- PacketHandler **_packetHandlers** [constants::MAX_INDEX_PACKET_TYPE]
- std::map< uint64_t, ComponentParser > **_componentParsers**
- DLLoader< createNetworkLib_t > **_networloader**
- DLLoader< createBuffer_t > **_bufferloader**
- DLLoader< createPacket_t > **_packetloader**
- std::shared_ptr< net::INetwork > **_network**
- std::shared_ptr< IBuffer > **_receptionBuffer**
- std::shared_ptr< IBuffer > **_sendBuffer**
- std::shared_ptr< pm::IPacketManager > **_packet**
- std::shared_ptr< ResourceManager > **_resourceManager**
- std::shared_ptr< gsm::IGameStateMachine > **_gsm**
- uint32_t **_sequenceNumber**
- uint16_t **_port**
- std::string **_ip**
- std::string **_name**
- std::vector< std::string > **_clientNames**
- std::vector< std::pair< std::string, std::string > > **_lastMessages**
- bool **_isDebug**
- bool **_expectingLoginResponse** = false
- bool **_expectingProfileResponse** = false
- bool **_expectingRegisterResponse** = false
- uint8_t **_idClient**
- std::shared_ptr< net::INetworkEndpoint > **_serverEndpoint**
- std::queue< NetworkEvent > **_eventQueue**
- std::mutex **_queueMutex**
- std::condition_variable **_queueCond**
- std::unordered_map< size_t, ecs::Entity > **_serverToLocalEntityMap**
- std::unordered_map< ecs::Entity, std::string > **_lastReceivedAnimationState**
- std::string **_lobbyCode**
- int **_retryCount**
- bool **_shouldConnect**
- std::vector< std::pair< std::string, std::string > > **_leaderboardData**
- bool **_leaderboardDataUpdated** = false
- std::vector< std::string > **_profileData**
- bool **_profileDataUpdated** = false
- std::chrono::steady_clock::time_point **_connectionAttemptTime**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/ClientNetwork.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/ClientGameStateConversions.cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/ClientLibLoading.cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/ClientNetwork.cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/ClientReceivedPacket.cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/ClientSentPacket.cpp

4.32 err::ClientNetworkError Class Reference

Inheritance diagram for err::ClientNetworkError:



Public Types

- enum **ErrorCode** {
  UNKNOWN = 1000 , **CONNECTION_FAILED** = 1001 , **TIMEOUT** = 1002 , **INVALID_REQUEST** = 1003 ,
  INTERNAL_ERROR = 1004 , **LIBRARY_LOAD_FAILED** = 1005 , **CONFIG_ERROR** = 1006 }

Public Member Functions

- **ClientNetworkError** (const std::string &message, ErrorCode code=UNKNOWN)
- std::string getType () const noexcept override

Public Member Functions inherited from err::AError

- **AError** (const std::string &message, int code=0)
- const char * what () const noexcept override
- int getCode () const noexcept override
- std::string getDetails () const noexcept override

Additional Inherited Members

Protected Attributes inherited from err::AError

- std::string **m_message**
- int **m_code**

4.32.1 Member Function Documentation

4.32.1.1 getType()

```
std::string err::ClientNetworkError::getType () const  [override], [virtual], [noexcept]
```

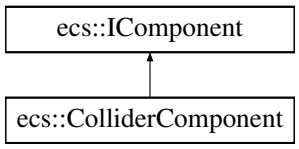
Implements err::AError.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Error/ClientNetworkError.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/Error/ClientNetworkError.cpp

4.33 ecs::ColliderComponent Class Reference

Inheritance diagram for ecs::ColliderComponent:



Public Member Functions

- **ColliderComponent** (math::Vector2f offset=math::Vector2f(0.0f, 0.0f), math::Vector2f size=math::Vector2f(0.↩ 0f, 0.0f), CollisionType type=CollisionType::Solid)
- math::Vector2f **getOffset** () const
- void **setOffset** (math::Vector2f offset)
- math::Vector2f **getSize** () const
- void **setSize** (math::Vector2f size)
- CollisionType **getType** () const
- void **setType** (CollisionType type)
- math::FRect **getHitbox** (math::Vector2f entityPosition, math::Vector2f scale=math::Vector2f(1.0f, 1.0f)) const
- math::FRect **getScaledHitbox** (math::Vector2f entityPosition, math::Vector2f scale) const
- math::OrientedRect **getOrientedHitbox** (math::Vector2f entityPosition, math::Vector2f scale, float rotation) const
- math::FRect **getHitbox** (math::Vector2f entityPosition, math::Vector2f scale, float rotation) const

Private Attributes

- math::Vector2f **_offset**
- math::Vector2f **_size**
- CollisionType **_type**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/permanent/ColliderComponent.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/permanent/ColliderComponent.cpp

4.34 ecs::CollisionRule Struct Reference

Public Attributes

- std::vector< std::string > **groupA**
- std::vector< std::string > **groupB**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/CollisionRules/CollisionRulesData.hpp

4.35 ecs::CollisionRules Class Reference

Public Member Functions

- bool **canCollide** (CollisionType type, const std::vector< std::string > &tagsA, const std::vector< std::string > &tagsB) const

Static Public Member Functions

- static const CollisionRules & **getInstance** ()
- static void **initWithData** (const CollisionRulesData &data)

Private Member Functions

- **CollisionRules** (const CollisionRules &)=delete
- CollisionRules & **operator=** (const CollisionRules &)=delete
- const std::vector< CollisionRule > & **getAllowRules** (CollisionType type) const
- bool **entityMatchesGroup** (const std::vector< std::string > &entityTags, const std::vector< std::string > &group) const
- bool **ruleMatches** (const CollisionRule &rule, const std::vector< std::string > &tagsA, const std::vector< std::string > &tagsB) const

Private Attributes

- std::shared_ptr< std::vector< CollisionRule > > **_solidAllowRules**
- std::shared_ptr< std::vector< CollisionRule > > **_triggerAllowRules**
- std::shared_ptr< std::vector< CollisionRule > > **_pushAllowRules**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/CollisionRules/CollisionRules.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/CollisionRules/CollisionRules.cpp

4.36 ecs::CollisionRulesData Struct Reference

Public Attributes

- std::shared_ptr< std::vector< CollisionRule > > **solidAllowRules**
- std::shared_ptr< std::vector< CollisionRule > > **triggerAllowRules**
- std::shared_ptr< std::vector< CollisionRule > > **pushAllowRules**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/CollisionRules/CollisionRulesData.hpp

4.37 ecs::CollisionRulesParser Class Reference

Static Public Member Functions

- static CollisionRulesData **parseFromFile** (const std::string &filePath)
- static CollisionRulesData **parseFromJsonString** (const std::string &jsonString)
- static CollisionRulesData **parseFromJson** (const nlohmann::json &json)

Static Private Member Functions

- static void **parseRulesForType** (const nlohmann::json &typeJson, std::shared_ptr< std::vector< CollisionRule > > allowRules)

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/CollisionRulesParser.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/CollisionRulesParser.cpp

4.38 gfx::color_t Struct Reference

Public Attributes

- uint8_t **r**
- uint8_t **g**
- uint8_t **b**
- uint8_t **a** = 255

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/interfaces/IWindow.hpp

4.39 rserv::ComponentDeltaTracker Class Reference

Public Member Functions

- std::vector< uint64_t > **createEntityDelta** (uint8_t clientId, uint32_t entityId, const EntitySnapshot &currentSnapshot)
- std::vector< uint64_t > **createMultiEntityDelta** (uint8_t clientId, const std::vector< EntitySnapshot > &entities)
- EntitySnapshot **applyDelta** (uint8_t clientId, const std::vector< uint64_t > &deltaPayload)
- void **clearClientCache** (uint8_t clientId)
- void **clearEntityCache** (uint8_t clientId, uint32_t entityId)
- void **clearAllCaches** ()
- void **clearDeadEntities** (const std::set< uint32_t > &aliveEntityIds)

Private Member Functions

- std::vector< uint64_t > **serializeFullSnapshot** (uint32_t entityId, const EntitySnapshot &snapshot)
- std::vector< uint64_t > **serializeDelta** (uint32_t entityId, uint32_t changedMask, const std::map< uint8_t, std::vector< uint64_t > > &changedComponents)

Private Attributes

- std::unordered_map< uint8_t, std::unordered_map< uint32_t, EntitySnapshot > > **_clientEntityCache**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/server/deltaTracker/ComponentDeltaTracker.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/deltaTracker/ComponentDeltaTracker.cpp

4.40 parser::ComponentMetadata Struct Reference

Public Attributes

- std::string **name**
- std::type_index **typeIndex**
- std::vector< Field > **fields**
- ComponentCreator **creator**
- ComponentAdder **adder**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/ComponentRegistry/ComponentMetadata.hpp

4.41 parser::ComponentRegistrar< T > Class Template Reference

Public Member Functions

- **ComponentRegistrar** (const std::string &name, const std::vector< Field > &fields, const ComponentCreator &creator)

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/ComponentRegistry/ComponentRegistrar.hpp

4.42 parser::ComponentRegistry Class Reference

Public Member Functions

- void **registerComponent** (const ComponentMetadata &metadata)
- bool **hasComponent** (const std::string &name) const
- const std::map< std::string, ComponentMetadata > & **getAllComponents** () const
- std::shared_ptr< std::map< std::string, std::pair< std::type_index, std::vector< Field > > > > **getComponentDefinitions** () const
- std::map< std::type_index, ComponentCreator > **getComponentCreators** () const
- std::map< std::type_index, ComponentAdder > **getComponentAdders** () const

Static Public Member Functions

- static ComponentRegistry & **getInstance** ()

Private Member Functions

- **ComponentRegistry** (const ComponentRegistry &)=delete
- ComponentRegistry & **operator=** (const ComponentRegistry &)=delete

Private Attributes

- std::map< std::string, ComponentMetadata > **_components**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/ComponentRegistry/ComponentRegistry.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/ComponentRegistry/ComponentRegistry.cpp

4.43 rserv::ComponentSerializer Class Reference

Static Public Member Functions

- static EntitySnapshot **createSnapshotFromComponents** (uint32_t entityId, const std::vector< uint64_t > &componentData)

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

- /home/albane/epitech/tech3/r-type/ryanR-type/server/deltaTracker/ComponentSerializer.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/deltaTracker/ComponentSerializer.cpp

4.44 ComposantParser Class Reference

Public Types

- using **ShouldParseComponentCallback** = std::function<bool(const std::map<std::string, std::shared_ptr<FieldValue>>&)>

Public Member Functions

- **ComposantParser** (std::shared_ptr< const std::map< std::string, std::pair< std::type_index, std::vector< Field > > > > componentDefinitions, const std::map< std::type_index, ComponentCreator > &componentCreators, const ShouldParseComponentCallback &shouldParseCallback=nullptr)
- std::pair< std::shared_ptr< ecs::IComponent >, std::type_index > **parseComponent** (const std::string &componentName, const nlohmann::json &componentData)

Private Member Functions

- std::shared_ptr< FieldValue > **parseFieldValue** (const nlohmann::json &jsonValue, FieldType type)

Private Attributes

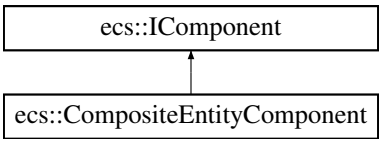
- std::shared_ptr< const std::map< std::string, std::pair< std::type_index, std::vector< Field > > > > **componentDefinitions**
- const std::map< std::type_index, ComponentCreator > & **_componentCreators**
- ShouldParseComponentCallback **_shouldParseCallback**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/ComposantParser/ComposantParser.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/ComposantParser/ComposantParser.cpp

4.45 ecs::CompositeEntityComponent Class Reference

Inheritance diagram for ecs::CompositeEntityComponent:



Public Member Functions

- **CompositeEntityComponent** (size_t parent_id)
- size_t **getParentId** () const
- void **setParentId** (size_t id)

Private Attributes

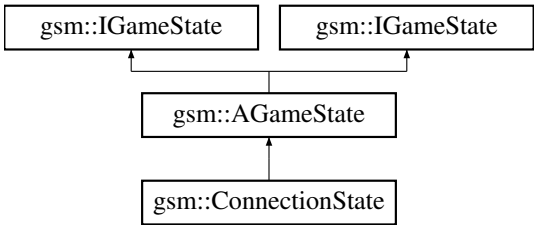
- size_t **parentId**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/permanent/CompositeEntityComponent.↩ hpp

4.46 gsm::ConnectionState Class Reference

Inheritance diagram for gsm::ConnectionState:



Public Member Functions

- **ConnectionState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void enter () override
- void update (float deltaTime) override
- void exit () override
- std::string getStateName () const override

Public Member Functions inherited from gsm::AGameState

- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override
- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override

Private Member Functions

- void **renderUI** ()
- void **updateUIStatus** ()

Private Attributes

- std::unique_ptr< ui::UIManager > **_uiManager**
- std::unique_ptr< MouseInputHandler > **_mouseHandler**
- std::shared_ptr< ui::Background > **_background**
- std::shared_ptr< ui::TextInput > **_ipInput**
- std::shared_ptr< ui::TextInput > **_portInput**
- std::shared_ptr< ui::Button > **_connectButton**
- std::shared_ptr< ui::Button > **_levelEditorButton**
- std::shared_ptr< ui::Button > **_quitButton**
- std::shared_ptr< ui::Text > **_spacer**
- std::shared_ptr< ui::UILayout > **_layout**
- std::shared_ptr< ui::SpritePreview > **_loadingAnimation**
- std::shared_ptr< ui::UILayout > **_loadingLayout**
- bool **_wasConnected** = false

Additional Inherited Members

Protected Member Functions inherited from gsm::AGameState

- void addSystem (std::shared_ptr< ecs::ISystem > system) override
- void addSystem (std::shared_ptr< ecs::ISystem > system) override

Protected Attributes inherited from gsm::AGameState

- std::weak_ptr< IGameStateMachine > **_gsm**
- std::shared_ptr< ResourceManager > **_resourceManager**
- std::vector< std::shared_ptr< ecs::ISystem > > **_systems**

4.46.1 Member Function Documentation

4.46.1.1 enter()

```
void gsm::ConnectionState::enter ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.46.1.2 exit()

```
void gsm::ConnectionState::exit ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.46.1.3 getStateName()

```
std::string gsm::ConnectionState::getStateName () const  [inline], [override], [virtual]
```

Implements gsm::AGameState.

4.46.1.4 update()

```
void gsm::ConnectionState::update (
            float deltaTime)  [override], [virtual]
```

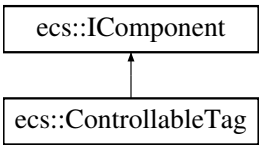
Reimplemented from gsm::AGameState.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/Connection/ConnectionState.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/Connection/ConnectionState.cpp

4.47 ecs::ControllableTag Class Reference

Inheritance diagram for ecs::ControllableTag:



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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/tags/ControllableTag.hpp

4.48 Core Class Reference

Public Member Functions

- void **initFirstScene** ()
- void **run** ()
- void **startNetwork** ()
- std::shared_ptr< ClientNetwork > **getNetwork** ()

Private Member Functions

- void **initNetwork** ()
- void **initLibraries** ()
- void **networkLoop** ()

Private Attributes

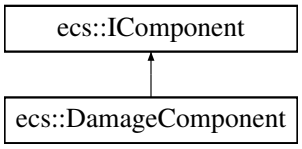
- std::shared_ptr< DLLoader< gfx::createWindow_t > > **_windowLoader**
- std::shared_ptr< DLLoader< gfx::createEvent_t > > **_eventLoader**
- std::shared_ptr< DLLoader< gfx::createAudio_t > > **_audioLoader**
- std::shared_ptr< ResourceManager > **_resourceManager**
- std::shared_ptr< gsm::GameStateMachine > **_gsm**
- std::shared_ptr< ecs::Registry > **_registry**
- std::shared_ptr< ClientNetwork > **_clientNetwork**
- std::shared_ptr< Parser > **_parser**
- std::thread **_networkThread**
- std::chrono::steady_clock::time_point **_lastHealthcheckTime**
- float **_healthcheckInterval** = constants::HEALTHCHECK_INTERVAL

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/Core.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/Core.cpp

4.49 ecs::DamageComponent Class Reference

Inheritance diagram for ecs::DamageComponent:



Public Member Functions

- **DamageComponent** (float damage=0.0f)
- float **getDamage** () const
- void **setDamage** (float damage)

Private Attributes

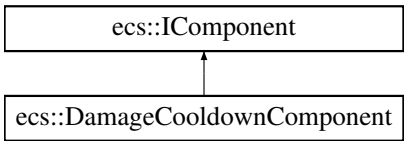
- float **_damage**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/permanent/DamageComponent.hpp

4.50 ecs::DamageCooldownComponent Class Reference

Inheritance diagram for ecs::DamageCooldownComponent:



Public Member Functions

- **DamageCooldownComponent** (float cooldown=constants::TRIGGER_DAMAGE_COOLDOWN)
- float **getCooldown** () const
- void **setCooldown** (float cooldown)
- float **getLastDamageTime** () const
- void **setLastDamageTime** (float time)

Private Attributes

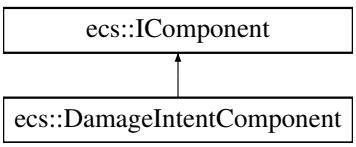
- float **_cooldown**
- float **_lastDamageTime**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/permanent/DamageCooldown↩ Component.hpp

4.51 ecs::DamageIntentComponent Class Reference

Inheritance diagram for ecs::DamageIntentComponent:



Public Member Functions

- **DamageIntentComponent** (float damages=0.0f, ecs::Entity source=0)
- float **getDamages** ()
- void **setDamages** (float damages)
- ecs::Entity **getSource** () const
- void **setSource** (ecs::Entity source)

Private Attributes

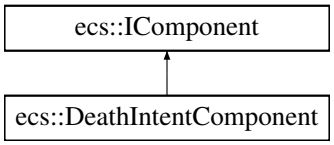
- float **_damages**
- ecs::Entity **_source**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/temporary/DamageIntentComponent.hpp

4.52 ecs::DeathIntentComponent Class Reference

Inheritance diagram for ecs::DeathIntentComponent:



Public Member Functions

- **DeathIntentComponent** (ecs::Entity source=0)
- ecs::Entity **getSource** () const
- void **setSource** (ecs::Entity source)

Private Attributes

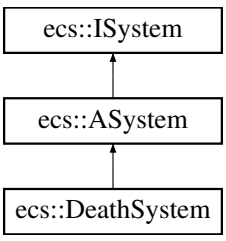
- ecs::Entity **_source**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/temporary/DeathIntentComponent.hpp

4.53 ecs::DeathSystem Class Reference

Inheritance diagram for ecs::DeathSystem:



Public Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

4.53.1 Member Function Documentation

4.53.1.1 update()

```
void ecs::DeathSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/death/DeathSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/death/DeathSystem.cpp

4.54 debug::Debug Class Reference

Static Public Member Functions

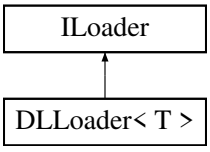
- static void **printDebug** (const bool isDebug, const std::string &message, debugType type, debugLevel level)

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/debug.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/debug.cpp

4.55 DLLoader< T > Class Template Reference

Inheritance diagram for DLLoader< T >:



Public Member Functions

- void * getHandler () const override
- void * Open (const char *path, int flag=RTLD_LAZY) override
- void * Symbol (const char *symbolName) override
- T **getSymbol** (const char *symbolName)
- int Close () override
- const char * Error () override

Private Attributes

- void * **_handler** = nullptr

4.55.1 Member Function Documentation

4.55.1.1 Close()

```
template<typename T>
int DLLoader< T >::Close ()  [inline], [override], [virtual]
```

Implements ILoader.

4.55.1.2 Error()

```
template<typename T>
const char * DLLoader< T >::Error ()  [inline], [override], [virtual]
```

Implements ILoader.

4.55.1.3 getHandler()

```
template<typename T>
void * DLLoader< T >::getHandler () const  [inline], [override], [virtual]
```

Implements ILoader.

4.55.1.4 Open()

```
template<typename T>
void * DLLoader< T >::Open (
            const char * path,
            int flag = RTLD_LAZY)  [inline], [override], [virtual]
```

Implements ILoader.

4.55.1.5 Symbol()

```
template<typename T>
void * DLLoader< T >::Symbol (
            const char * symbolName)  [inline], [override], [virtual]
```

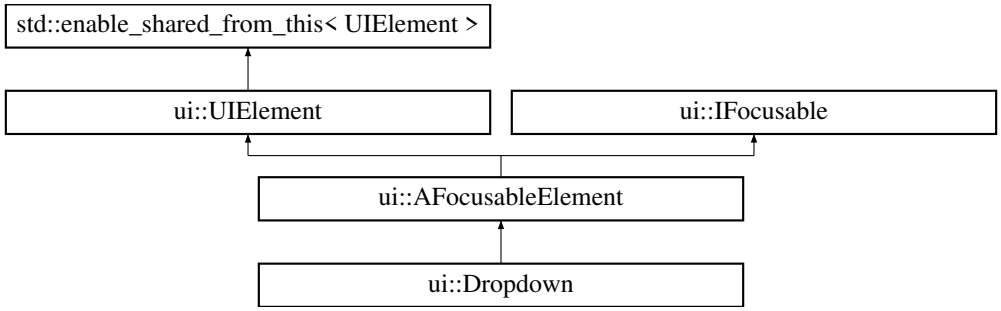
Implements ILoader.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/DLLoader/DLLoader.hpp

4.56 ui::Dropdown Class Reference

Inheritance diagram for ui::Dropdown:



Public Member Functions

- **Dropdown** (std::shared_ptr< ResourceManager > resourceManager)
- virtual void render () override
- virtual void handleInput (const math::Vector2f &mousePos, bool mousePressed) override
- virtual void update (float deltaTime) override
- virtual bool containsPoint (const math::Vector2f &point) const override
- void **setOptions** (const std::vector< std::string > &options)
- void **addOption** (const std::string &option)
- void **clearOptions** ()
- const std::vector< std::string > & **getOptions** () const
- const std::string & **getSelectedOption** () const
- void **setSelectedIndex** (size_t index)
- size_t **getSelectedIndex** () const
- const std::string & **getSelectedValue** () const
- void **setDirection** (DropdownDirection direction)
- DropdownDirection **getDirection** () const
- void **setPlaceholder** (const std::string &placeholder)
- const std::string & **getPlaceholder** () const
- void **setTextColor** (const gfx::color_t &color)
- void **setPlaceholderColor** (const gfx::color_t &color)
- void **setFontPath** (const std::string &fontPath)
- void **setBaseFontSize** (size_t fontSize)
- size_t **getBaseFontSize** () const
- void **setOnSelectionChanged** (std::function< void(const std::string &, size_t)> callback)
- void **open** ()
- void **close** ()
- bool **isOpen** () const

Public Member Functions inherited from ui::AFocusableElement

- **AFocusableElement** (std::shared_ptr< ResourceManager > resourceManager)
- virtual void setFocused (bool focused) override
- virtual bool isFocused () const override
- virtual bool canBeFocused () const override
- virtual void onFocusGained () override
- virtual void onFocusLost () override
- virtual void onActivated () override
- void **setOnFocusGained** (std::function< void()> callback)
- void **setOnFocusLost** (std::function< void()> callback)
- void **setOnActivated** (std::function< void()> callback)

Public Member Functions inherited from ui::UIElement

- **UIElement** (std::shared_ptr< ResourceManager > resourceManager)
- void **setPosition** (const math::Vector2f &position)
- void **setSize** (const math::Vector2f &size)
- math::Vector2f **getPosition** () const
- math::Vector2f **getSize** () const
- math::Vector2f **getAbsolutePosition** () const
- math::Vector2f **getAbsoluteSize** () const
- void **setVisible** (bool visible)
- bool **isVisible** () const
- void **setScalingEnabled** (bool enabled)
- bool **isScalingEnabled** () const
- void **setFocusEnabled** (bool enabled)
- bool **isFocusEnabled** () const
- void **setState** (UIState state)
- UIState **getState** () const
- virtual void **setScale** (UIScale scale)
- UIScale **getScale** () const
- void **setParent** (std::weak_ptr< UIElement > parent)
- std::shared_ptr< UIElement > **getParent** () const
- void **addChild** (std::shared_ptr< UIElement > child)
- void **removeChild** (std::shared_ptr< UIElement > child)
- const std::vector< std::shared_ptr< UIElement > > & **getChildren** () const
- void **setOnClick** (std::function< void()> callback)
- void **setOnHover** (std::function< void()> callback)
- void **setOnRelease** (std::function< void()> callback)

Public Member Functions inherited from ui::IFocusable

- virtual bool **onNavigateLeft** ()
- virtual bool **onNavigateRight** ()

Private Member Functions

- void **renderClosed** ()
- void **renderOpen** ()
- void **drawArrow** (float arrowX, float arrowY, float arrowSize)
- bool **isMouseOverOption** (const math::Vector2f &mousePos, size_t optionIndex) const
- math::Vector2f **getOptionPosition** (size_t optionIndex) const
- math::Vector2f **getDropdownSize** () const
- size_t **getFontSize** () const
- gfx::color_t **_getCurrentColor** () const

Private Attributes

- std::vector< std::string > **_options**
- size_t **_selectedIndex** = 0
- bool **_hasSelection** = false
- DropdownDirection **_direction** = DropdownDirection::Down
- bool **_isOpen** = false
- int **_hoveredOptionIndex** = -1
- std::string **_placeholder** = "Select..."
- std::function< void(const std::string &, size_t)> **_onSelectionChanged**
- gfx::color_t **_textColor** = colors::UI_TEXT
- gfx::color_t **_placeholderColor** = colors::UI_DISABLED
- std::string **_fontPath** = constants::MAIN_FONT
- size_t **_baseFontSize** = constants::BUTTON_FONT_SIZE_BASE
- gfx::color_t **_normalColor** = colors::BUTTON_PRIMARY
- gfx::color_t **_hoveredColor** = colors::BUTTON_PRIMARY_HOVER
- gfx::color_t **_pressedColor** = colors::BUTTON_PRIMARY_PRESSED
- gfx::color_t **_disabledColor** = colors::UI_DISABLED
- gfx::color_t **_focusedColor** = colors::UI_FOCUSED
- bool **_dropdownWasPressed** = false

Additional Inherited Members

Protected Member Functions inherited from ui::AFocusableElement

- virtual void **onFocusStateChanged** (bool focused)

Protected Member Functions inherited from ui::UIElement

- std::pair< int, int > **getWindowSize** () const
- std::pair< int, int > **getLogicalSize** () const
- float **getScaleFactor** () const

Protected Attributes inherited from ui::AFocusableElement

- bool **_focused** = false
- bool **_pressedInside** = false
- bool **_wasPressed** = false
- std::function< void()> **_onFocusGained**
- std::function< void()> **_onFocusLost**
- std::function< void()> **_onActivated**

Protected Attributes inherited from ui::UIElement

- std::weak_ptr< ResourceManager > **_resourceManager**
- math::Vector2f **_position**
- math::Vector2f **_size**
- bool **_visible** = true
- bool **_scalingEnabled** = true
- bool **_focusEnabled** = true
- UIState **_state** = UIState::Normal
- UIScale **_scale** = UIScale::Normal
- std::weak_ptr< UIElement > **_parent**
- std::vector< std::shared_ptr< UIElement > > **_children**
- bool **_pressedInside** = false
- bool **_wasPressed** = false
- std::function< void()> **_onClick**
- std::function< void()> **_onHover**
- std::function< void()> **_onRelease**

4.56.1 Member Function Documentation

4.56.1.1 containsPoint()

```
bool ui::Dropdown::containsPoint (
            const math::Vector2f & point) const  [override], [virtual]
```

Reimplemented from ui::UIElement.

4.56.1.2 handleInput()

```
void ui::Dropdown::handleInput (
            const math::Vector2f & mousePos,
            bool mousePressed)  [override], [virtual]
```

Reimplemented from ui::AFocusableElement.

4.56.1.3 render()

```
void ui::Dropdown::render ()  [override], [virtual]
```

Reimplemented from ui::UIElement.

4.56.1.4 update()

```
void ui::Dropdown::update (
            float deltaTime)  [override], [virtual]
```

Reimplemented from ui::UIElement.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/elements/focusable/Dropdown.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/elements/focusable/Dropdown.cpp

4.57 utils::Encryption Class Reference

Static Public Member Functions

- static std::string **encrypt** (const std::string &data)
- static std::string **decrypt** (const std::string &encryptedData)
- static std::string **base64Encode** (const std::vector< unsigned char > &data)
- static std::vector< unsigned char > **base64Decode** (const std::string &encoded)

Static Private Attributes

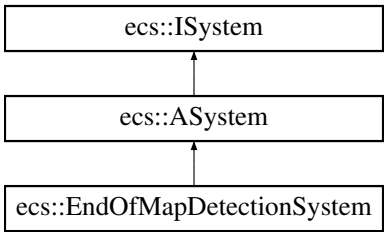
- static const std::string **_key** = "R-Type_Secure_Key_2026"

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/utils/Encryption.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/utils/Encryption.cpp

4.58 ecs::EndOfMapDetectionSystem Class Reference

Inheritance diagram for ecs::EndOfMapDetectionSystem:



Public Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

4.58.1 Member Function Documentation

4.58.1.1 update()

```
void ecs::EndOfMapDetectionSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

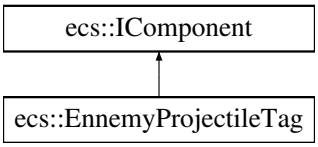
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/server/systems/gameEnd/EndOfMapDetectionSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/systems/gameEnd/EndOfMapDetectionSystem.cpp

4.59 ecs::EnnemyProjectileTag Class Reference

Inheritance diagram for ecs::EnnemyProjectileTag:



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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/tags/EnnemyProjectileTag.hpp

4.60 ecs::EntityCreationContext Struct Reference

Static Public Member Functions

- static EntityCreationContext **forServer** ()
- static EntityCreationContext **forLocalClient** ()

Public Attributes

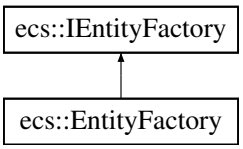
- EntityCreationOrigin **origin** = EntityCreationOrigin::CLIENT_LOCAL

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/ECS/entity/EntityCreationContext.hpp

4.61 ecs::EntityFactory Class Reference

Inheritance diagram for ecs::EntityFactory:



Public Member Functions

- Entity createEntity (const std::shared_ptr< Registry > &registry, const EntityCreationContext &context=EntityCreationContext::forLocalClient()) override

Private Attributes

- std::atomic< size_t > **_nextLocalId**

4.61.1 Member Function Documentation

4.61.1.1 createEntity()

```
Entity ecs::EntityFactory::createEntity (
            const std::shared_ptr< Registry > & registry,
            const EntityCreationContext & context = EntityCreationContext::forLocalClient())
[override], [virtual]
```

Implements ecs::IEntityFactory.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/ECS/entity/factory/EntityFactory.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/ECS/entity/factory/EntityFactory.cpp

4.62 EntityParser Class Reference

Public Types

- using **ShouldParseComponentCallback** = ComposantParser::ShouldParseComponentCallback

Public Member Functions

- **EntityParser** (std::shared_ptr< const std::map< std::string, std::pair< std::type_index, std::vector< Field > > > > componentDefinitions, const std::map< std::type_index, ComponentCreator > &componentCreators, const std::map< std::type_index, ComponentAdder > &componentAdders, const ShouldParseComponentCallback &shouldParseCallback=nullptr)
- std::shared_ptr< IPrefab > **parseEntity** (const std::string &filePath)

Private Attributes

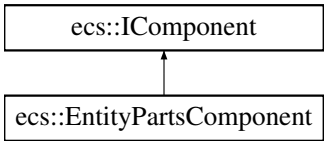
- ComposantParser **_composantParser**
- const std::map< std::type_index, ComponentAdder > & **_componentAdders**
- ShouldParseComponentCallback **_shouldParseCallback**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/EntityParser/EntityParser.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/EntityParser/EntityParser.cpp

4.63 ecs::EntityPartsComponent Class Reference

Inheritance diagram for ecs::EntityPartsComponent:



Public Attributes

- std::vector< size_t > **partIds**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/permanent/EntityPartsComponent.hpp

4.64 EntityPrefabManager Class Reference

Public Member Functions

- void **registerPrefab** (const std::string &name, const std::shared_ptr< IPrefab > &prefab)
- std::shared_ptr< IPrefab > **getPrefab** (const std::string &name) const
- ecs::Entity **createEntityFromPrefab** (const std::string &prefabName, const std::shared_ptr< ecs::Registry > &registry, const ecs::EntityCreationContext &context)
- ecs::Entity **createEntityFromPrefab** (const std::string &prefabName, const std::shared_ptr< ecs::Registry > &registry)
- bool **hasPrefab** (const std::string &name) const
- void **deletePrefab** (const std::string &name)
- void **clearPrefabs** ()
- std::shared_ptr< ecs::IEntityFactory > **getEntityFactory** () const
- void **setEntityFactory** (std::shared_ptr< ecs::IEntityFactory > factory)
- void **setOnEntityCreated** (std::function< void(ecs::Entity, const std::string &)> callback)

Private Attributes

- std::map< std::string, std::shared_ptr< IPrefab > > **_prefabs**
- std::shared_ptr< ecs::IEntityFactory > **_entityFactory**
- std::function< void(ecs::Entity, const std::string &)> **_onEntityCreated**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Prefab/entityPrefabManager/EntityPrefabManager.↩ hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/Prefab/entityPrefabManager/EntityPrefabManager.↩ cpp

4.65 rserv::EntitySnapshot Struct Reference

Public Attributes

- uint32_t **entityId**
- uint32_t **componentMask**
- std::map< uint8_t, std::vector< uint64_t > > **components**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/server/deltaTracker/ComponentDeltaTracker.hpp

4.66 Field Struct Reference

Public Member Functions

- **Field** (std::string n, FieldType t, bool opt=false, std::shared_ptr< FieldValue > def=nullptr)

Public Attributes

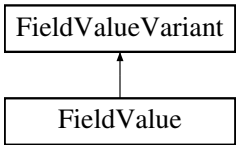
- std::string **name** = ""
- FieldType **type**
- bool **optional** = false
- std::shared_ptr< FieldValue > **defaultValue** = nullptr

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/ParserParam.hpp

4.67 FieldValue Struct Reference

Inheritance diagram for FieldValue:



Public Member Functions

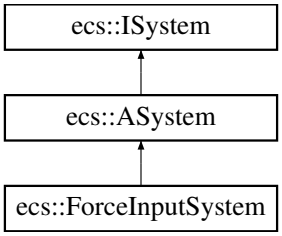
- template<typename T>
  FieldValue (T &&value)

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/ParserParam.hpp

4.68 ecs::ForceInputSystem Class Reference

Inheritance diagram for ecs::ForceInputSystem:



Public Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Private Member Functions

- bool **isPlayerAlive** (std::shared_ptr< Registry > registry, Entity entityId) const

Private Attributes

- float **_lastForceTime**

4.68.1 Member Function Documentation

4.68.1.1 update()

```
void ecs::ForceInputSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

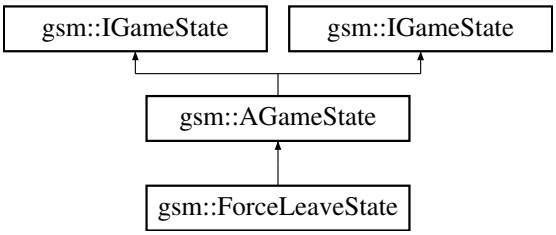
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/input/ForceInputSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/input/ForceInputSystem.cpp

4.69 gsm::ForceLeaveState Class Reference

Inheritance diagram for gsm::ForceLeaveState:



Public Member Functions

- **ForceLeaveState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager, constants::ForceLeaveType leaveType)
- void enter () override
- void update (float deltaTime) override
- void exit () override
- std::string getStateName () const override

Public Member Functions inherited from gsm::AGameState

- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override
- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override

Private Member Functions

- void **renderUI** ()

Private Attributes

- constants::ForceLeaveType **_leaveType**
- std::unique_ptr< ui::UIManager > **_uiManager**
- std::unique_ptr< MouseInputHandler > **_mouseHandler**
- std::shared_ptr< ui::Text > **_reasonText**
- std::shared_ptr< ui::Button > **_leaveButton**
- std::shared_ptr< ui::UILayout > **_bottomRightLayout**

Additional Inherited Members

Protected Member Functions inherited from gsm::AGameState

- void addSystem (std::shared_ptr< ecs::ISystem > system) override
- void addSystem (std::shared_ptr< ecs::ISystem > system) override

Protected Attributes inherited from gsm::AGameState

- std::weak_ptr< IGameStateMachine > **_gsm**
- std::shared_ptr< ResourceManager > **_resourceManager**
- std::vector< std::shared_ptr< ecs::ISystem > > **_systems**

4.69.1 Member Function Documentation

4.69.1.1 enter()

```
void gsm::ForceLeaveState::enter ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.69.1.2 exit()

```
void gsm::ForceLeaveState::exit ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.69.1.3 getStateName()

```
std::string gsm::ForceLeaveState::getStateName () const  [inline], [override], [virtual]
```

Implements gsm::AGameState.

4.69.1.4 update()

```
void gsm::ForceLeaveState::update (
            float deltaTime)  [override], [virtual]
```

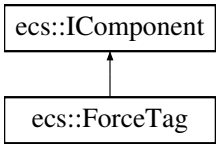
Reimplemented from gsm::AGameState.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/ForceLeave/ForceLeaveState.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/ForceLeave/ForceLeaveState.cpp

4.70 ecs::ForceTag Class Reference

Inheritance diagram for ecs::ForceTag:



Public Member Functions

- **ForceTag** (const std::string &type="")
- std::string **getForceType** () const
- void **setForceType** (const std::string &type)

Private Attributes

- std::string **forceType**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/tags/ForceTag.hpp

4.71 math::FRect Class Reference

Public Member Functions

- **FRect** (float left, float top, float width, float height)
- **FRect** (FRect const &other)
- float **getLeft** () const
- void **setLeft** (float left)
- float **getTop** () const
- void **setTop** (float top)
- float **getWidth** () const
- void **setWidth** (float width)
- float **getHeight** () const
- void **setHeight** (float height)
- bool **contains** (float x, float y) const
- bool **intersects** (FRect const &other) const
- bool **intersects** (FRect const &other, FRect &intersection) const
- FRect & **operator=** (FRect const &other)
- bool **operator==** (FRect const &other) const
- bool **operator!=** (FRect const &other) const

Private Attributes

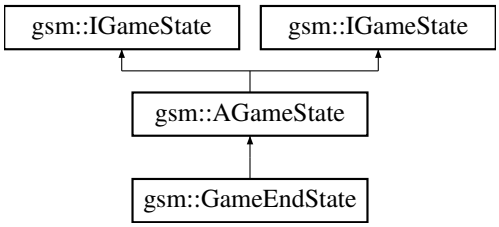
- float **left**
- float **top**
- float **width**
- float **height**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/types/FRect.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/types/FRect.cpp

4.72 gsm::GameEndState Class Reference

Inheritance diagram for gsm::GameEndState:



Public Member Functions

- **GameEndState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void update (float deltaTime) override
- std::string getStateName () const override

Public Member Functions inherited from gsm::AGameState

- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void enter () override
- void exit () override
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override
- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void enter () override
- void exit () override
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override

Additional Inherited Members

Protected Member Functions inherited from gsm::AGameState

- void addSystem (std::shared_ptr< ecs::ISystem > system) override
- void addSystem (std::shared_ptr< ecs::ISystem > system) override

Protected Attributes inherited from gsm::AGameState

- std::weak_ptr< IGameStateMachine > **_gsm**
- std::shared_ptr< ResourceManager > **_resourceManager**
- std::vector< std::shared_ptr< ecs::ISystem > > **_systems**

4.72.1 Member Function Documentation

4.72.1.1 getStateName()

```
std::string gsm::GameEndState::getStateName () const  [inline], [override], [virtual]
```

Implements gsm::AGameState.

4.72.1.2 update()

```
void gsm::GameEndState::update (
            float deltaTime)  [override], [virtual]
```

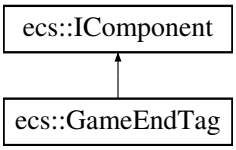
Reimplemented from gsm::AGameState.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/server/gsm/states/scenes/GameEnd/GameEndState.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/gsm/states/scenes/GameEnd/GameEndState.cpp

4.73 ecs::GameEndTag Class Reference

Inheritance diagram for ecs::GameEndTag:



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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/tags/GameEndTag.hpp

4.74 GameRules Class Reference

Public Member Functions

- void **setGamemode** (GameRulesNS::Gamemode gamemode)
- GameRulesNS::Gamemode **getGamemode** () const
- void **setDifficulty** (GameRulesNS::Difficulty difficulty)
- GameRulesNS::Difficulty **getDifficulty** () const
- void **setCrossfire** (bool crossfire)
- bool **getCrossfire** () const

Private Attributes

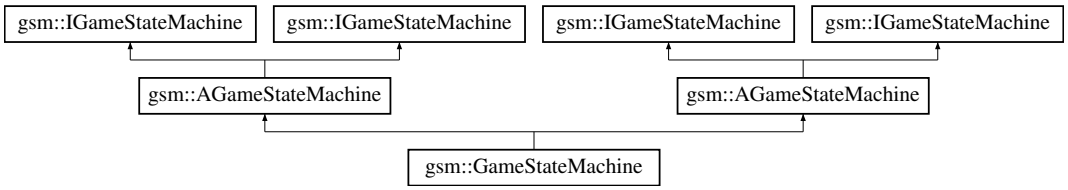
- GameRulesNS::Gamemode **_gamemode**
- GameRulesNS::Difficulty **_difficulty**
- bool **_crossfire**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/GameRules.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/GameRules.cpp

4.75 gsm::GameStateMachine Class Reference

Inheritance diagram for gsm::GameStateMachine:



Public Member Functions

- void requestStateChange (std::shared_ptr< IGameState > newState) override
- void requestStatePush (std::shared_ptr< IGameState > newState) override
- void requestStatePop () override

Public Member Functions inherited from gsm::AGameStateMachine

- void changeState (std::shared_ptr< IGameState > newState) override
- void pushState (std::shared_ptr< IGameState > newState) override
- void popState () override
- void update (float deltaTime) override
- void changeState (std::shared_ptr< IGameState > newState) override
- void pushState (std::shared_ptr< IGameState > newState) override
- void popState () override
- void update (float deltaTime) override
- std::string **getCurrentStateName** () const

Additional Inherited Members

Protected Attributes inherited from gsm::AGameStateMachine

- std::stack< std::shared_ptr< IGameState > > **_states**
- std::shared_ptr< IGameState > **_pendingChangeState**
- std::shared_ptr< IGameState > **_pendingPushState**
- bool **_pendingPopState** = false

4.75.1 Member Function Documentation

4.75.1.1 requestStateChange()

```
void gsm::GameStateMachine::requestStateChange (
            std::shared_ptr< IGameState > newState)  [override], [virtual]
```

Reimplemented from gsm::AGameStateMachine.

4.75.1.2 requestStatePop()

```
void gsm::GameStateMachine::requestStatePop ()  [override], [virtual]
```

Reimplemented from gsm::AGameStateMachine.

4.75.1.3 requestStatePush()

```
void gsm::GameStateMachine::requestStatePush (
            std::shared_ptr< IGameState > newState)  [override], [virtual]
```

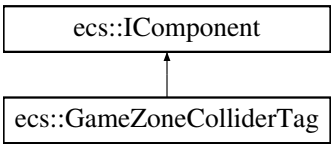
Reimplemented from gsm::AGameStateMachine.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/machine/GameStateMachine.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/gsm/machine/GameStateMachine.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/machine/GameStateMachine.cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/gsm/machine/GameStateMachine.cpp

4.76 ecs::GameZoneColliderTag Class Reference

Inheritance diagram for ecs::GameZoneColliderTag:

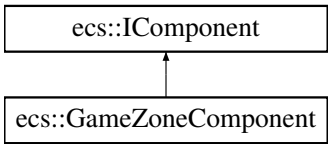


The documentation for this class was generated from the following file:

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/tags/GameZoneColliderTag.hpp

4.77 ecs::GameZoneComponent Class Reference

Inheritance diagram for ecs::GameZoneComponent:



Public Member Functions

- **GameZoneComponent** (math::FRect zone=math::FRect(0.0f, 0.0f, constants::MAX_WIDTH, constants::↩ MAX_HEIGHT))
- math::FRect **getZone** () const
- void **setZone** (math::FRect zone)

Private Attributes

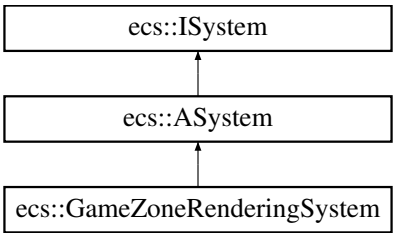
- math::FRect **_zone**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/permanent/GameZoneComponent.hpp

4.78 ecs::GameZoneRenderingSystem Class Reference

Inheritance diagram for ecs::GameZoneRenderingSystem:



Protected Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Additional Inherited Members

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

4.78.1 Member Function Documentation

4.78.1.1 update()

```
void ecs::GameZoneRenderingSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [protected], [virtual]
```

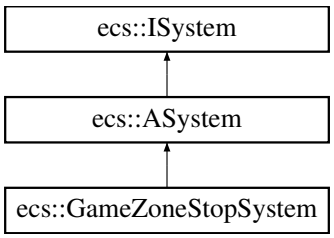
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/rendering/GameZoneRenderingSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/rendering/GameZoneRenderingSystem.cpp

4.79 ecs::GameZoneStopSystem Class Reference

Inheritance diagram for ecs::GameZoneStopSystem:



Public Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

4.79.1 Member Function Documentation

4.79.1.1 update()

```
void ecs::GameZoneStopSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

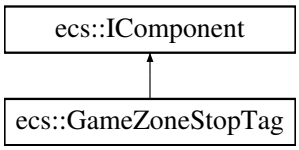
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/bounds/GameZoneStopSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/bounds/GameZoneStopSystem.cpp

4.80 ecs::GameZoneStopTag Class Reference

Inheritance diagram for ecs::GameZoneStopTag:

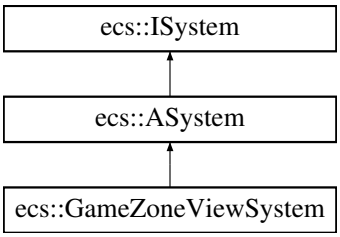


The documentation for this class was generated from the following file:

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/tags/GameZoneStopTag.hpp

4.81 ecs::GameZoneViewSystem Class Reference

Inheritance diagram for ecs::GameZoneViewSystem:



Protected Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Additional Inherited Members

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

4.81.1 Member Function Documentation

4.81.1.1 update()

```
void ecs::GameZoneViewSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [protected], [virtual]
```

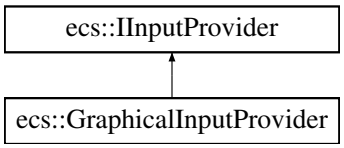
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/rendering/GameZoneViewSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/rendering/GameZoneViewSystem.cpp

4.82 ecs::GraphicalInputProvider Class Reference

Inheritance diagram for ecs::GraphicalInputProvider:



Public Member Functions

- **GraphicalInputProvider** (std::shared_ptr< gfx::IEvent > eventSystem, std::shared_ptr< InputMappingManager > mappingManager)
- float getAxisValue (event_t axis, size_t clientID=0) override
- bool isActionPressed (InputAction action, size_t clientID=0) override
- float getActionAxis (InputAction action, size_t clientID=0) override
- InputMapping getInputMapping (size_t clientID=0) const override
- void **setToggleMode** (bool enabled)
- bool **isToggleMode** () const

Private Attributes

- std::shared_ptr< gfx::IEvent > **_eventSystem**
- std::shared_ptr< InputMappingManager > **_mappingManager**
- bool **_toggleMode**
- std::map< InputAction, bool > **_toggledStates**
- std::map< InputAction, bool > **_lastKeyState**
- std::map< std::pair< InputAction, gfx::EventType >, bool > **_keyPressedState**
- std::map< std::pair< InputAction, gfx::EventType >, bool > **_toggledKeyStates**
- std::map< std::pair< InputAction, gfx::EventType >, int > **_lastToggleFrame**
- int **_currentFrame**

Additional Inherited Members

Public Types inherited from ecs::IInputProvider

- using **event_t** = gfx::EventType

4.82.1 Member Function Documentation

4.82.1.1 getActionAxis()

```
float ecs::GraphicalInputProvider::getActionAxis (
            InputAction action,
            size_t clientID = 0)  [override], [virtual]
```

Implements ecs::IInputProvider.

4.82.1.2 getAxisValue()

```
float ecs::GraphicalInputProvider::getAxisValue (
            event_t axis,
            size_t clientID = 0)  [override], [virtual]
```

Implements ecs::IInputProvider.

4.82.1.3 getInputMapping()

```
InputMapping ecs::GraphicalInputProvider::getInputMapping (
            size_t clientID = 0) const  [override], [virtual]
```

Implements ecs::IInputProvider.

4.82.1.4 isActionPressed()

```
bool ecs::GraphicalInputProvider::isActionPressed (
            InputAction action,
            size_t clientID = 0)  [override], [virtual]
```

Implements ecs::IInputProvider.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/initResourcesManager/GraphicalInputProvider.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/initResourcesManager/GraphicalInputProvider.cpp

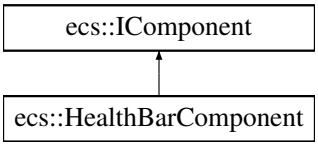
4.83 ecs::Group< Components > Class Template Reference

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/ECS/entity/registry/Registry.hpp

4.84 ecs::HealthBarComponent Class Reference

Inheritance diagram for ecs::HealthBarComponent:

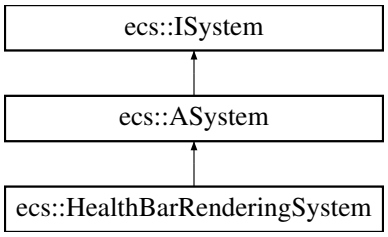


The documentation for this class was generated from the following file:

- /home/albane/epitech/tech3/r-type/ryanR-type/client/components/rendering/HealthBarComponent.hpp

4.85 ecs::HealthBarRenderingSystem Class Reference

Inheritance diagram for ecs::HealthBarRenderingSystem:



Protected Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Additional Inherited Members

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

4.85.1 Member Function Documentation

4.85.1.1 update()

```
void ecs::HealthBarRenderingSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [protected], [virtual]
```

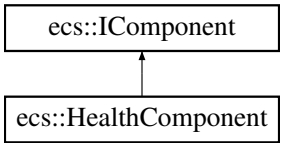
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/rendering/HealthBarRenderingSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/rendering/HealthBarRenderingSystem.cpp

4.86 ecs::HealthComponent Class Reference

Inheritance diagram for ecs::HealthComponent:



Public Member Functions

- **HealthComponent** (float health=100)
- float **getHealth** () const
- void **setHealth** (float health)
- float **getBaseHealth** () const
- void **setBaseHealth** (float health)
- ecs::Entity **getLastDamageSource** () const
- void **setLastDamageSource** (ecs::Entity source)

Private Attributes

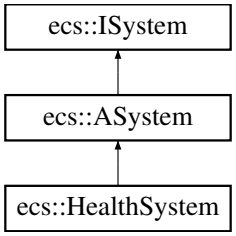
- float **_health**
- float **_baseHealth**
- ecs::Entity **_lastDamageSource**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/permanent/HealthComponent.hpp

4.87 ecs::HealthSystem Class Reference

Inheritance diagram for ecs::HealthSystem:



Public Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Private Member Functions

- void **_handleDamageUpdates** (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry)
- void **_handleHealthUpdates** (std::shared_ptr< Registry > registry)

4.87.1 Member Function Documentation

4.87.1.1 update()

```
void ecs::HealthSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

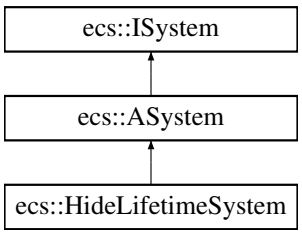
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/health/HealthSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/health/HealthSystem.cpp

4.88 ecs::HideLifetimeSystem Class Reference

Inheritance diagram for ecs::HideLifetimeSystem:



Public Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

4.88.1 Member Function Documentation

4.88.1.1 update()

```
void ecs::HideLifetimeSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

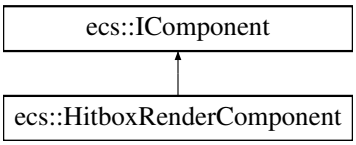
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/effects/HideLifetimeSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/effects/HideLifetimeSystem.cpp

4.89 ecs::HitboxRenderComponent Class Reference

Inheritance diagram for ecs::HitboxRenderComponent:



Public Member Functions

- **HitboxRenderComponent** (gfx::color_t color, float outlineThickness=1.0f)
- const gfx::color_t & **getColor** () const
- void **setColor** (const gfx::color_t &color)
- float **getOutlineThickness** () const
- void **setOutlineThickness** (float thickness)

Private Attributes

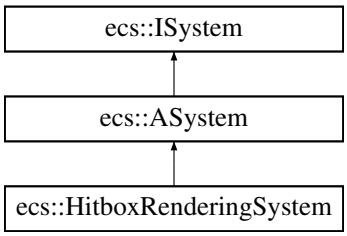
- gfx::color_t **_color**
- float **_outlineThickness**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/components/rendering/HitboxRenderComponent.hpp

4.90 ecs::HitboxRenderingSystem Class Reference

Inheritance diagram for ecs::HitboxRenderingSystem:



Protected Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Additional Inherited Members

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

4.90.1 Member Function Documentation

4.90.1.1 update()

```
void ecs::HitboxRenderingSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [protected], [virtual]
```

Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/rendering/HitboxRenderingSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/rendering/HitboxRenderingSystem.cpp

4.91 gsm::HorizontalLineObstacle Struct Reference

Public Attributes

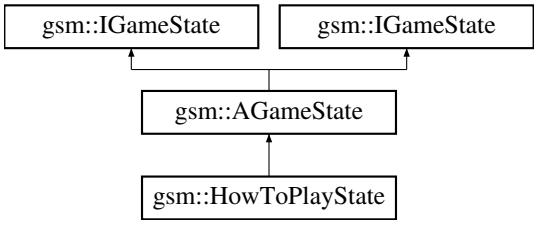
- float **fromX**
- float **posY**
- int **count**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditor/LevelEditorState.hpp

4.92 gsm::HowToPlayState Class Reference

Inheritance diagram for gsm::HowToPlayState:



Public Member Functions

- **HowToPlayState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void enter () override
- void update (float deltaTime) override
- void exit () override
- std::string getStateName () const override

Public Member Functions inherited from gsm::AGameState

- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override
- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override

Private Member Functions

- void **renderUI** ()

Private Attributes

- std::unique_ptr< MouseInputHandler > **_mouseHandler**
- std::unique_ptr< ui::UIManager > **_uiManager**
- std::shared_ptr< ui::Background > **_background**
- std::shared_ptr< ui::Text > **_titleText**
- std::shared_ptr< ui::Button > **_backButton**
- std::vector< std::shared_ptr< ui::Text > > **_controlTexts**
- std::vector< std::shared_ptr< ui::Text > > **_objectiveTexts**

Additional Inherited Members

Protected Member Functions inherited from gsm::AGameState

- void addSystem (std::shared_ptr< ecs::ISystem > system) override
- void addSystem (std::shared_ptr< ecs::ISystem > system) override

Protected Attributes inherited from gsm::AGameState

- std::weak_ptr< IGameStateMachine > **_gsm**
- std::shared_ptr< ResourceManager > **_resourceManager**
- std::vector< std::shared_ptr< ecs::ISystem > > **_systems**

4.92.1 Member Function Documentation

4.92.1.1 enter()

```
void gsm::HowToPlayState::enter ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.92.1.2 exit()

```
void gsm::HowToPlayState::exit ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.92.1.3 getStateName()

```
std::string gsm::HowToPlayState::getStateName () const  [inline], [override], [virtual]
```

Implements gsm::AGameState.

4.92.1.4 update()

```
void gsm::HowToPlayState::update (
            float deltaTime)  [override], [virtual]
```

Reimplemented from gsm::AGameState.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/HowToPlay/HowToPlayState.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/HowToPlay/HowToPlayState.cpp

4.93 rserv::HttpServer Class Reference

Public Member Functions

- **HttpServer** (std::function< bool()> statusChecker, std::function< ServerInfo()> infoGetter, std::shared_ptr< ServerConfig > serverConfig, std::function< std::string(const std::string &)> commandExecutor)
- void **start** ()
- void **stop** ()
- void **statusEndpoint** (const httplib::Request &, httplib::Response &res)
- void **infoEndpoint** (const httplib::Request &, httplib::Response &res)
- void **configEndpoint** (const httplib::Request &, httplib::Response &res)
- void **commandsSuggestionsEndpoint** (const httplib::Request &, httplib::Response &res)
- void **commandsExecuteEndpoint** (const httplib::Request &, httplib::Response &res)

Private Member Functions

- void **httpLoop** ()
- void **loadEnv** ()
- bool **checkAuth** (const httplib::Request &req)

Private Attributes

- std::thread **_httpThread**
- std::atomic_bool **_running**
- std::function< bool()> **_statusChecker**
- std::function< ServerInfo()> **_infoGetter**
- std::shared_ptr< ServerConfig > **_serverConfig**
- std::function< std::string(const std::string &)> **_commandExecutor**
- std::unique_ptr< httplib::Server > **_server**
- std::string **_password**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/server/http/HttpServer.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/http/HttpServer.cpp

4.94 gfx::IAudio Class Reference

Public Member Functions

- virtual void **playMusic** (const std::string &musicPath, bool loop=true)=0
- virtual void **stopMusic** ()=0
- virtual void **pauseMusic** ()=0
- virtual void **resumeMusic** ()=0
- virtual void **setMusicVolume** (float volume)=0
- virtual float **getMusicVolume** () const =0
- virtual bool **isMusicPlaying** () const =0
- virtual void **playSound** (const std::string &soundPath, float volume=100.0f, float pitch=1.0f)=0
- virtual void **setSoundVolume** (float volume)=0
- virtual float **getSoundVolume** () const =0
- virtual void **stopAllSounds** ()=0

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/interfaces/IAudio.hpp

4.95 IBuffer Class Reference

Public Member Functions

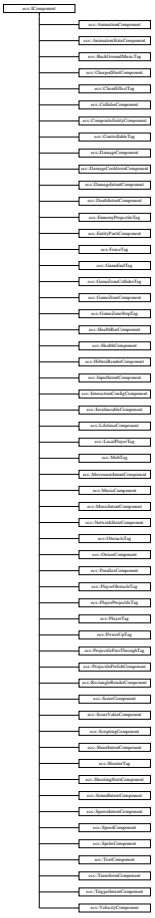
- virtual void **createBuffer** (size_t size)=0
- virtual void **deleteBuffer** ()=0
- virtual void **clear** ()=0
- virtual bool **writeBuffer** (const std::vector< uint64_t > &data, size_t size)=0
- virtual std::shared_ptr< std::vector< uint64_t > > **readBuffer** (size_t size)=0
- virtual size_t **getCapacity** () const =0
- virtual size_t **getUsedSize** () const =0
- virtual size_t **getAvailableSize** () const =0
- virtual bool **isEmpty** () const =0
- virtual bool **isFull** () const =0
- virtual std::vector< uint64_t > **getBuffer** () const =0

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/interfaces/IBuffer.hpp

4.96 ecs::IComponent Class Reference

Inheritance diagram for ecs::IComponent:

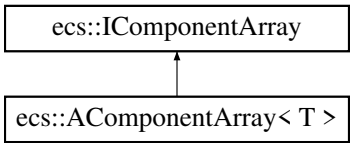


The documentation for this class was generated from the following file:

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/base/IComponent.hpp

4.97 ecs::IComponentArray Class Reference

Inheritance diagram for ecs::IComponentArray:



Public Member Functions

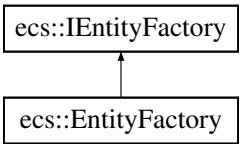
- virtual Entity **getMaxEntityId** () const =0
- virtual void **removeComponents** (Entity entityId)=0
- virtual void **removeOneComponent** (Entity entityId)=0
- virtual void **clear** ()=0

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/ECS/entity/componentArray/IComponentArray.hpp

4.98 ecs::IEntityFactory Class Reference

Inheritance diagram for ecs::IEntityFactory:



Public Member Functions

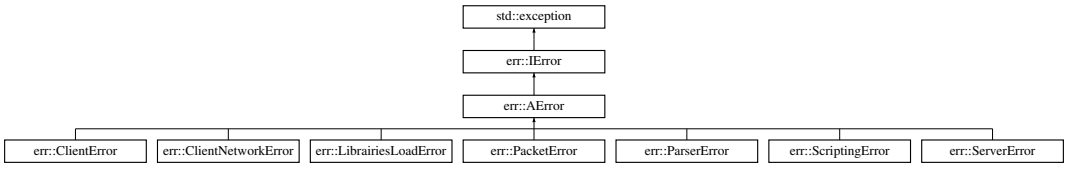
- virtual Entity **createEntity** (const std::shared_ptr< Registry > &registry, const EntityCreationContext &context=EntityCreationContext::forLocalClient())=0

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/ECS/entity/factory/IEntityFactory.hpp

4.99 err::IError Class Reference

Inheritance diagram for err::IError:



Public Member Functions

- virtual const char * **what** () const noexcept override=0
- virtual int **getCode** () const noexcept=0
- virtual std::string **getType** () const noexcept=0
- virtual std::string **getDetails** () const noexcept=0

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Error/IError.hpp

4.100 gfx::IEvent Class Reference

Public Types

- using **event_t** = EventType

Public Member Functions

- virtual void **init** ()=0
- virtual event_t **pollEvents** ()=0
- virtual std::string **getLastTextInput** ()=0
- virtual void **cleanup** ()=0
- virtual std::pair< int, int > **getMousePos** ()=0
- virtual bool **isKeyPressed** (event_t key)=0
- virtual bool **isMouseButtonPressed** (int button)=0
- virtual float **getAxisValue** (event_t axis)=0

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/interfaces/IEvent.hpp

4.101 net::IEventLoop Class Reference

Public Member Functions

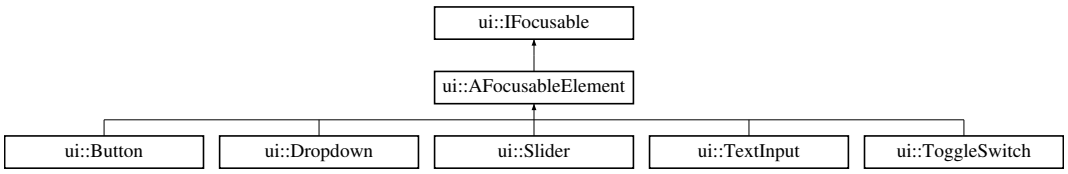
- virtual void **run** ()=0
- virtual void **runOne** ()=0
- virtual void **stop** ()=0
- virtual bool **stopped** () const =0
- virtual void **post** (std::function< void()> task)=0
- virtual void **restart** ()=0

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/interfaces/IEventLoop.hpp

4.102 ui::IFocusable Class Reference

Inheritance diagram for ui::IFocusable:



Public Member Functions

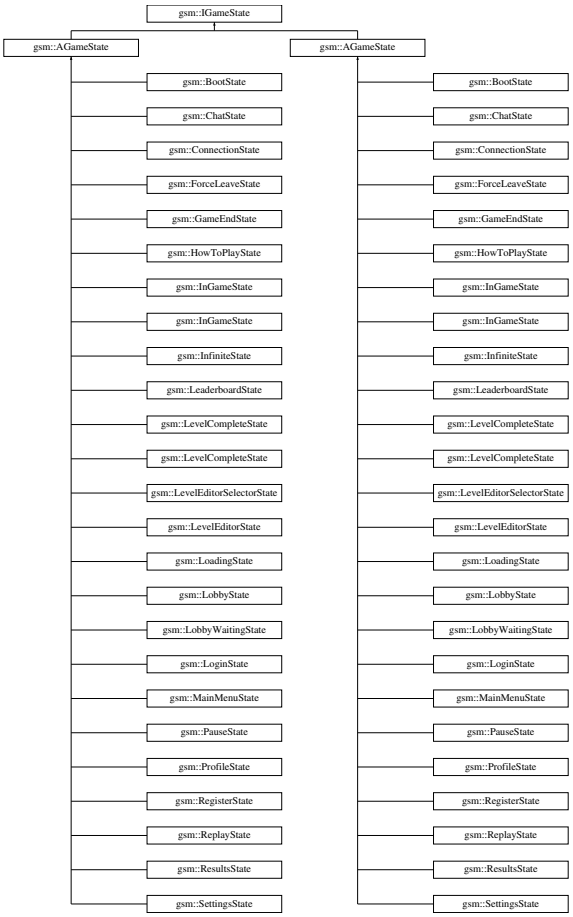
- virtual void **setFocused** (bool focused)=0
- virtual bool **isFocused** () const =0
- virtual bool **canBeFocused** () const =0
- virtual void **onFocusGained** ()=0
- virtual void **onFocusLost** ()=0
- virtual void **onActivated** ()=0
- virtual bool **onNavigateLeft** ()
- virtual bool **onNavigateRight** ()

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/core/IFocusable.hpp

4.103 gsm::IGameState Class Reference

Inheritance diagram for gsm::IGameState:



Public Member Functions

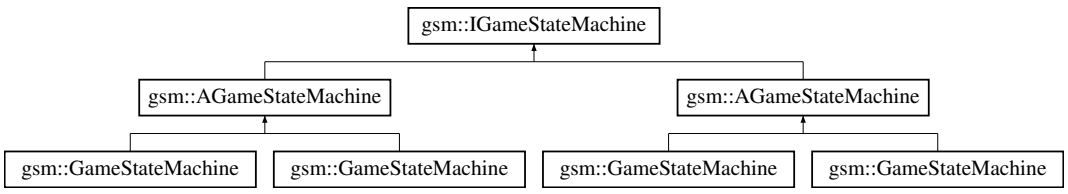
- virtual void **enter** ()=0
- virtual void **update** (float deltaTime)=0
- virtual void **exit** ()=0
- virtual void **addSystem** (std::shared_ptr< ecs::ISystem > system)=0
- virtual std::vector< std::shared_ptr< ecs::ISystem > > **getSystems** () const =0
- virtual std::string **getStateName** () const =0

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/gsm/IGameState.hpp

4.104 gsm::IGameStateMachine Class Reference

Inheritance diagram for gsm::IGameStateMachine:



Public Member Functions

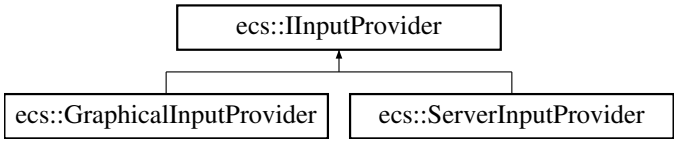
- virtual void **changeState** (std::shared_ptr< IGameState > newState)=0
- virtual void **pushState** (std::shared_ptr< IGameState > newState)=0
- virtual void **popState** ()=0
- virtual void **requestStateChange** (std::shared_ptr< IGameState > newState)=0
- virtual void **requestStatePush** (std::shared_ptr< IGameState > newState)=0
- virtual void **requestStatePop** ()=0
- virtual void **update** (float deltaTime)=0

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/gsm/IGameStateMachine.hpp

4.105 ecs::IInputProvider Class Reference

Inheritance diagram for ecs::IInputProvider:



Public Types

- using **event_t** = gfx::EventType

Public Member Functions

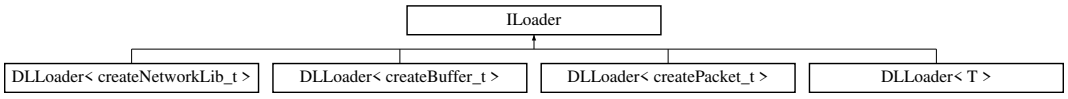
- virtual float **getAxisValue** (event_t axis, size_t clientID=0)=0
- virtual bool **isActionPressed** (InputAction action, size_t clientID=0)=0
- virtual float **getActionAxis** (InputAction action, size_t clientID=0)=0
- virtual InputMapping **getInputMapping** (size_t clientID=0) const =0

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/InputMapping/IInputProvider.hpp

4.106 ILoader Class Reference

Inheritance diagram for ILoader:



Public Member Functions

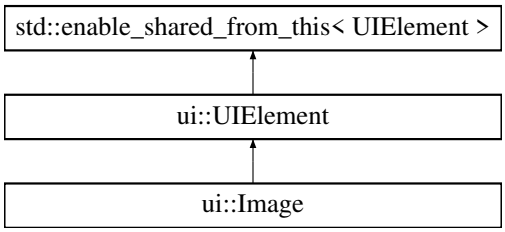
- virtual void * **Open** (const char *path, int flag)=0
- virtual void * **Symbol** (const char *symbolName)=0
- virtual int **Close** ()=0
- virtual const char * **Error** ()=0
- virtual void * **getHandler** () const =0

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/DLLoader/ILoader.hpp

4.107 ui::Image Class Reference

Inheritance diagram for ui::Image:



Public Member Functions

- **Image** (std::shared_ptr< ResourceManager > resourceManager)
- void **setTexturePath** (const std::string &path)
- void render () override
- void update (float deltaTime) override

Public Member Functions inherited from ui::UIElement

- **UIElement** (std::shared_ptr< ResourceManager > resourceManager)
- void **setPosition** (const math::Vector2f &position)
- void **setSize** (const math::Vector2f &size)
- math::Vector2f **getPosition** () const
- math::Vector2f **getSize** () const
- math::Vector2f **getAbsolutePosition** () const
- math::Vector2f **getAbsoluteSize** () const
- void **setVisible** (bool visible)
- bool **isVisible** () const
- void **setScalingEnabled** (bool enabled)
- bool **isScalingEnabled** () const
- void **setFocusEnabled** (bool enabled)
- bool **isFocusEnabled** () const
- void **setState** (UIState state)
- UIState **getState** () const
- virtual void **setScale** (UIScale scale)

- UIScale **getScale** () const
- void **setParent** (std::weak_ptr< UIElement > parent)
- std::shared_ptr< UIElement > **getParent** () const
- void **addChild** (std::shared_ptr< UIElement > child)
- void **removeChild** (std::shared_ptr< UIElement > child)
- const std::vector< std::shared_ptr< UIElement > > & **getChildren** () const
- virtual void **handleInput** (const math::Vector2f &mousePos, bool mousePressed)
- virtual bool **containsPoint** (const math::Vector2f &point) const
- void **setOnClick** (std::function< void()> callback)
- void **setOnHover** (std::function< void()> callback)
- void **setOnRelease** (std::function< void()> callback)

Private Attributes

- std::string **_texturePath**

Additional Inherited Members

Protected Member Functions inherited from ui::UIElement

- std::pair< int, int > **getWindowSize** () const
- std::pair< int, int > **getLogicalSize** () const
- float **getScaleFactor** () const

Protected Attributes inherited from ui::UIElement

- std::weak_ptr< ResourceManager > **_resourceManager**
- math::Vector2f **_position**
- math::Vector2f **_size**
- bool **_visible** = true
- bool **_scalingEnabled** = true
- bool **_focusEnabled** = true
- UIState **_state** = UIState::Normal
- UIScale **_scale** = UIScale::Normal
- std::weak_ptr< UIElement > **_parent**
- std::vector< std::shared_ptr< UIElement > > **_children**
- bool **_pressedInside** = false
- bool **_wasPressed** = false
- std::function< void()> **_onClick**
- std::function< void()> **_onHover**
- std::function< void()> **_onRelease**

4.107.1 Member Function Documentation

4.107.1.1 render()

```
void ui::Image::render ()  [override], [virtual]
```

Reimplemented from ui::UIElement.

4.107.1.2 update()

```
void ui::Image::update (
            float deltaTime)  [override], [virtual]
```

Reimplemented from ui::UIElement.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/elements/Image.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/elements/Image.cpp

4.108 net::INetwork Class Reference

Public Member Functions

- virtual void **init** (uint16_t port, const std::string host)=0
- virtual void **stop** ()=0
- virtual bool **sendTo** (const INetworkEndpoint &endpoint, std::vector< uint8_t > packet)=0
- virtual bool **broadcast** (const std::vector< std::shared_ptr< INetworkEndpoint > > &endpoints, const std↩ ::vector< uint8_t > &data)=0
- virtual bool **hasIncomingData** () const =0
- virtual std::vector< uint8_t > **receiveFrom** (const uint8_t &connectionId)=0
- virtual std::pair< std::shared_ptr< INetworkEndpoint >, std::vector< uint8_t > > **receiveAny** ()=0
- virtual void **setConnectionCallback** (std::function< void(int)> onConnect)=0
- virtual void **setDisconnectionCallback** (std::function< void(int)> onDisconnect)=0
- virtual ConnectionState **getConnectionState** () const =0
- virtual void **setConnectionState** (ConnectionState state)=0

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/interfaces/INetwork.hpp

4.109 net::INetworkAddress Class Reference

Public Member Functions

- virtual bool **isV4** () const =0
- virtual bool **isV6** () const =0
- virtual std::string **toString** () const =0
- virtual std::shared_ptr< INetworkAddress > **operator=** (const INetworkAddress &other)=0
- virtual std::shared_ptr< void > **getInternalAddress** ()=0
- virtual std::shared_ptr< const void > **getInternalAddress** () const =0
- virtual void **setFromInternal** (std::shared_ptr< void > internalAddr)=0

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/interfaces/INetworkAddress.hpp

4.110 net::INetworkEndpoint Class Reference

Public Member Functions

- virtual const std::string & **getAddress** () const =0
- virtual uint16_t **getPort** () const =0
- virtual void **setAddress** (const std::string &address)=0
- virtual void **setPort** (uint16_t port)=0
- virtual bool **operator==** (const INetworkEndpoint &other) const =0
- virtual bool **operator!=** (const INetworkEndpoint &other) const =0
- virtual bool **operator<** (const INetworkEndpoint &other) const =0

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/interfaces/INetworkEndpoint.hpp

4.111 net::INetworkErrorCode Class Reference

Public Member Functions

- virtual void **clear** ()=0
- virtual bool **hasError** () const =0
- virtual **operator bool** () const =0
- virtual std::string **message** () const =0
- virtual NetworkError **getError** () const =0
- virtual void **setError** (NetworkError error, const std::string &msg="")=0
- virtual bool **operator==** (NetworkError error) const =0
- virtual bool **operator!=** (NetworkError error) const =0
- virtual std::shared_ptr< void > **getInternalErrorCode** ()=0
- virtual std::shared_ptr< const void > **getInternalErrorCode** () const =0
- virtual void **setFromInternal** (std::shared_ptr< void > internalEc)=0

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/interfaces/INetworkErrorCode.hpp

4.112 net::INetworkFactory Class Reference

Public Member Functions

- virtual std::shared_ptr< IEventLoop > **createEventLoop** ()=0
- virtual std::shared_ptr< INetworkSocket > **createSocket** (std::shared_ptr< IEventLoop > eventLoop)=0
- virtual std::shared_ptr< INetworkResolver > **createResolver** (std::shared_ptr< IEventLoop > eventLoop)=0

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/interfaces/INetworkFactory.hpp

4.113 net::INetworkResolver Class Reference

Public Member Functions

- virtual std::vector< std::shared_ptr< INetworkEndpoint > > **resolve** (const std::string &host, const std::string &port, std::shared_ptr< INetworkErrorCode > ec)=0

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/interfaces/INetworkResolver.hpp

4.114 net::INetworkSocket Class Reference

Public Member Functions

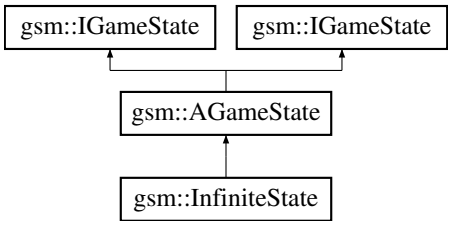
- virtual bool **open** (std::shared_ptr< INetworkErrorCode > ec)=0
- virtual bool **bind** (const INetworkEndpoint &endpoint, std::shared_ptr< INetworkErrorCode > ec)=0
- virtual std::size_t **sendTo** (const std::vector< uint8_t > &data, const INetworkEndpoint &endpoint, int flags, std::shared_ptr< INetworkErrorCode > ec)=0
- virtual std::size_t **receiveFrom** (std::shared_ptr< std::vector< uint8_t > > buffer, std::shared_ptr< INetworkEndpoint > sender, int flags, std::shared_ptr< INetworkErrorCode > ec)=0
- virtual bool **setNonBlocking** (bool nonBlocking, std::shared_ptr< INetworkErrorCode > ec)=0
- virtual bool **close** (std::shared_ptr< INetworkErrorCode > ec)=0
- virtual bool **isOpen** () const =0

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/interfaces/INetworkSocket.hpp

4.115 gsm::InfiniteState Class Reference

Inheritance diagram for gsm::InfiniteState:



Public Member Functions

- **InfiniteState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void enter () override
- void update (float deltaTime) override
- void exit () override
- std::string getStateName () const override

Public Member Functions inherited from gsm::AGameState

- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override
- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override

Additional Inherited Members

Protected Member Functions inherited from gsm::AGameState

- void addSystem (std::shared_ptr< ecs::ISystem > system) override
- void addSystem (std::shared_ptr< ecs::ISystem > system) override

Protected Attributes inherited from gsm::AGameState

- std::weak_ptr< IGameStateMachine > **_gsm**
- std::shared_ptr< ResourceManager > **_resourceManager**
- std::vector< std::shared_ptr< ecs::ISystem > > **_systems**

4.115.1 Member Function Documentation

4.115.1.1 enter()

```
void gsm::InfiniteState::enter ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.115.1.2 exit()

```
void gsm::InfiniteState::exit ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.115.1.3 getStateName()

```
std::string gsm::InfiniteState::getStateName () const  [inline], [override], [virtual]
```

Implements gsm::AGameState.

4.115.1.4 update()

```
void gsm::InfiniteState::update (
            float deltaTime)  [override], [virtual]
```

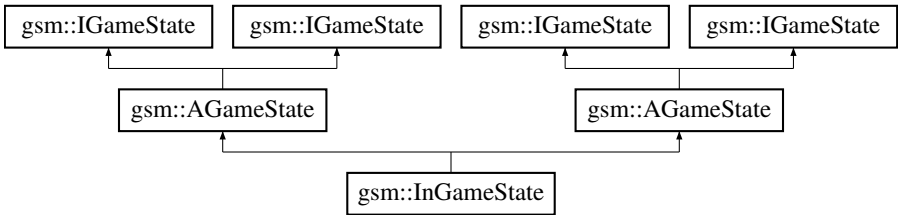
Reimplemented from gsm::AGameState.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/server/gsm/states/scenes/Infinite/InfiniteState.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/gsm/states/scenes/Infinite/InfiniteState.cpp

4.116 gsm::InGameState Class Reference

Inheritance diagram for gsm::InGameState:



Public Member Functions

- **InGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void enter () override
- void update (float deltaTime) override
- void exit () override
- std::string getStateName () const override
- **InGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void enter () override
- void exit () override
- void update (float deltaTime) override
- std::string getStateName () const override

Public Member Functions inherited from gsm::AGameState

- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override
- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override

Private Member Functions

- void **renderHUD** (float deltaTime)
- void **drawHealthHUD** (std::shared_ptr< gfx::IWindow > window, float health, float maxHealth)
- void **drawScoreHUD** (std::shared_ptr< gfx::IWindow > window, int score)
- void **drawShotChargeHUD** (std::shared_ptr< gfx::IWindow > window, float shotCharge, float maxShot↩ Charge)
- void **drawInGameMetrics** (std::shared_ptr< gfx::IWindow > window, float deltaTime)

Private Attributes

- std::shared_ptr< ecs::Registry > **_registry**
- std::shared_ptr< EntityPrefabManager > **_prefabManager**
- std::shared_ptr< Parser > **_parser**
- int **_previousScore**
- int **_previousHealth**
- std::vector< ScoreFeedback > **_scoreFeedbacks**
- std::vector< ScoreFeedback > **_healthFeedbacks**
- float **_whoAmITimer** = 0.0f
- bool **_localPlayerFound** = false

Additional Inherited Members

Protected Member Functions inherited from gsm::AGameState

- void addSystem (std::shared_ptr< ecs::ISystem > system) override
- void addSystem (std::shared_ptr< ecs::ISystem > system) override

Protected Attributes inherited from gsm::AGameState

- std::weak_ptr< IGameStateMachine > **_gsm**
- std::shared_ptr< ResourceManager > **_resourceManager**
- std::vector< std::shared_ptr< ecs::ISystem > > **_systems**

4.116.1 Member Function Documentation

4.116.1.1 enter() [1/2]

```
void gsm::InGameState::enter ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.116.1.2 enter() [2/2]

```
void gsm::InGameState::enter ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.116.1.3 exit() [1/2]

```
void gsm::InGameState::exit ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.116.1.4 exit() [2/2]

```
void gsm::InGameState::exit ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.116.1.5 getStateName() [1/2]

```
std::string gsm::InGameState::getStateName () const  [inline], [override], [virtual]
```

Implements gsm::AGameState.

4.116.1.6 getStateName() [2/2]

```
std::string gsm::InGameState::getStateName () const  [inline], [override], [virtual]
```

Implements gsm::AGameState.

4.116.1.7 update() [1/2]

```
void gsm::InGameState::update (
            float deltaTime)  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.116.1.8 update() [2/2]

```
void gsm::InGameState::update (
            float deltaTime)  [override], [virtual]
```

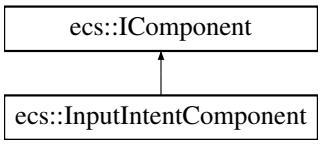
Reimplemented from gsm::AGameState.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/InGame/InGameState.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/gsm/states/scenes/InGame/InGameState.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/InGame/InGameState.cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/gsm/states/scenes/InGame/InGameState.cpp

4.117 ecs::InputIntentComponent Class Reference

Inheritance diagram for ecs::InputIntentComponent:



Public Member Functions

- **InputIntentComponent** (const math::Vector2f &direction=math::Vector2f(0.0f, 0.0f))
- math::Vector2f **getDirection** () const
- void **setDirection** (const math::Vector2f &direction)

Private Attributes

- math::Vector2f **_direction**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/temporary/InputIntentComponent.hpp

4.118 ecs::InputMapping Struct Reference

Public Member Functions

- std::map< InputAction, std::map< gfx::EventType, float > > **getAllMappings** () const

Public Attributes

- std::map< RemappableAction, RemappableKeyBinding > **remappableKeys**
- std::map< InputAction, std::map< gfx::EventType, float > > **fixedMappings**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/InputMapping/InputMapping.hpp

4.119 ecs::InputMappingManager Class Reference

Public Member Functions

- void **loadDefault** ()
- void **setMapping** (const InputMapping &mapping)
- const InputMapping & **getMapping** () const
- InputMapping & **getMutableMapping** ()
- gfx::EventType **getKeyForRemappableAction** (RemappableAction action, bool getPrimary=true) const
- void **remapKey** (RemappableAction action, gfx::EventType newKey, bool setPrimary)
- bool **isKeyboardKey** (gfx::EventType eventType)

Static Public Member Functions

- static std::string **eventTypeToString** (gfx::EventType eventType)
- static gfx::EventType **stringToEventType** (const std::string &str)
- static std::string **remappableActionToString** (RemappableAction action)
- static RemappableAction **stringToRemappableAction** (const std::string &str)

Private Attributes

- InputMapping **_mapping**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/InputMapping/InputMappingManager.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/InputMapping/InputMappingManager.cpp

4.120 ecs::InputNormalizer Class Reference

Static Public Member Functions

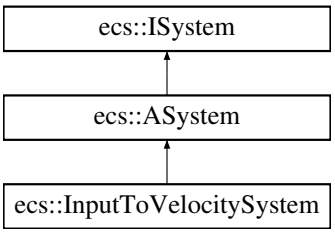
- static math::Vector2f **normalizeDirection** (const math::Vector2f &direction)
- static math::Vector2f **normalizeAnalogInput** (float rawX, float rawY)

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/input/InputNormalizer.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/input/InputNormalizer.cpp

4.121 ecs::InputToVelocitySystem Class Reference

Inheritance diagram for ecs::InputToVelocitySystem:



Public Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

4.121.1 Member Function Documentation

4.121.1.1 update()

```
void ecs::InputToVelocitySystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

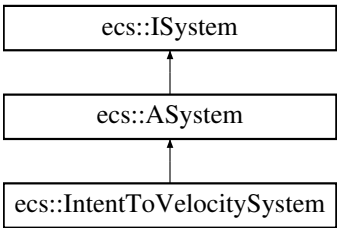
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/movement/InputToVelocitySystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/movement/InputToVelocitySystem.cpp

4.122 ecs::IntentToVelocitySystem Class Reference

Inheritance diagram for ecs::IntentToVelocitySystem:



Public Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

4.122.1 Member Function Documentation

4.122.1.1 update()

```
void ecs::IntentToVelocitySystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

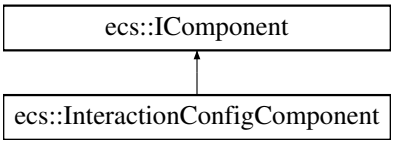
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/movement/IntentToVelocitySystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/movement/IntentToVelocitySystem.cpp

4.123 ecs::InteractionConfigComponent Class Reference

Inheritance diagram for ecs::InteractionConfigComponent:



Public Member Functions

- **InteractionConfigComponent** (const std::vector< InteractionMapping > &mappings)
- const std::vector< InteractionMapping > & **getMappings** () const
- void **setMappings** (const std::vector< InteractionMapping > &mappings)
- void **addMapping** (const InteractionMapping &mapping)

Private Attributes

- std::vector< InteractionMapping > **_mappings**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/permanent/InteractionConfigComponent.hpp

4.124 ecs::InteractionMapping Struct Reference

Public Attributes

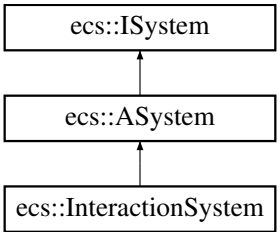
- std::vector< std::string > **targetTags**
- std::vector< std::string > **actionsToOther**
- std::vector< std::string > **actionsToSelf**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/permanent/InteractionConfigComponent.↩ hpp

4.125 ecs::InteractionSystem Class Reference

Inheritance diagram for ecs::InteractionSystem:



Public Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Private Member Functions

- std::vector< std::string > **filterDamageActions** (const std::vector< std::string > &actions, std::shared_ptr< Registry > registry, Entity targetEntity)

4.125.1 Member Function Documentation

4.125.1.1 update()

```
void ecs::InteractionSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

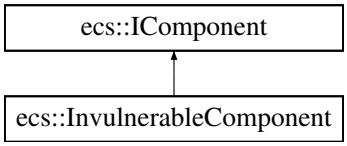
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/interactions/InteractionSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/interactions/InteractionSystem.cpp

4.126 ecs::InvulnerableComponent Class Reference

Inheritance diagram for ecs::InvulnerableComponent:



Public Member Functions

- **InvulnerableComponent** (bool active=false)
- bool **isActive** () const
- void **setActive** (bool active)

Private Attributes

- bool **_active**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/permanent/InvulnerableComponent.↩ hpp

4.127 pm::IPacketManager Class Reference

Public Member Functions

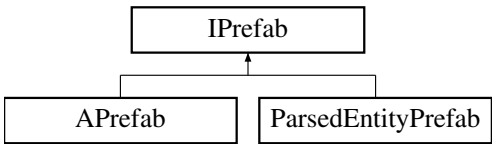
- virtual uint32_t **getLength** () const =0
- virtual uint32_t **getSequenceNumber** () const =0
- virtual uint8_t **getType** () const =0
- virtual std::vector< uint64_t > **getPayload** () const =0
- virtual std::vector< std::vector< uint64_t > > **getBatchedPayloads** () const =0
- virtual uint8_t **getIdClient** () const =0
- virtual void **setType** (uint8_t type)=0
- virtual void **setLength** (uint32_t length)=0
- virtual void **setSequenceNumber** (uint32_t sequenceNumber)=0
- virtual void **setPayload** (std::vector< uint64_t > payload)=0
- virtual void **setIdClient** (uint8_t idClient)=0
- virtual std::vector< uint64_t > **formatString** (const std::string str)=0
- virtual std::vector< uint8_t > **pack** (uint8_t idClient, uint32_t sequenceNumber, uint8_t type, std::vector< uint64_t > payload)=0
- virtual std::vector< uint8_t > **packBatchedGameState** (uint8_t idClient, uint32_t sequenceNumber, const std::vector< std::vector< uint64_t > > &entities)=0
- virtual bool **unpack** (std::vector< uint8_t > data)=0
- virtual void **reset** ()=0
- virtual void **registerBuilder** (uint8_t type, std::function< std::vector< uint8_t >(std::vector< uint64_t >)> builder)=0
- virtual void **registerParser** (uint8_t type, std::function< bool(const std::vector< uint8_t >)> parser)=0
- virtual void **registerLength** (uint8_t type, uint32_t length)=0
- virtual void **registerGameStatePackFunction** (std::function< std::vector< uint8_t >(std::vector< uint64_t >, std::shared_ptr< unsigned int >)> func)=0
- virtual void **registerGameStateUnpackFunction** (std::function< unsigned int(const std::vector< uint8_t >, unsigned int)> func)=0
- virtual void **clearAllHandlers** ()=0

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/interfaces/IPacketManager.hpp

4.128 IPrefab Class Reference

Inheritance diagram for IPrefab:



Public Member Functions

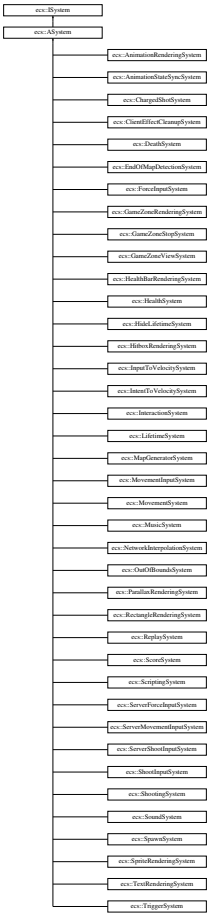
- virtual ecs::Entity **instantiate** (const std::shared_ptr< ecs::Registry > &registry, const std::shared_ptr< ecs::IEntityFactory > &factory, const ecs::EntityCreationContext &context=ecs::EntityCreationContext::for↩ LocalClient())=0
- virtual ecs::Entity **instantiate** (const std::shared_ptr< ecs::Registry > &registry)=0

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Prefab/IPrefab.hpp

4.129 ecs::ISystem Class Reference

Inheritance diagram for ecs::ISystem:



Public Member Functions

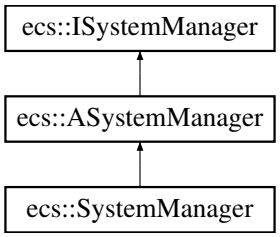
- virtual void **updateSystem** (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime)=0

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/base/ISystem.hpp

4.130 ecs::ISystemManager Class Reference

Inheritance diagram for ecs::ISystemManager:



Public Member Functions

- virtual void **updateAllSystems** (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime)=0
- virtual void **addSystem** (std::shared_ptr< ISystem > system)=0
- virtual void **removeSystem** (std::shared_ptr< ISystem > system)=0
- virtual void **clearAllSystems** ()=0

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/systemManager/ISystemManager.hpp

4.131 ecs::View< Components >::Iterator Class Reference

Public Member Functions

- **Iterator** (std::shared_ptr< Registry > registry, size_t entityId, size_t maxEntityId)
- bool **operator!=** (const Iterator &other) const
- Iterator & **operator++** ()
- size_t **operator*** () const
- bool **operator==** (const Iterator &other) const

Private Member Functions

- bool **hasAllComponents** () const

Private Attributes

- std::shared_ptr< Registry > **_registry**
- size_t **_entityId**
- size_t **_maxEntityId**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/ECS/view/View.hpp

4.132 gfx::IWindow Class Reference

Public Member Functions

- virtual void **init** ()=0
- virtual void **display** ()=0
- virtual void **closeWindow** ()=0
- virtual bool **isOpen** ()=0
- virtual void **clear** ()=0
- virtual void **resizeWindow** (size_t x, size_t y)=0
- virtual void **drawSprite** (std::string asset, color_t color, std::pair< size_t, size_t > position)=0
- virtual void **drawText** (std::string text, color_t color, std::pair< size_t, size_t > position, const std::string &fontPath, size_t fontSize=24, color_t outlineColor={0, 0, 0}, float outlineThickness=0.0f)=0
- virtual std::pair< size_t, size_t > **getTextSize** (const std::string &text, const std::string &fontPath, size_t fontSize=24)=0
- virtual void **drawRectangleOutline** (color_t color, std::pair< size_t, size_t > position, std::pair< size_t, size_t > size, size_t borderWidth=5)=0
- virtual void **drawFilledRectangle** (color_t color, std::pair< size_t, size_t > position, std::pair< size_t, size_t > size)=0
- virtual void **drawRoundedRectangleFilled** (color_t color, std::pair< size_t, size_t > position, std::pair< size_t, size_t > size, float radius)=0
- virtual void **drawRoundedRectangleOutline** (color_t color, std::pair< size_t, size_t > position, std::pair< size_t, size_t > size, float radius)=0
- virtual bool **isMouseOver** (std::pair< size_t, size_t > position, std::pair< size_t, size_t > size)=0
- virtual std::pair< int, int > **getWindowSize** ()=0
- virtual void **drawSprite** (const std::string &texturePath, float x, float y, float scaleX=1.0f, float scaleY=1.0f, float rotation=0.0f)=0
- virtual void **drawSprite** (const std::string &texturePath, float x, float y, const math::FRect frameRect, float scaleX=1.0f, float scaleY=1.0f, float rotation=0.0f)=0
- virtual void **drawSprite** (const std::string &texturePath, float x, float y, float scaleX, float scaleY, float rotation, color_t color)=0
- virtual void **drawSprite** (const std::string &texturePath, float x, float y, const math::FRect frameRect, float scaleX, float scaleY, float rotation, color_t color)=0
- virtual void **updateView** ()=0
- virtual void **setViewCenter** (float x, float y)=0
- virtual math::Vector2f **getViewCenter** ()=0
- virtual math::Vector2f **mapPixelToCoords** (int x, int y)=0
- virtual std::pair< int, int > **getLogicalSize** () const =0
- virtual float **getScaleFactor** () const =0
- virtual void **addShaderFilter** (const std::string &path)=0
- virtual void **removeShaderFilter** (const std::string &path)=0
- virtual void **setShaderUniform** (const std::string &filterPath, const std::string &name, float value)=0
- virtual void **setFramerateLimit** (unsigned int fps)=0
- virtual void **setFullscreen** (bool fullscreen)=0
- virtual void **setRenderQuality** (float quality)=0
- virtual void **setCursor** (bool isHand)=0
- virtual std::string **getClipboardText** ()=0
- virtual void **setClipboardText** (const std::string &text)=0

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/interfaces/IWindow.hpp

4.133 parser::JsonLoader Class Reference

Classes

- struct LoadResult

Static Public Member Functions

- static nlohmann::json **loadFromFile** (const std::string &filePath)
- static LoadResult **tryLoadFromFile** (const std::string &filePath) noexcept
- static nlohmann::json **parseFromString** (const std::string &jsonString, const std::string &source↩ Name="string")
- static LoadResult **tryParseFromString** (const std::string &jsonString) noexcept
- static bool **fileExists** (const std::string &filePath)
- static std::string **getExtension** (const std::string &filePath)

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/Utils/JsonLoader.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/Utils/JsonLoader.cpp

4.134 parser::JsonValidation Class Reference

Static Public Member Functions

- static ValidationResult **hasRequiredFields** (const nlohmann::json &json, const std::vector< std::string > &requiredFields, const std::string &contextName="")
- static bool **hasFieldOfType** (const nlohmann::json &json, const std::string &fieldName, const std::string &expectedType)
- template<typename T>
  static T **getOrDefault** (const nlohmann::json &json, const std::string &fieldName, const T &defaultValue)
- static std::optional< nlohmann::json > **getNestedField** (const nlohmann::json &json, const std::string &path)

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/Utils/JsonValidation.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/Utils/JsonValidation.cpp

4.135 ui::Background::Layer Struct Reference

Public Attributes

- std::string **texturePath**
- float **speedX**
- float **speedY**
- math::Vector2f **sourceSize**
- float **offsetX** = 0.0f
- float **offsetY** = 0.0f

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/elements/Background.hpp

4.136 ui::LayoutConfig Struct Reference

Classes

- struct BackgroundConfig

Public Attributes

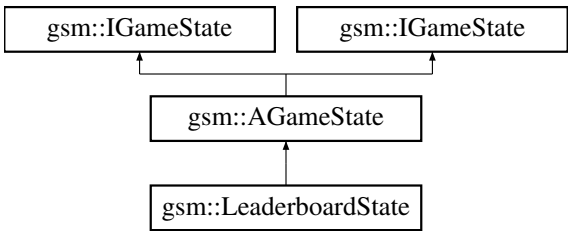
- LayoutDirection **direction** = LayoutDirection::Vertical
- LayoutAlignment **alignment** = LayoutAlignment::Start
- float **spacing** = 0.0f
- math::Vector2f **padding** = math::Vector2f(0.0f, 0.0f)
- math::Vector2f **offset** = math::Vector2f(0.0f, 0.0f)
- bool **autoResize** = false
- AnchorX **anchorX** = AnchorX::None
- AnchorY **anchorY** = AnchorY::None
- struct ui::LayoutConfig::BackgroundConfig **background**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/core/UILayout.hpp

4.137 gsm::LeaderboardState Class Reference

Inheritance diagram for gsm::LeaderboardState:



Public Member Functions

- **LeaderboardState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void enter () override
- void update (float deltaTime) override
- void exit () override
- std::string getStateName () const override

Public Member Functions inherited from gsm::AGameState

- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override
- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override

Private Member Functions

- void **loadLeaderboardData** ()
- void **renderUI** ()

Private Attributes

- std::unique_ptr< MouseInputHandler > **_mouseHandler**
- std::unique_ptr< ui::UIManager > **_uiManager**
- std::shared_ptr< ui::Background > **_background**
- std::shared_ptr< ui::UILayout > **_mainLayout**
- std::shared_ptr< ui::Text > **_titleText**
- std::vector< std::shared_ptr< ui::Text > > **_leaderTexts**
- std::shared_ptr< ui::Button > **_backButton**

Additional Inherited Members

Protected Member Functions inherited from gsm::AGameState

- void addSystem (std::shared_ptr< ecs::ISystem > system) override
- void addSystem (std::shared_ptr< ecs::ISystem > system) override

Protected Attributes inherited from gsm::AGameState

- std::weak_ptr< IGameStateMachine > **_gsm**
- std::shared_ptr< ResourceManager > **_resourceManager**
- std::vector< std::shared_ptr< ecs::ISystem > > **_systems**

4.137.1 Member Function Documentation

4.137.1.1 enter()

```
void gsm::LeaderboardState::enter ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.137.1.2 exit()

```
void gsm::LeaderboardState::exit ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.137.1.3 getStateName()

```
std::string gsm::LeaderboardState::getStateName () const  [inline], [override], [virtual]
```

Implements gsm::AGameState.

4.137.1.4 update()

```
void gsm::LeaderboardState::update (
            float deltaTime)  [override], [virtual]
```

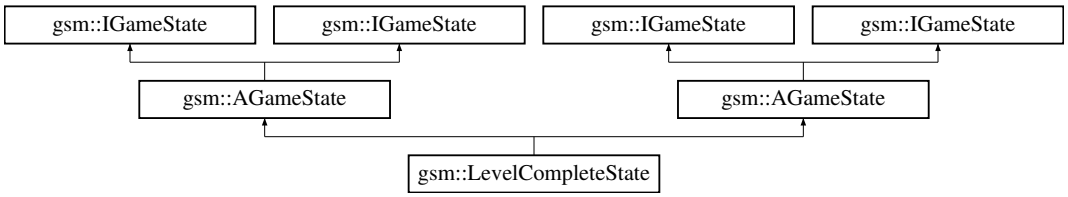
Reimplemented from gsm::AGameState.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/Leaderboard/LeaderboardState.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/Leaderboard/LeaderboardState.cpp

4.138 gsm::LevelCompleteState Class Reference

Inheritance diagram for gsm::LevelCompleteState:



Public Member Functions

- **LevelCompleteState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void enter () override
- void update (float deltaTime) override
- void exit () override
- std::string getStateName () const override
- void **onNextLevel** ()
- **LevelCompleteState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void enter () override
- void update (float deltaTime) override
- std::string getStateName () const override

Public Member Functions inherited from gsm::AGameState

- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override
- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override

Private Attributes

- std::unique_ptr< ui::UIManager > **_uiManager**
- std::shared_ptr< ui::Text > **_subtitleText**
- bool **_waitingForNextLevel**
- std::shared_ptr< ui::SpritePreview > **_victoryAnimation**
- float **_transitionTimer**
- std::vector< int > **_savedPlayerScores**

Static Private Attributes

- static constexpr float **TRANSITION_DELAY** = 2.0f

Additional Inherited Members

Protected Member Functions inherited from gsm::AGameState

- void addSystem (std::shared_ptr< ecs::ISystem > system) override
- void addSystem (std::shared_ptr< ecs::ISystem > system) override

Protected Attributes inherited from gsm::AGameState

- std::weak_ptr< IGameStateMachine > **_gsm**
- std::shared_ptr< ResourceManager > **_resourceManager**
- std::vector< std::shared_ptr< ecs::ISystem > > **_systems**

4.138.1 Member Function Documentation

4.138.1.1 enter() [1/2]

```
void gsm::LevelCompleteState::enter ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.138.1.2 enter() [2/2]

```
void gsm::LevelCompleteState::enter ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.138.1.3 exit()

```
void gsm::LevelCompleteState::exit ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.138.1.4 getStateName() [1/2]

```
std::string gsm::LevelCompleteState::getStateName () const  [inline], [override], [virtual]
```

Implements gsm::AGameState.

4.138.1.5 getStateName() [2/2]

```
std::string gsm::LevelCompleteState::getStateName () const  [inline], [override], [virtual]
```

Implements gsm::AGameState.

4.138.1.6 update() [1/2]

```
void gsm::LevelCompleteState::update (
            float deltaTime)  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.138.1.7 update() [2/2]

```
void gsm::LevelCompleteState::update (
            float deltaTime)  [override], [virtual]
```

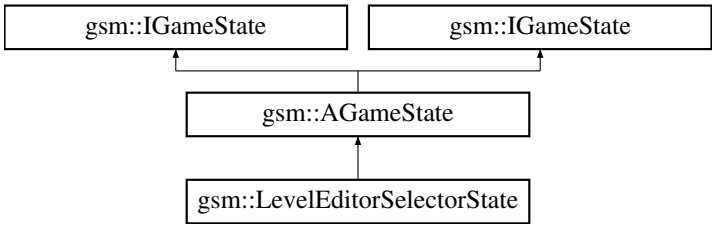
Reimplemented from gsm::AGameState.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelComplete/LevelComplete↩ State.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/gsm/states/scenes/LevelComplete/LevelComplete↩ State.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelComplete/LevelComplete↩ State.cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/gsm/states/scenes/LevelComplete/LevelComplete↩ State.cpp

4.139 gsm::LevelEditorSelectorState Class Reference

Inheritance diagram for gsm::LevelEditorSelectorState:



Public Member Functions

- **LevelEditorSelectorState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void enter () override
- void update (float deltaTime) override
- void exit () override
- std::string getStateName () const override

Public Member Functions inherited from gsm::AGameState

- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override
- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override

Private Member Functions

- void **renderUI** ()
- void **createLevelSelectionUI** ()
- std::vector< std::pair< std::filesystem::path, int > > **getAvailableLevels** ()
- std::optional< std::filesystem::path > **createNewLevel** ()
- void **swapLevels** (const std::filesystem::path &path1, const std::filesystem::path &path2)
- void **showDeleteConfirmation** (const std::filesystem::path &levelPath, const std::string &levelName)
- void **showDeleteConfirmationPopup** (const std::filesystem::path &levelPath, const std::string &levelName)
- void **hideDeleteConfirmationPopup** ()
- void **confirmDelete** ()
- void **showDuplicateConfirmation** (const std::filesystem::path &levelPath, const std::string &levelName)
- void **showDuplicateConfirmationPopup** (const std::filesystem::path &levelPath, const std::string &levelName)
- void **hideDuplicateConfirmationPopup** ()
- void **confirmDuplicate** ()
- void **setMainButtonsEnabled** (bool enabled)

Private Attributes

- std::unique_ptr< MouseInputHandler > **_mouseHandler**
- std::unique_ptr< ui::UIManager > **_uiManager**
- std::shared_ptr< ui::Background > **_background**
- std::shared_ptr< ui::Button > **_backButton**
- std::vector< std::shared_ptr< ui::Button > > **_levelButtons**
- std::vector< std::shared_ptr< ui::Text > > **_indexLabels**
- std::shared_ptr< ui::Button > **_addLevelButton**
- std::vector< std::shared_ptr< ui::Button > > **_upButtons**
- std::vector< std::shared_ptr< ui::Button > > **_downButtons**
- std::vector< std::shared_ptr< ui::Button > > **_duplicateButtons**
- std::vector< std::shared_ptr< ui::Button > > **_deleteButtons**
- std::shared_ptr< ui::Button > **_prevButton**
- std::shared_ptr< ui::Button > **_nextButton**
- bool **_shouldUpdateUI**

- size_t **_lastLevelCount**
- int **_currentPage**
- std::shared_ptr< ui::UILayout > **_deletePopupOverlay**
- std::shared_ptr< ui::UILayout > **_deletePopupLayout**
- std::shared_ptr< ui::Text > **_deletePopupText**
- std::shared_ptr< ui::Button > **_deleteCancelButton**
- std::shared_ptr< ui::Button > **_deleteConfirmButton**
- std::filesystem::path **_pendingDeletePath**
- bool **_shouldHideDeletePopup**
- std::shared_ptr< ui::UILayout > **_duplicatePopupOverlay**
- std::shared_ptr< ui::UILayout > **_duplicatePopupLayout**
- std::shared_ptr< ui::Text > **_duplicatePopupText**
- std::shared_ptr< ui::Button > **_duplicateCancelButton**
- std::shared_ptr< ui::Button > **_duplicateConfirmButton**
- std::filesystem::path **_pendingDuplicatePath**
- std::string **_pendingDuplicateName**
- bool **_shouldHideDuplicatePopup**

Static Private Attributes

- static constexpr int **_levelsPerPage** = 8

Additional Inherited Members

Protected Member Functions inherited from gsm::AGameState

- void addSystem (std::shared_ptr< ecs::ISystem > system) override
- void addSystem (std::shared_ptr< ecs::ISystem > system) override

Protected Attributes inherited from gsm::AGameState

- std::weak_ptr< IGameStateMachine > **_gsm**
- std::shared_ptr< ResourceManager > **_resourceManager**
- std::vector< std::shared_ptr< ecs::ISystem > > **_systems**

4.139.1 Member Function Documentation

4.139.1.1 enter()

```
void gsm::LevelEditorSelectorState::enter ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.139.1.2 exit()

```
void gsm::LevelEditorSelectorState::exit ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.139.1.3 getStateName()

```
std::string gsm::LevelEditorSelectorState::getStateName () const  [inline], [override], [virtual]
```

Implements gsm::AGameState.

4.139.1.4 update()

```
void gsm::LevelEditorSelectorState::update (
            float deltaTime)  [override], [virtual]
```

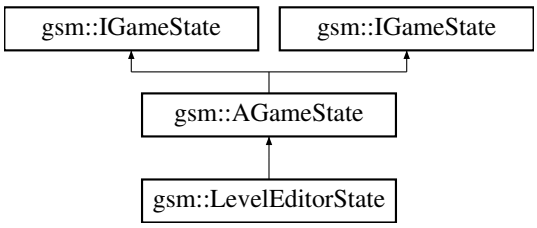
Reimplemented from gsm::AGameState.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditorSelector/LevelEditor↩ SelectorState.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditorSelector/LevelEditor↩ SelectorState.cpp

4.140 gsm::LevelEditorState Class Reference

Inheritance diagram for gsm::LevelEditorState:



Classes

- struct WorldCoordinates

Public Member Functions

- **LevelEditorState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager, std::optional< std::filesystem::path > levelPath=std::nullopt)
- void enter () override
- void update (float deltaTime) override
- void exit () override
- std::string getStateName () const override

Public Member Functions inherited from gsm::AGameState

- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override
- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override

Private Member Functions

- void **renderUI** ()
- void **createUI** ()
- void **createBottomPanel** ()
- void **createBottomPanelBase** ()
- void **createEditorModeDropdown** ()
- void **createObstacleUI** ()
- void **createPowerUpUI** ()
- void **createWaveUI** ()
- void **createEnemyUI** ()
- void **updateSaveButtonText** ()
- bool **validateFields** ()
- std::vector< std::string > **loadAvailableMusics** ()
- std::vector< std::string > **loadAvailableBackgrounds** ()
- std::vector< std::string > **loadAvailableObstacles** ()
- std::vector< std::string > **loadAvailablePowerUps** ()
- std::vector< std::string > **loadAvailableEnemies** ()
- void **saveToHistory** ()
- void **loadFromHistory** (size_t index)
- void **updateHistoryButtons** ()
- void **initializeViewport** ()
- void **handleZoom** (float deltaTime, gfx::EventType eventResult)
- void **handleCanvasDrag** (float deltaTime)
- void **renderLevelPreview** ()
- void **renderSpriteInLevelPreview** (const LevelPreviewSprite &spriteData, const std::string &prefabName, float screenX, float screenY, float canvasLeft, float canvasRight, float canvasTop, float canvasBottom)
- LevelPreviewSprite **extractSpriteDataFromPrefab** (const std::string &prefabPath)
- void **hideAllUIElements** ()
- void **hideObstacleUI** ()
- void **hidePowerUpUI** ()
- void **hideWaveUI** ()
- void **hideEnemyUI** ()
- void **showObstacleUI** (bool showCount)
- void **showPowerUpUI** ()
- void **showWaveUI** ()
- WorldCoordinates **extractWorldCoordinates** (const math::Vector2f &mousePos, float sidePanelWidth) const
- void **parseObstacles** ()
- void **renderAllObstacles** (float levelX, float levelY, float canvasLeft, float canvasRight, float canvasTop, float canvasBottom)
- void **saveObstacles** ()
- void **handleObstacleClick** (float mouseX, float mouseY, float levelX, float levelY)
- void **startObstacleDrag** (math::Vector2f mousePos, float viewportZoom, float sidePanelWidth)
- void **handleObstacleDrag** (math::Vector2f mousePos, float viewportZoom, float sidePanelWidth)

- std::optional< ObstacleSelection > **getObstacleAtPosition** (float mouseX, float mouseY, float levelX, float levelY)
- void **parsePowerUps** ()
- void **renderAllPowerUps** (float levelX, float levelY, float canvasLeft, float canvasRight, float canvasTop, float canvasBottom)
- void **savePowerUps** ()
- void **handlePowerUpClick** (float mouseX, float mouseY, float levelX, float levelY)
- void **startPowerUpDrag** (math::Vector2f mousePos, float viewportZoom, float sidePanelWidth)
- void **handlePowerUpDrag** (math::Vector2f mousePos, float viewportZoom, float sidePanelWidth)
- std::optional< PowerUpSelection > **getPowerUpAtPosition** (float mouseX, float mouseY, float levelX, float levelY)
- void **parseWaves** ()
- void **renderAllWaves** (float levelX, float levelY, float canvasLeft, float canvasRight, float canvasTop, float canvasBottom)
- void **saveWaves** ()
- void **handleWaveClick** (float mouseX, float mouseY, float levelX, float levelY)
- void **startWaveDrag** (math::Vector2f mousePos, float viewportZoom, float sidePanelWidth)
- void **handleWaveDrag** (math::Vector2f mousePos, float viewportZoom, float sidePanelWidth)
- std::optional< WaveSelection > **getWaveAtPosition** (float mouseX, float mouseY, float levelX, float levelY)
- void **updateEnemyUI** ()

Private Attributes

- std::unique_ptr< MouseInputHandler > **_mouseHandler**
- std::unique_ptr< ui::UIManager > **_uiManager**
- std::shared_ptr< ui::Background > **_background**
- std::shared_ptr< ui::Panel > **_sidePanel**
- std::shared_ptr< ui::Panel > **_bottomPanel**
- std::shared_ptr< ui::Panel > **_canvasPanel**
- std::shared_ptr< ui::Dropdown > **_editorModeDropdown**
- std::shared_ptr< ui::Text > **_obstaclePrefabLabel**
- std::shared_ptr< ui::Dropdown > **_obstaclePrefabDropdown**
- std::shared_ptr< ui::Text > **_powerUpPrefabLabel**
- std::shared_ptr< ui::Dropdown > **_powerUpPrefabDropdown**
- std::shared_ptr< ui::Panel > **_spritePreviewPanel**
- std::shared_ptr< ui::SpritePreview > **_spritePreview**
- std::shared_ptr< ui::Text > **_spriteWidthLabel**
- std::shared_ptr< ui::Text > **_spriteHeightLabel**
- std::shared_ptr< ui::Button > **_saveButton**
- std::shared_ptr< ui::Button > **_backButton**
- std::shared_ptr< ui::Text > **_nameLabel**
- std::shared_ptr< ui::TextInput > **_levelNameInput**
- std::shared_ptr< ui::Text > **_mapLengthLabel**
- std::shared_ptr< ui::TextInput > **_mapLengthInput**
- std::shared_ptr< ui::Text > **_scrollSpeedLabel**
- std::shared_ptr< ui::TextInput > **_scrollSpeedInput**
- std::shared_ptr< ui::Text > **_musicLabel**
- std::shared_ptr< ui::Dropdown > **_musicDropdown**
- std::shared_ptr< ui::Text > **_backgroundLabel**
- std::shared_ptr< ui::Dropdown > **_backgroundDropdown**
- std::shared_ptr< ui::Button > **_undoButton**
- std::shared_ptr< ui::Button > **_redoButton**
- std::shared_ptr< ui::Text > **_cursorPosLabel**
- std::shared_ptr< ui::Text > **_cursorPosYLabel**

- std::shared_ptr< ui::Button > **_resetViewButton**
- std::shared_ptr< ui::Button > **_filterButton**
- std::shared_ptr< ui::Button > **_showHitboxesButton**
- std::shared_ptr< ui::Text > **_obstacleTypeLabel**
- std::shared_ptr< ui::Dropdown > **_obstacleTypeDropdown**
- std::shared_ptr< ui::Text > **_obstaclePosXLabel**
- std::shared_ptr< ui::TextInput > **_obstaclePosXInput**
- std::shared_ptr< ui::Text > **_obstaclePosYLabel**
- std::shared_ptr< ui::TextInput > **_obstaclePosYInput**
- std::shared_ptr< ui::Text > **_obstacleCountLabel**
- std::shared_ptr< ui::TextInput > **_obstacleCountInput**
- std::shared_ptr< ui::Button > **_obstacleDeleteButton**
- std::shared_ptr< ui::Button > **_obstacleDuplicateButton**
- std::shared_ptr< ui::Text > **_powerUpPosXLabel**
- std::shared_ptr< ui::TextInput > **_powerUpPosXInput**
- std::shared_ptr< ui::Text > **_powerUpPosYLabel**
- std::shared_ptr< ui::TextInput > **_powerUpPosYInput**
- std::shared_ptr< ui::Button > **_powerUpDeleteButton**
- std::shared_ptr< ui::Button > **_powerUpDuplicateButton**
- std::shared_ptr< ui::Text > **_waveLabel**
- std::shared_ptr< ui::Text > **_waveIndexLabel**
- std::shared_ptr< ui::Button > **_wavePrevButton**
- std::shared_ptr< ui::Button > **_waveNextButton**
- std::shared_ptr< ui::Button > **_waveDeleteButton**
- std::shared_ptr< ui::Button > **_waveDuplicateButton**
- std::shared_ptr< ui::Text > **_waveTriggerXLabel**
- std::shared_ptr< ui::TextInput > **_waveTriggerXInput**
- std::shared_ptr< ui::Text > **_enemyLabel**
- std::shared_ptr< ui::Text > **_enemyIndexLabel**
- std::shared_ptr< ui::Button > **_enemyPrevButton**
- std::shared_ptr< ui::Button > **_enemyNextButton**
- std::shared_ptr< ui::Button > **_enemyAddButton**
- std::shared_ptr< ui::Button > **_enemyDeleteButton**
- std::shared_ptr< ui::Text > **_enemyTypeLabel**
- std::shared_ptr< ui::TextInput > **_enemyTypeInput**
- std::shared_ptr< ui::Button > **_enemyApplyTypeButton**
- std::shared_ptr< ui::Text > **_enemyAppliedTypeLabel**
- std::shared_ptr< ui::Text > **_enemyCountLabel**
- std::shared_ptr< ui::TextInput > **_enemyCountInput**
- std::shared_ptr< ui::Text > **_enemyDistXMinLabel**
- std::shared_ptr< ui::TextInput > **_enemyDistXMinInput**
- std::shared_ptr< ui::Text > **_enemyDistXMaxLabel**
- std::shared_ptr< ui::TextInput > **_enemyDistXMaxInput**
- std::shared_ptr< ui::Text > **_enemyDistYMinLabel**
- std::shared_ptr< ui::TextInput > **_enemyDistYMinInput**
- std::shared_ptr< ui::Text > **_enemyDistYMaxLabel**
- std::shared_ptr< ui::TextInput > **_enemyDistYMaxInput**
- std::shared_ptr< ui::Text > **_enemyDistXTypeLabel**
- std::shared_ptr< ui::Dropdown > **_enemyDistXTypeDropdown**
- std::shared_ptr< ui::Text > **_enemyDistYTypeLabel**
- std::shared_ptr< ui::Dropdown > **_enemyDistYTypeDropdown**
- bool **_hasUnsavedChanges** = false
- bool **_showHitboxes** = false
- std::string **_displayFilter** = "All"
- std::optional< std::filesystem::path > **_levelPath**

- nlohmann::json **_levelData**
- std::vector< nlohmann::json > **_history**
- size_t **_currentHistoryIndex** = 0
- float **_lastChangeTime** = constants::CHANGE_DEBOUNCE_TIME + 1.0f
- float **_currentDebounceTime** = constants::CHANGE_DEBOUNCE_TIME
- bool **_hasPendingChange** = false
- bool **_isLoadingFromHistory** = false
- bool **_isSelectingObject** = false
- bool **_undoPressedLastFrame** = false
- bool **_redoPressedLastFrame** = false
- bool **_leftMousePressedLastFrame** = false
- math::Vector2f **_viewportOffset**
- float **_viewportZoom** = 1.0f
- float **_minZoom** = 0.1f
- float **_maxZoom** = 2.0f
- bool **_isDragging** = false
- math::Vector2f **_lastMousePos**
- math::Vector2f **_dragStartPos**
- bool **_isDraggingObstacle** = false
- math::Vector2f **_dragObstacleOffset**
- bool **_isDraggingPowerUp** = false
- math::Vector2f **_dragPowerUpOffset**
- bool **_isDraggingWave** = false
- float **_dragWaveOffsetX** = 0.0f
- std::map< std::string, ObstacleGroup > **_obstaclesByName**
- std::optional< ObstacleSelection > **_selectedObstacle**
- std::map< std::string, std::vector< PowerUpData > > **_powerUpsByName**
- std::optional< PowerUpSelection > **_selectedPowerUp**
- std::vector< Wave > **_waves**
- std::optional< WaveSelection > **_selectedWave**
- int **_currentWaveIndex** = 0
- int **_currentEnemyIndex** = 0
- std::vector< std::string > **_availableEnemies**
- std::map< std::string, LevelPreviewSprite > **_obstacleAnimationData**
- std::map< std::string, float > **_obstacleAnimationFrames**
- std::map< std::string, float > **_obstacleAnimationTimes**
- std::map< std::string, LevelPreviewSprite > **_powerUpAnimationData**
- std::map< std::string, float > **_powerUpAnimationFrames**
- std::map< std::string, float > **_powerUpAnimationTimes**
- std::optional< ObstacleSelection > **_clipboardObstacle**
- std::optional< PowerUpSelection > **_clipboardPowerUp**
- std::optional< WaveSelection > **_clipboardWave**
- bool **_copyPressedLastFrame** = false
- bool **_pastePressedLastFrame** = false

Additional Inherited Members

Protected Member Functions inherited from gsm::AGameState

- void addSystem (std::shared_ptr< ecs::ISystem > system) override
- void addSystem (std::shared_ptr< ecs::ISystem > system) override

Protected Attributes inherited from gsm::AGameState

- std::weak_ptr< IGameStateMachine > **_gsm**
- std::shared_ptr< ResourceManager > **_resourceManager**
- std::vector< std::shared_ptr< ecs::ISystem > > **_systems**

4.140.1 Member Function Documentation

4.140.1.1 enter()

```
void gsm::LevelEditorState::enter ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.140.1.2 exit()

```
void gsm::LevelEditorState::exit ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.140.1.3 getStateName()

```
std::string gsm::LevelEditorState::getStateName () const  [inline], [override], [virtual]
```

Implements gsm::AGameState.

4.140.1.4 update()

```
void gsm::LevelEditorState::update (
            float deltaTime)  [override], [virtual]
```

Reimplemented from gsm::AGameState.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditor/LevelEditorState.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditor/LevelEditorHistory.cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditor/LevelEditorObstacles.↩ cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditor/LevelEditorPowerup.cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditor/LevelEditorRender.cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditor/LevelEditorState.cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditor/LevelEditorUtils.cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditor/LevelEditorWaves.cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditor/UI/LevelEditorUI.cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditor/UI/LevelEditorUIEnemies.↩ cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditor/UI/LevelEditorUIObstacles.↩ cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditor/UI/LevelEditorUIPanel.↩ cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditor/UI/LevelEditorUIPower↩ Ups.cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditor/UI/LevelEditorUIWaves.↩ cpp

4.141 gsm::LevelPreviewSprite Struct Reference

Public Attributes

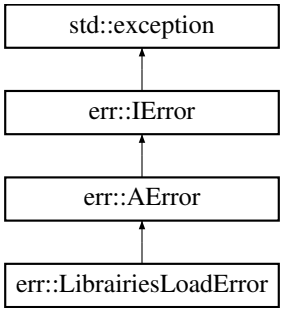
- std::string **texturePath**
- float **width**
- float **height**
- float **posX**
- float **posY**
- float **scale**
- float **rotation**
- bool **isAnimation** = false
- float **frameCount** = 0.0f
- float **frameWidth** = 0.0f
- float **frameHeight** = 0.0f
- float **animationSpeed** = 0.1f
- bool **animationLoop** = true

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditor/LevelEditorState.hpp

4.142 err::LibrairiesLoadError Class Reference

Inheritance diagram for err::LibrairiesLoadError:



Public Types

- enum **ErrorCode** { **UNKNOWN** = 1000 , **LIBRARY_NOT_FOUND** = 1001 , **SYMBOL_NOT_FOUND** = 1002 }

Public Member Functions

- **LibrairiesLoadError** (const std::string &message, ErrorCode code=UNKNOWN)
- std::string getType () const noexcept override

Public Member Functions inherited from err::AError

- **AError** (const std::string &message, int code=0)
- const char * what () const noexcept override
- int getCode () const noexcept override
- std::string getDetails () const noexcept override

Additional Inherited Members

Protected Attributes inherited from err::AError

- std::string **m_message**
- int **m_code**

4.142.1 Member Function Documentation

4.142.1.1 getType()

```
std::string err::LibrairiesLoadError::getType () const  [override], [virtual], [noexcept]
```

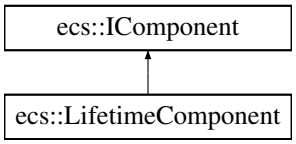
Implements err::AError.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Error/LibrairiesLoadError.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/Error/LibrairiesLoadError.cpp

4.143 ecs::LifetimeComponent Class Reference

Inheritance diagram for ecs::LifetimeComponent:



Public Member Functions

- **LifetimeComponent** (float lifetime=0.0f)
- float **getLifetime** () const
- void **setLifetime** (float lifetime)

Private Attributes

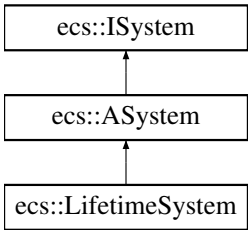
- float **_lifetime**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/permanent/LifetimeComponent.hpp

4.144 ecs::LifetimeSystem Class Reference

Inheritance diagram for ecs::LifetimeSystem:



Public Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

4.144.1 Member Function Documentation

4.144.1.1 update()

```
void ecs::LifetimeSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

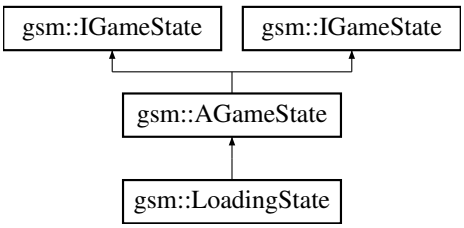
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/lifetime/LifetimeSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/lifetime/LifetimeSystem.cpp

4.145 gsm::LoadingState Class Reference

Inheritance diagram for gsm::LoadingState:



Public Member Functions

- **LoadingState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void enter () override
- std::string getStateName () const override

Public Member Functions inherited from gsm::AGameState

- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void update (float deltaTime) override
- void exit () override
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override
- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void update (float deltaTime) override
- void exit () override
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override

Additional Inherited Members

Protected Member Functions inherited from gsm::AGameState

- void addSystem (std::shared_ptr< ecs::ISystem > system) override
- void addSystem (std::shared_ptr< ecs::ISystem > system) override

Protected Attributes inherited from gsm::AGameState

- std::weak_ptr< IGameStateMachine > **_gsm**
- std::shared_ptr< ResourceManager > **_resourceManager**
- std::vector< std::shared_ptr< ecs::ISystem > > **_systems**

4.145.1 Member Function Documentation

4.145.1.1 enter()

```
void gsm::LoadingState::enter ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.145.1.2 getStateName()

```
std::string gsm::LoadingState::getStateName () const  [inline], [override], [virtual]
```

Implements gsm::AGameState.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/server/gsm/states/scenes/Loading/LoadingState.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/gsm/states/scenes/Loading/LoadingState.cpp

4.146 parser::JsonLoader::LoadResult Struct Reference

Public Attributes

- nlohmann::json **data**
- bool **success**
- std::string **errorMessage**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/Utils/JsonLoader.hpp

4.147 rserv::Lobby Class Reference

Public Member Functions

- **Lobby** (std::shared_ptr< net::INetwork > network, std::vector< std::tuple< uint8_t, std::shared_ptr< net::INetworkEndpoint >, std::string > > lobbyPlayerInfo, std::string lobbyCode, bool debug, int64_t tps)
- void **stop** ()
- void **startNetworkThread** ()
- void **startGameThread** ()
- void **networkLoop** ()
- void **gameLoop** ()
- void **setIsDebug** (bool debug)
- bool **getIsDebug** () const
- std::shared_ptr< ResourceManager > **getResourceManager** () const
- const std::map< uint8_t, ecs::Entity > & **getClientToEntity** () const
- std::vector< uint8_t > **getConnectedClients** () const

- std::vector< std::tuple< uint8_t, std::string > > **getConnectedClientDetails** () const
- std::vector< std::shared_ptr< net::INetworkEndpoint > > **getConnectedClientEndpoints** () const
- size_t **getClientCount** () const
- bool **isRunning** () const
- void **addClient** (std::tuple< uint8_t, std::shared_ptr< net::INetworkEndpoint >, std::string > client)
- void **removeClient** (uint8_t clientId)
- void **resetClientHeartbeats** ()
- void **createPlayerEntityForClient** (uint8_t clientId)
- void **syncExistingEntitiesToClient** (std::shared_ptr< net::INetworkEndpoint > clientEndpoint)
- std::string **getLobbyCode** () const
- std::shared_ptr< net::INetwork > **getNetwork** () const
- std::string **getGameState** () const
- std::string **getGameRules** () const
- std::shared_ptr< std::queue< std::tuple< uint8_t, constants::EventType, double > > > **getEventQueue** ()
- bool **hasEvents** () const
- void **enqueuePacket** (std::pair< std::shared_ptr< net::INetworkEndpoint >, std::vector< uint8_t > > packet)
- void **processIncomingPackets** ()
- bool **processDisconnections** (uint8_t idClient)
- bool **processEvents** (uint8_t idClient)
- bool **processWhoAmI** (uint8_t idClient)
- bool **gameStatePacket** ()
- bool **endGamePacket** (bool isWin)
- std::vector< uint64_t > **spawnPacket** (size_t entity, const std::string prefabName)
- std::vector< uint64_t > **deathPacket** (size_t entity)
- bool **serverStatusPacket** ()
- bool **ackLeaveLobbyPacket** (const net::INetworkEndpoint &endpoint, bool canDisconnect)
- bool **levelCompletePacket** ()
- bool **nextLevelPacket** ()
- bool **gameRulesPacket** ()
- bool **isGameStarted** () const
- bool **allClientsReady** () const
- uint32_t **getSequenceNumber** () const
- void **setPacketManager** (std::shared_ptr< pm::IPacketManager > packet)
- std::shared_ptr< pm::IPacketManager > **getPacketManager** () const
- void **incrementSequenceNumber** ()
- void **setResourceManager** (std::shared_ptr< ResourceManager > resourceManager)
- void **clearEntityDeltaCache** (uint8_t clientId, uint32_t entityId)
- void **clearDeltaTrackerCaches** ()
- void **createPlayerEntities** ()
- void **processLobbyEvents** ()

Protected Member Functions

- std::vector< uint64_t > **convertTransformComponent** (std::shared_ptr< ecs::Registry > registry, ecs::↩ Entity i)
- std::vector< uint64_t > **convertHealthComponent** (std::shared_ptr< ecs::Registry > registry, ecs::Entity i)
- std::vector< uint64_t > **convertScoreComponent** (std::shared_ptr< ecs::Registry > registry, ecs::Entity i)
- std::vector< uint64_t > **convertAnimationStateComponent** (std::shared_ptr< ecs::Registry > registry, ecs::Entity i)
- std::vector< uint64_t > **convertChargedShotComponent** (std::shared_ptr< ecs::Registry > registry, ecs↩ ::Entity i)

Private Attributes

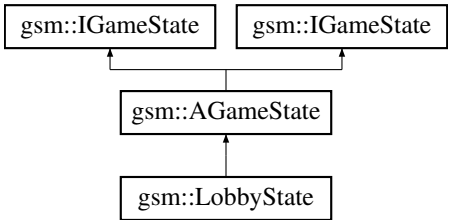
- bool **_isDebug**
- int64_t **_tps**
- std::shared_ptr< net::INetwork > **_network**
- std::vector< std::tuple< uint8_t, std::shared_ptr< net::INetworkEndpoint >, std::string > > **_clients**
- std::string **_lobbyCode**
- std::map< uint8_t, bool > **_clientsReady**
- std::map< uint8_t, ecs::Entity > **_clientToEntity**
- std::shared_ptr< pm::IPacketManager > **_packet**
- uint32_t **_sequenceNumber**
- std::shared_ptr< std::queue< std::tuple< uint8_t, constants::EventType, double > > > **_eventQueue**
- std::queue< std::pair< std::shared_ptr< net::INetworkEndpoint >, std::vector< uint8_t > > > **_incoming↩ Packets**
- std::mutex **_packetMutex**
- std::mutex **_clientsMutex**
- bool **_gameStarted**
- bool **_playerEntitiesCreated**
- std::shared_ptr< ResourceManager > **_resourceManager**
- std::shared_ptr< gsm::GameStateMachine > **_gsm**
- std::chrono::steady_clock::time_point **_lastGameStateTime**
- float **_statusUpdateTimer**
- std::atomic_bool **_running**
- std::thread **_networkThread**
- std::thread **_gameThread**
- std::mutex **_eventMutex**
- ComponentDeltaTracker **_deltaTracker**
- std::vector< std::function< std::vector< uint64_t >(std::shared_ptr< ecs::Registry >, ecs::Entity)> > _↩ **convertFunctions**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/server/Lobby.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/ECSConversions.cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/Lobby.cpp

4.148 gsm::LobbyState Class Reference

Inheritance diagram for gsm::LobbyState:



Public Member Functions

- **LobbyState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void enter () override
- void update (float deltaTime) override
- std::string getStateName () const override

Public Member Functions inherited from gsm::AGameState

- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void exit () override
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override
- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void exit () override
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override

Additional Inherited Members

Protected Member Functions inherited from gsm::AGameState

- void addSystem (std::shared_ptr< ecs::ISystem > system) override
- void addSystem (std::shared_ptr< ecs::ISystem > system) override

Protected Attributes inherited from gsm::AGameState

- std::weak_ptr< IGameStateMachine > **_gsm**
- std::shared_ptr< ResourceManager > **_resourceManager**
- std::vector< std::shared_ptr< ecs::ISystem > > **_systems**

4.148.1 Member Function Documentation

4.148.1.1 enter()

```
void gsm::LobbyState::enter ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.148.1.2 getStateName()

```
std::string gsm::LobbyState::getStateName () const  [inline], [override], [virtual]
```

Implements gsm::AGameState.

4.148.1.3 update()

```
void gsm::LobbyState::update (
            float deltaTime)  [override], [virtual]
```

Reimplemented from gsm::AGameState.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/server/gsm/states/scenes/Lobby/LobbyState.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/gsm/states/scenes/Lobby/LobbyState.cpp

4.149 rserv::LobbyStruct Struct Reference

Public Attributes

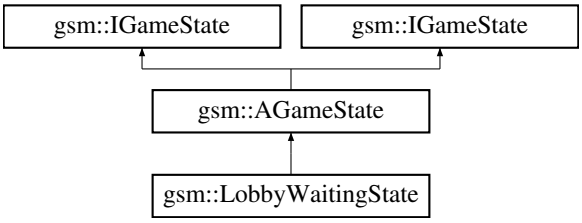
- std::string **_lobbyCode**
- std::vector< std::tuple< uint8_t, std::shared_ptr< net::INetworkEndpoint >, std::string > > **_clients**
- std::shared_ptr< Lobby > **_lobby**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/server/LobbyStruct.hpp

4.150 gsm::LobbyWaitingState Class Reference

Inheritance diagram for gsm::LobbyWaitingState:



Public Member Functions

- **LobbyWaitingState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager, bool isLobbyMaster)
- void enter () override
- void update (float deltaTime) override
- void exit () override
- std::string getStateName () const override

Public Member Functions inherited from gsm::AGameState

- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override
- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override

Private Member Functions

- void **renderUI** ()
- void **updateUIStatus** ()
- void **setupLobbyMasterUI** ()
- void **setupPlayerUI** ()

Private Attributes

- std::unique_ptr< MouseInputHandler > **_mouseHandler**
- std::unique_ptr< ui::UIManager > **_uiManager**
- std::shared_ptr< ui::UILayout > **_centerLayout**
- std::shared_ptr< ui::Text > **_lobbyCodeText**
- std::shared_ptr< ui::Text > **_statusText**
- std::shared_ptr< ui::Button > **_startGameButton**
- std::shared_ptr< ui::UILayout > **_topLeftLayout**
- std::shared_ptr< ui::Text > **_gamemodeLabel**
- std::shared_ptr< ui::Button > **_gamemodeButton**
- std::shared_ptr< ui::Text > **_difficultyLabel**
- std::shared_ptr< ui::Button > **_difficultyButton**
- std::shared_ptr< ui::Text > **_crossfireLabel**
- std::shared_ptr< ui::Button > **_crossfireButton**
- std::shared_ptr< ui::Button > **_leaveButton**
- std::shared_ptr< ui::UILayout > **_topRightLayout**
- std::shared_ptr< ui::Button > **_chatButton**
- std::shared_ptr< ui::SpritePreview > **_loadingAnimation**
- std::shared_ptr< ui::UILayout > **_loadingLayout**
- std::shared_ptr< ui::UILayout > **_bottomLeftLayout**
- std::shared_ptr< ui::Button > **_copyCodeButton**
- bool **_isLobbyMaster**

Additional Inherited Members

Protected Member Functions inherited from gsm::AGameState

- void addSystem (std::shared_ptr< ecs::ISystem > system) override
- void addSystem (std::shared_ptr< ecs::ISystem > system) override

Protected Attributes inherited from gsm::AGameState

- std::weak_ptr< IGameStateMachine > **_gsm**
- std::shared_ptr< ResourceManager > **_resourceManager**
- std::vector< std::shared_ptr< ecs::ISystem > > **_systems**

4.150.1 Member Function Documentation

4.150.1.1 enter()

```
void gsm::LobbyWaitingState::enter ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.150.1.2 exit()

```
void gsm::LobbyWaitingState::exit ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.150.1.3 getStateName()

```
std::string gsm::LobbyWaitingState::getStateName () const  [inline], [override], [virtual]
```

Implements gsm::AGameState.

4.150.1.4 update()

```
void gsm::LobbyWaitingState::update (
            float deltaTime)  [override], [virtual]
```

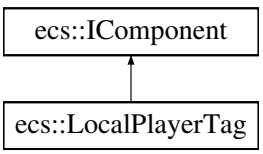
Reimplemented from gsm::AGameState.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LobbyWaiting/LobbyWaitingState.↩ hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LobbyWaiting/LobbyWaitingState.↩ cpp

4.151 ecs::LocalPlayerTag Class Reference

Inheritance diagram for ecs::LocalPlayerTag:

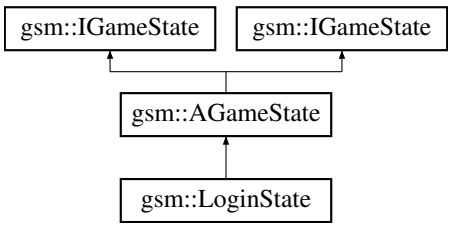


The documentation for this class was generated from the following file:

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/tags/LocalPlayerTag.hpp

4.152 gsm::LoginState Class Reference

Inheritance diagram for gsm::LoginState:



Public Member Functions

- **LoginState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void enter () override
- void update (float deltaTime) override
- void exit () override
- std::string getStateName () const override

Public Member Functions inherited from gsm::AGameState

- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override
- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override

Private Member Functions

- void **renderUI** ()

Private Attributes

- std::unique_ptr< MouseInputHandler > **_mouseHandler**
- std::unique_ptr< ui::UIManager > **_uiManager**
- std::shared_ptr< ui::Background > **_background**
- std::shared_ptr< ui::UILayout > **_mainLayout**
- std::shared_ptr< ui::TextInput > **_usernameInput**
- std::shared_ptr< ui::TextInput > **_passwordInput**
- std::shared_ptr< ui::Button > **_loginButton**
- std::shared_ptr< ui::Button > **_backButton**

Additional Inherited Members

Protected Member Functions inherited from gsm::AGameState

- void addSystem (std::shared_ptr< ecs::ISystem > system) override
- void addSystem (std::shared_ptr< ecs::ISystem > system) override

Protected Attributes inherited from gsm::AGameState

- std::weak_ptr< IGameStateMachine > **_gsm**
- std::shared_ptr< ResourceManager > **_resourceManager**
- std::vector< std::shared_ptr< ecs::ISystem > > **_systems**

4.152.1 Member Function Documentation

4.152.1.1 enter()

```
void gsm::LoginState::enter ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.152.1.2 exit()

```
void gsm::LoginState::exit ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.152.1.3 getStateName()

```
std::string gsm::LoginState::getStateName () const  [inline], [override], [virtual]
```

Implements gsm::AGameState.

4.152.1.4 update()

```
void gsm::LoginState::update (
            float deltaTime)  [override], [virtual]
```

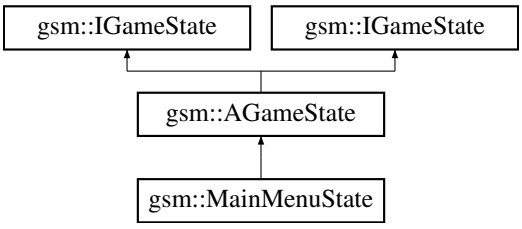
Reimplemented from gsm::AGameState.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/Login/LoginState.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/Login/LoginState.cpp

4.153 gsm::MainMenuState Class Reference

Inheritance diagram for gsm::MainMenuState:



Public Member Functions

- **MainMenuState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void enter () override
- void update (float deltaTime) override
- void exit () override
- std::string getStateName () const override

Public Member Functions inherited from gsm::AGameState

- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override
- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override

Private Member Functions

- void **renderUI** ()
- void **updateUIStatus** ()
- void **checkLobbyConnectionTransition** ()

Private Attributes

- std::unique_ptr< MouseInputHandler > **_mouseHandler**
- std::shared_ptr< ui::Button > **_usernameButton**
- std::shared_ptr< ui::Button > **_settingsButton**
- std::shared_ptr< ui::Button > **_quitButton**
- std::shared_ptr< ui::Button > **_requestCodeButton**
- std::shared_ptr< ui::Button > **_lobbyConnectButton**
- std::unique_ptr< ui::UIManager > **_uiManager**
- std::shared_ptr< ui::UILayout > **_mainMenuLayout**
- std::shared_ptr< ui::UILayout > **_headerLayout**
- std::shared_ptr< ui::UILayout > **_topLeftLayout**
- std::shared_ptr< ui::UILayout > **_rightLayout**
- std::shared_ptr< ui::Button > **_devButton**
- std::shared_ptr< ui::Button > **_howToPlayButton**
- std::shared_ptr< ui::Button > **_leaderboardButton**
- std::shared_ptr< ui::Button > **_registerButton**
- std::shared_ptr< ui::Button > **_loginButton**
- std::shared_ptr< ui::Button > **_disconnectButton**
- std::shared_ptr< ui::Button > **_chatButton**
- std::shared_ptr< ui::TextInput > **_lobbyCodeInput**
- std::shared_ptr< ui::Background > **_background**
- bool **_previousLobbyConnectedState**
- bool **_previousLobbyMasterState**

Additional Inherited Members

Protected Member Functions inherited from gsm::AGameState

- void addSystem (std::shared_ptr< ecs::ISystem > system) override
- void addSystem (std::shared_ptr< ecs::ISystem > system) override

Protected Attributes inherited from gsm::AGameState

- std::weak_ptr< IGameStateMachine > **_gsm**
- std::shared_ptr< ResourceManager > **_resourceManager**
- std::vector< std::shared_ptr< ecs::ISystem > > **_systems**

4.153.1 Member Function Documentation

4.153.1.1 enter()

```
void gsm::MainMenuState::enter ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.153.1.2 exit()

```
void gsm::MainMenuState::exit ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.153.1.3 getStateName()

```
std::string gsm::MainMenuState::getStateName () const  [inline], [override], [virtual]
```

Implements gsm::AGameState.

4.153.1.4 update()

```
void gsm::MainMenuState::update (
            float deltaTime)  [override], [virtual]
```

Reimplemented from gsm::AGameState.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/MainMenu/MainMenuState.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/MainMenu/MainMenuState.cpp

4.154 MapData Struct Reference

Public Attributes

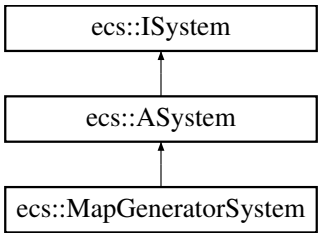
- int **index**
- std::string **name**
- std::string **filePath**
- nlohmann::json **jsonData**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/MapParser/MapHandler.hpp

4.155 ecs::MapGeneratorSystem Class Reference

Inheritance diagram for ecs::MapGeneratorSystem:



Public Member Functions

- **MapGeneratorSystem** (unsigned int seed=42)
- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Private Member Functions

- void **generateObstaclesAt** (float x, std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry)
- void **generateRandomWave** (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float currentX)
- void **generateRandomPowerUp** (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float currentX)
- float **noise** (float x)

Private Attributes

- unsigned int **_seed**
- std::mt19937 **_rng**
- float **_lastGeneratedX**
- const float **_generationStep**
- const float **_startGenerationX**
- float **_waveTimer**
- const float **_waveInterval**
- float **_powerUpTimer**
- const float **_powerUpInterval**

4.155.1 Member Function Documentation

4.155.1.1 update()

```
void ecs::MapGeneratorSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/map/MapGeneratorSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/map/MapGeneratorSystem.cpp

4.156 MapHandler Class Reference

Public Member Functions

- void **parseAllLevels** (const std::string &directoryPath)
- const MapData & **getCurrentMap** () const
- nlohmann::json **getCurrentMapJson** () const
- size_t **getCurrentMapIndex** () const
- size_t **getTotalMaps** () const
- bool **hasMaps** () const
- bool **isLastMap** () const
- bool **advanceToNextMap** ()
- bool **hasCompletedAllMaps** () const
- void **resetToFirstMap** ()
- const std::vector< MapData > & **getAllMaps** () const

Private Member Functions

- void **sortMapsByIndex** ()

Private Attributes

- std::vector< MapData > **_maps**
- size_t **_currentMapIndex**
- bool **_completedAllMaps**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/MapParser/MapHandler.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/MapParser/MapHandler.cpp

4.157 MapParser Class Reference

Public Member Functions

- **MapParser** (std::shared_ptr< EntityPrefabManager > prefabManager, std::shared_ptr< ecs::Registry > registry)
- void **parseMapFromFile** (const std::string &filePath)
- void **parseMap** (const nlohmann::json &mapJson)
- void **generateMapEntities** ()
- nlohmann::json **getMapJson** () const
- void **setMapJson** (const nlohmann::json &mapJson)
- void **setCreationContext** (const ecs::EntityCreationContext &context)
- ecs::EntityCreationContext **getCreationContext** () const

Private Member Functions

- void **createBackgroundEntity** (const std::string &entityName)
- void **createMusicEntity** (const std::string &prefabName)
- void **createGameZoneEntity** (float scrollSpeed)
- void **createGameEndEntity** (float mapLength)
- void **createGameZoneStopEntity** (float stopAtX)
- void **parsePowerUps** (const nlohmann::json &powerUps)
- void **parseObstacles** (const nlohmann::json &obstacles)
- void **parseWaves** (const nlohmann::json &waves)
- std::vector< float > **getPositionsFromDistrib** (int count, const nlohmann::json &distribution, float limit)
- ecs::Entity **createEntityFromPrefab** (const std::string &prefabName, float x, float y)

Private Attributes

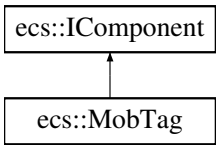
- std::shared_ptr< EntityPrefabManager > **_prefabManager**
- std::shared_ptr< ecs::Registry > **_registry**
- ecs::EntityCreationContext **_creationContext**
- nlohmann::json **_mapJson**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/MapParser/MapParser.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/MapParser/MapParser.cpp

4.158 ecs::MobTag Class Reference

Inheritance diagram for ecs::MobTag:



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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/tags/MobTag.hpp

4.159 MouseClickInfo Struct Reference

Public Attributes

- math::Vector2f **position**
- constants::MouseButton **button**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/input/MouseInputHandler.hpp

4.160 MouseInputHandler Class Reference

Public Member Functions

- **MouseInputHandler** (std::shared_ptr< ResourceManager > resourceManager)
- std::optional< MouseClickInfo > **pollMouseClick** ()
- math::Vector2f **getMousePosition** () const
- math::Vector2f **getWorldMousePosition** () const
- math::Vector2f **getNormalizedMousePosition** () const
- bool **isMouseButtonPressed** (int button) const

Private Attributes

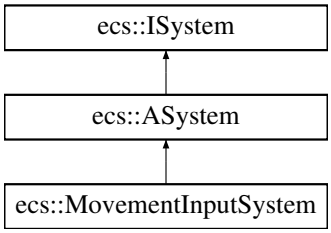
- std::weak_ptr< ResourceManager > **_resourceManager**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/input/MouseInputHandler.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/input/MouseInputHandler.cpp

4.161 ecs::MovementInputSystem Class Reference

Inheritance diagram for ecs::MovementInputSystem:



Public Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Private Member Functions

- math::Vector2f **getMovementDirection** (std::shared_ptr< ResourceManager > resourceManager) const
- void **updateInputIntent** (std::shared_ptr< Registry > registry, Entity entityId, const math::Vector2f &direction)
- math::Vector2f **getAnalogStickInput** (std::shared_ptr< IInputProvider > inputProvider) const
- void **sendAxisEvents** (std::shared_ptr< ResourceManager > resourceManager, const math::Vector2f &direction)
- bool **isPlayerAlive** (std::shared_ptr< Registry > registry, Entity entityId) const

Private Attributes

- bool **_wasMovingLastFrame** = false

4.161.1 Member Function Documentation

4.161.1.1 update()

```
void ecs::MovementInputSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

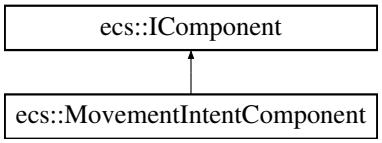
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/input/MovementInputSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/input/MovementInputSystem.cpp

4.162 ecs::MovementIntentComponent Class Reference

Inheritance diagram for ecs::MovementIntentComponent:



Public Member Functions

- **MovementIntentComponent** (const math::Vector2f &direction=math::Vector2f(0.0f, 0.0f), bool active=false)
- math::Vector2f **getDirection** () const
- void **setDirection** (const math::Vector2f &direction)
- bool **isActive** () const
- void **setActive** (bool active)

Private Attributes

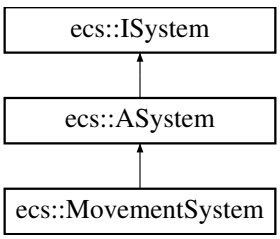
- math::Vector2f **_direction**
- bool **_active**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/temporary/MovementIntentComponent.↩ hpp

4.163 ecs::MovementSystem Class Reference

Inheritance diagram for ecs::MovementSystem:



Public Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Private Member Functions

- void **buildSpatialGrid** (std::shared_ptr< Registry > registry)
- bool **checkCollision** (std::shared_ptr< Registry > registry, size_t entityId, math::Vector2f newPos)
- math::Vector2f **calculateSmoothMovement** (std::shared_ptr< Registry > registry, size_t entityId, math::Vector2f startPos, math::Vector2f desiredPos)
- math::Vector2f **calculateSlidingMovement** (std::shared_ptr< Registry > registry, size_t entityId, math::Vector2f basePos, math::Vector2f desiredPos)
- math::Vector2f **calculateSmoothSlidingPosition** (std::shared_ptr< Registry > registry, size_t entityId, math::Vector2f startPos, math::Vector2f desiredPos)
- void **handlePushCollision** (std::shared_ptr< Registry > registry, size_t entityId, math::Vector2f finalPos, float deltaTime)
- bool **shouldCollide** (std::shared_ptr< Registry > registry, size_t entityA, const ColliderComponent &colliderA, size_t entityB)
- bool **checkCollisionWithBoundaries** (std::shared_ptr< Registry > registry, size_t entityId, math::Vector2f newPos)

Private Attributes

- SpatialGrid **_spatialGrid**
- std::vector< Entity > **_boundaryEntities**

4.163.1 Member Function Documentation

4.163.1.1 update()

```
void ecs::MovementSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/movement/MovementSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/movement/MovementSystem.cpp

4.164 ecs::MultiShotPattern Struct Reference

Public Attributes

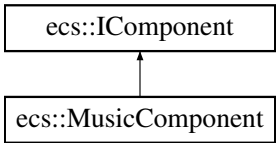
- int **shotCount** = 1
- float **angleSpread** = 0.0f
- float **offsetDistance** = 0.0f
- float **angleOffset** = 0.0f

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/permanent/ShootingStatsComponent.↩ hpp

4.165 ecs::MusicComponent Class Reference

Inheritance diagram for ecs::MusicComponent:



Public Member Functions

- **MusicComponent** (std::string musicFile="", MusicState initialState=STOPPED, float volume=100.0f, bool loop=false)
- void **playMusic** ()
- void **pauseMusic** ()
- void **stopMusic** ()
- bool **isPlaying** () const
- MusicState **getState** () const
- void **playNewMusic** (const std::string &musicFile)
- std::string **getCurrentMusic** () const
- void **setCurrentMusic** (const std::string &musicFile)
- float **getVolume** () const
- void **setVolume** (float volume)
- bool **isLooping** () const
- void **setLoop** (bool loop)

Private Attributes

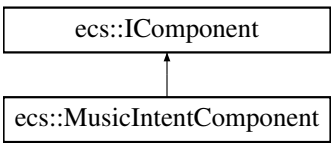
- std::string **_currentMusic**
- MusicState **_state**
- float **_volume**
- bool **_loop**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/components/rendering/MusicComponent.hpp

4.166 ecs::MusicIntentComponent Class Reference

Inheritance diagram for ecs::MusicIntentComponent:



Public Member Functions

- **MusicIntentComponent** (MusicAction action=PLAY, const std::string &musicPath="", float volume=100.0f)
- MusicAction **getAction** () const
- void **setAction** (MusicAction action)
- std::string **getMusicPath** () const
- void **setMusicPath** (const std::string &musicPath)
- float **getVolume** () const
- void **setVolume** (float volume)

Private Attributes

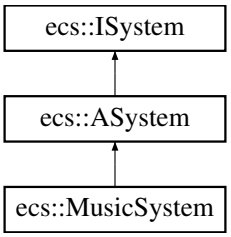
- MusicAction **_action**
- std::string **_musicPath**
- float **_volume**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/components/temporary/MusicIntentComponent.hpp

4.167 ecs::MusicSystem Class Reference

Inheritance diagram for ecs::MusicSystem:



Protected Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Additional Inherited Members

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

4.167.1 Member Function Documentation

4.167.1.1 update()

```
void ecs::MusicSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [protected], [virtual]
```

Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/audio/MusicSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/audio/MusicSystem.cpp

4.168 NetworkEvent Struct Reference

Public Attributes

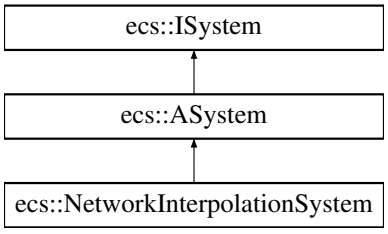
- constants::EventType **eventType**
- double **depth**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/ClientNetwork.hpp

4.169 ecs::NetworkInterpolationSystem Class Reference

Inheritance diagram for ecs::NetworkInterpolationSystem:



Public Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Private Member Functions

- void **interpolateTransform** (std::shared_ptr< NetworkStateComponent > networkState, std::shared_ptr< TransformComponent > transform)
- float **getTransformInterpolationFactor** (std::shared_ptr< NetworkStateComponent > networkState) const

4.169.1 Member Function Documentation

4.169.1.1 update()

```
void ecs::NetworkInterpolationSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

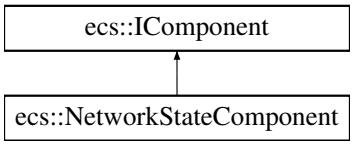
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/network/NetworkInterpolationSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/network/NetworkInterpolationSystem.cpp

4.170 ecs::NetworkStateComponent Class Reference

Inheritance diagram for ecs::NetworkStateComponent:



Public Member Functions

- void **setCurrentTransform** (const math::Vector2f &pos, float rot, const math::Vector2f &scale)
- bool **hasTransform** () const
- const NetworkTransformState & **getPreviousTransform** () const
- const NetworkTransformState & **getCurrentTransform** () const
- void **setInterpolationTime** (float time)
- float **getInterpolationTime** () const

Private Attributes

- NetworkTransformState **_previousTransform**
- NetworkTransformState **_currentTransform**
- bool **_hasTransform**
- float **_interpolationTime**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/components/permanent/NetworkStateComponent.hpp

4.171 ecs::NetworkTransformState Struct Reference

Public Attributes

- math::Vector2f **position**
- float **rotation**
- math::Vector2f **scale**
- std::chrono::steady_clock::time_point **timestamp**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/components/permanent/NetworkStateComponent.hpp

4.172 gsm::ObstacleGroup Struct Reference

Public Attributes

- std::vector< HorizontalLineObstacle > **horizontalLines**
- std::vector< VerticalLineObstacle > **verticalLines**
- std::vector< UniqueObstacle > **uniques**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditor/LevelEditorState.hpp

4.173 gsm::ObstacleSelection Struct Reference

Public Attributes

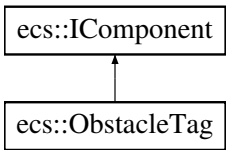
- std::string **prefabName**
- std::string **type**
- int **index**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditor/LevelEditorState.hpp

4.174 ecs::ObstacleTag Class Reference

Inheritance diagram for ecs::ObstacleTag:



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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/tags/ObstacleTag.hpp

4.175 math::OrientedRect Class Reference

Public Member Functions

- **OrientedRect** (Vector2f center, Vector2f size, float rotation)
- **OrientedRect** (OrientedRect const &other)
- Vector2f **getCenter** () const
- void **setCenter** (Vector2f center)
- Vector2f **getSize** () const
- void **setSize** (Vector2f size)
- float **getRotation** () const
- void **setRotation** (float rotation)
- std::vector< Vector2f > **getCorners** () const
- Vector2f **getAxisX** () const
- Vector2f **getAxisY** () const
- bool **intersects** (OrientedRect const &other) const
- OrientedRect & **operator=** (OrientedRect const &other)

Private Member Functions

- float **projectPoint** (Vector2f point, Vector2f axis) const
- bool **overlapOnAxis** (OrientedRect const &other, Vector2f axis) const

Private Attributes

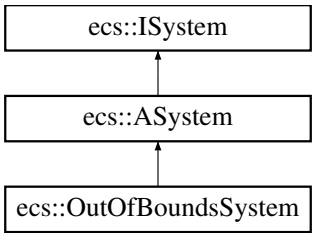
- Vector2f **_center**
- Vector2f **_size**
- float **_rotation**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/types/OrientedRect.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/types/OrientedRect.cpp

4.176 ecs::OutOfBoundsSystem Class Reference

Inheritance diagram for ecs::OutOfBoundsSystem:



Public Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Private Attributes

- float **_margin**

4.176.1 Member Function Documentation

4.176.1.1 update()

```
void ecs::OutOfBoundsSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

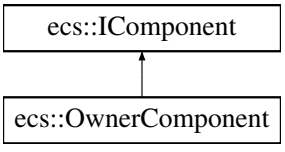
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/bounds/OutOfBoundsSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/bounds/OutOfBoundsSystem.cpp

4.177 ecs::OwnerComponent Class Reference

Inheritance diagram for ecs::OwnerComponent:



Public Member Functions

- **OwnerComponent** (Entity owner=0)
- Entity **getOwner** () const
- void **setOwner** (Entity owner)

Private Attributes

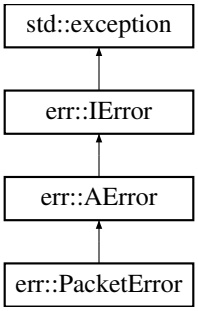
- Entity **_owner**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/permanent/OwnerComponent.hpp

4.178 err::PacketError Class Reference

Inheritance diagram for err::PacketError:



Public Types

- enum **ErrorCode** { **UNKNOWN** = 1000 , **SERIALIZER_ATTRIBUTION_FAILED** = 1001 , **STRING_↩ FORMATTING_ERROR** = 1002 }

Public Member Functions

- **PacketError** (const std::string &message, ErrorCode code=UNKNOWN)
- std::string getType () const noexcept override

Public Member Functions inherited from err::AError

- **AError** (const std::string &message, int code=0)
- const char * what () const noexcept override
- int getCode () const noexcept override
- std::string getDetails () const noexcept override

Additional Inherited Members

Protected Attributes inherited from err::AError

- std::string **m_message**
- int **m_code**

4.178.1 Member Function Documentation

4.178.1.1 getType()

```
std::string err::PacketError::getType () const  [override], [virtual], [noexcept]
```

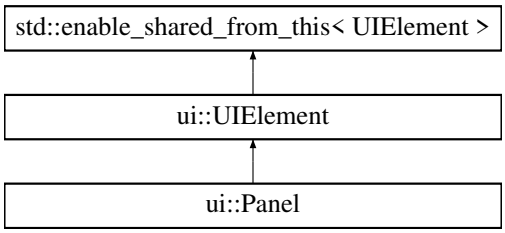
Implements err::AError.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Error/PacketError.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/Error/PacketError.cpp

4.179 ui::Panel Class Reference

Inheritance diagram for ui::Panel:



Public Member Functions

- **Panel** (std::shared_ptr< ResourceManager > resourceManager)
- void setScale (UIScale scale) override
- void **setBackgroundColor** (gfx::color_t color)
- void **setBorderColor** (gfx::color_t color)
- void **setBorderThickness** (float thickness)
- void render () override
- void update (float deltaTime) override

Public Member Functions inherited from ui::UIElement

- **UIElement** (std::shared_ptr< ResourceManager > resourceManager)
- void **setPosition** (const math::Vector2f &position)
- void **setSize** (const math::Vector2f &size)
- math::Vector2f **getPosition** () const
- math::Vector2f **getSize** () const
- math::Vector2f **getAbsolutePosition** () const
- math::Vector2f **getAbsoluteSize** () const
- void **setVisible** (bool visible)
- bool **isVisible** () const
- void **setScalingEnabled** (bool enabled)
- bool **isScalingEnabled** () const
- void **setFocusEnabled** (bool enabled)
- bool **isFocusEnabled** () const
- void **setState** (UIState state)
- UIState **getState** () const
- UIScale **getScale** () const
- void **setParent** (std::weak_ptr< UIElement > parent)
- std::shared_ptr< UIElement > **getParent** () const
- void **addChild** (std::shared_ptr< UIElement > child)
- void **removeChild** (std::shared_ptr< UIElement > child)
- const std::vector< std::shared_ptr< UIElement > > & **getChildren** () const
- virtual void **handleInput** (const math::Vector2f &mousePos, bool mousePressed)
- virtual bool **containsPoint** (const math::Vector2f &point) const
- void **setOnClick** (std::function< void()> callback)
- void **setOnHover** (std::function< void()> callback)
- void **setOnRelease** (std::function< void()> callback)

Private Attributes

- gfx::color_t **_backgroundColor** = {50, 50, 50, 255}
- gfx::color_t **_borderColor** = {100, 100, 100, 255}
- float **_borderThickness** = 2.0f

Additional Inherited Members

Protected Member Functions inherited from ui::UIElement

- std::pair< int, int > **getWindowSize** () const
- std::pair< int, int > **getLogicalSize** () const
- float **getScaleFactor** () const

Protected Attributes inherited from ui::UIElement

- std::weak_ptr< ResourceManager > **_resourceManager**
- math::Vector2f **_position**
- math::Vector2f **_size**
- bool **_visible** = true
- bool **_scalingEnabled** = true
- bool **_focusEnabled** = true
- UIState **_state** = UIState::Normal
- UIScale **_scale** = UIScale::Normal
- std::weak_ptr< UIElement > **_parent**
- std::vector< std::shared_ptr< UIElement > > **_children**
- bool **_pressedInside** = false
- bool **_wasPressed** = false
- std::function< void()> **_onClick**
- std::function< void()> **_onHover**
- std::function< void()> **_onRelease**

4.179.1 Member Function Documentation

4.179.1.1 render()

```
void ui::Panel::render ()  [override], [virtual]
```

Reimplemented from ui::UIElement.

4.179.1.2 setScale()

```
void ui::Panel::setScale (
            UIScale scale)  [override], [virtual]
```

Reimplemented from ui::UIElement.

4.179.1.3 update()

```
void ui::Panel::update (
            float deltaTime)  [override], [virtual]
```

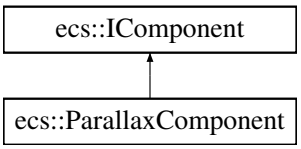
Reimplemented from ui::UIElement.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/elements/Panel.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/elements/Panel.cpp

4.180 ecs::ParallaxComponent Class Reference

Inheritance diagram for ecs::ParallaxComponent:



Public Member Functions

- float **getBaseScrollSpeed** () const
- const math::Vector2f & **getDirection** () const
- std::vector< std::shared_ptr< ParallaxLayer > > & **getLayers** ()
- void **setBaseScrollSpeed** (float speed)
- void **setDirection** (const math::Vector2f &direction)
- void **addLayer** (std::shared_ptr< ParallaxLayer > layer)

Private Attributes

- float **_baseScrollSpeed**
- math::Vector2f **_direction**
- std::vector< std::shared_ptr< ParallaxLayer > > **_layers**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/components/rendering/ParallaxComponent.hpp

4.181 ecs::ParallaxLayer Struct Reference

Public Attributes

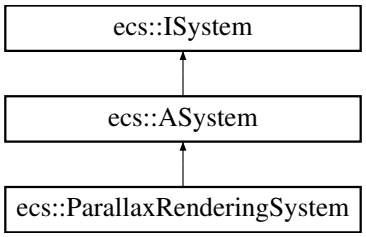
- std::string **name**
- std::string **filePath**
- float **speedMultiplier**
- math::Vector2f **scale**
- ParallaxScaleMode **scaleMode**
- math::Vector2f **sourceSize**
- bool **repeat**
- int **zIndex**
- math::Vector2f **currentOffset**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/components/rendering/ParallaxComponent.hpp

4.182 ecs::ParallaxRenderingSystem Class Reference

Inheritance diagram for ecs::ParallaxRenderingSystem:



Protected Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Private Member Functions

- math::Vector2f **calculateScale** (std::shared_ptr< ParallaxLayer > layer, float screenWidth, float screen↩ Height)
- void **renderLayer** (std::shared_ptr< ParallaxLayer > layer, std::shared_ptr< ResourceManager > resourceManager, const math::Vector2f &basePosition, float screenWidth, float screenHeight)

Additional Inherited Members

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

4.182.1 Member Function Documentation

4.182.1.1 update()

```
void ecs::ParallaxRenderingSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [protected], [virtual]
```

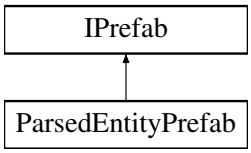
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/rendering/ParallaxRenderingSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/rendering/ParallaxRenderingSystem.cpp

4.183 ParsedEntityPrefab Class Reference

Inheritance diagram for ParsedEntityPrefab:



Public Member Functions

- **ParsedEntityPrefab** (const std::string &name, const std::map< std::type_index, ComponentAdder > &adders)
- void **addComponent** (std::shared_ptr< ecs::IComponent > component, std::type_index typeIndex)
- const std::vector< std::shared_ptr< ecs::IComponent > > & **getComponents** () const
- std::string **getName** () const
- ecs::Entity instantiate (const std::shared_ptr< ecs::Registry > &registry, const std::shared_ptr< ecs::IEntityFactory > &factory, const ecs::EntityCreationContext &context=ecs::EntityCreationContext↩ ::forLocalClient()) override
- ecs::Entity instantiate (const std::shared_ptr< ecs::Registry > &registry) override

Private Member Functions

- void **addParsedComponents** (const std::shared_ptr< ecs::Registry > &registry, ecs::Entity entity)

Private Attributes

- std::string **_name**
- std::vector< std::pair< std::shared_ptr< ecs::IComponent >, std::type_index > > **_components**
- const std::map< std::type_index, ComponentAdder > & **_componentAdders**

4.183.1 Member Function Documentation

4.183.1.1 instantiate() [1/2]

```
ecs::Entity ParsedEntityPrefab::instantiate (
            const std::shared_ptr< ecs::Registry > & registry)  [override], [virtual]
```

Implements IPrefab.

4.183.1.2 instantiate() [2/2]

```
ecs::Entity ParsedEntityPrefab::instantiate (
            const std::shared_ptr< ecs::Registry > & registry,
            const std::shared_ptr< ecs::IEntityFactory > & factory,
            const ecs::EntityCreationContext & context = ecs::EntityCreationContext::forLocalClient())
[override], [virtual]
```

Implements IPrefab.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Prefab/ParsedEntityPrefab.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/Prefab/ParsedEntityPrefab.cpp

4.184 Parser Class Reference

Public Member Functions

- **Parser** (std::shared_ptr< EntityPrefabManager > prefab, ParsingType type, std::shared_ptr< ecs::Registry > registry)
- std::shared_ptr< EntityPrefabManager > **getPrefabManager** () const
- void **setPrefabManager** (std::shared_ptr< EntityPrefabManager > prefab)
- void **parseAllEntities** (std::string directoryPath)
- void **parseEntity** (std::string entityPath)
- const std::map< std::type_index, ComponentAdder > & **getComponentAdders** () const
- ParsingType **getParsingType** () const
- bool **isClientParsing** () const
- bool **isServerParsing** () const
- bool **shouldParseComponent** (std::map< std::string, std::shared_ptr< FieldValue > > fields) const
- void **parseMapFromFile** (const std::string &filePath)
- void **parseMapFromJson** (const nlohmann::json &mapJson)
- std::shared_ptr< MapParser > **getMapParser** () const
- void **setRegistry** (std::shared_ptr< ecs::Registry > registry)

Private Attributes

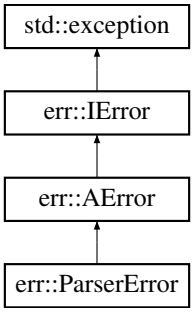
- std::shared_ptr< EntityParser > **_entityParser**
- std::shared_ptr< MapParser > **_mapParser**
- std::shared_ptr< EntityPrefabManager > **_prefabManager**
- std::shared_ptr< std::map< std::string, std::pair< std::type_index, std::vector< Field > > > > **_componentDefinitions**
- std::map< std::type_index, ComponentCreator > **_componentCreators**
- std::map< std::type_index, ComponentAdder > **_componentAdders**
- ParsingType **_parsingType**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/Parser.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/Parser.cpp

4.185 err::ParserError Class Reference

Inheritance diagram for err::ParserError:



Public Types

- enum **ErrorCode** {
  UNKNOWN = 1000 , **FILE_NOT_FOUND** = 1001 , **INVALID_FORMAT** = 1002 , **MISSING_FIELD** = 1003 ,
  TYPE_MISMATCH = 1004 }

Public Member Functions

- **ParserError** (const std::string &message, ErrorCode code=UNKNOWN)
- std::string getType () const noexcept override

Public Member Functions inherited from err::AError

- **AError** (const std::string &message, int code=0)
- const char * what () const noexcept override
- int getCode () const noexcept override
- std::string getDetails () const noexcept override

Additional Inherited Members

Protected Attributes inherited from err::AError

- std::string **m_message**
- int **m_code**

4.185.1 Member Function Documentation

4.185.1.1 getType()

```
std::string err::ParserError::getType () const  [override], [virtual], [noexcept]
```

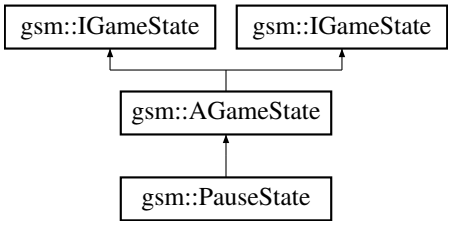
Implements err::AError.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Error/ParserError.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/Error/ParserError.cpp

4.186 gsm::PauseState Class Reference

Inheritance diagram for gsm::PauseState:



Public Member Functions

- **PauseState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void enter () override
- void update (float deltaTime) override
- void exit () override
- std::string getStateName () const override

Public Member Functions inherited from gsm::AGameState

- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override
- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override

Private Member Functions

- void **renderUI** ()

Private Attributes

- std::unique_ptr< MouseInputHandler > **_mouseHandler**
- std::unique_ptr< ui::UIManager > **_uiManager**
- std::shared_ptr< ui::UILayout > **_menuLayout**
- std::shared_ptr< ui::Button > **_resumeButton**
- std::shared_ptr< ui::Button > **_settingsButton**
- std::shared_ptr< ui::Button > **_leaveButton**

Additional Inherited Members

Protected Member Functions inherited from gsm::AGameState

- void addSystem (std::shared_ptr< ecs::ISystem > system) override
- void addSystem (std::shared_ptr< ecs::ISystem > system) override

Protected Attributes inherited from gsm::AGameState

- std::weak_ptr< IGameStateMachine > **_gsm**
- std::shared_ptr< ResourceManager > **_resourceManager**
- std::vector< std::shared_ptr< ecs::ISystem > > **_systems**

4.186.1 Member Function Documentation

4.186.1.1 enter()

```
void gsm::PauseState::enter ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.186.1.2 exit()

```
void gsm::PauseState::exit ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.186.1.3 getStateName()

```
std::string gsm::PauseState::getStateName () const  [inline], [override], [virtual]
```

Implements gsm::AGameState.

4.186.1.4 update()

```
void gsm::PauseState::update (
            float deltaTime)  [override], [virtual]
```

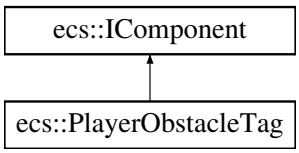
Reimplemented from gsm::AGameState.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/Pause/PauseState.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/Pause/PauseState.cpp

4.187 ecs::PlayerObstacleTag Class Reference

Inheritance diagram for ecs::PlayerObstacleTag:

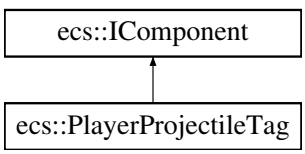


The documentation for this class was generated from the following file:

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/tags/PlayerObstacleTag.hpp

4.188 ecs::PlayerProjectileTag Class Reference

Inheritance diagram for ecs::PlayerProjectileTag:

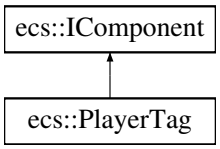


The documentation for this class was generated from the following file:

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/tags/PlayerProjectileTag.hpp

4.189 ecs::PlayerTag Class Reference

Inheritance diagram for ecs::PlayerTag:



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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/tags/PlayerTag.hpp

4.190 gsm::PowerUpData Struct Reference

Public Attributes

- float **posX**
- float **posY**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditor/LevelEditorState.hpp

4.191 gsm::PowerUpSelection Struct Reference

Public Attributes

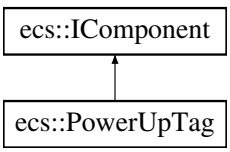
- std::string **prefabName**
- int **index**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditor/LevelEditorState.hpp

4.192 ecs::PowerUpTag Class Reference

Inheritance diagram for ecs::PowerUpTag:

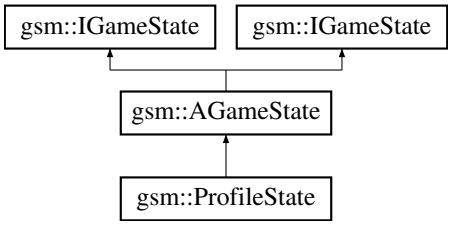


The documentation for this class was generated from the following file:

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/tags/PowerUpTag.hpp

4.193 gsm::ProfileState Class Reference

Inheritance diagram for gsm::ProfileState:



Public Member Functions

- **ProfileState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void enter () override
- void update (float deltaTime) override
- void exit () override
- std::string getStateName () const override

Public Member Functions inherited from gsm::AGameState

- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override
- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override

Private Member Functions

- void **renderUI** ()
- void **loadUserData** ()

Private Attributes

- std::unique_ptr< MouseInputHandler > **_mouseHandler**
- std::unique_ptr< ui::UIManager > **_uiManager**
- std::shared_ptr< ui::Background > **_background**
- std::shared_ptr< ui::UILayout > **_mainLayout**
- std::shared_ptr< ui::Text > **_titleText**
- std::shared_ptr< ui::Text > **_usernameText**
- std::shared_ptr< ui::Text > **_gamesPlayedText**
- std::shared_ptr< ui::Text > **_winsText**
- std::shared_ptr< ui::Text > **_highScoreText**
- std::shared_ptr< ui::Button > **_button1**
- std::shared_ptr< ui::Button > **_button2**
- std::shared_ptr< ui::Button > **_backButton**

Additional Inherited Members

Protected Member Functions inherited from gsm::AGameState

- void addSystem (std::shared_ptr< ecs::ISystem > system) override
- void addSystem (std::shared_ptr< ecs::ISystem > system) override

Protected Attributes inherited from gsm::AGameState

- std::weak_ptr< IGameStateMachine > **_gsm**
- std::shared_ptr< ResourceManager > **_resourceManager**
- std::vector< std::shared_ptr< ecs::ISystem > > **_systems**

4.193.1 Member Function Documentation

4.193.1.1 enter()

```
void gsm::ProfileState::enter ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.193.1.2 exit()

```
void gsm::ProfileState::exit ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.193.1.3 getStateName()

```
std::string gsm::ProfileState::getStateName () const  [inline], [override], [virtual]
```

Implements gsm::AGameState.

4.193.1.4 update()

```
void gsm::ProfileState::update (
            float deltaTime)  [override], [virtual]
```

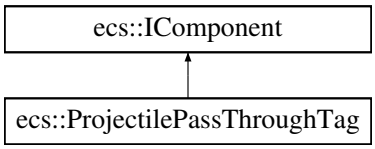
Reimplemented from gsm::AGameState.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/Profile/ProfileState.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/Profile/ProfileState.cpp

4.194 ecs::ProjectilePassThroughTag Class Reference

Inheritance diagram for ecs::ProjectilePassThroughTag:

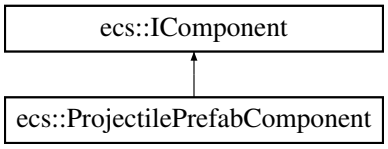


The documentation for this class was generated from the following file:

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/tags/ProjectilePassThroughTag.hpp

4.195 ecs::ProjectilePrefabComponent Class Reference

Inheritance diagram for ecs::ProjectilePrefabComponent:



Public Member Functions

- **ProjectilePrefabComponent** (const std::string &prefabName="")
- std::string **getPrefabName** () const
- void **setPrefabName** (const std::string &prefabName)

Private Attributes

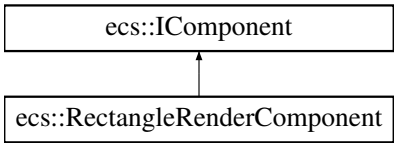
- std::string **_prefabName**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/permanent/ProjectilePrefabComponent.↩ hpp

4.196 ecs::RectangleRenderComponent Class Reference

Inheritance diagram for ecs::RectangleRenderComponent:



Public Member Functions

- **RectangleRenderComponent** (gfx::color_t color, float width, float height)
- const gfx::color_t & **getColor** () const
- void **setColor** (const gfx::color_t &color)
- float **getWidth** () const
- float **getHeight** () const
- void **setSize** (float width, float height)

Private Attributes

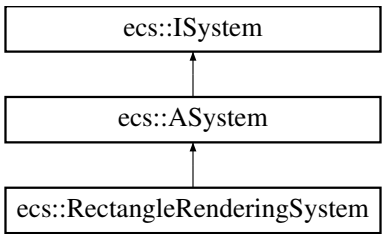
- gfx::color_t **_color**
- std::pair< float, float > **_size**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/components/rendering/RectangleRenderComponent.↩ hpp

4.197 ecs::RectangleRenderingSystem Class Reference

Inheritance diagram for ecs::RectangleRenderingSystem:



Protected Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Additional Inherited Members

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

4.197.1 Member Function Documentation

4.197.1.1 update()

```
void ecs::RectangleRenderingSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [protected], [virtual]
```

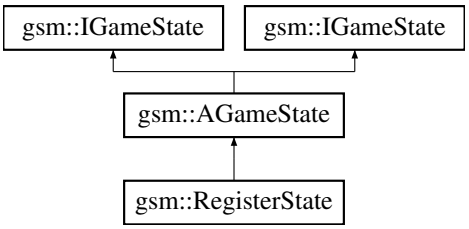
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/rendering/RectangleRenderingSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/rendering/RectangleRenderingSystem.cpp

4.198 gsm::RegisterState Class Reference

Inheritance diagram for gsm::RegisterState:



Public Member Functions

- **RegisterState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void enter () override
- void update (float deltaTime) override
- void exit () override
- std::string getStateName () const override

Public Member Functions inherited from gsm::AGameState

- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override
- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override

Private Member Functions

- void **renderUI** ()

Private Attributes

- std::unique_ptr< MouseInputHandler > **_mouseHandler**
- std::unique_ptr< ui::UIManager > **_uiManager**
- std::shared_ptr< ui::Background > **_background**
- std::shared_ptr< ui::UILayout > **_mainLayout**
- std::shared_ptr< ui::TextInput > **_usernameInput**
- std::shared_ptr< ui::TextInput > **_passwordInput**
- std::shared_ptr< ui::TextInput > **_confirmPasswordInput**
- std::shared_ptr< ui::Text > **_errorMessage**
- std::shared_ptr< ui::Button > **_registerButton**
- std::shared_ptr< ui::Button > **_backButton**

Additional Inherited Members

Protected Member Functions inherited from gsm::AGameState

- void addSystem (std::shared_ptr< ecs::ISystem > system) override
- void addSystem (std::shared_ptr< ecs::ISystem > system) override

Protected Attributes inherited from gsm::AGameState

- std::weak_ptr< IGameStateMachine > **_gsm**
- std::shared_ptr< ResourceManager > **_resourceManager**
- std::vector< std::shared_ptr< ecs::ISystem > > **_systems**

4.198.1 Member Function Documentation

4.198.1.1 enter()

```
void gsm::RegisterState::enter ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.198.1.2 exit()

```
void gsm::RegisterState::exit ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.198.1.3 getStateName()

```
std::string gsm::RegisterState::getStateName () const  [inline], [override], [virtual]
```

Implements gsm::AGameState.

4.198.1.4 update()

```
void gsm::RegisterState::update (
            float deltaTime)  [override], [virtual]
```

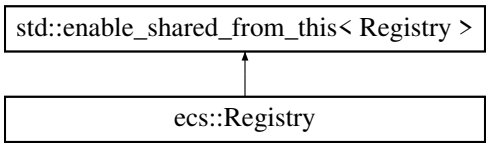
Reimplemented from gsm::AGameState.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/Register/RegisterState.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/Register/RegisterState.cpp

4.199 ecs::Registry Class Reference

Inheritance diagram for ecs::Registry:



Public Member Functions

- **Registry** (Entity nextEntityId)
- template<typename T>
  void **registerComponent** ()
- template<typename T>
  void **addComponent** (Entity entityId, std::shared_ptr< T > component)
- template<typename T>
  std::shared_ptr< T > **getComponent** (Entity entityId) const
- template<typename T>
  std::vector< std::shared_ptr< T > > **getComponents** (Entity entityId) const
- template<typename T>
  void **removeAllComponents** (Entity entityId)
- template<typename T>
  void **removeOneComponent** (Entity entityId)
- template<typename T>
  bool **hasComponent** (Entity entityId) const
- template<typename... Components>
  View< Components... > **view** ()
- Entity **getMaxEntityId** () const
- Entity **createEntity** ()
- void **destroyEntity** (Entity entityId)
- void **clearAllEntities** ()
- void **setOnEntityDestroyed** (std::function< void(Entity)> callback)

Private Attributes

- Entity **_nextEntityId**
- std::unordered_map< std::string, std::shared_ptr< IComponentArray > > **_components**
- std::function< void(Entity)> **_onEntityDestroyed**
- std::recursive_mutex **_mutex**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/ECS/entity/registry/Registry.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/ECS/entity/registry/Registry.cpp

4.200 ecs::RemappableKeyBinding Struct Reference

Public Member Functions

- **RemappableKeyBinding** (gfx::EventType p, gfx::EventType s)

Public Attributes

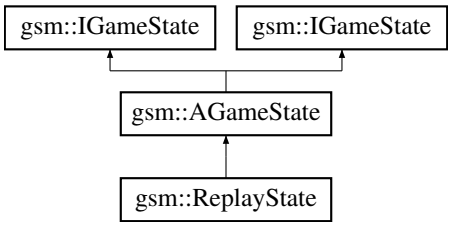
- gfx::EventType **primary**
- gfx::EventType **secondary**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/InputMapping/InputMapping.hpp

4.201 gsm::ReplayState Class Reference

Inheritance diagram for gsm::ReplayState:



Public Member Functions

- **ReplayState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void enter () override
- void update (float deltaTime) override
- void exit () override
- std::string getStateName () const override

Public Member Functions inherited from gsm::AGameState

- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override
- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override

Private Member Functions

- void **renderUI** ()
- void **renderReplaySprites** ()
- void **renderParallaxBackground** (const nlohmann::json &parallaxData, std::shared_ptr< gfx::IWindow > window)
- void **renderHealthBar** (const nlohmann::json &healthBarData, std::shared_ptr< gfx::IWindow > window)
- void **renderText** (const nlohmann::json &textData, std::shared_ptr< gfx::IWindow > window)
- void **renderRectangle** (const nlohmann::json &rectangleData, std::shared_ptr< gfx::IWindow > window)
- void **renderHitbox** (const nlohmann::json &hitboxData, std::shared_ptr< gfx::IWindow > window)
- void **updateViewForFrame** (const nlohmann::json &frame)
- void **loadReplay** (const std::filesystem::path &replayFile)
- void **playReplay** (float deltaTime)
- void **processAudioForFrame** (const nlohmann::json &frame)
- std::vector< std::filesystem::path > **getAvailableReplays** ()
- void **createReplaySelectionUI** ()
- void **createPlaybackControlsUI** ()
- void **updatePlaybackControls** ()

Private Attributes

- std::unique_ptr< MouseInputHandler > **_mouseHandler**
- std::unique_ptr< ui::UIManager > **_uiManager**
- std::unique_ptr< ui::UIManager > **_playbackUIManager**
- std::shared_ptr< ui::Background > **_background**
- std::shared_ptr< ui::Button > **_backButton**
- std::vector< std::shared_ptr< ui::Button > > **_replayButtons**
- std::shared_ptr< ui::Button > **_playPauseButton**
- std::shared_ptr< ui::Button > **_replayBackButton**
- std::shared_ptr< ui::Button > **_increaseSpeedButton**
- std::shared_ptr< ui::Button > **_decreaseSpeedButton**
- std::shared_ptr< ui::Slider > **_progressSlider**
- std::shared_ptr< ui::Text > **_timeText**
- std::shared_ptr< ui::Text > **_speedText**
- std::shared_ptr< ui::UILayout > **_playbackLayout**
- std::vector< nlohmann::json > **_frames**
- float **_replayTime**
- float **_totalReplayTime**
- size_t **_currentFrameIndex**
- bool **_isPlaying**
- bool **_isPaused**
- bool **_shouldSwitch**
- float **_spacePressCooldown**
- float **_playbackSpeed**
- float **_renderOffsetX**
- float **_renderOffsetY**

Additional Inherited Members

Protected Member Functions inherited from gsm::AGameState

- void addSystem (std::shared_ptr< ecs::ISystem > system) override
- void addSystem (std::shared_ptr< ecs::ISystem > system) override

Protected Attributes inherited from gsm::AGameState

- std::weak_ptr< IGameStateMachine > **_gsm**
- std::shared_ptr< ResourceManager > **_resourceManager**
- std::vector< std::shared_ptr< ecs::ISystem > > **_systems**

4.201.1 Member Function Documentation

4.201.1.1 enter()

```
void gsm::ReplayState::enter ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.201.1.2 exit()

```
void gsm::ReplayState::exit ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.201.1.3 getStateName()

```
std::string gsm::ReplayState::getStateName () const  [inline], [override], [virtual]
```

Implements gsm::AGameState.

4.201.1.4 update()

```
void gsm::ReplayState::update (
            float deltaTime)  [override], [virtual]
```

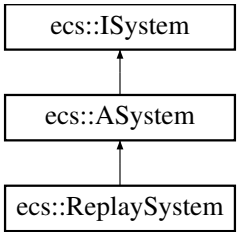
Reimplemented from gsm::AGameState.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/Replay/ReplayState.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/Replay/ReplayState.cpp

4.202 ecs::ReplaySystem Class Reference

Inheritance diagram for ecs::ReplaySystem:



Public Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Private Member Functions

- void **saveReplayToFile** (const nlohmann::json &frameData)
- std::string **getSpriteId** (const std::string &texturePath)
- std::filesystem::path **getNextReplayFile** ()

Private Attributes

- float **_totalElapsedTime**
- std::filesystem::path **_currentReplayFile**

4.202.1 Member Function Documentation

4.202.1.1 update()

```
void ecs::ReplaySystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/replay/ReplaySystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/replay/ReplaySystem.cpp

4.203 ResourceManager Class Reference

Public Member Functions

- template<typename T>
  void **add** (std::shared_ptr< T > resource)
- template<typename T>
  std::shared_ptr< T > **get** ()
- template<typename T>
  bool **has** ()
- void **clear** ()
- template<typename T>
  void **remove** ()

Private Attributes

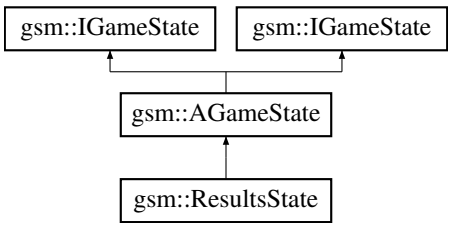
- std::unordered_map< size_t, std::shared_ptr< void > > **resources**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/resourceManager/ResourceManager.hpp

4.204 gsm::ResultsState Class Reference

Inheritance diagram for gsm::ResultsState:



Public Member Functions

- **ResultsState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager, bool isWin)
- void enter () override
- void update (float deltaTime) override
- void exit () override
- std::string getStateName () const override

Public Member Functions inherited from gsm::AGameState

- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override
- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override

Private Member Functions

- void **updateUserStats** ()
- void **renderUI** ()

Private Attributes

- bool **_isWin**
- std::unique_ptr< ui::UIManager > **_uiManager**
- std::unique_ptr< MouseInputHandler > **_mouseHandler**
- std::shared_ptr< ui::Text > **_resultText**
- std::shared_ptr< ui::SpritePreview > **_victoryAnimation**
- std::shared_ptr< ui::SpritePreview > **_youDiedAnimation**
- std::shared_ptr< ui::Button > **_leaveButton**
- std::shared_ptr< ui::UILayout > **_bottomRightLayout**

Additional Inherited Members

Protected Member Functions inherited from gsm::AGameState

- void addSystem (std::shared_ptr< ecs::ISystem > system) override
- void addSystem (std::shared_ptr< ecs::ISystem > system) override

Protected Attributes inherited from gsm::AGameState

- std::weak_ptr< IGameStateMachine > **_gsm**
- std::shared_ptr< ResourceManager > **_resourceManager**
- std::vector< std::shared_ptr< ecs::ISystem > > **_systems**

4.204.1 Member Function Documentation

4.204.1.1 enter()

```
void gsm::ResultsState::enter ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.204.1.2 exit()

```
void gsm::ResultsState::exit ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.204.1.3 getStateName()

```
std::string gsm::ResultsState::getStateName () const  [inline], [override], [virtual]
```

Implements gsm::AGameState.

4.204.1.4 update()

```
void gsm::ResultsState::update (
            float deltaTime)  [override], [virtual]
```

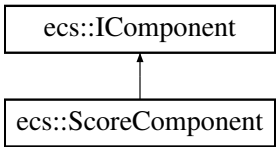
Reimplemented from gsm::AGameState.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/Results/ResultsState.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/Results/ResultsState.cpp

4.205 ecs::ScoreComponent Class Reference

Inheritance diagram for ecs::ScoreComponent:



Public Member Functions

- **ScoreComponent** (int score=0)
- int **getScore** () const
- void **setScore** (int score)

Private Attributes

- int **_score**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/permanent/ScoreComponent.hpp

4.206 gsm::ScoreFeedback Struct Reference

Public Attributes

- std::string **text**
- float **lifetime**
- float **maxLifetime**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/InGame/InGameState.hpp

4.207 ScoreIntentComponent Class Reference

Public Member Functions

- **ScoreIntentComponent** (int score=0)
- int **getScore** () const
- void **setScore** (int newScore)

Private Attributes

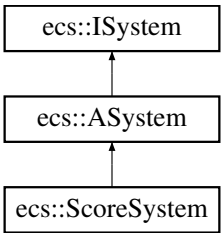
- int **_score**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/temporary/ScoreIntentComponent.hpp

4.208 ecs::ScoreSystem Class Reference

Inheritance diagram for ecs::ScoreSystem:



Public Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

4.208.1 Member Function Documentation

4.208.1.1 update()

```
void ecs::ScoreSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

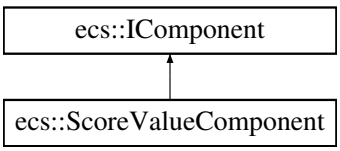
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/score/ScoreSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/score/ScoreSystem.cpp

4.209 ecs::ScoreValueComponent Class Reference

Inheritance diagram for ecs::ScoreValueComponent:



Public Member Functions

- **ScoreValueComponent** (int scoreValue=0)
- int **getScoreValue** () const
- void **setScoreValue** (int scoreValue)

Private Attributes

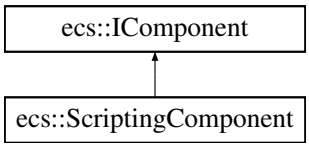
- int **_scoreValue**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/permanent/ScoreValueComponent.hpp

4.210 ecs::ScriptingComponent Class Reference

Inheritance diagram for ecs::ScriptingComponent:



Public Member Functions

- **ScriptingComponent** (std::string script_name="", std::vector< std::string > additionalFunctions=std↩ ::vector< std::string >(), std::shared_ptr< sol::state > lua=nullptr, size_t entityId=0)
- void **init** (sol::state &lua, size_t entityId)
- const std::string & **getScriptName** () const
- void **setEnvironment** (const sol::table &table)
- sol::table **getEnvironment** () const
- bool **hasFunction** (const std::string &name) const
- sol::function **getFunction** (const std::string &name) const
- void **addFunction** (const std::string &name, const sol::function &function)
- void **removeFunction** (const std::string &name)
- std::vector< std::string > **getFunctionNames** () const
- bool **isInitialized** () const
- void **setInitialized** (bool value)
- void **clearLuaReferences** ()

Private Attributes

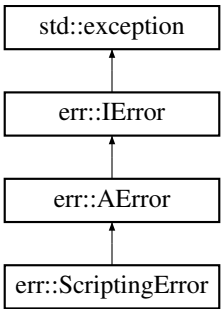
- std::string **_scriptName**
- std::vector< std::string > **_additionalFunctions**
- sol::table **_env**
- std::map< std::string, sol::function > **_functions**
- bool **_initialized** = false

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/permanent/ScriptingComponent.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/permanent/ScriptingComponent.cpp

4.211 err::ScriptingError Class Reference

Inheritance diagram for err::ScriptingError:



Public Types

- enum **ErrorCode** { **UNKNOWN** = 1000 , **LOAD_FAILED** = 1001 , **RUN_FAILED** = 1002 }

Public Member Functions

- **ScriptingError** (const std::string &message, ErrorCode code=UNKNOWN)
- std::string getType () const noexcept override

Public Member Functions inherited from err::AError

- **AError** (const std::string &message, int code=0)
- const char * what () const noexcept override
- int getCode () const noexcept override
- std::string getDetails () const noexcept override

Additional Inherited Members

Protected Attributes inherited from err::AError

- std::string **m_message**
- int **m_code**

4.211.1 Member Function Documentation

4.211.1.1 getType()

```
std::string err::ScriptingError::getType () const  [override], [virtual], [noexcept]
```

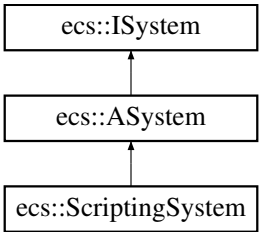
Implements err::AError.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Error/ScriptingError.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/Error/ScriptingError.cpp

4.212 ecs::ScriptingSystem Class Reference

Inheritance diagram for ecs::ScriptingSystem:



Public Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > reg, float deltaTime) override

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Private Member Functions

- void **bindAPI** ()

Private Attributes

- sol::state **lua**
- std::shared_ptr< Registry > **registry**
- std::shared_ptr< ResourceManager > **resourceManager**

4.212.1 Member Function Documentation

4.212.1.1 update()

```
void ecs::ScriptingSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > reg,
            float deltaTime)  [override], [virtual]
```

Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/scripting/ScriptingSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/scripting/ScriptingApiFunctions.cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/scripting/ScriptingSystem.cpp

4.213 utils::SecureJsonManager Class Reference

Static Public Member Functions

- static nlohmann::json **readSecureJson** (const std::string &filepath)
- static bool **writeSecureJson** (const std::string &filepath, const nlohmann::json &data)

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/utils/SecureJsonManager.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/utils/SecureJsonManager.cpp

4.214 rserv::Server Class Reference

Public Member Functions

- void **init** ()
- void **start** ()
- void **stop** ()
- void **setConfig** (std::shared_ptr< ServerConfig > config)
- std::shared_ptr< ServerConfig > **getConfig** () const
- uint16_t **getPort** () const
- void **setPort** (uint16_t port)
- int **getState** () const
- void **setState** (int state)
- void **initRessourceManager** (std::shared_ptr< Lobby > lobby)
- **operator int** () const noexcept
- std::shared_ptr< net::INetwork > **getNetwork** () const
- void **setNetwork** (std::shared_ptr< net::INetwork > network)
- void **onClientConnected** (uint8_t idClient)
- void **onClientDisconnected** (uint8_t idClient)
- void **onPacketReceived** (uint8_t idClient, const pm::IPacketManager &packet)

- std::vector< uint8_t > **getConnectedClients** () const
- std::vector< std::shared_ptr< net::INetworkEndpoint > > **getConnectedClientEndpoints** () const
- size_t **getClientCount** () const
- uint8_t **findClientIdByEndpoint** (const net::INetworkEndpoint &endpoint) const
- ServerInfo **getServerInfo** () const
- std::map< std::string, int > **loadUserStats** (const std::string &username) const
- void **saveUserBannedStatus** (const std::string &username, bool banned) const
- void **processIncomingPackets** ()
- bool **processConnections** (std::pair< std::shared_ptr< net::INetworkEndpoint >, std::vector< uint8_t > > client)
- bool **processDisconnections** (uint8_t idClient)
- bool **requestCode** (const net::INetworkEndpoint &endpoint)
- bool **processConnectToLobby** (std::pair< std::shared_ptr< net::INetworkEndpoint >, std::vector< uint8_t > > payload)
- bool **processMasterStart** (std::pair< std::shared_ptr< net::INetworkEndpoint >, std::vector< uint8_t > > payload)
- bool **processRegistration** (std::pair< std::shared_ptr< net::INetworkEndpoint >, std::vector< uint8_t > > client)
- bool **processLogin** (std::pair< std::shared_ptr< net::INetworkEndpoint >, std::vector< uint8_t > > client)
- bool **processLeaderboardRequest** (std::shared_ptr< net::INetworkEndpoint > client)
- bool **processProfileRequest** (std::shared_ptr< net::INetworkEndpoint > client)
- bool **processNewChatMessage** (std::pair< std::shared_ptr< net::INetworkEndpoint >, std::vector< uint8_t > > payload)
- bool **processRequestGameRulesUpdate** (std::pair< std::shared_ptr< net::INetworkEndpoint >, std::vector< uint8_t > > payload)
- void **cleanupClosedLobbies** ()
- void **checkClientTimeouts** ()
- bool **connectionPacket** (const net::INetworkEndpoint &endpoint)
- bool **canStartPacket** (std::vector< std::shared_ptr< net::INetworkEndpoint > > endpoints)
- bool **serverStatusPacket** ()
- bool **sendCodeLobbyPacket** (const net::INetworkEndpoint &endpoint)
- bool **lobbyConnectValuePacket** (const net::INetworkEndpoint &endpoint, bool canConnect)
- bool **connectUserPacket** (const net::INetworkEndpoint &endpoint, const std::string &username)
- bool **leaderboardPacket** (const net::INetworkEndpoint &endpoint)
- bool **profilePacket** (const net::INetworkEndpoint &endpoint)
- bool **newChatMessagePacket** (const net::INetworkEndpoint &endpoint, std::vector< uint8_t > message)
- bool **forceLeavePacket** (const net::INetworkEndpoint &endpoint, constants::ForceLeaveType leaveType)
- uint32_t **getSequenceNumber** () const
- std::shared_ptr< pm::IPacketManager > **getPacketManager** () const
- std::shared_ptr< pm::IPacketManager > **createNewPacketManager** ()
- uint32_t **getNextEntityId** ()
- void **incrementSequenceNumber** ()
- std::string **executeCommand** (const std::string &command)
- std::string **closeLobby** (const std::string &lobbyId)
- std::string **kickPlayer** (const std::string &playerId)
- std::string **banPlayer** (const std::string &playerId)
- std::string **unbanPlayer** (const std::string &playerId)
- std::string **toggleGodmod** (const std::string &playerId)

Private Member Functions

- void **loadNetworkLibrary** ()
- void **loadBufferLibrary** ()
- void **loadPacketLibrary** ()

Private Attributes

- DLLoader< createNetworkLib_t > **_networloader**
- DLLoader< createBuffer_t > **_bufferloader**
- DLLoader< createPacket_t > **_packetloader**
- std::shared_ptr< ServerConfig > **_config**
- std::shared_ptr< net::INetwork > **_network**
- std::shared_ptr< IBuffer > **_buffer**
- std::shared_ptr< pm::IPacketManager > **_packet**
- uint8_t **_nextClientId**
- uint32_t **_sequenceNumber**
- uint32_t **_nextEntityId**
- std::mutex **_clientsMutex**
- std::vector< std::tuple< uint8_t, std::shared_ptr< net::INetworkEndpoint >, std::string > > **_clients**
- std::map< uint8_t, bool > **_clientsReady**
- std::vector< std::shared_ptr< LobbyStruct > > **_lobbyThreads**
- std::vector< std::shared_ptr< Lobby > > **_lobbies**
- std::map< uint8_t, std::shared_ptr< Lobby > > **_clientToLobby**
- std::unique_ptr< HttpServer > **_httpServer**
- std::map< uint8_t, std::chrono::steady_clock::time_point > **_clientLastHeartbeat**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/server/Server.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/Server.cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/ServerLibsLoading.cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/ServerReceivePacket.cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/ServerSentPacket.cpp

4.215 rserv::ServerConfig Class Reference

Public Member Functions

- int **getState** () const
- void **setPort** (uint16_t port)
- uint16_t **getPort** () const
- void **setState** (int state)
- std::string **getIp** () const
- void **setIp** (std::string ip)
- void **setIsDebug** (bool isDebug)
- bool **getIsDebug** () const
- void **setTps** (int64_t tps)
- int64_t **getTps** () const

Private Attributes

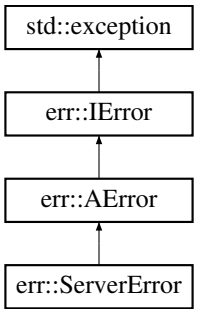
- int **_state**
- uint16_t **_port**
- std::string **_ip**
- bool **_isDebug**
- int64_t **_tps**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/server/ServerConfig.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/ServerConfig.cpp

4.216 err::ServerError Class Reference

Inheritance diagram for err::ServerError:



Public Types

- enum **ErrorCode** {
  UNKNOWN = 1000 , **CONNECTION_FAILED** = 1001 , **TIMEOUT** = 1002 , **INVALID_REQUEST** = 1003 ,
  INTERNAL_ERROR = 1004 , **LIBRARY_LOAD_FAILED** = 1005 , **CONFIG_ERROR** = 1006 }

Public Member Functions

- **ServerError** (const std::string &message, ErrorCode code=UNKNOWN)
- std::string getType () const noexcept override

Public Member Functions inherited from err::AError

- **AError** (const std::string &message, int code=0)
- const char * what () const noexcept override
- int getCode () const noexcept override
- std::string getDetails () const noexcept override

Additional Inherited Members

Protected Attributes inherited from err::AError

- std::string **m_message**
- int **m_code**

4.216.1 Member Function Documentation

4.216.1.1 getType()

```
std::string err::ServerError::getType () const  [override], [virtual], [noexcept]
```

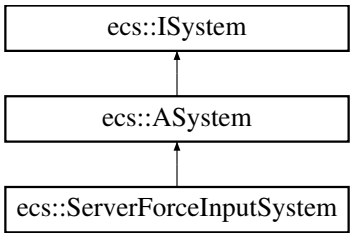
Implements err::AError.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Error/ServerErrror.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/Error/ServerErrror.cpp

4.217 ecs::ServerForceInputSystem Class Reference

Inheritance diagram for ecs::ServerForceInputSystem:



Public Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Private Member Functions

- void **callActivateOrDeactivateForce** (std::shared_ptr< Registry > registry, Entity partId, Entity entityId)

4.217.1 Member Function Documentation

4.217.1.1 update()

```
void ecs::ServerForceInputSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/server/systems/input/ServerForceInputSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/systems/input/ServerForceInputSystem.cpp

4.218 rserv::ServerInfo Struct Reference

Public Attributes

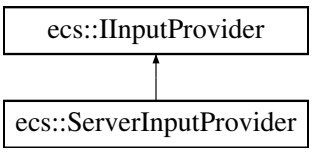
- int **connectedClients**
- std::chrono::seconds **uptime**
- int **activeLobbies**
- size_t **totalPlayers**
- std::vector< std::string > **lobbyDetails**
- std::vector< std::string > **playerDetails**
- std::vector< std::vector< std::string > > **lobbyPlayerDetails**
- std::vector< std::map< std::string, int > > **playerStats**
- std::vector< std::string > **inGamePlayers**
- std::vector< std::string > **bannedPlayers**
- int64_t **tps**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/server/http/HttpServer.hpp

4.219 ecs::ServerInputProvider Class Reference

Inheritance diagram for ecs::ServerInputProvider:



Public Member Functions

- float getAxisValue (event_t axis, size_t clientID=0) override
- bool isActionPressed (InputAction action, size_t clientID=0) override
- float getActionAxis (InputAction action, size_t clientID=0) override
- InputMapping getInputMapping (size_t clientID=0) const override
- void **setAxisValue** (ecs::InputAction action, float value, size_t clientID=0)
- void **addClientInputMapping** (size_t clientID, size_t identity, const InputMapping &mapping)
- void **registerClient** (size_t clientID)
- void **registerEntityForClient** (size_t entityId, size_t clientID)
- size_t **getClientIdForEntity** (size_t entityId) const
- void **updateInputFromEvent** (size_t clientID, constants::EventType eventType, float value)
- std::vector< size_t > **getConnectedClients** () const

Private Types

- using **InputHandler** = void (ServerInputProvider::*)(size_t, float)

Private Member Functions

- void **handleUp** (size_t clientID, float value)
- void **handleDown** (size_t clientID, float value)
- void **handleLeft** (size_t clientID, float value)
- void **handleRight** (size_t clientID, float value)
- void **handleStop** (size_t clientID, float value)
- void **handleShoot** (size_t clientID, float value)
- void **handleForce** (size_t clientID, float value)
- void **handleHealthCheck** (size_t clientID, float value)

Private Attributes

- std::vector< std::tuple< size_t, size_t, InputMapping > > **_inputMapping**
- std::map< size_t, std::map< ecs::InputAction, float > > **_clientAxisValues**
- std::map< size_t, std::map< ecs::InputAction, std::chrono::steady_clock::time_point > > **_clientInputTimestamps**
- std::set< size_t > **_registeredClients**
- std::map< size_t, size_t > **_entityToClientId**
- std::vector< InputHandler > **_inputHandlers**

Static Private Attributes

- static constexpr std::chrono::milliseconds **INPUT_TIMEOUT** = std::chrono::milliseconds(200)

Additional Inherited Members

Public Types inherited from ecs::IInputProvider

- using **event_t** = gfx::EventType

4.219.1 Member Function Documentation

4.219.1.1 getActionAxis()

```
float ecs::ServerInputProvider::getActionAxis (
            InputAction action,
            size_t clientID = 0)  [override], [virtual]
```

Implements ecs::IInputProvider.

4.219.1.2 getAxisValue()

```
float ecs::ServerInputProvider::getAxisValue (
            event_t axis,
            size_t clientID = 0)  [override], [virtual]
```

Implements ecs::IInputProvider.

4.219.1.3 getInputMapping()

```
InputMapping ecs::ServerInputProvider::getInputMapping (
            size_t clientID = 0) const  [override], [virtual]
```

Implements ecs::IInputProvider.

4.219.1.4 isActionPressed()

```
bool ecs::ServerInputProvider::isActionPressed (
            InputAction action,
            size_t clientID = 0)  [override], [virtual]
```

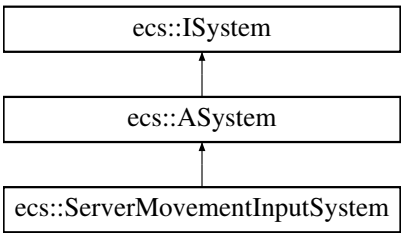
Implements ecs::IInputProvider.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/server/initResourcesManager/ServerInputProvider.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/initResourcesManager/ServerInputProvider.cpp

4.220 ecs::ServerMovementInputSystem Class Reference

Inheritance diagram for ecs::ServerMovementInputSystem:



Public Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Private Member Functions

- math::Vector2f **getMovementDirection** (std::shared_ptr< IInputProvider > inputProvider, size_t clientID) const
- void **updateInputIntent** (std::shared_ptr< Registry > registry, Entity entityId, const math::Vector2f &direction)
- math::Vector2f **getAnalogStickInput** (std::shared_ptr< IInputProvider > inputProvider, size_t clientID) const
- math::Vector2f **normalizeDirection** (const math::Vector2f &direction) const

4.220.1 Member Function Documentation

4.220.1.1 update()

```
void ecs::ServerMovementInputSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

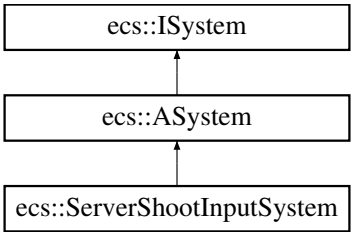
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/server/systems/input/ServerMovementInputSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/systems/input/ServerMovementInputSystem.cpp

4.221 ecs::ServerShootInputSystem Class Reference

Inheritance diagram for ecs::ServerShootInputSystem:



Public Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Private Member Functions

- void **updateShootIntent** (std::shared_ptr< Registry > registry, Entity entityId)

Private Attributes

- std::map< ecs::Entity, float > **_cooldowns**

4.221.1 Member Function Documentation

4.221.1.1 update()

```
void ecs::ServerShootInputSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/server/systems/input/ServerShootInputSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/systems/input/ServerShootInputSystem.cpp

4.222 SettingsConfig Class Reference

Public Types

- enum class **ScreenResolution** {
  RES_800x600 = 0 , **RES_1024x768** = 1 , **RES_1280x720** = 2 , **RES_1920x1080** = 3 ,
  FULLSCREEN = 4 }

Public Member Functions

- int **getColorBlindnessState** () const
- void **setColorBlindnessState** (int state)
- float **getBrightnessValue** () const
- void **setBrightnessValue** (float value)
- bool **isHighContrastEnabled** () const
- void **setHighContrastEnabled** (bool enabled)
- ui::UIScale **getUIScale** () const
- void **setUIScale** (ui::UIScale scale)
- float **getMusicVolume** () const
- void **setMusicVolume** (float volume)
- float **getSoundVolume** () const
- void **setSoundVolume** (float volume)
- ScreenResolution **getScreenResolution** () const
- void **setScreenResolution** (ScreenResolution resolution)
- int **getTargetFPS** () const
- void **setTargetFPS** (int fps)
- float **getRenderQuality** () const
- void **setRenderQuality** (float quality)
- bool **isInGameMetricsEnabled** () const
- void **setInGameMetricsEnabled** (bool enabled)
- std::string **getUsername** () const
- void **setUsername** (const std::string &username)
- std::string **getScreenResolutionName** (ScreenResolution resolution) const
- std::pair< int, int > **getScreenResolutionSize** (ScreenResolution resolution) const
- bool **isFullscreen** (ScreenResolution resolution) const
- void **saveAccessibility** (const std::string &filepath=constants::ACCESSIBILITY_FILE_PATH)
- void **loadAccessibility** (const std::string &filepath=constants::ACCESSIBILITY_FILE_PATH)
- void **saveSettings** (const std::string &filepath=constants::SETTINGS_FILE_PATH)
- void **loadSettings** (const std::string &filepath=constants::SETTINGS_FILE_PATH)

Private Attributes

- int **_colorBlindnessState** = 0
- float **_brightnessValue** = 1.0f
- bool **_highContrastEnabled** = false
- ui::UIScale **_uiScale** = ui::UIScale::Normal
- float **_musicVolume** = 100.0f
- float **_soundVolume** = 100.0f
- ScreenResolution **_screenResolution** = ScreenResolution::RES_1920x1080
- int **_targetFPS** = 60
- float **_renderQuality** = 1.0f
- bool **_inGameMetricsEnabled** = false
- std::string **_username** = ""

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/SettingsConfig.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/SettingsConfig.cpp

4.223 SettingsManager Class Reference

Public Member Functions

- **SettingsManager** (std::shared_ptr< ecs::InputMappingManager > mappingManager, std::shared_ptr< ecs::IInputProvider > inputProvider, std::shared_ptr< SettingsConfig > settingsConfig)
- void **loadAll** ()
- void **saveAll** ()
- void **saveKeybinds** ()
- void **loadKeybinds** ()
- void **saveAccessibility** ()
- void **loadAccessibility** ()
- void **saveSettings** ()
- void **loadSettings** ()
- void **applyAccessibilityToWindow** (std::shared_ptr< gfx::IWindow > window)

Private Attributes

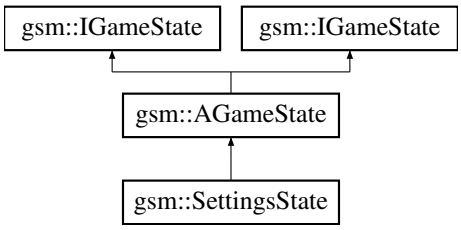
- std::shared_ptr< ecs::InputMappingManager > **_mappingManager**
- std::shared_ptr< ecs::IInputProvider > **_inputProvider**
- std::shared_ptr< SettingsConfig > **_settingsConfig**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/SettingsManager.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/SettingsManager.cpp

4.224 gsm::SettingsState Class Reference

Inheritance diagram for gsm::SettingsState:



Public Member Functions

- **SettingsState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- void enter () override
- void update (float deltaTime) override
- void exit () override
- std::string getStateName () const override

Public Member Functions inherited from gsm::AGameState

- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override
- **AGameState** (std::shared_ptr< IGameStateMachine > gsm, std::shared_ptr< ResourceManager > resourceManager)
- std::vector< std::shared_ptr< ecs::ISystem > > getSystems () const override

Private Member Functions

- void **renderUI** ()
- void **cycleColorBlindnessFilter** ()
- void **toggleHighContrastFilter** ()
- void **updateBrightnessFilter** (float value)
- void **applyColorBlindnessFilter** (int state)
- void **applyHighContrastFilter** (bool enabled)
- void **cycleUIScale** ()
- void **updateMusicVolume** (float value)
- void **updateSoundVolume** (float value)
- void **updateToggleValue** (bool value)
- void **cycleScreenResolution** ()
- void **updateTargetFPS** (int fps)
- void **updateRenderQuality** (float quality)
- void **updateInGameMetrics** (bool enabled)
- void **setScreenResolution** (SettingsConfig::ScreenResolution resolution)
- void **updateResolutionButtonColors** (SettingsConfig::ScreenResolution current)
- void **startKeyRebind** (ecs::RemappableAction action, bool rebindPrimary, std::shared_ptr< ui::Button > button)
- void **handleKeyRebind** (gfx::EventType newKey)
- void **updateKeyBindingButtonText** (std::shared_ptr< ui::Button > button, ecs::RemappableAction action, bool isPrimary)
- std::string **getRemappableActionName** (ecs::RemappableAction action) const
- std::string **getScreenResolutionText** (SettingsConfig::ScreenResolution resolution)
- std::string **getColorBlindnessText** (int state)
- std::string **getUIScaleText** (ui::UIScale scale)

Private Attributes

- std::unique_ptr< MouseInputHandler > **_mouseHandler**
- std::shared_ptr< ui::Button > **_backButton**
- std::shared_ptr< ui::Button > **_highContrastButton**
- std::shared_ptr< ui::Button > **_colorBlindnessButton**
- std::shared_ptr< ui::Slider > **_brightnessSlider**
- std::shared_ptr< ui::Slider > **_musicVolumeSlider**
- std::shared_ptr< ui::Slider > **_soundVolumeSlider**
- std::shared_ptr< ui::ToggleSwitch > **_toggleSwitch**
- std::shared_ptr< ui::Text > **_toggleLabel**
- std::shared_ptr< ui::UILayout > **_toggleLayout**
- std::vector< std::shared_ptr< ui::Button > > **_resolutionButtons**
- std::shared_ptr< ui::Slider > **_fpsSlider**
- std::shared_ptr< ui::Slider > **_renderQualitySlider**
- std::shared_ptr< ui::Button > **_scaleButton**
- std::unique_ptr< ui::UIManager > **_uiManager**
- std::shared_ptr< ui::UILayout > **_settingsLayout**
- std::shared_ptr< ui::UILayout > **_leftColumnLayout**
- std::shared_ptr< ui::UILayout > **_rightColumnLayout**
- std::shared_ptr< ui::UILayout > **_centerColumnLayout**
- std::shared_ptr< ui::UILayout > **_titleLayout**
- std::shared_ptr< ui::Background > **_background**
- math::Vector2f **_savedViewCenter**
- std::shared_ptr< SettingsManager > **_settingsManager**
- std::shared_ptr< ui::UILayout > **_moveUpLayout**
- std::shared_ptr< ui::Text > **_moveUpLabel**
- std::shared_ptr< ui::Button > **_moveUpPrimaryButton**
- std::shared_ptr< ui::Button > **_moveUpSecondaryButton**
- std::shared_ptr< ui::UILayout > **_moveDownLayout**
- std::shared_ptr< ui::Text > **_moveDownLabel**
- std::shared_ptr< ui::Button > **_moveDownPrimaryButton**
- std::shared_ptr< ui::Button > **_moveDownSecondaryButton**
- std::shared_ptr< ui::UILayout > **_moveLeftLayout**
- std::shared_ptr< ui::Text > **_moveLeftLabel**
- std::shared_ptr< ui::Button > **_moveLeftPrimaryButton**
- std::shared_ptr< ui::Button > **_moveLeftSecondaryButton**
- std::shared_ptr< ui::UILayout > **_moveRightLayout**
- std::shared_ptr< ui::Text > **_moveRightLabel**
- std::shared_ptr< ui::Button > **_moveRightPrimaryButton**
- std::shared_ptr< ui::Button > **_moveRightSecondaryButton**
- std::shared_ptr< ui::UILayout > **_shootLayout**
- std::shared_ptr< ui::Text > **_shootLabel**
- std::shared_ptr< ui::Button > **_shootPrimaryButton**
- std::shared_ptr< ui::Button > **_shootSecondaryButton**
- std::shared_ptr< ui::UILayout > **_forceLayout**
- std::shared_ptr< ui::Text > **_forceLabel**
- std::shared_ptr< ui::Button > **_forcePrimaryButton**
- std::shared_ptr< ui::Button > **_forceSecondaryButton**
- bool **_isWaitingForKey** = false
- std::optional< ecs::RemappableAction > **_actionToRebind**
- bool **_rebindingPrimary** = true
- std::string **_rebindLabel**
- std::shared_ptr< ui::Button > **_buttonToUpdate**
- gfx::EventType **_originalKey** = gfx::EventType::NOTHING
- std::shared_ptr< ui::ToggleSwitch > **_inGameMetricsToggle**
- std::shared_ptr< ui::Text > **_inGameMetricsLabel**

Additional Inherited Members

Protected Member Functions inherited from gsm::AGameState

- void addSystem (std::shared_ptr< ecs::ISystem > system) override
- void addSystem (std::shared_ptr< ecs::ISystem > system) override

Protected Attributes inherited from gsm::AGameState

- std::weak_ptr< IGameStateMachine > **_gsm**
- std::shared_ptr< ResourceManager > **_resourceManager**
- std::vector< std::shared_ptr< ecs::ISystem > > **_systems**

4.224.1 Member Function Documentation

4.224.1.1 enter()

```
void gsm::SettingsState::enter ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.224.1.2 exit()

```
void gsm::SettingsState::exit ()  [override], [virtual]
```

Reimplemented from gsm::AGameState.

4.224.1.3 getStateName()

```
std::string gsm::SettingsState::getStateName () const  [inline], [override], [virtual]
```

Implements gsm::AGameState.

4.224.1.4 update()

```
void gsm::SettingsState::update (
            float deltaTime)  [override], [virtual]
```

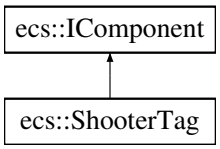
Reimplemented from gsm::AGameState.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/Settings/SettingsState.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/Settings/SettingsState.cpp

4.225 ecs::ShooterTag Class Reference

Inheritance diagram for ecs::ShooterTag:

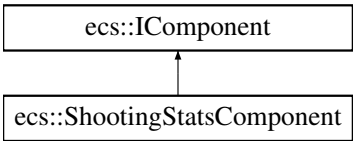


The documentation for this class was generated from the following file:

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/tags/ShooterTag.hpp

4.226 ecs::ShootingStatsComponent Class Reference

Inheritance diagram for ecs::ShootingStatsComponent:



Public Member Functions

- **ShootingStatsComponent** (float fireRate=1.0f, const MultiShotPattern &pattern=MultiShotPattern())
- float **getFireRate** () const
- void **setFireRate** (float fireRate)
- MultiShotPattern **getMultiShotPattern** () const
- void **setMultiShotPattern** (const MultiShotPattern &pattern)
- float **getCooldownTimer** () const
- void **setCooldownTimer** (float timer)

Private Attributes

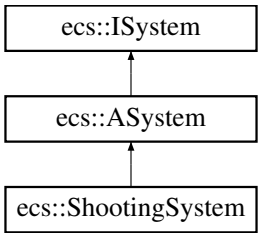
- float **_fireRate**
- MultiShotPattern **_multiShotPattern**
- float **_cooldownTimer**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/permanent/ShootingStatsComponent.↩ hpp

4.227 ecs::ShootingSystem Class Reference

Inheritance diagram for ecs::ShootingSystem:



Public Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Private Member Functions

- void **spawnProjectile** (std::shared_ptr< Registry > registry, std::shared_ptr< ResourceManager > resourceManager, const std::string &prefabName, const math::Vector2f &position, float angle, ecs::Entity shooterEntity)
- math::Vector2f **calculateProjectileVelocity** (float angle, float speed)

4.227.1 Member Function Documentation

4.227.1.1 update()

```
void ecs::ShootingSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

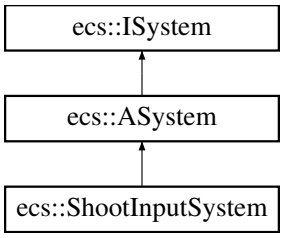
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/shooting/ShootingSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/shooting/ShootingSystem.cpp

4.228 ecs::ShootInputSystem Class Reference

Inheritance diagram for ecs::ShootInputSystem:



Public Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Private Member Functions

- bool **isPlayerAlive** (std::shared_ptr< Registry > registry, Entity entityId) const

4.228.1 Member Function Documentation

4.228.1.1 update()

```
void ecs::ShootInputSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

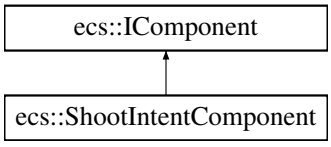
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/input/ShootInputSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/input/ShootInputSystem.cpp

4.229 ecs::ShootIntentComponent Class Reference

Inheritance diagram for ecs::ShootIntentComponent:



Public Member Functions

- **ShootIntentComponent** (float angle=0.0f)
- void **setAngle** (float angle)
- float **getAngle** () const

Private Attributes

- float **_angle**
- math::Vector2f **_position**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/temporary/ShootIntentComponent.hpp

4.230 Signal Class Reference

Static Public Member Functions

- static void **signalHandler** (int signum)
- static void **setupSignalHandlers** ()

Static Public Attributes

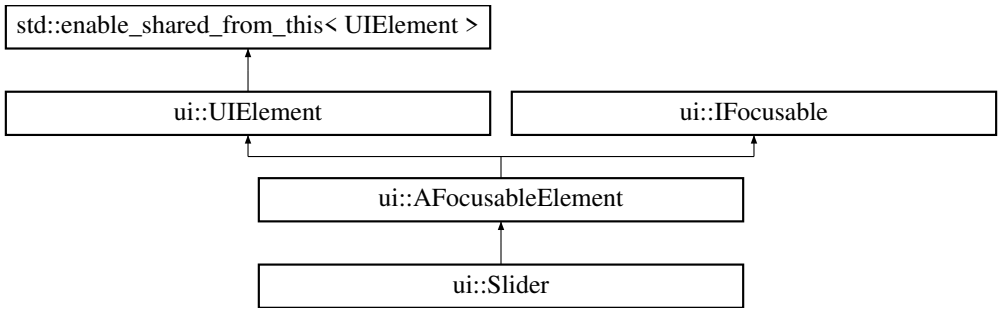
- static volatile sig_atomic_t **stopFlag** = 0

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Signal/Signal.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/Signal/Signal.cpp

4.231 ui::Slider Class Reference

Inheritance diagram for ui::Slider:



Public Member Functions

- **Slider** (std::shared_ptr< ResourceManager > resourceManager)
- void **setMinValue** (float minValue)
- void **setMaxValue** (float maxValue)
- void **setValue** (float value)
- float **getValue** () const
- float **getMinValue** () const
- float **getMaxValue** () const
- void **setStep** (float step)
- float **getStep** () const
- void **setLabel** (const std::string &label)
- const std::string & **getLabel** () const
- void **setLabelColor** (const gfx::color_t &color)
- void **setFontPath** (const std::string &fontPath)
- void **setBaseFontSize** (size_t fontSize)
- size_t **getBaseFontSize** () const
- void **setShowPercentage** (bool show)
- void **setTrackColor** (const gfx::color_t &color)
- void **setFillColor** (const gfx::color_t &color)
- void **setHandleColor** (const gfx::color_t &color)
- void **setHandleHoveredColor** (const gfx::color_t &color)
- void **setHandleFocusedColor** (const gfx::color_t &color)
- void **setOnValueChanged** (std::function< void(float)> callback)
- virtual void render () override
- virtual void handleInput (const math::Vector2f &mousePos, bool mousePressed) override
- virtual void onActivated () override
- virtual bool onNavigateLeft () override
- virtual bool onNavigateRight () override
- void **incrementValue** ()
- void **decrementValue** ()

Public Member Functions inherited from ui::AFocusableElement

- **AFocusableElement** (std::shared_ptr< ResourceManager > resourceManager)
- virtual void setFocused (bool focused) override
- virtual bool isFocused () const override
- virtual bool canBeFocused () const override
- virtual void onFocusGained () override
- virtual void onFocusLost () override
- void **setOnFocusGained** (std::function< void()> callback)
- void **setOnFocusLost** (std::function< void()> callback)
- void **setOnActivated** (std::function< void()> callback)

Public Member Functions inherited from ui::UIElement

- **UIElement** (std::shared_ptr< ResourceManager > resourceManager)
- void **setPosition** (const math::Vector2f &position)
- void **setSize** (const math::Vector2f &size)
- math::Vector2f **getPosition** () const
- math::Vector2f **getSize** () const
- math::Vector2f **getAbsolutePosition** () const
- math::Vector2f **getAbsoluteSize** () const
- void **setVisible** (bool visible)
- bool **isVisible** () const
- void **setScalingEnabled** (bool enabled)
- bool **isScalingEnabled** () const
- void **setFocusEnabled** (bool enabled)
- bool **isFocusEnabled** () const
- void **setState** (UIState state)
- UIState **getState** () const
- virtual void **setScale** (UIScale scale)
- UIScale **getScale** () const
- void **setParent** (std::weak_ptr< UIElement > parent)
- std::shared_ptr< UIElement > **getParent** () const
- void **addChild** (std::shared_ptr< UIElement > child)
- void **removeChild** (std::shared_ptr< UIElement > child)
- const std::vector< std::shared_ptr< UIElement > > & **getChildren** () const
- virtual bool **containsPoint** (const math::Vector2f &point) const
- void **setOnClick** (std::function< void()> callback)
- void **setOnHover** (std::function< void()> callback)
- void **setOnRelease** (std::function< void()> callback)
- virtual void **update** (float deltaTime)

Private Member Functions

- float **getNormalizedValue** () const
- void **setNormalizedValue** (float normalized)
- gfx::color_t **getCurrentHandleColor** () const
- size_t **getFontSize** () const
- float **getHandleRadius** () const
- float **getTrackHeight** () const
- float **getLabelHeight** () const

Private Attributes

- float **_minValue** = 0.0f
- float **_maxValue** = 1.0f
- float **_value** = 0.5f
- float **_step** = 0.1f
- float **_visualNormalizedValue** = 0.5f
- std::string **_label**
- gfx::color_t **_labelColor** = colors::SLIDER_LABEL
- std::string **_fontPath** = constants::MAIN_FONT
- size_t **_baseFontSize** = constants::BUTTON_FONT_SIZE_BASE
- float **_outlineThickness** = 2.0f
- bool **_showPercentage** = true

- gfx::color_t **_trackColor** = colors::SLIDER_TRACK
- gfx::color_t **_fillColor** = colors::SLIDER_FILL
- gfx::color_t **_handleColor** = colors::SLIDER_HANDLE
- gfx::color_t **_handleHoveredColor** = colors::SLIDER_HANDLE_HOVER
- gfx::color_t **_handleFocusedColor** = colors::SLIDER_HANDLE_FOCUSED
- std::function< void(float)> **_onValueChanged**
- bool **_isDragging** = false
- bool **_wasMousePressed** = false

Additional Inherited Members

Protected Member Functions inherited from ui::AFocusableElement

- virtual void **onFocusStateChanged** (bool focused)

Protected Member Functions inherited from ui::UIElement

- std::pair< int, int > **getWindowSize** () const
- std::pair< int, int > **getLogicalSize** () const
- float **getScaleFactor** () const

Protected Attributes inherited from ui::AFocusableElement

- bool **_focused** = false
- bool **_pressedInside** = false
- bool **_wasPressed** = false
- std::function< void()> **_onFocusGained**
- std::function< void()> **_onFocusLost**
- std::function< void()> **_onActivated**

Protected Attributes inherited from ui::UIElement

- std::weak_ptr< ResourceManager > **_resourceManager**
- math::Vector2f **_position**
- math::Vector2f **_size**
- bool **_visible** = true
- bool **_scalingEnabled** = true
- bool **_focusEnabled** = true
- UIState **_state** = UIState::Normal
- UIScale **_scale** = UIScale::Normal
- std::weak_ptr< UIElement > **_parent**
- std::vector< std::shared_ptr< UIElement > > **_children**
- bool **_pressedInside** = false
- bool **_wasPressed** = false
- std::function< void()> **_onClick**
- std::function< void()> **_onHover**
- std::function< void()> **_onRelease**

4.231.1 Member Function Documentation

4.231.1.1 handleInput()

```
void ui::Slider::handleInput (
            const math::Vector2f & mousePos,
            bool mousePressed)  [override], [virtual]
```

Reimplemented from ui::AFocusableElement.

4.231.1.2 onActivated()

```
void ui::Slider::onActivated ()  [override], [virtual]
```

Reimplemented from ui::AFocusableElement.

4.231.1.3 onNavigateLeft()

```
bool ui::Slider::onNavigateLeft ()  [override], [virtual]
```

Reimplemented from ui::IFocusable.

4.231.1.4 onNavigateRight()

```
bool ui::Slider::onNavigateRight ()  [override], [virtual]
```

Reimplemented from ui::IFocusable.

4.231.1.5 render()

```
void ui::Slider::render ()  [override], [virtual]
```

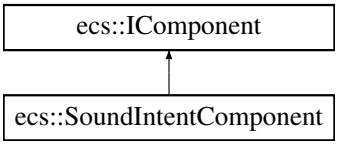
Reimplemented from ui::UIElement.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/elements/focusable/Slider.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/elements/focusable/Slider.cpp

4.232 ecs::SoundIntentComponent Class Reference

Inheritance diagram for ecs::SoundIntentComponent:



Public Member Functions

- **SoundIntentComponent** (const std::string &soundPath="", float volume=100.0f)
- std::string **getSoundPath** () const
- void **setSoundPath** (const std::string &soundPath)
- float **getVolume** () const
- void **setVolume** (float volume)

Private Attributes

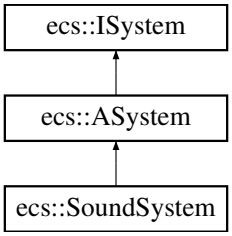
- std::string **_soundPath**
- float **_volume**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/components/temporary/SoundIntentComponent.hpp

4.233 ecs::SoundSystem Class Reference

Inheritance diagram for ecs::SoundSystem:



Protected Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Additional Inherited Members

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

4.233.1 Member Function Documentation

4.233.1.1 update()

```
void ecs::SoundSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [protected], [virtual]
```

Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/audio/SoundSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/audio/SoundSystem.cpp

4.234 ecs::SpatialGrid Class Reference

Public Member Functions

- **SpatialGrid** (float worldWidth=constants::MAX_WIDTH, float worldHeight=constants::MAX_HEIGHT, float cellSize=constants::SPATIAL_GRID_CELL_SIZE, float padding=constants::SPATIAL_GRID_PADDING)
- void **clear** ()
- void **insert** (Entity entityId, const math::FRect &bounds)
- std::vector< Entity > **query** (const math::FRect &bounds) const
- std::vector< std::pair< Entity, Entity > > **getPotentialPairs** () const
- void **setCellSize** (float cellSize)
- void **setOffset** (float offsetX, float offsetY)
- float **getCellSize** () const
- size_t **getNumCols** () const
- size_t **getNumRows** () const
- float **getOffsetX** () const
- float **getOffsetY** () const

Private Member Functions

- size_t **getCellIndex** (float x, float y) const
- std::vector< size_t > **getCellIndices** (const math::FRect &bounds) const

Private Attributes

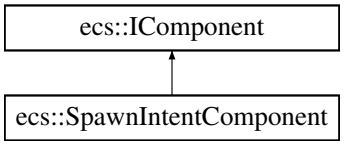
- float **_worldWidth**
- float **_worldHeight**
- float **_cellSize**
- float **_padding**
- float **_offsetX**
- float **_offsetY**
- size_t **_numCols**
- size_t **_numRows**
- std::vector< std::vector< Entity > > **_cells**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/SpatialGrid/SpatialGrid.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/SpatialGrid/SpatialGrid.cpp

4.235 ecs::SpawnIntentComponent Class Reference

Inheritance diagram for ecs::SpawnIntentComponent:



Public Member Functions

- **SpawnIntentComponent** (const std::string &prefabName, const math::Vector2f &position, float gameView↩ XTrigger=0.0f)
- **SpawnIntentComponent** (const std::string &prefabName, const math::Vector2f &position, const EntityCreationContext &context, float gameViewXTrigger=0.0f)
- void **setPrefabName** (const std::string &prefabName)
- std::string **getPrefabName** () const
- void **setPosition** (const math::Vector2f &position)
- math::Vector2f **getPosition** () const
- void **setCreationContext** (const EntityCreationContext &context)
- EntityCreationContext **getCreationContext** () const
- void **setGameViewXTrigger** (const float &gameViewXTrigger)
- float **getGameViewXTrigger** () const

Private Attributes

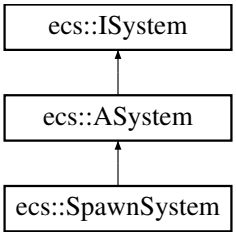
- std::string **_prefabName**
- math::Vector2f **_position**
- EntityCreationContext **_creationContext**
- float **_gameViewXTrigger**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/temporary/SpawnIntentComponent.↩ hpp

4.236 ecs::SpawnSystem Class Reference

Inheritance diagram for ecs::SpawnSystem:



Public Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Private Member Functions

- bool **isPositionFree** (Entity newEntity, const math::Vector2f &position, const std::vector< std::shared_ptr< ColliderComponent > > &newColliders, std::shared_ptr< TransformComponent > newTransform, std::shared_ptr< Registry > registry)
- math::Vector2f **findNearestFreePosition** (Entity newEntity, const math::Vector2f &originalPosition, std::shared_ptr< Registry > registry, float stepSize=10.0f)

4.236.1 Member Function Documentation

4.236.1.1 update()

```
void ecs::SpawnSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

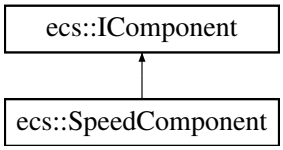
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/spawn/SpawnSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/spawn/SpawnSystem.cpp

4.237 ecs::SpeedComponent Class Reference

Inheritance diagram for ecs::SpeedComponent:



Public Member Functions

- **SpeedComponent** (float speed=constants::BASE_SPEED)
- float **getSpeed** () const
- void **setSpeed** (float speed)

Private Attributes

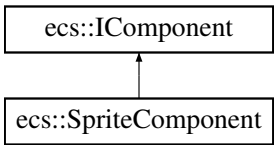
- float **_speed**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/permanent/SpeedComponent.hpp

4.238 ecs::SpriteComponent Class Reference

Inheritance diagram for ecs::SpriteComponent:



Public Member Functions

- **SpriteComponent** (const std::string &texturePath)
- const std::string & **getTexturePath** () const
- void **setTexturePath** (const std::string &path)
- bool **isValid** () const

Private Attributes

- std::string **_texturePath**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/components/rendering/SpriteComponent.hpp

4.239 ui::SpritePreview::SpriteData Struct Reference

Public Attributes

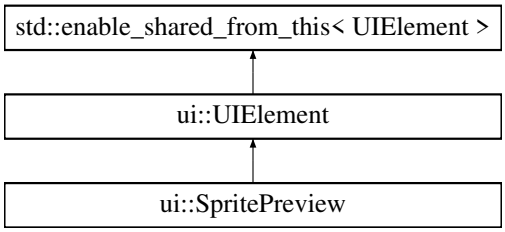
- std::string **texturePath**
- math::Vector2f **frameSize**
- math::Vector2f **position**
- math::Vector2f **scale**
- float **rotation** = 0.0f

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/elements/SpritePreview.hpp

4.240 ui::SpritePreview Class Reference

Inheritance diagram for ui::SpritePreview:



Classes

- struct AnimationData
- struct SpriteData

Public Member Functions

- **SpritePreview** (std::shared_ptr< ResourceManager > resourceManager)
- bool **loadPrefab** (const std::filesystem::path &prefabPath)
- void **setDisplayBounds** (const math::Vector2f &position, const math::Vector2f &size)
- void **setTransform** (float scale, float rotation)
- void render () override
- void update (float deltaTime) override
- void **clear** ()

Public Member Functions inherited from ui::UIElement

- **UIElement** (std::shared_ptr< ResourceManager > resourceManager)
- void **setPosition** (const math::Vector2f &position)
- void **setSize** (const math::Vector2f &size)
- math::Vector2f **getPosition** () const
- math::Vector2f **getSize** () const
- math::Vector2f **getAbsolutePosition** () const
- math::Vector2f **getAbsoluteSize** () const
- void **setVisible** (bool visible)
- bool **isVisible** () const
- void **setScalingEnabled** (bool enabled)
- bool **isScalingEnabled** () const
- void **setFocusEnabled** (bool enabled)
- bool **isFocusEnabled** () const
- void **setState** (UIState state)
- UIState **getState** () const
- virtual void **setScale** (UIScale scale)
- UIScale **getScale** () const
- void **setParent** (std::weak_ptr< UIElement > parent)
- std::shared_ptr< UIElement > **getParent** () const
- void **addChild** (std::shared_ptr< UIElement > child)
- void **removeChild** (std::shared_ptr< UIElement > child)
- const std::vector< std::shared_ptr< UIElement > > & **getChildren** () const
- virtual void **handleInput** (const math::Vector2f &mousePos, bool mousePressed)
- virtual bool **containsPoint** (const math::Vector2f &point) const
- void **setOnClick** (std::function< void()> callback)
- void **setOnHover** (std::function< void()> callback)
- void **setOnRelease** (std::function< void()> callback)

Private Types

- enum class **PreviewType** { **None** , **Sprite** , **Animation** }

Private Member Functions

- bool **extractSpriteFromPrefab** (const nlohmann::json &prefab)
- bool **extractAnimationFromPrefab** (const nlohmann::json &prefab)
- void **renderSprite** ()
- void **renderAnimation** ()

Private Attributes

- PreviewType **_type** = PreviewType::None
- SpriteData **_spriteData**
- AnimationData **_animationData**
- float **_currentFrame** = 0.0f
- float **_animationTime** = 0.0f
- bool **_loaded** = false

Additional Inherited Members

Protected Member Functions inherited from ui::UIElement

- std::pair< int, int > **getWindowSize** () const
- std::pair< int, int > **getLogicalSize** () const
- float **getScaleFactor** () const

Protected Attributes inherited from ui::UIElement

- std::weak_ptr< ResourceManager > **_resourceManager**
- math::Vector2f **_position**
- math::Vector2f **_size**
- bool **_visible** = true
- bool **_scalingEnabled** = true
- bool **_focusEnabled** = true
- UIState **_state** = UIState::Normal
- UIScale **_scale** = UIScale::Normal
- std::weak_ptr< UIElement > **_parent**
- std::vector< std::shared_ptr< UIElement > > **_children**
- bool **_pressedInside** = false
- bool **_wasPressed** = false
- std::function< void()> **_onClick**
- std::function< void()> **_onHover**
- std::function< void()> **_onRelease**

4.240.1 Member Function Documentation

4.240.1.1 render()

```
void ui::SpritePreview::render ()  [override], [virtual]
```

Reimplemented from ui::UIElement.

4.240.1.2 update()

```
void ui::SpritePreview::update (
            float deltaTime)  [override], [virtual]
```

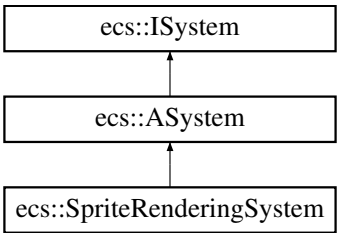
Reimplemented from ui::UIElement.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/elements/SpritePreview.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/elements/SpritePreview.cpp

4.241 ecs::SpriteRenderingSystem Class Reference

Inheritance diagram for ecs::SpriteRenderingSystem:



Protected Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Additional Inherited Members

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

4.241.1 Member Function Documentation

4.241.1.1 update()

```
void ecs::SpriteRenderingSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [protected], [virtual]
```

Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/rendering/SpriteRenderingSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/rendering/SpriteRenderingSystem.cpp

4.242 SystemConfig Class Reference

Static Public Member Functions

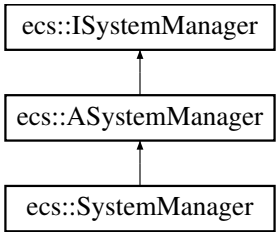
- static long **getPageSize** ()

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/InGame/InGameState.hpp

4.243 ecs::SystemManager Class Reference

Inheritance diagram for ecs::SystemManager:



Additional Inherited Members

Public Member Functions inherited from ecs::ASystemManager

- void updateAllSystems (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override
- void addSystem (std::shared_ptr< ISystem > system) override
- void removeSystem (std::shared_ptr< ISystem > system) override
- void clearAllSystems () override

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/systemManager/SystemManager.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/systemManager/SystemManager.cpp

4.244 parser::TagComponentRegistrar< T > Class Template Reference

Public Member Functions

- **TagComponentRegistrar** (const std::string &name)

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/ComponentRegistry/ComponentRegistrar.↩ hpp

4.245 TagRegistry Class Reference

Public Member Functions

- template<typename T>
  void **registerTag** (const std::string &tagName)
- bool **hasTag** (std::shared_ptr< ecs::Registry > registry, ecs::Entity entity, const std::string &tagName) const
- std::vector< std::string > **getTags** (std::shared_ptr< ecs::Registry > registry, ecs::Entity entity) const

Static Public Member Functions

- static const TagRegistry & **getInstance** ()

Private Member Functions

- **TagRegistry** (const TagRegistry &)=delete
- TagRegistry & **operator=** (const TagRegistry &)=delete
- void **initializeTags** ()

Private Attributes

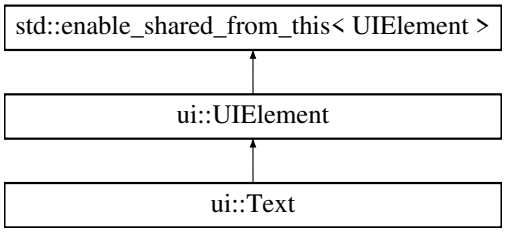
- std::unordered_map< std::string, std::function< bool(std::shared_ptr< ecs::Registry >, ecs::Entity)> > _↩ **tagCheckers**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/interactions/TagRegistry.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/interactions/TagRegistry.cpp

4.246 ui::Text Class Reference

Inheritance diagram for ui::Text:



Public Member Functions

- **Text** (std::shared_ptr< ResourceManager > resourceManager)
- void render () override
- void update (float deltaTime) override
- void setScale (UIScale scale) override
- void **setText** (const std::string &text)
- std::string **getText** () const
- void **setTextColor** (const gfx::color_t &color)
- void **setFontSize** (unsigned int size)
- void **setFontPath** (const std::string &path)
- void **setOutlineColor** (const gfx::color_t &color)
- void **setOutlineThickness** (float thickness)

Public Member Functions inherited from ui::UIElement

- **UIElement** (std::shared_ptr< ResourceManager > resourceManager)
- void **setPosition** (const math::Vector2f &position)
- void **setSize** (const math::Vector2f &size)
- math::Vector2f **getPosition** () const
- math::Vector2f **getSize** () const
- math::Vector2f **getAbsolutePosition** () const
- math::Vector2f **getAbsoluteSize** () const
- void **setVisible** (bool visible)
- bool **isVisible** () const
- void **setScalingEnabled** (bool enabled)
- bool **isScalingEnabled** () const
- void **setFocusEnabled** (bool enabled)
- bool **isFocusEnabled** () const
- void **setState** (UIState state)
- UIState **getState** () const
- UIScale **getScale** () const
- void **setParent** (std::weak_ptr< UIElement > parent)
- std::shared_ptr< UIElement > **getParent** () const
- void **addChild** (std::shared_ptr< UIElement > child)
- void **removeChild** (std::shared_ptr< UIElement > child)
- const std::vector< std::shared_ptr< UIElement > > & **getChildren** () const
- virtual void **handleInput** (const math::Vector2f &mousePos, bool mousePressed)
- virtual bool **containsPoint** (const math::Vector2f &point) const
- void **setOnClick** (std::function< void()> callback)
- void **setOnHover** (std::function< void()> callback)
- void **setOnRelease** (std::function< void()> callback)

Private Attributes

- std::string **_text**
- gfx::color_t **_textColor**
- unsigned int **_fontSize**
- unsigned int **_baseFontSize**
- std::string **_fontPath**
- gfx::color_t **_outlineColor**
- float **_outlineThickness**

Additional Inherited Members

Protected Member Functions inherited from ui::UIElement

- std::pair< int, int > **getWindowSize** () const
- std::pair< int, int > **getLogicalSize** () const
- float **getScaleFactor** () const

Protected Attributes inherited from ui::UIElement

- std::weak_ptr< ResourceManager > **_resourceManager**
- math::Vector2f **_position**
- math::Vector2f **_size**
- bool **_visible** = true
- bool **_scalingEnabled** = true
- bool **_focusEnabled** = true
- UIState **_state** = UIState::Normal
- UIScale **_scale** = UIScale::Normal
- std::weak_ptr< UIElement > **_parent**
- std::vector< std::shared_ptr< UIElement > > **_children**
- bool **_pressedInside** = false
- bool **_wasPressed** = false
- std::function< void()> **_onClick**
- std::function< void()> **_onHover**
- std::function< void()> **_onRelease**

4.246.1 Member Function Documentation

4.246.1.1 render()

```
void ui::Text::render ()  [override], [virtual]
```

Reimplemented from ui::UIElement.

4.246.1.2 setScale()

```
void ui::Text::setScale (
            UIScale scale)  [override], [virtual]
```

Reimplemented from ui::UIElement.

4.246.1.3 update()

```
void ui::Text::update (
            float deltaTime)  [override], [virtual]
```

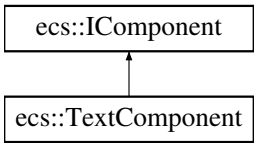
Reimplemented from ui::UIElement.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/elements/Text.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/elements/Text.cpp

4.247 ecs::TextComponent Class Reference

Inheritance diagram for ecs::TextComponent:



Public Member Functions

- **TextComponent** (const std::string &text, const std::string &fontPath, gfx::color_t color=colors::WHITE)
- const std::string & **getText** () const
- const std::string & **getFontPath** () const
- const gfx::color_t & **getColor** () const
- void **setText** (const std::string &text)
- void **setFontPath** (const std::string &fontPath)
- void **setColor** (const gfx::color_t &color)

Private Attributes

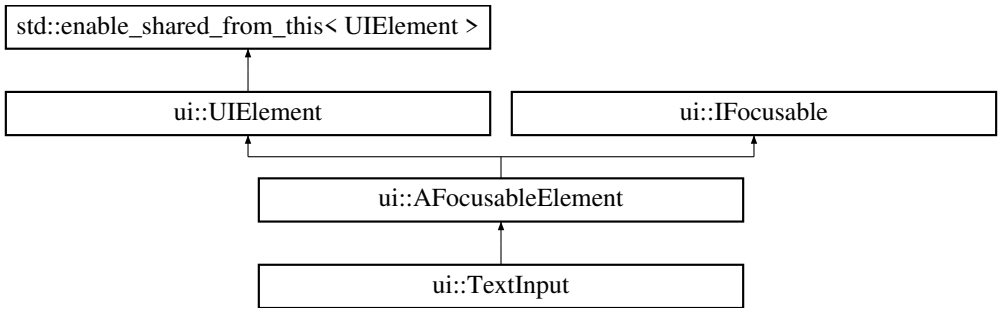
- std::string **_text**
- std::string **_fontPath**
- gfx::color_t **_color**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/components/rendering/TextComponent.hpp

4.248 ui::TextInput Class Reference

Inheritance diagram for ui::TextInput:



Public Member Functions

- **TextInput** (std::shared_ptr< ResourceManager > resourceManager)
- virtual void render () override
- void **setText** (const std::string &text)
- const std::string & **getText** () const
- void **setPlaceholder** (const std::string &placeholder)
- const std::string & **getPlaceholder** () const
- void **setTextColor** (const gfx::color_t &color)
- void **setPlaceholderColor** (const gfx::color_t &color)
- void **setFontPath** (const std::string &fontPath)
- void **setBaseFontSize** (size_t fontSize)
- size_t **getBaseFontSize** () const
- void **setOnTextChanged** (std::function< void(const std::string &)> callback)
- void **setOnSubmit** (std::function< void(const std::string &)> callback)
- void **setOnFocusLost** (std::function< void()> callback)
- void **setOnNavigate** (std::function< void(bool up)> callback)
- void **setMaxLength** (size_t maxLength)
- virtual void handleInput (const math::Vector2f &mousePos, bool mousePressed) override
- void **handleKeyboardInput** (gfx::EventType event)
- void **handleTextInput** (const std::string &text)
- virtual void update (float deltaTime) override

Public Member Functions inherited from ui::AFocusableElement

- **AFocusableElement** (std::shared_ptr< ResourceManager > resourceManager)
- virtual void setFocused (bool focused) override
- virtual bool isFocused () const override
- virtual bool canBeFocused () const override
- virtual void onFocusGained () override
- virtual void onFocusLost () override
- virtual void onActivated () override
- void **setOnFocusGained** (std::function< void()> callback)
- void **setOnFocusLost** (std::function< void()> callback)
- void **setOnActivated** (std::function< void()> callback)

Public Member Functions inherited from ui::UIElement

- **UIElement** (std::shared_ptr< ResourceManager > resourceManager)
- void **setPosition** (const math::Vector2f &position)
- void **setSize** (const math::Vector2f &size)
- math::Vector2f **getPosition** () const
- math::Vector2f **getSize** () const
- math::Vector2f **getAbsolutePosition** () const
- math::Vector2f **getAbsoluteSize** () const
- void **setVisible** (bool visible)
- bool **isVisible** () const
- void **setScalingEnabled** (bool enabled)
- bool **isScalingEnabled** () const
- void **setFocusEnabled** (bool enabled)
- bool **isFocusEnabled** () const
- void **setState** (UIState state)
- UIState **getState** () const
- virtual void **setScale** (UIScale scale)
- UIScale **getScale** () const
- void **setParent** (std::weak_ptr< UIElement > parent)
- std::shared_ptr< UIElement > **getParent** () const
- void **addChild** (std::shared_ptr< UIElement > child)
- void **removeChild** (std::shared_ptr< UIElement > child)
- const std::vector< std::shared_ptr< UIElement > > & **getChildren** () const
- virtual bool **containsPoint** (const math::Vector2f &point) const
- void **setOnClick** (std::function< void()> callback)
- void **setOnHover** (std::function< void()> callback)
- void **setOnRelease** (std::function< void()> callback)

Public Member Functions inherited from ui::IFocusable

- virtual bool **onNavigateLeft** ()
- virtual bool **onNavigateRight** ()

Private Member Functions

- void **insertChar** (char c)
- void **deleteChar** ()
- void **moveCursorLeft** ()
- void **moveCursorRight** ()
- size_t **getFontSize** () const
- void **updateCursorBlink** (float deltaTime)
- gfx::color_t **_getCurrentColor** () const

Private Attributes

- std::string **_text**
- std::string **_placeholder**
- size_t **_cursorPosition** = 0
- float **_cursorBlinkTimer** = 0.0f
- bool **_showCursor** = true
- size_t **_maxLength** = 0
- gfx::color_t **_textColor** = {0, 0, 0}
- gfx::color_t **_placeholderColor** = {128, 128, 128}
- std::string **_fontPath** = constants::MAIN_FONT
- size_t **_baseFontSize** = 24
- std::function< void(const std::string &)> **_onTextChanged**
- std::function< void(const std::string &)> **_onSubmit**
- std::function< void()> **_onFocusLost**
- std::function< void(bool)> **_onNavigate**
- gfx::color_t **_normalColor** = colors::WHITE
- gfx::color_t **_hoveredColor** = colors::LIGHT_GRAY
- gfx::color_t **_pressedColor** = colors::DARK_GRAY
- gfx::color_t **_disabledColor** = colors::UI_DISABLED
- gfx::color_t **_focusedColor** = colors::UI_FOCUSED

Additional Inherited Members

Protected Member Functions inherited from ui::AFocusableElement

- virtual void **onFocusStateChanged** (bool focused)

Protected Member Functions inherited from ui::UIElement

- std::pair< int, int > **getWindowSize** () const
- std::pair< int, int > **getLogicalSize** () const
- float **getScaleFactor** () const

Protected Attributes inherited from ui::AFocusableElement

- bool **_focused** = false
- bool **_pressedInside** = false
- bool **_wasPressed** = false
- std::function< void()> **_onFocusGained**
- std::function< void()> **_onFocusLost**
- std::function< void()> **_onActivated**

Protected Attributes inherited from ui::UIElement

- std::weak_ptr< ResourceManager > **_resourceManager**
- math::Vector2f **_position**
- math::Vector2f **_size**
- bool **_visible** = true
- bool **_scalingEnabled** = true
- bool **_focusEnabled** = true
- UIState **_state** = UIState::Normal
- UIScale **_scale** = UIScale::Normal
- std::weak_ptr< UIElement > **_parent**
- std::vector< std::shared_ptr< UIElement > > **_children**
- bool **_pressedInside** = false
- bool **_wasPressed** = false
- std::function< void()> **_onClick**
- std::function< void()> **_onHover**
- std::function< void()> **_onRelease**

4.248.1 Member Function Documentation

4.248.1.1 handleInput()

```
void ui::TextInput::handleInput (
            const math::Vector2f & mousePos,
            bool mousePressed)  [override], [virtual]
```

Reimplemented from ui::AFocusableElement.

4.248.1.2 render()

```
void ui::TextInput::render ()  [override], [virtual]
```

Reimplemented from ui::UIElement.

4.248.1.3 update()

```
void ui::TextInput::update (
            float deltaTime)  [override], [virtual]
```

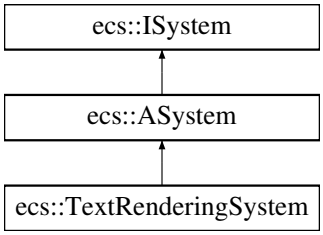
Reimplemented from ui::UIElement.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/elements/focusable/TextInput.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/elements/focusable/TextInput.cpp

4.249 ecs::TextRenderingSystem Class Reference

Inheritance diagram for ecs::TextRenderingSystem:



Protected Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Additional Inherited Members

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

4.249.1 Member Function Documentation

4.249.1.1 update()

```
void ecs::TextRenderingSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [protected], [virtual]
```

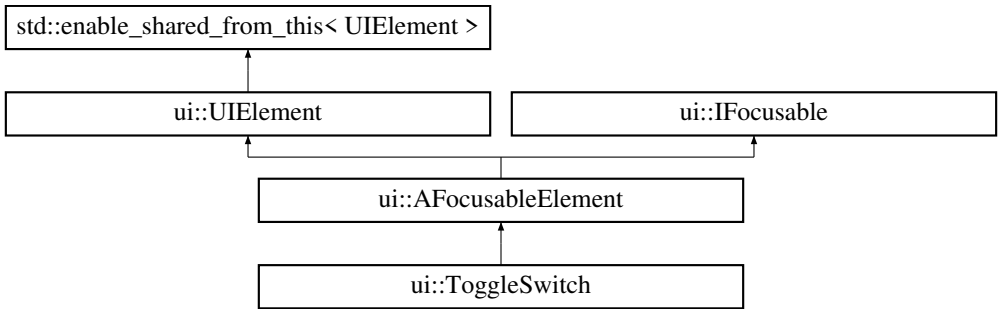
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/rendering/TextRenderingSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/systems/rendering/TextRenderingSystem.cpp

4.250 ui::ToggleSwitch Class Reference

Inheritance diagram for ui::ToggleSwitch:



Public Member Functions

- **ToggleSwitch** (std::shared_ptr< ResourceManager > resourceManager)
- void **setValue** (bool value)
- bool **getValue** () const
- void **setFontPath** (const std::string &fontPath)
- void **setBaseFontSize** (size_t fontSize)
- size_t **getBaseFontSize** () const
- void **setOnText** (const std::string &text)
- void **setOffText** (const std::string &text)
- void **setTrackColor** (const gfx::color_t &color)
- void **setHandleColor** (const gfx::color_t &color)
- void **setHandleHoveredColor** (const gfx::color_t &color)
- void **setHandleFocusedColor** (const gfx::color_t &color)
- void **setOnColor** (const gfx::color_t &color)
- void **setOffColor** (const gfx::color_t &color)
- void **setOnValueChanged** (std::function< void(bool)> callback)
- virtual void render () override
- virtual void handleInput (const math::Vector2f &mousePos, bool mousePressed) override
- virtual bool containsPoint (const math::Vector2f &point) const override

Public Member Functions inherited from ui::AFocusableElement

- **AFocusableElement** (std::shared_ptr< ResourceManager > resourceManager)
- virtual void setFocused (bool focused) override
- virtual bool isFocused () const override
- virtual bool canBeFocused () const override
- virtual void onFocusGained () override
- virtual void onFocusLost () override
- virtual void onActivated () override
- void **setOnFocusGained** (std::function< void()> callback)
- void **setOnFocusLost** (std::function< void()> callback)
- void **setOnActivated** (std::function< void()> callback)

Public Member Functions inherited from ui::UIElement

- **UIElement** (std::shared_ptr< ResourceManager > resourceManager)
- void **setPosition** (const math::Vector2f &position)
- void **setSize** (const math::Vector2f &size)
- math::Vector2f **getPosition** () const
- math::Vector2f **getSize** () const
- math::Vector2f **getAbsolutePosition** () const
- math::Vector2f **getAbsoluteSize** () const
- void **setVisible** (bool visible)
- bool **isVisible** () const
- void **setScalingEnabled** (bool enabled)
- bool **isScalingEnabled** () const
- void **setFocusEnabled** (bool enabled)
- bool **isFocusEnabled** () const
- void **setState** (UIState state)
- UIState **getState** () const
- virtual void **setScale** (UIScale scale)
- UIScale **getScale** () const

- void **setParent** (std::weak_ptr< UIElement > parent)
- std::shared_ptr< UIElement > **getParent** () const
- void **addChild** (std::shared_ptr< UIElement > child)
- void **removeChild** (std::shared_ptr< UIElement > child)
- const std::vector< std::shared_ptr< UIElement > > & **getChildren** () const
- void **setOnClick** (std::function< void()> callback)
- void **setOnHover** (std::function< void()> callback)
- void **setOnRelease** (std::function< void()> callback)
- virtual void **update** (float deltaTime)

Public Member Functions inherited from ui::IFocusable

- virtual bool **onNavigateLeft** ()
- virtual bool **onNavigateRight** ()

Private Attributes

- bool **_value** = false
- std::string **_fontPath** = constants::MAIN_FONT
- size_t **_baseFontSize** = constants::BUTTON_FONT_SIZE_BASE
- std::string **_onText** = "ON"
- std::string **_offText** = "OFF"
- gfx::color_t **_trackColor** = colors::TOGGLE_TRACK
- gfx::color_t **_handleColor** = colors::TOGGLE_HANDLE
- gfx::color_t **_handleHoveredColor** = colors::TOGGLE_HANDLE_HOVER
- gfx::color_t **_handleFocusedColor** = colors::TOGGLE_HANDLE_FOCUSED
- gfx::color_t **_onColor** = colors::TOGGLE_ON
- gfx::color_t **_offColor** = colors::TOGGLE_OFF
- std::function< void(bool)> **_onValueChanged**
- bool **_isHovered** = false

Additional Inherited Members

Protected Member Functions inherited from ui::AFocusableElement

- virtual void **onFocusStateChanged** (bool focused)

Protected Member Functions inherited from ui::UIElement

- std::pair< int, int > **getWindowSize** () const
- std::pair< int, int > **getLogicalSize** () const
- float **getScaleFactor** () const

Protected Attributes inherited from ui::AFocusableElement

- bool **_focused** = false
- bool **_pressedInside** = false
- bool **_wasPressed** = false
- std::function< void()> **_onFocusGained**
- std::function< void()> **_onFocusLost**
- std::function< void()> **_onActivated**

Protected Attributes inherited from ui::UIElement

- std::weak_ptr< ResourceManager > **_resourceManager**
- math::Vector2f **_position**
- math::Vector2f **_size**
- bool **_visible** = true
- bool **_scalingEnabled** = true
- bool **_focusEnabled** = true
- UIState **_state** = UIState::Normal
- UIScale **_scale** = UIScale::Normal
- std::weak_ptr< UIElement > **_parent**
- std::vector< std::shared_ptr< UIElement > > **_children**
- bool **_pressedInside** = false
- bool **_wasPressed** = false
- std::function< void()> **_onClick**
- std::function< void()> **_onHover**
- std::function< void()> **_onRelease**

4.250.1 Member Function Documentation

4.250.1.1 containsPoint()

```
bool ui::ToggleSwitch::containsPoint (
            const math::Vector2f & point) const  [override], [virtual]
```

Reimplemented from ui::UIElement.

4.250.1.2 handleInput()

```
void ui::ToggleSwitch::handleInput (
            const math::Vector2f & mousePos,
            bool mousePressed)  [override], [virtual]
```

Reimplemented from ui::AFocusableElement.

4.250.1.3 render()

```
void ui::ToggleSwitch::render ()  [override], [virtual]
```

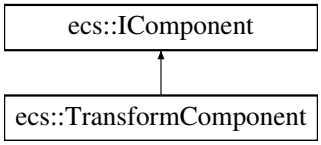
Reimplemented from ui::UIElement.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/elements/focusable/ToggleSwitch.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/elements/focusable/ToggleSwitch.cpp

4.251 ecs::TransformComponent Class Reference

Inheritance diagram for ecs::TransformComponent:



Public Member Functions

- **TransformComponent** (math::Vector2f position=math::Vector2f(0.0f, 0.0f), float rotation=0.0f, math::Vector2f scale=math::Vector2f(1.0f, 1.0f))
- math::Vector2f **getPosition** () const
- void **setPosition** (math::Vector2f position)
- float **getRotation** () const
- void **setRotation** (float rotation)
- math::Vector2f **getScale** () const
- void **setScale** (math::Vector2f scale)

Private Attributes

- math::Vector2f **_position**
- float **_rotation**
- math::Vector2f **_scale**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/permanent/TransformComponent.hpp

4.252 ecs::Transition Struct Reference

Public Attributes

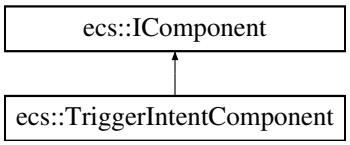
- std::string **from**
- std::string **to**
- std::vector< AnimationCondition > **conditions**
- bool **playRewind** = false

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/components/rendering/AnimationComponent.hpp

4.253 ecs::TriggerIntentComponent Class Reference

Inheritance diagram for ecs::TriggerIntentComponent:



Public Member Functions

- **TriggerIntentComponent** (Entity self=0, Entity other=0)
- Entity **getSelf** () const
- void **setSelf** (Entity self)
- Entity **getOther** () const
- void **setOther** (Entity other)

Private Attributes

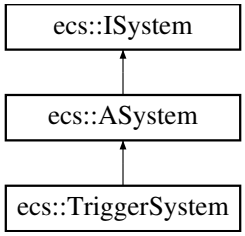
- Entity **_self**
- Entity **_other**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/temporary/TriggerIntentComponent.↩ hpp

4.254 ecs::TriggerSystem Class Reference

Inheritance diagram for ecs::TriggerSystem:



Public Member Functions

- void update (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Public Member Functions inherited from ecs::ASystem

- void updateSystem (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, float deltaTime) override

Private Member Functions

- void **buildSpatialGrid** (std::shared_ptr< Registry > registry)
- bool **checkCollision** (const TransformComponent &transformA, const ColliderComponent &colliderA, const TransformComponent &transformB, const ColliderComponent &colliderB)
- bool **shouldCollide** (std::shared_ptr< ResourceManager > resourceManager, std::shared_ptr< Registry > registry, size_t entityA, const ColliderComponent &colliderA, size_t entityB)

Private Attributes

- SpatialGrid **_spatialGrid**

4.254.1 Member Function Documentation

4.254.1.1 update()

```
void ecs::TriggerSystem::update (
            std::shared_ptr< ResourceManager > resourceManager,
            std::shared_ptr< Registry > registry,
            float deltaTime)  [override], [virtual]
```

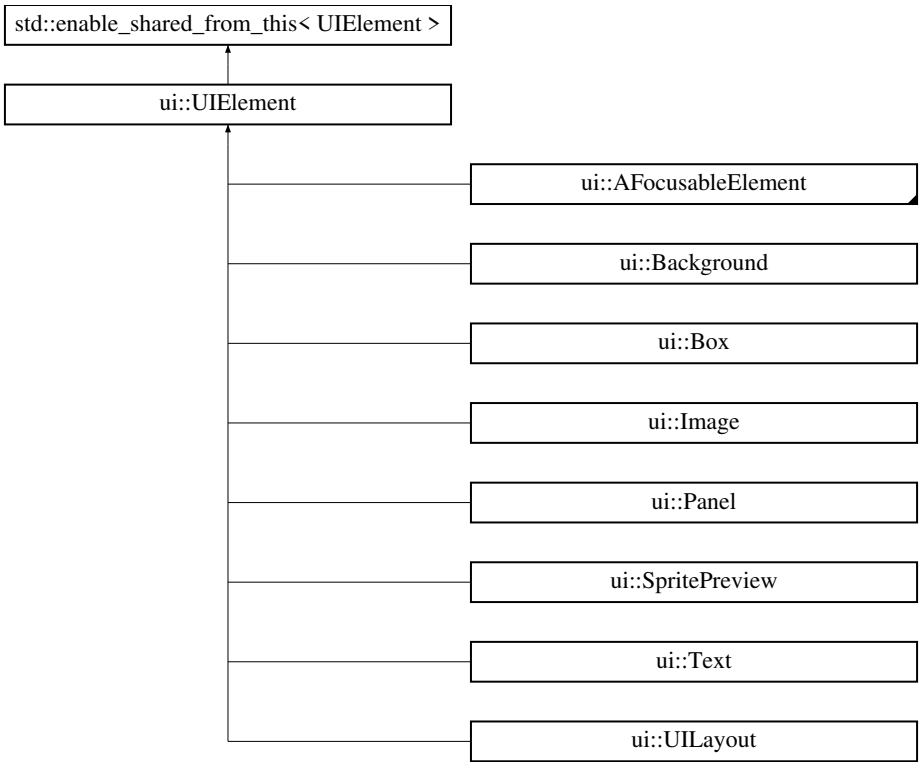
Implements ecs::ASystem.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/interactions/TriggerSystem.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/systems/interactions/TriggerSystem.cpp

4.255 ui::UIElement Class Reference

Inheritance diagram for ui::UIElement:



Public Member Functions

- **UIElement** (std::shared_ptr< ResourceManager > resourceManager)
- void **setPosition** (const math::Vector2f &position)
- void **setSize** (const math::Vector2f &size)
- math::Vector2f **getPosition** () const
- math::Vector2f **getSize** () const
- math::Vector2f **getAbsolutePosition** () const
- math::Vector2f **getAbsoluteSize** () const
- void **setVisible** (bool visible)
- bool **isVisible** () const
- void **setScalingEnabled** (bool enabled)
- bool **isScalingEnabled** () const
- void **setFocusEnabled** (bool enabled)
- bool **isFocusEnabled** () const
- void **setState** (UIState state)
- UIState **getState** () const
- virtual void **setScale** (UIScale scale)
- UIScale **getScale** () const
- void **setParent** (std::weak_ptr< UIElement > parent)
- std::shared_ptr< UIElement > **getParent** () const
- void **addChild** (std::shared_ptr< UIElement > child)
- void **removeChild** (std::shared_ptr< UIElement > child)
- const std::vector< std::shared_ptr< UIElement > > & **getChildren** () const
- virtual void **handleInput** (const math::Vector2f &mousePos, bool mousePressed)
- virtual bool **containsPoint** (const math::Vector2f &point) const
- void **setOnClick** (std::function< void()> callback)
- void **setOnHover** (std::function< void()> callback)
- void **setOnRelease** (std::function< void()> callback)
- virtual void **render** ()
- virtual void **update** (float deltaTime)

Protected Member Functions

- std::pair< int, int > **getWindowSize** () const
- std::pair< int, int > **getLogicalSize** () const
- float **getScaleFactor** () const

Protected Attributes

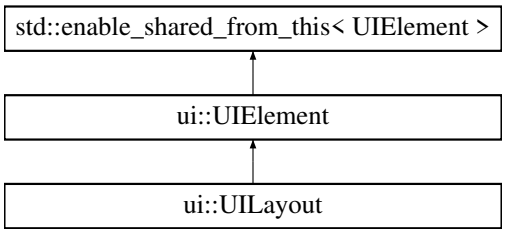
- std::weak_ptr< ResourceManager > **_resourceManager**
- math::Vector2f **_position**
- math::Vector2f **_size**
- bool **_visible** = true
- bool **_scalingEnabled** = true
- bool **_focusEnabled** = true
- UIState **_state** = UIState::Normal
- UIScale **_scale** = UIScale::Normal
- std::weak_ptr< UIElement > **_parent**
- std::vector< std::shared_ptr< UIElement > > **_children**
- bool **_pressedInside** = false
- bool **_wasPressed** = false
- std::function< void()> **_onClick**
- std::function< void()> **_onHover**
- std::function< void()> **_onRelease**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/elements/base/UIElement.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/elements/base/UIElement.cpp

4.256 ui::UILayout Class Reference

Inheritance diagram for ui::UILayout:



Public Member Functions

- **UILayout** (std::shared_ptr< ResourceManager > resourceManager, const LayoutConfig &config=LayoutConfig())
- void **addElement** (std::shared_ptr< UIElement > element)
- void **removeElement** (std::shared_ptr< UIElement > element)
- void **clearElements** ()
- void **setDirection** (LayoutDirection direction)
- void **setAlignment** (LayoutAlignment alignment)
- void **setSpacing** (float spacing)
- void **setPadding** (const math::Vector2f &padding)
- void **setOffset** (const math::Vector2f &offset)
- void **setAutoResize** (bool autoResize)
- void **setAnchor** (AnchorX anchorX, AnchorY anchorY)
- void **setBackgroundEnabled** (bool enabled)
- void **setBackgroundFillColor** (const color_t &color)
- void **setBackgroundOutlineColor** (const color_t &color)
- void **setBackgroundCornerRadius** (float radius)
- LayoutDirection **getDirection** () const
- LayoutAlignment **getAlignment** () const
- float **getSpacing** () const
- math::Vector2f **getPadding** () const
- bool **isAutoResize** () const
- void **updateLayout** ()
- void setScale (UIScale scale) override
- void render () override
- void update (float deltaTime) override
- float **getScaledSpacing** () const
- void **applyAnchor** ()

Public Member Functions inherited from ui::UIElement

- **UIElement** (std::shared_ptr< ResourceManager > resourceManager)
- void **setPosition** (const math::Vector2f &position)
- void **setSize** (const math::Vector2f &size)
- math::Vector2f **getPosition** () const
- math::Vector2f **getSize** () const
- math::Vector2f **getAbsolutePosition** () const
- math::Vector2f **getAbsoluteSize** () const
- void **setVisible** (bool visible)
- bool **isVisible** () const
- void **setScalingEnabled** (bool enabled)
- bool **isScalingEnabled** () const
- void **setFocusEnabled** (bool enabled)
- bool **isFocusEnabled** () const
- void **setState** (UIState state)
- UIState **getState** () const
- UIScale **getScale** () const
- void **setParent** (std::weak_ptr< UIElement > parent)
- std::shared_ptr< UIElement > **getParent** () const
- void **addChild** (std::shared_ptr< UIElement > child)
- void **removeChild** (std::shared_ptr< UIElement > child)
- const std::vector< std::shared_ptr< UIElement > > & **getChildren** () const
- virtual void **handleInput** (const math::Vector2f &mousePos, bool mousePressed)
- virtual bool **containsPoint** (const math::Vector2f &point) const
- void **setOnClick** (std::function< void()> callback)
- void **setOnHover** (std::function< void()> callback)
- void **setOnRelease** (std::function< void()> callback)

Private Member Functions

- void **calculatePositions** ()
- float **getTotalSize** () const
- math::Vector2f **calculateElementPosition** (size_t index, float totalSize) const

Private Attributes

- LayoutConfig **_config**
- std::vector< std::shared_ptr< UIElement > > **_layoutElements**

Additional Inherited Members

Protected Member Functions inherited from ui::UIElement

- std::pair< int, int > **getWindowSize** () const
- std::pair< int, int > **getLogicalSize** () const
- float **getScaleFactor** () const

Protected Attributes inherited from ui::UIElement

- std::weak_ptr< ResourceManager > **_resourceManager**
- math::Vector2f **_position**
- math::Vector2f **_size**
- bool **_visible** = true
- bool **_scalingEnabled** = true
- bool **_focusEnabled** = true
- UIState **_state** = UIState::Normal
- UIScale **_scale** = UIScale::Normal
- std::weak_ptr< UIElement > **_parent**
- std::vector< std::shared_ptr< UIElement > > **_children**
- bool **_pressedInside** = false
- bool **_wasPressed** = false
- std::function< void()> **_onClick**
- std::function< void()> **_onHover**
- std::function< void()> **_onRelease**

4.256.1 Member Function Documentation

4.256.1.1 render()

```
void ui::UILayout::render ()  [override], [virtual]
```

Reimplemented from ui::UIElement.

4.256.1.2 setScale()

```
void ui::UILayout::setScale (
            UIScale scale)  [override], [virtual]
```

Reimplemented from ui::UIElement.

4.256.1.3 update()

```
void ui::UILayout::update (
            float deltaTime)  [override], [virtual]
```

Reimplemented from ui::UIElement.

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/core/UILayout.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/core/UILayout.cpp

4.257 ui::UIManager Class Reference

Public Member Functions

- void **addElement** (std::shared_ptr< UIElement > element)
- void **removeElement** (std::shared_ptr< UIElement > element)
- void **clearElements** ()
- void **update** (float deltaTime)
- void **render** ()
- void **handleMouseInput** (const math::Vector2f &mousePos, bool mousePressed)
- bool **handleNavigationInput** (ecs::InputAction action)
- bool **handleNavigationInputs** (std::shared_ptr< ecs::IInputProvider > inputProvider, float deltaTime)
- void **handleKeyboardInput** (gfx::EventType event)
- void **handleTextInput** (const std::string &text)
- std::shared_ptr< UINavigationManager > **getNavigationManager** ()
- void **setNavigationEnabled** (bool enabled)
- bool **isNavigationEnabled** () const
- bool **focusFirstElement** ()
- void **clearFocus** ()
- std::shared_ptr< IFocusable > **getFocusedElement** () const
- void **setGlobalScale** (UIScale scale)
- void **cycleGlobalScale** ()
- UIScale **getGlobalScale** () const
- void **setOnBack** (std::function< void()> callback)
- void **setCursorCallback** (std::function< void(bool)> callback)
- bool **isMouseHoveringAnyElement** (const math::Vector2f &mousePos) const
- void **setResourceManager** (std::shared_ptr< ResourceManager > resourceManager)

Private Member Functions

- bool **hasMouseMoved** (const math::Vector2f &mousePos)
- void **refreshNavigationElements** ()

Private Attributes

- std::vector< std::shared_ptr< UIElement > > **_elements**
- std::shared_ptr< UINavigationManager > **_navigationManager**
- math::Vector2f **_lastMousePos**
- bool **_mouseMovementDetected**
- float **_navigationCooldown** = 0.0f
- UIScale **_globalScale** = UIScale::Normal
- std::function< void()> **_onBack**
- std::function< void(bool)> **_cursorCallback**
- std::set< gfx::EventType > **_consumedTextKeys**
- std::set< ecs::InputAction > **_blockedActions**
- std::shared_ptr< ResourceManager > **_resourceManager**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/manager/UIManager.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/manager/UIManager.cpp

4.258 ui::UINavigationManager Class Reference

Public Member Functions

- void **addFocusableElement** (std::shared_ptr< IFocusable > element)
- void **removeFocusableElement** (std::shared_ptr< IFocusable > element)
- void **clearFocusableElements** ()
- bool **handleNavigationInput** (ecs::InputAction action)
- bool **setFocus** (std::shared_ptr< IFocusable > element)
- std::shared_ptr< IFocusable > **getFocusedElement** () const
- void **clearFocus** ()
- bool **focusFirstElement** ()
- bool **focusNextElement** ()
- bool **focusPreviousElement** ()
- void **setNavigationEnabled** (bool enabled)
- bool **isNavigationEnabled** () const
- void **setOnFocusChanged** (std::function< void(std::shared_ptr< IFocusable >)> callback)
- void **onMouseMovement** ()
- void **enableFocus** ()
- bool **isFocusDisabled** () const

Private Member Functions

- void **cleanupExpiredElements** ()
- int **getCurrentFocusedIndex** () const
- bool **navigateInDirection** (NavigationDirection direction)

Private Attributes

- std::vector< std::weak_ptr< IFocusable > > **_focusableElements**
- std::weak_ptr< IFocusable > **_currentFocused**
- bool **_navigationEnabled**
- bool **_focusDisabled**
- std::function< void(std::shared_ptr< IFocusable >)> **_onFocusChanged**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/navigation/UINavigationManager.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/ui/navigation/UINavigationManager.cpp

4.259 gsm::UniqueObstacle Struct Reference

Public Attributes

- float **posX**
- float **posY**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditor/LevelEditorState.hpp

4.260 Utils Class Reference

Public Member Functions

- void **helper** ()
- void **parseCli** (int ac, char **av, std::shared_ptr< ClientNetwork > clientNetwork)
- void **helper** ()
- void **parsCli** (int ac, char **av, std::shared_ptr< rserv::ServerConfig > config)

Static Public Member Functions

- static std::string **createAlphaNumericCode** ()

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/Utils.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/Utils.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/client/Utils.cpp
- /home/albane/epitech/tech3/r-type/ryanR-type/server/Utils.cpp

4.261 parser::ValidationResult Struct Reference

Public Member Functions

- **operator bool** () const

Public Attributes

- bool **valid**
- std::vector< std::string > **errors**
- std::vector< std::string > **warnings**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/Parser/Utils/JsonValidation.hpp

4.262 math::Vector2f Class Reference

Public Member Functions

- **Vector2f** (float x=0.0f, float y=0.0f)
- **Vector2f** (Vector2f const &other)
- float **getX** () const
- void **setX** (float x)
- float **getY** () const
- void **setY** (float y)
- Vector2f **getVector** () const
- Vector2f **operator*** (float scalar) const
- Vector2f **operator-** (Vector2f const &other) const
- Vector2f **operator+** (Vector2f const &other) const
- void **operator=** (Vector2f const &other)
- void **operator+=** (Vector2f const &other)
- void **operator-=** (Vector2f const &other)
- void **operator*=** (float scalar)
- void **operator/=** (float scalar)

Private Attributes

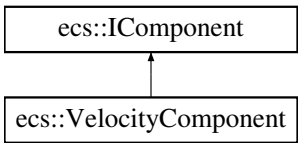
- float **_x**
- float **_y**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/types/Vector2f.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/types/Vector2f.cpp

4.263 ecs::VelocityComponent Class Reference

Inheritance diagram for ecs::VelocityComponent:



Public Member Functions

- **VelocityComponent** (math::Vector2f velocity=math::Vector2f(0.0f, 0.0f))
- math::Vector2f **getVelocity** () const
- void **setVelocity** (math::Vector2f velocity)

Private Attributes

- math::Vector2f **_velocity**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/components/permanent/VelocityComponent.hpp

4.264 gsm::VerticalLineObstacle Struct Reference

Public Attributes

- float **fromY**
- float **posX**
- int **count**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditor/LevelEditorState.hpp

4.265 ecs::View< Components > Class Template Reference

Classes

- class Iterator

Public Member Functions

- **View** (std::shared_ptr< Registry > registry)
- Iterator **begin** ()
- Iterator **end** ()

Private Attributes

- std::shared_ptr< Registry > **_registry**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/common/ECS/entity/registry/Registry.hpp
- /home/albane/epitech/tech3/r-type/ryanR-type/common/ECS/view/View.hpp

4.266 gsm::Wave Struct Reference

Public Attributes

- float **gameXTrigger**
- std::vector< WaveEnemy > **enemies**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditor/LevelEditorState.hpp

4.267 gsm::WaveDistribution Struct Reference

Public Attributes

- float **min**
- float **max**
- std::string **type**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditor/LevelEditorState.hpp

4.268 gsm::WaveEnemy Struct Reference

Public Attributes

- std::string **type**
- WaveDistribution **distributionX**
- WaveDistribution **distributionY**
- int **count**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditor/LevelEditorState.hpp

4.269 gsm::WaveSelection Struct Reference

Public Attributes

- int **waveIndex**
- int **enemyIndex**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditor/LevelEditorState.hpp

4.270 gsm::LevelEditorState::WorldCoordinates Struct Reference

Public Attributes

- float **worldX**
- float **worldY**
- float **levelX**
- float **levelY**

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

- /home/albane/epitech/tech3/r-type/ryanR-type/client/gsm/states/scenes/LevelEditor/LevelEditorState.hpp

Chapter 5

File Documentation

5.1 ClientNetwork.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ClientNetwork
00006 */
00007
00008
00009 #ifndef CLIENTNETWORK_HPP_
00010 #define CLIENTNETWORK_HPP_
00011
00012 #include <memory>
00013 #include <thread>
00014 #include <queue>
00015 #include <mutex>
00016 #include <condition_variable>
00017 #include <atomic>
00018 #include <map>
00019 #include <chrono>
00020
00021 #include "../common/DLLoader/DLLoader.hpp"
00022 #include "../common/DLLoader/LoaderType.hpp"
00023 #include "../common/interfaces/INetwork.hpp"
00024 #include "../common/constants.hpp"
00025 #include "../common/resourceManager/ResourceManager.hpp"
00026 #include "../common/gsm/IGameStateMachine.hpp"
00027 #include "gsm/states/scenes/MainMenu/MainMenuState.hpp"
00028
00029 namespace ecs {
00030     class Registry;
00031     using Entity = std::size_t;
00032 }
00033
00034 struct NetworkEvent {
00035     constants::EventType eventType;
00036     double depth;
00037 };
00038
00039 class ClientNetwork {
00040     public:
00041         ClientNetwork();
00042         ~ClientNetwork();
00043
00044         void init();
00045         void start();
00046         void stop();
00047         void connect();
00048
00049         uint16_t getPort() const;
00050         void setPort(int port);
00051
00052         std::string getIp() const;
00053         void setIp(const std::string &ip);
00054         std::shared_ptr<net::INetwork> getNetwork() const;
00055
00056         void setDebugMode(bool isDebug);
00057         bool isDebugMode() const;
```

```
00058
00059         void loadNetworkLibrary();
00060         void loadBufferLibrary();
00061         void loadPacketLibrary();
00062
00063         void sendConnectionData(std::vector<uint8_t> packet);
00064
00065         std::string getName() const;
00066         void setName(const std::string &name);
00067
00068         uint8_t getIdClient() const;
00069         void setIdClient(uint8_t idClient);
00070
00071         std::string getLobbyCode() const;
00072         void setLobbyCode(std::string lobbyCode);
00073
00074         net::ConnectionState getConnectionState() const;
00075
00076         /* Packet Handling */
00077         void eventPacket(const constants::EventType &eventType, double depth);
00078         void disconnectionPacket();
00079         void connectionPacket();
00080         void sendWhoAmI();
00081         void requestCode();
00082         void sendLobbyConnection(std::string lobbyCode);
00083         void sendMasterStartGame();
00084         void sendRegisterPacket(const std::string &username, const std::string &password);
00085         void sendLoginPacket(const std::string &username, const std::string &password);
00086         void sendRequestLeaderboardPacket();
00087         void sendRequestProfilePacket();
00088         void sendMessageToServer(const std::string &message);
00089         void sendRequestGameRulesUpdate(uint8_t ruleType, uint8_t value);
00090         void sendDisconnectFromLobby();
00091
00092         const std::vector<std::pair<std::string, std::string»& getLeaderboardData() const;
00093         bool isLeaderboardDataUpdated() const;
00094         void clearLeaderboardDataUpdateFlag();
00095
00096         const std::vector<std::string>& getProfileData() const;
00097         bool isProfileDataUpdated() const;
00098         void clearProfileDataUpdateFlag();
00099         const std::vector<std::pair<std::string, std::string»& getLastMessages() const;
00100
00101         void addToEventQueue(const NetworkEvent &event);
00102
00103         void clearEntitiesAndMappings();
00104
00105         bool isConnected() const;
00106         bool isReady() const;
00107
00108         size_t getConnectedClients() const;
00109         size_t getReadyClients() const;
00110         uint8_t getClientId() const;
00111         bool getClientReadyStatus() const;
00112
00113         bool isConnectedToLobby() const;
00114         bool isLobbyMaster() const;
00115
00116         std::atomic<bool> _isConnected;
00117         std::atomic<bool> _ready;
00118         std::atomic<bool> _isConnectedToLobby;
00119         std::atomic<bool> _isLobbyMaster;
00120
00121         std::atomic<size_t> _connectedClients;
00122         std::atomic<size_t> _readyClients;
00123         std::atomic<uint8_t> _clientId;
00124         std::atomic<bool> _clientReadyStatus;
00125         std::atomic<bool> _shouldDisconnect;
00126
00127         std::chrono::steady_clock::time_point _lastLeaveLobbyTime;
00128
00129         void setResourceManager(std::shared_ptr<ResourceManager> resourceManager);
00130         void setGameStateMachine(std::shared_ptr<gsm::IGameStateMachine> gsm);
00131         std::shared_ptr<gsm::IGameStateMachine> getGameStateMachine() const;
00132
00133         void redoServerEndpoint();
00134
00135     protected:
00136         std::pair<int, std::chrono::steady_clock::time_point> tryConnection(const int maxRetries,
      std::chrono::steady_clock::time_point lastRetryTime);
00137         void handlePacketType(uint8_t type);
00138
00139     private:
00140         typedef void (ClientNetwork::*PacketHandler)();
00141         PacketHandler _packetHandlers[constants::MAX_INDEX_PACKET_TYPE];
00142
00143         void handleNoOp();
```

```
00144         void handleConnectionAcceptation();
00145         void handleBatchedGameState();
00146         void handleEndGame();
00147         void handleCanStart();
00148         void handleEntitySpawn();
00149         void handleEntityDeath();
00150         void handleWhoAmI();
00151         void handleServerStatus();
00152         void handleCode();
00153         void handleLevelComplete();
00154         void handleNextLevel();
00155         void handleLobbyConnectValue();
00156         void handleConnectUser();
00157         void handleLeaderboard();
00158         void handleProfile();
00159         void handleRegisterFail();
00160         void handleBroadcastedChat();
00161         void handleGameRules();
00162         void handleForceLeave();
00163         void handleAckLeaveLobby();
00164
00165         typedef size_t (ClientNetwork::*ComponentParser)(const std::vector<uint64_t> &, size_t,
      ecs::Entity);
00166         std::map<uint64_t, ComponentParser> _componentParsers;
00167
00168         size_t parseTransformComponent(const std::vector<uint64_t> &payload, size_t index, ecs::Entity
      entityId);
00169         size_t parseHealthComponent(const std::vector<uint64_t> &payload, size_t index, ecs::Entity
      entityId);
00170         size_t parseScoreComponent(const std::vector<uint64_t> &payload, size_t index, ecs::Entity
      entityId);
00171         size_t parseChargedShotComponent(const std::vector<uint64_t> &payload, size_t index,
      ecs::Entity entityId);
00172         size_t parseAnimationStateComponent(const std::vector<uint64_t> &payload, size_t index,
      ecs::Entity entityId);
00173
00174         DLLoader<createNetworkLib_t> _networloader;
00175         DLLoader<createBuffer_t> _bufferloader;
00176         DLLoader<createPacket_t> _packetloader;
00177
00178         std::shared_ptr<net::INetwork> _network;
00179         std::shared_ptr<IBuffer> _receptionBuffer;
00180         std::shared_ptr<IBuffer> _sendBuffer;
00181         std::shared_ptr<pm::IPacketManager> _packet;
00182
00183         std::shared_ptr<ResourceManager> _resourceManager;
00184         std::shared_ptr<gsm::IGameStateMachine> _gsm;
00185
00186         uint32_t _sequenceNumber;
00187         uint16_t _port;
00188         std::string  _ip;
00189         std::string _name;
00190         std::vector<std::string> _clientNames;
00191         std::vector<std::pair<std::string, std::string» _lastMessages;
00192         bool _isDebug;
00193         bool _expectingLoginResponse = false;
00194         bool _expectingProfileResponse = false;
00195         bool _expectingRegisterResponse = false;
00196
00197
00198         uint8_t _idClient;
00199         std::shared_ptr<net::INetworkEndpoint> _serverEndpoint;
00200
00201         std::queue<NetworkEvent> _eventQueue;
00202         std::mutex _queueMutex;
00203         std::condition_variable _queueCond;
00204
00205         std::unordered_map<size_t, ecs::Entity> _serverToLocalEntityMap;
00206
00207         std::unordered_map<ecs::Entity, std::string> _lastReceivedAnimationState;
00208
00209         std::string _lobbyCode;
00210         int _retryCount;
00211         bool _shouldConnect;
00212
00213         std::vector<std::pair<std::string, std::string» _leaderboardData;
00214         bool _leaderboardDataUpdated = false;
00215         std::vector<std::string> _profileData;
00216         bool _profileDataUpdated = false;
00217
00218         std::chrono::steady_clock::time_point _connectionAttemptTime;
00219 };
00220
00221 #endif /* !CLIENTNETWORK_HPP_ */
```

5.2 colors.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** Colors
00006 */
00007
00008 #ifndef COLORS_HPP_
00009 #define COLORS_HPP_
00010
00011 #include "../common/interfaces/IWindow.hpp"
00012
00013 namespace colors {
00014     const gfx::color_t BLACK = {0, 0, 0};
00015     const gfx::color_t WHITE = {255, 255, 255};
00016     const gfx::color_t RED = {255, 0, 0};
00017     const gfx::color_t GREEN = {0, 255, 0};
00018     const gfx::color_t BLUE = {0, 0, 255};
00019     const gfx::color_t YELLOW = {255, 255, 0};
00020     const gfx::color_t CYAN = {0, 255, 255};
00021     const gfx::color_t MAGENTA = {255, 0, 255};
00022     const gfx::color_t GRAY = {128, 128, 128};
00023     const gfx::color_t LIGHT_GRAY = {192, 192, 192};
00024     const gfx::color_t DARK_GRAY = {64, 64, 64};
00025     const gfx::color_t ORANGE = {255, 165, 0};
00026     const gfx::color_t PURPLE = {128, 0, 128};
00027     const gfx::color_t PINK = {255, 192, 203};
00028     const gfx::color_t BROWN = {165, 42, 42};
00029
00030     // UI Colors
00031     const gfx::color_t UI_BACKGROUND = {0, 0, 0};                                  // Black background
00032     const gfx::color_t UI_ACCENT = {253, 208, 16};                                 // Yellow accent
00033     const gfx::color_t UI_TEXT = {255, 255, 255};                                  // White text
00034     const gfx::color_t UI_TEXT_SECONDARY = {253, 208, 16};                         // Yellow secondary
    text
00035     const gfx::color_t UI_OUTLINE = {0, 0, 0};                                     // Black outline
00036
00037     // Button Colors
00038     const gfx::color_t BUTTON_PRIMARY = {224,51,14};                               // Red primary button
00039     const gfx::color_t BUTTON_PRIMARY_HOVER = {245, 81, 47};                       // Bright red hover
00040     const gfx::color_t BUTTON_PRIMARY_PRESSED = {255, 48, 0};                      // Dark red pressed
00041     const gfx::color_t BUTTON_SECONDARY = {240, 207, 65};                          // Yellow secondary
    button
00042     const gfx::color_t BUTTON_SECONDARY_HOVER = {253, 208, 16};                    // Bright yellow hover
00043     const gfx::color_t BUTTON_SECONDARY_PRESSED = {252, 205, 3};                   // Dark yellow pressed
00044     const gfx::color_t BUTTON_DANGER = {222, 77, 47};                              // Red danger
00045     const gfx::color_t BUTTON_DANGER_HOVER = {224,51,14};                          // Bright red hover
00046     const gfx::color_t BUTTON_DANGER_PRESSED = {247, 47, 5};                       // Dark red pressed
00047
00048     // Panel/Frame Colors
00049     const gfx::color_t PANEL_BACKGROUND = {20, 20, 50};                            // Slightly lighter
    than UI background
00050     const gfx::color_t PANEL_BORDER = {0, 200, 255};                               // Cyan border
00051
00052     // Slider Colors
00053     const gfx::color_t SLIDER_TRACK = {50, 50, 50};                                // Dark gray track
00054     const gfx::color_t SLIDER_FILL = {253, 208, 16};                               // Yellow fill
00055     const gfx::color_t SLIDER_HANDLE = {240, 207, 65};                             // Light yellow handle
00056     const gfx::color_t SLIDER_HANDLE_HOVER = {253, 208, 16};                       // Bright yellow hover
00057     const gfx::color_t SLIDER_HANDLE_FOCUSED = {252, 205, 3};                      // Dark yellow focused
00058     const gfx::color_t SLIDER_LABEL = {255, 255, 255};                             // White label
00059
00060     // Toggle Switch Colors
00061     const gfx::color_t TOGGLE_TRACK = {100, 100, 100};                             // Gray track
00062     const gfx::color_t TOGGLE_HANDLE = {224, 51, 14};                              // Red handle
00063     const gfx::color_t TOGGLE_HANDLE_HOVER = {245, 81, 47};                        // Bright red hover
00064     const gfx::color_t TOGGLE_HANDLE_FOCUSED = {255, 48, 0};                       // Dark red focused
00065     const gfx::color_t TOGGLE_ON = {253, 208, 16};                                 // Yellow on
00066     const gfx::color_t TOGGLE_OFF = {100, 100, 100};                               // Gray off
00067
00068     const gfx::color_t UI_HOVER = {0, 150, 255};                                   // Cyan hover
00069     const gfx::color_t UI_FOCUSED = {175, 175, 175};                               // Light gray focused
00070     const gfx::color_t UI_DISABLED = {100, 100, 100};                              // Gray disabled
00071
00072     const gfx::color_t LEVEL_EDITOR_PANEL_BACKGROUND = {15, 10, 25};               // Dark with slight
    purple tint
00073     const gfx::color_t LEVEL_EDITOR_PANEL_BORDER = {200, 200, 200};                // Light gray border
00074
00075     /* Player colors */
00076     const gfx::color_t PLAYER_LOCAL = {255, 255, 255, 255};                        // White for local
    player
00077     const gfx::color_t PLAYER_REMOTE = {150, 150, 150, 200};                       // Grayish for remote
    players
00078 }
```

```
00079
00080 #endif  // COLORS_HPP_
```

5.3 NetworkStateComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** NetworkStateComponent
00006 */
00007
00008 #ifndef NETWORKSTATECOMPONENT_HPP_
00009 #define NETWORKSTATECOMPONENT_HPP_
00010
00011 #include "../../common/components/base/IComponent.hpp"
00012 #include "../../common/types/Vector2f.hpp"
00013 #include <chrono>
00014
00015 namespace ecs {
00016
00017 struct NetworkTransformState {
00018     math::Vector2f position;
00019     float rotation;
00020     math::Vector2f scale;
00021     std::chrono::steady_clock::time_point timestamp;
00022
00023     NetworkTransformState()
00024         : position(0.0f, 0.0f)
00025         , rotation(0.0f)
00026         , scale(1.0f, 1.0f)
00027         , timestamp(std::chrono::steady_clock::now()) {}
00028 };
00029
00030 class NetworkStateComponent : public IComponent {
00031     public:
00032         NetworkStateComponent()
00033             : _hasTransform(false)
00034             , _interpolationTime(0.1f) {}
00035
00036         ~NetworkStateComponent() = default;
00037
00038         void setCurrentTransform(const math::Vector2f& pos, float rot, const math::Vector2f& scale) {
00039             if (_hasTransform) {
00040                 _previousTransform = _currentTransform;
00041             }
00042             _currentTransform.position = pos;
00043             _currentTransform.rotation = rot;
00044             _currentTransform.scale = scale;
00045             _currentTransform.timestamp = std::chrono::steady_clock::now();
00046             _hasTransform = true;
00047         }
00048
00049         bool hasTransform() const { return _hasTransform; }
00050
00051         const NetworkTransformState& getPreviousTransform() const { return _previousTransform; }
00052         const NetworkTransformState& getCurrentTransform() const { return _currentTransform; }
00053
00054         void setInterpolationTime(float time) { _interpolationTime = time; }
00055         float getInterpolationTime() const { return _interpolationTime; }
00056
00057     private:
00058         NetworkTransformState _previousTransform;
00059         NetworkTransformState _currentTransform;
00060         bool _hasTransform;
00061         float _interpolationTime;
00062 };
00063
00064 } // namespace ecs
00065
00066 #endif /* !NETWORKSTATECOMPONENT_HPP_ */
```

5.4 AnimationComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
```

```
00005 ** AnimationComponent
00006 */
00007
00008 #ifndef ANIMATIONCOMPONENT_HPP_
00009 #define ANIMATIONCOMPONENT_HPP_
00010
00011 #include <unordered_map>
00012 #include <vector>
00013 #include <functional>
00014 #include <string>
00015 #include <memory>
00016 #include <algorithm>
00017 #include "../../../common/components/base/IComponent.hpp"
00018 #include "../../../common/types/FRect.hpp"
00019 #include "../../../common/ECS/entity/Entity.hpp"
00020
00021 namespace ecs {
00022
00023 class Registry;
00024
00025 struct AnimationClip {
00026     std::string texturePath;
00027     float frameWidth;
00028     float frameHeight;
00029     int frameCount;
00030     float startWidth;
00031     float startHeight;
00032     float speed;
00033     bool loop;
00034 };
00035
00036 struct AnimationCondition {
00037     std::string param;
00038     bool equals;
00039 };
00040
00041 struct Transition {
00042     std::string from;
00043     std::string to;
00044     std::vector<AnimationCondition> conditions;
00045     bool playRewind = false;
00046 };
00047
00048 class AnimationComponent : public IComponent {
00049     public:
00050         AnimationComponent()
00051             : _currentState(""), _timer(0.f), _isPlaying(false), _currentFrame(0),
      _rewindStartFrame(-1) {
00052                 this->_states = {};
00053                 this->_playRewind = false;
00054                 this->_transitions = {};
00055                 this->_frameRect = math::FRect();
00056             }
00057
00058         void addState(const std::string& name, std::shared_ptr<AnimationClip> clip) {
00059             _states[name] = clip;
00060         }
00061
00062         void addTransition(const std::string& from, const std::string& to,
00063             const std::vector<AnimationCondition>& conditions,
00064             bool playRewind = false) {
00065             _transitions.push_back({from, to, conditions, playRewind});
00066         }
00067
00068         void setCurrentState(const std::string& state) {
00069             if (_states.find(state) != _states.end()) {
00070                 _currentState = state;
00071                 _timer = 0.f;
00072                 _isPlaying = true;
00073                 _currentFrame = 0;
00074                 _playRewind = false;
00075                 _stateJustChanged = true;
00076
00077                 auto clip = _states[state];
00078                 _minAnimationTime = clip->speed;
00079                 _frameRect = math::FRect(clip->startWidth, clip->startHeight, clip->frameWidth,
      clip->frameHeight);
00080             }
00081         }
00082
00083         const std::string& getCurrentState() const { return _currentState; }
00084         float getTimer() const { return _timer; }
00085         void setTimer(float timer) { _timer = timer; }
00086         bool isPlaying() const { return _isPlaying; }
00087         void setPlaying(bool playing) { _isPlaying = playing; }
00088         bool isPlayingRewind() const { return _playRewind; }
00089         void setPlayingRewind(bool rewind) { _playRewind = rewind; }
```

```
00090
00091         int getRewindStartFrame() const { return _rewindStartFrame; }
00092         void setRewindStartFrame(int frame) { _rewindStartFrame = frame; }
00093
00094         std::shared_ptr<const AnimationClip> getCurrentClip() const {
00095             auto it = _states.find(_currentState);
00096             return it != _states.end() ? it->second : nullptr;
00097         }
00098
00099         const std::vector<Transition>& getTransitions() const { return _transitions; }
00100
00101         int getCurrentFrame() const { return _currentFrame; }
00102         void setCurrentFrame(int frame) { _currentFrame = frame; }
00103
00104         const math::FRect& getFrameRect() const { return _frameRect; }
00105         void setFrameRect(const math::FRect& rect) { _frameRect = rect; }
00106
00107         bool isValid() const { return !_states.empty(); }
00108
00109         bool isAnimationFinished() const {
00110             auto clip = getCurrentClip();
00111             if (!clip) return true;
00112             if (clip->loop) return false;
00113             int currentFrame = static_cast<int>(_timer / clip->speed);
00114             if (_playRewind) {
00115                 return currentFrame >= clip->frameCount;
00116             } else {
00117                 return currentFrame >= clip->frameCount - 1;
00118             }
00119         }
00120
00121         void setStateJustChanged(bool changed) { _stateJustChanged = changed; }
00122         bool getStateJustChanged() const { return _stateJustChanged; }
00123
00124         void setMinAnimationTime(float time) { _minAnimationTime = time; }
00125         float getMinAnimationTime() const { return _minAnimationTime; }
00126
00127         std::unordered_map<std::string, std::shared_ptr<AnimationClip» getStates() const {
00128             return _states;
00129         }
00130     private:
00131         std::unordered_map<std::string, std::shared_ptr<AnimationClip» _states;
00132         std::vector<Transition> _transitions;
00133         std::string _currentState;
00134         float _timer;
00135         bool _isPlaying;
00136         bool _playRewind;
00137         int _currentFrame;
00138         int _rewindStartFrame;
00139         math::FRect _frameRect;
00140         bool _stateJustChanged = false;
00141         float _minAnimationTime = 0.0f;
00142 };
00143
00144 }  // namespace ecs
00145
00146 #endif /* !ANIMATIONCOMPONENT_HPP_ */
```

5.5 HealthBarComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** HealthBarComponent
00006 */
00007
00008 #ifndef HEALTHBARCOMPONENT_HPP_
00009 #define HEALTHBARCOMPONENT_HPP_
00010
00011 #include "../../../common/components/base/IComponent.hpp"
00012
00013 namespace ecs {
00014
00015 class HealthBarComponent : public IComponent {
00016     public:
00017         HealthBarComponent() = default;
00018         ~HealthBarComponent() = default;
00019 };
00020
00021 } // namespace ecs
00022
00023 #endif /* !HEALTHBARCOMPONENT_HPP_ */
```

5.6 HitboxRenderComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** HitboxRenderComponent
00006 */
00007
00008 #ifndef HITBOXRENDERCOMPONENT_HPP_
00009 #define HITBOXRENDERCOMPONENT_HPP_
00010
00011 #include "../../../common/components/base/IComponent.hpp"
00012 #include "../../../common/interfaces/IWindow.hpp"
00013 #include "../../colors.hpp"
00014
00015 namespace ecs {
00016
00017 class HitboxRenderComponent : public IComponent {
00018     public:
00019         HitboxRenderComponent() : _color{colors::WHITE}, _outlineThickness(1.0f) {}
00020         HitboxRenderComponent(
00021             gfx::color_t color, float outlineThickness = 1.0f
00022         ) : _color(color), _outlineThickness(outlineThickness) {}
00023
00024         ~HitboxRenderComponent() = default;
00025
00026         const gfx::color_t& getColor() const { return _color; }
00027         void setColor(const gfx::color_t& color) { _color = color; }
00028
00029         float getOutlineThickness() const { return _outlineThickness; }
00030         void setOutlineThickness(float thickness) { _outlineThickness = thickness; }
00031
00032     private:
00033         gfx::color_t _color;
00034         float _outlineThickness;
00035 };
00036
00037 }  // namespace ecs
00038
00039 #endif /* !HITBOXRENDERCOMPONENT_HPP_ */
```

5.7 MusicComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** MusicComponent
00006 */
00007
00008 #ifndef MUSICCOMPONENT_HPP_
00009 #define MUSICCOMPONENT_HPP_
00010
00011 #include "../../../common/components/base/IComponent.hpp"
00012 #include <string>
00013
00014
00015 namespace ecs {
00016
00017 typedef enum MusicState {
00018     PLAYING = 0,
00019     PAUSED = 1,
00020     CHANGING = 2,
00021     STOPPED = 3
00022 } MusicState;
00023
00024 class MusicComponent : public IComponent {
00025     public:
00026         MusicComponent(
00027             std::string musicFile = "", MusicState initialState = STOPPED, float volume = 100.0f, bool
      loop = false
00028         ) : _currentMusic(musicFile), _state(initialState), _volume(volume), _loop(loop) {};
00029         ~MusicComponent() = default;
00030
00031         void playMusic() { _state = PLAYING; };
00032         void pauseMusic() { _state = PAUSED; };
00033         void stopMusic() { _state = STOPPED; };
00034         bool isPlaying() const { return _state == PLAYING; };
00035         MusicState getState() const { return _state; };
00036         void playNewMusic(const std::string& musicFile) {
00037             _currentMusic = musicFile;
```

```
00038             _state = CHANGING;
00039         };
00040
00041         std::string getCurrentMusic() const { return _currentMusic; };
00042         void setCurrentMusic(const std::string& musicFile) { _currentMusic = musicFile; };
00043
00044         float getVolume() const { return _volume; };
00045         void setVolume(float volume) { _volume = volume; };
00046
00047         bool isLooping() const { return _loop; };
00048         void setLoop(bool loop) { _loop = loop; };
00049
00050     protected:
00051     private:
00052         std::string _currentMusic;
00053         MusicState _state;
00054         float _volume;
00055         bool _loop;
00056 };
00057
00058 }  // namespace ecs
00059
00060 #endif /* !MUSICCOMPONENT_HPP_ */
```

5.8 ParallaxComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ParallaxComponent
00006 */
00007
00008 #ifndef PARALLAXCOMPONENT_HPP_
00009 #define PARALLAXCOMPONENT_HPP_
00010
00011 #include "../../../common/components/base/IComponent.hpp"
00012 #include "../../../common/types/Vector2f.hpp"
00013 #include "../../../common/constants.hpp"
00014 #include <string>
00015 #include <vector>
00016 #include <algorithm>
00017
00018 namespace ecs {
00019
00020 enum class ParallaxScaleMode {
00021     FIT_SCREEN = 0,
00022     STRETCH = 1,
00023     MANUAL = 2,
00024 };
00025
00026 struct ParallaxLayer {
00027     std::string name;
00028     std::string filePath;
00029     float speedMultiplier;
00030     math::Vector2f scale;
00031     ParallaxScaleMode scaleMode;
00032     math::Vector2f sourceSize;
00033     bool repeat;
00034     int zIndex;
00035     math::Vector2f currentOffset;
00036
00037     ParallaxLayer()
00038         : name(""),
00039           filePath(""),
00040           speedMultiplier(1.0f),
00041           scale(1.0f, 1.0f),
00042           scaleMode(ParallaxScaleMode::FIT_SCREEN),
00043           sourceSize(constants::DEFAULT_TEXTURE_WIDTH, constants::DEFAULT_TEXTURE_HEIGHT),
00044           repeat(true),
00045           zIndex(0),
00046           currentOffset(0.0f, 0.0f) {}
00047 };
00048
00049 class ParallaxComponent : public IComponent {
00050     public:
00051         ParallaxComponent()
00052             : _baseScrollSpeed(1.0f),
00053               _direction(constants::BACKGROUND_PARALLAX_DIRECTION_X,
    constants::BACKGROUND_PARALLAX_DIRECTION_Y),
00054               _layers() {}
00055
00056         ~ParallaxComponent() = default;
```

```
00057
00058         float getBaseScrollSpeed() const { return _baseScrollSpeed; }
00059         const math::Vector2f& getDirection() const { return _direction; }
00060         std::vector<std::shared_ptr<ParallaxLayer» &getLayers() { return _layers; }
00061
00062         void setBaseScrollSpeed(float speed) { _baseScrollSpeed = speed; }
00063         void setDirection(const math::Vector2f& direction) { _direction = direction; }
00064         void addLayer(std::shared_ptr<ParallaxLayer> layer) { _layers.push_back(layer); }
00065
00066     private:
00067         float _baseScrollSpeed;
00068         math::Vector2f _direction;
00069         std::vector<std::shared_ptr<ParallaxLayer» _layers;
00070 };
00071
00072 }  // namespace ecs
00073
00074 #endif /* !PARALLAXCOMPONENT_HPP_ */
```

5.9 RectangleRenderComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** RectangleRenderComponent
00006 */
00007
00008 #ifndef RECTANGLERENDERCOMPONENT_HPP_
00009 #define RECTANGLERENDERCOMPONENT_HPP_
00010
00011 #include "../../../common/components/base/IComponent.hpp"
00012 #include "../../../common/interfaces/IWindow.hpp"
00013 #include "../../colors.hpp"
00014
00015 namespace ecs {
00016
00017 class RectangleRenderComponent : public IComponent {
00018     public:
00019         RectangleRenderComponent() : _color(colors::WHITE), _size{10.0f, 10.0f} {}
00020         RectangleRenderComponent(gfx::color_t color, float width, float height)
00021             : _color(color), _size{width, height} {}
00022
00023         ~RectangleRenderComponent() = default;
00024
00025         const gfx::color_t& getColor() const { return _color; }
00026         void setColor(const gfx::color_t& color) { _color = color; }
00027
00028         float getWidth() const { return _size.first; }
00029         float getHeight() const { return _size.second; }
00030         void setSize(float width, float height) { _size = {width, height}; }
00031
00032     private:
00033         gfx::color_t _color;
00034         std::pair<float, float> _size;
00035 };
00036
00037 }  // namespace ecs
00038
00039 #endif /* !RECTANGLERENDERCOMPONENT_HPP_ */
```

5.10 SpriteComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** SpriteComponent
00006 */
00007
00008 #ifndef SPRITECOMPONENT_HPP_
00009 #define SPRITECOMPONENT_HPP_
00010
00011 #include "../../../common/components/base/IComponent.hpp"
00012 #include "../../../common/types/FRect.hpp"
00013 #include <string>
00014
00015 namespace ecs {
```

```
00016
00017 class SpriteComponent : public IComponent {
00018     public:
00019         SpriteComponent() : _texturePath("") {}
00020         SpriteComponent(const std::string& texturePath)
00021             : _texturePath(texturePath) {}
00022
00023         ~SpriteComponent() = default;
00024         const std::string& getTexturePath() const { return _texturePath; }
00025         void setTexturePath(const std::string& path) { _texturePath = path; }
00026         bool isValid() const { return !_texturePath.empty(); }
00027
00028     private:
00029         std::string _texturePath;
00030 };
00031
00032 } // namespace ecs
00033
00034 #endif /* !SPRITECOMPONENT_HPP_ */
```

5.11 TextComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** TextComponent
00006 */
00007
00008 #ifndef TEXTCOMPONENT_HPP_
00009 #define TEXTCOMPONENT_HPP_
00010
00011 #include <string>
00012 #include "../../../common/components/base/IComponent.hpp"
00013 #include "../../../common/interfaces/IWindow.hpp"
00014 #include "../../colors.hpp"
00015
00016 namespace ecs {
00017
00018 class TextComponent : public IComponent {
00019     public:
00020         TextComponent(
00021             const std::string& text,
00022             const std::string& fontPath,
00023             gfx::color_t color = colors::WHITE
00024         ) : _text(text), _fontPath(fontPath), _color(color) {};
00025         ~TextComponent() = default;
00026
00027         const std::string& getText() const { return _text; }
00028         const std::string& getFontPath() const { return _fontPath; }
00029         const gfx::color_t& getColor() const { return _color; }
00030
00031         void setText(const std::string& text) { _text = text; }
00032         void setFontPath(const std::string& fontPath) { _fontPath = fontPath; }
00033         void setColor(const gfx::color_t& color) { _color = color; }
00034
00035     protected:
00036     private:
00037         std::string _text;
00038         std::string _fontPath;
00039         gfx::color_t _color;
00040 };
00041
00042 } // namespace ecs
00043
00044 #endif /* !TEXTCOMPONENT_HPP_ */
```

5.12 BackGroundMusicTag.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** BackGroundMusicTag
00006 */
00007
00008 #ifndef BACKGROUNDMUSICTAG_HPP_
00009 #define BACKGROUNDMUSICTAG_HPP_
```

```
00010
00011 #include "../../../common/components/base/IComponent.hpp"
00012
00013 namespace ecs {
00014
00015 class BackGroundMusicTag : public IComponent {
00016     public:
00017         BackGroundMusicTag() = default;
00018         ~BackGroundMusicTag() = default;
00019 };
00020
00021 } // namespace ecs
00022
00023 #endif /* !BACKGROUNDMUSICTAG_HPP_ */
```

5.13 MusicIntentComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** MusicIntentComponent
00006 */
00007
00008 #ifndef MUSICINTENTCOMPONENT_HPP_
00009 #define MUSICINTENTCOMPONENT_HPP_
00010
00011 #include "../../../common/components/base/IComponent.hpp"
00012 #include <string>
00013
00014 namespace ecs {
00015
00016 typedef enum MusicAction {
00017     PLAY = 0,
00018     PAUSE = 1,
00019     CHANGE = 2
00020 } MusicAction;
00021
00022 class MusicIntentComponent : public IComponent {
00023     public:
00024         MusicIntentComponent(MusicAction action = PLAY, const std::string &musicPath = "", float
      volume = 100.0f)
00025             : _action(action), _musicPath(musicPath), _volume(volume) {};
00026         ~MusicIntentComponent() = default;
00027
00028         MusicAction getAction() const { return _action; };
00029         void setAction(MusicAction action) { _action = action; };
00030
00031         std::string getMusicPath() const { return _musicPath; };
00032         void setMusicPath(const std::string &musicPath) { _musicPath = musicPath; };
00033
00034         float getVolume() const { return _volume; };
00035         void setVolume(float volume) { _volume = volume; };
00036
00037     private:
00038         MusicAction _action;
00039         std::string _musicPath;
00040         float _volume;
00041 };
00042
00043 }  // namespace ecs
00044
00045 #endif /* !MUSICINTENTCOMPONENT_HPP_ */
```

5.14 SoundIntentComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** SoundIntentComponent
00006 */
00007
00008 #ifndef SOUNDINTENTCOMPONENT_HPP_
00009 #define SOUNDINTENTCOMPONENT_HPP_
00010
00011 #include "../../../common/components/base/IComponent.hpp"
00012 #include <string>
```

```
00013
00014 namespace ecs {
00015
00016 class SoundIntentComponent : public IComponent {
00017     public:
00018         SoundIntentComponent(const std::string &soundPath = "", float volume = 100.0f)
00019             : _soundPath(soundPath), _volume(volume) {};
00020         ~SoundIntentComponent() = default;
00021
00022         std::string getSoundPath() const { return _soundPath; };
00023         void setSoundPath(const std::string &soundPath) { _soundPath = soundPath; };
00024
00025         float getVolume() const { return _volume; };
00026         void setVolume(float volume) { _volume = volume; };
00027
00028     private:
00029         std::string _soundPath;
00030         float _volume;
00031 };
00032
00033 }  // namespace ecs
00034
00035 #endif /* !SOUNDINTENTCOMPONENT_HPP_ */
```

5.15 constants.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** Client Constants
00006 */
00007
00008 #ifndef CLIENT_CONSTANTS_HPP_
00009 #define CLIENT_CONSTANTS_HPP_
00010
00011 #include "../common/constants.hpp"
00012
00013 namespace constants {
00014     /* Timeout */
00015     constexpr int NETWORK_TIMEOUT = 5;
00016     constexpr int CONNECTION_ATTEMPT_TIMEOUT = 10;
00017     constexpr float HEALTHCHECK_INTERVAL = 5.0f;
00018     constexpr float WHOAMI_REQUEST_INTERVAL = 0.25f;
00019
00020     /* Window */
00021     constexpr int WINDOW_WIDTH = 1920;
00022     constexpr int WINDOW_HEIGHT = 1080;
00023
00024     /* UI */
00025     constexpr float INVALID_MOUSE_POSITION = -1.0f;
00026     constexpr float NAVIGATION_COOLDOWN_TIME = 0.1f;
00027     constexpr float UI_SCALE_SMALL = 0.75f;
00028     constexpr float UI_SCALE_NORMAL = 1.0f;
00029     constexpr float UI_SCALE_LARGE = 1.25f;
00030     constexpr size_t BUTTON_FONT_SIZE_BASE = 24;
00031     constexpr double SMALL_FONT_SIZE_MULTIPLIER = 0.5;
00032
00033     /* View */
00034     constexpr float VIEW_SMOOTHING_SPEED = 15.0f;
00035
00036     enum MouseButton {
00037         LEFT = 0,
00038         RIGHT = 1,
00039         MIDDLE = 2,
00040     };
00041
00042     /* Accessibility Filters */
00043     const std::string FILTER_HIGH_CONTRAST_SHADER_PATH = "assets/shaders/highcontrast.frag";
00044     const std::string FILTER_PROTANOPIA_SHADER_PATH = "assets/shaders/protanopia.frag";
00045     const std::string FILTER_DEUTERANOPIA_SHADER_PATH = "assets/shaders/deuteranopia.frag";
00046     const std::string FILTER_TRITANOPIA_SHADER_PATH = "assets/shaders/tritanopia.frag";
00047     const std::string FILTER_BRIGHTNESS_SHADER_PATH = "assets/shaders/brightness.frag";
00048     const std::string FILTER_BRIGHTNESS_UNIFORM_NAME = "brightness";
00049
00050     /* Health Bar */
00051     constexpr float HEALTH_BAR_OFFSET_Y = -10.0f;
00052     constexpr float HEALTH_BAR_HEIGHT = 5.0f;
00053     constexpr float HEALTH_BAR_OUTLINE_THICKNESS = 2.0f;
00054
00055     /* Settings Parsing Constants */
00056     const std::string ACCESSIBILITY_COLOR_BLINDNESS_STATE = "colorBlindnessState";
00057     const std::string ACCESSIBILITY_BRIGHTNESS_VALUE = "brightnessValue";
```

```
00058     const std::string ACCESSIBILITY_HIGH_CONTRAST_ENABLED = "highContrastEnabled";
00059     const std::string SETTINGS_UI_SCALE = "uiScale";
00060     const std::string SETTINGS_MUSIC_VOLUME = "musicVolume";
00061     const std::string SETTINGS_SOUND_VOLUME = "soundVolume";
00062     const std::string SETTINGS_SCREEN_RESOLUTION = "screenResolution";
00063     const std::string SETTINGS_TARGET_FPS = "targetFPS";
00064     const std::string SETTINGS_RENDER_QUALITY = "renderQuality";
00065     const std::string SETTINGS_USERNAME = "username";
00066     const std::string SETTINGS_IN_GAME_METRICS_ENABLED = "inGameMetricsEnabled";
00067     const std::string KEYBIND_PRIMARY = "primary";
00068     const std::string KEYBIND_SECONDARY = "secondary";
00069     const std::string KEYBIND_TOGGLE_MODE = "toggle_mode";
00070
00071     /* Paths */
00072     const std::string SAVES_DIRECTORY = "saves";
00073     const std::string KEYBINDS_FILE_PATH = "saves/keybinds.json";
00074     const std::string ACCESSIBILITY_FILE_PATH = "saves/accessibility.json";
00075     const std::string SETTINGS_FILE_PATH = "saves/settings.json";
00076     const std::string UI_BACKGROUND_EARTH_PATH = "assets/ui/background-home.png";
00077     const std::string UI_BACKGROUND_CHAT = "assets/sprites/menu/chatBackgorund.png";
00078     const std::string HOW_TO_PLAY_PATH = "assets/sprites/menu/how_to_play.png";
00079     const std::string LEADERBOARD_PATH = "assets/sprites/menu/leaderboard.png";
00080     const std::string CHAT_PATH = "assets/sprites/menu/chat.png";
00081     const std::string LEADERBOARD_PLACEHOLDER_PATH =
      "assets/sprites/menu/leaderboard-placeholder.png";
00082
00083     const std::string LOADING_PREFAVB = "configs/prefab/loading_animation.json";
00084     const std::string VICTORY_PREFAB = "configs/prefab/vicotry.json";
00085     const std::string LOSE_PREFAB = "configs/prefab/youdied.json";
00086
00087     const std::string MENU_MUSIC_PATH = "assets/musics/menu.wav";
00088     const std::string VICTORY_MUSIC_PATH = "assets/musics/victory-music.wav";
00089     const std::string DEATH_MUSIC_PATH = "assets/musics/playerDeath.wav";
00090
00091     const std::string MAIN_FONT = "assets/fonts/cuphead_font.ttf";
00092
00093     const std::string LEVEL_COMPLETE_SUB_TITLE_TEXT = "Going to the next level...";
00094
00095     /* Profile Constants */
00096     const std::string PROFILE_TITLE_TEXT = "YOUR PROFILE";
00097     const std::string USERNAME_LABEL = "Username: ";
00098     const std::string GAMES_PLAYED_LABEL = "Games Played: ";
00099     const std::string WINS_LABEL = "Wins: ";
00100     const std::string HIGH_SCORE_LABEL = "High Score: ";
00101     const std::string REFRESH_PROFILE_BUTTON_TEXT = "Refresh Profile";
00102     const std::string BACK_BUTTON_TEXT = "Back";
00103
00104     /* Chat Constants */
00105     const std::string CHAT_TITLE_TEXT = "CHAT ROOM";
00106     const std::string CHAT_NO_MESSAGES_TEXT = "No messages yet. Start the conversation!";
00107     const std::string CHAT_PLACEHOLDER_TEXT = "Type your message...";
00108     const std::string SEND_BUTTON_TEXT = "Send";
00109     const std::string BACK_BUTTON_TEXT_UPPER = "BACK";
00110
00111     /* How To Play Constants */
00112     const std::string HOW_TO_PLAY_TITLE_TEXT = "HOW TO PLAY";
00113     const std::string CONTROLS_TITLE_TEXT = "CONTROLS";
00114     const std::string OBJECTIVES_TITLE_TEXT = "OBJECTIVES";
00115     const std::string OBJECTIVE_DESTROY_ENEMIES = "Destroy Enemy Ships";
00116     const std::string OBJECTIVE_SURVIVE_WAVES = "Survive the Waves";
00117     const std::string OBJECTIVE_COLLECT_POWERUPS = "Collect Power-ups";
00118     const std::string OBJECTIVE_BEAT_HIGH_SCORE = "Beat High Score";
00119
00120     /* Leaderboard Constants */
00121     const std::string LEADERBOARD_TITLE_TEXT = "LEADERBOARD";
00122     const std::string LEADERBOARD_DEFAULT_NAME_PREFIX = "Player ";
00123     const std::string LEADERBOARD_EMPTY_NAME = "---";
00124     const std::string LEADERBOARD_DEFAULT_SCORE = "0";
00125
00126     /* Home page input place holders */
00127     const std::string IP_PLACEHOLDER = "Enter an IP address";
00128     const std::string PORT_PLACEHOLDER = "Enter a port";
00129
00130      /* Replay Constants */
00131      constexpr int MAX_REPLAY_FILES = 5;
00132     const std::string REPLAY_DIRECTORY = "saves/replays";
00133     const std::string REPLAY_FILE_PREFIX = "replay";
00134     const std::string REPLAY_BUTTON_TEXT = "Replay";
00135     const std::string REPLAY_TOTAL_TIME = "totalTime";
00136     const std::string REPLAY_RENDERABLES = "renderables";
00137     const std::string REPLAY_GAMEZONE = "gamezone";
00138     const std::string REPLAY_AUDIO = "audio";
00139     const std::string REPLAY_TYPE = "type";
00140     const std::string REPLAY_TRANSFORM = "transform";
00141     const std::string REPLAY_X = "x";
00142     const std::string REPLAY_Y = "y";
00143     const std::string REPLAY_WIDTH = "width";
```

```
00144     const std::string REPLAY_HEIGHT = "height";
00145     const std::string REPLAY_ROTATION = "r";
00146     const std::string REPLAY_SCALE_X = "sx";
00147     const std::string REPLAY_SCALE_Y = "sy";
00148     const std::string REPLAY_COLOR = "color";
00149     const std::string REPLAY_RED = "r";
00150     const std::string REPLAY_GREEN = "g";
00151     const std::string REPLAY_BLUE = "b";
00152     const std::string REPLAY_ALPHA = "a";
00153     const std::string REPLAY_SPRITE = "sprite";
00154     const std::string REPLAY_TEXTURE = "texture";
00155     const std::string REPLAY_OFFSET_X = "offsetX";
00156     const std::string REPLAY_OFFSET_Y = "offsetY";
00157     const std::string REPLAY_PARALLAX = "parallax";
00158     const std::string REPLAY_BASE_SCROLL_SPEED = "baseScrollSpeed";
00159     const std::string REPLAY_DIRECTION = "direction";
00160     const std::string REPLAY_LAYERS = "layers";
00161     const std::string REPLAY_NAME = "name";
00162     const std::string REPLAY_FILE_PATH = "filePath";
00163     const std::string REPLAY_SPEED_MULTIPLIER = "speedMultiplier";
00164     const std::string REPLAY_SCALE = "scale";
00165     const std::string REPLAY_SCALE_MODE = "scaleMode";
00166     const std::string REPLAY_SOURCE_SIZE = "sourceSize";
00167     const std::string REPLAY_REPEAT = "repeat";
00168     const std::string REPLAY_Z_INDEX = "zIndex";
00169     const std::string REPLAY_CURRENT_OFFSET = "currentOffset";
00170     const std::string REPLAY_HEALTH = "health";
00171     const std::string REPLAY_CURRENT = "current";
00172     const std::string REPLAY_MAX = "max";
00173     const std::string REPLAY_COLLIDER = "collider";
00174     const std::string REPLAY_SIZE_X = "sizeX";
00175     const std::string REPLAY_SIZE_Y = "sizeY";
00176     const std::string REPLAY_TEXT = "text";
00177     const std::string REPLAY_CONTENT = "content";
00178     const std::string REPLAY_FONT_PATH = "fontPath";
00179     const std::string REPLAY_RECTANGLE = "rectangle";
00180     const std::string REPLAY_HITBOX = "hitbox";
00181     const std::string REPLAY_SOUND_PATH = "soundPath";
00182     const std::string REPLAY_VOLUME = "volume";
00183     const std::string REPLAY_TYPE_PARALLAX = "parallax";
00184     const std::string REPLAY_TYPE_HEALTHBAR = "healthbar";
00185     const std::string REPLAY_TYPE_TEXT = "text";
00186     const std::string REPLAY_TYPE_RECTANGLE = "rectangle";
00187     const std::string REPLAY_TYPE_HITBOX = "hitbox";
00188     const std::string REPLAY_TYPE_SOUND = "sound";
00189
00190     /* Level Constants */
00191     const size_t MAX_HISTORY_SIZE = 50;
00192     const float CHANGE_DEBOUNCE_TIME = 1.0f;
00193     const float CHANGE_DEBOUNCE_TIME_SHORT = 0.25f;
00194
00195     const std::string LEVEL_FILE_PREFIX = "level";
00196     const std::string LEVEL_FILE_EXTENSION = ".json";
00197     const std::string LEVEL_DIRECTORY = "configs/map";
00198     const std::string MUSIC_DIRECTORY = "configs/entities/musics";
00199     const std::string BACKGROUNDS_DIRECTORY = "configs/entities/backgrounds";
00200     const std::string OBSTACLES_DIRECTORY = "configs/entities/obstacles";
00201     const std::string POWERUPS_DIRECTORY = "configs/entities/powerUp";
00202     const std::string ENEMIES_DIRECTORY = "configs/entities/enemies";
00203 }
00204
00205 #endif /* !CLIENT_CONSTANTS_HPP_ */
```

5.16 constants.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** Constants
00006 */
00007
00008 #ifndef CONSTANTS_HPP_
00009 #define CONSTANTS_HPP_
00010
00011 #include <cstdint>
00012 #include <string>
00013 #include "types/Vector2f.hpp"
00014
00015 namespace constants {
00016     /* Network Defaults */
00017     constexpr int DEFAULT_SERVER_PORT = 4242;
00018     const std::string DEFAULT_SERVER_IP = "127.0.0.1";
```

```
00019
00020     /* Game Defaults */
00021     constexpr float BASE_SPEED = 100.0f;
00022     constexpr float EPS = 1e-6f;
00023     constexpr float PLAYER_BASE_SPEED = 300.0f;
00024     constexpr float GAMEPAD_DEADZONE = 0.15f;
00025     constexpr float AXIS_MAX_VALUE = 100.0f;
00026     constexpr int SMOOTH_MOVEMENT_ITERATIONS = 4;
00027
00028     /* Texture Defaults */
00029     constexpr float DEFAULT_TEXTURE_WIDTH = 1920.0f;
00030     constexpr float DEFAULT_TEXTURE_HEIGHT = 1080.0f;
00031     constexpr int LOBBY_CODE_LENGTH = 8;
00032
00033     enum class EventType {
00034         UP = 0,
00035         DOWN = 1,
00036         LEFT = 2,
00037         RIGHT = 3,
00038         SHOOT = 4,
00039         STOP = 5,
00040         FORCE = 6,
00041         HEALTHCHECK = 7,
00042     };
00043
00044     constexpr char END_OFSTRING_ST = '\r';
00045     constexpr char END_OFSTRING_ND = '\n';
00046     constexpr char END_OFSTRING_TRD = '\0';
00047
00048     constexpr int MAX_RETRY_CONNECTIONS = 3;
00049     /* Paths */
00050     const std::string CONFIG_PATH = "configs/entities/";
00051     const std::string COLLISION_RULES_PATH = "configs/rules/collision_rules.json";
00052     const std::string MAPS_PATH = "configs/map/";
00053     const std::string INFINITE_MAP_PATH = "configs/map/infinite.json";
00054
00055     /* Collision Rules JSON Keys */
00056     const std::string COLLISION_SOLID_KEY = "solid";
00057     const std::string COLLISION_TRIGGER_KEY = "trigger";
00058     const std::string COLLISION_PUSH_KEY = "push";
00059     const std::string COLLISION_ALLOW_KEY = "allow";
00060
00061     /* Parsing constants */
00062     const std::string SERVER_VALUE = "server";
00063     const std::string CLIENT_VALUE = "client";
00064     const std::string BOTH_VALUE = "both";
00065
00066     /* Components */
00067     const std::string TRANSFORMCOMPONENT = "TransformComponent";
00068     const std::string VELOCITYCOMPONENT = "VelocityComponent";
00069     const std::string SPEEDCOMPONENT = "SpeedComponent";
00070     const std::string SPRITECOMPONENT = "SpriteComponent";
00071     const std::string ANIMATIONCOMPONENT = "AnimationComponent";
00072     const std::string SHOOTINGSTATSCOMPONENT = "ShootingStatsComponent";
00073     const std::string RECTANGLERENDERCOMPONENT = "RectangleRenderComponent";
00074     const std::string PROJECTILEPREFABCOMPONENT = "ProjectilePrefabComponent";
00075     const std::string TEXTCOMPONENT = "TextComponent";
00076     const std::string HEALTHBARCOMPONENT = "HealthBarComponent";
00077     const std::string SCORECOMPONENT = "ScoreComponent";
00078     const std::string SCOREVALUECOMPONENT = "ScoreValueComponent";
00079     const std::string DAMAGECOMPONENT = "DamageComponent";
00080     const std::string DAMAGECOOLDOWNCOMPONENT = "DamageCooldownComponent";
00081     const std::string HEALTHCOMPONENT = "HealthComponent";
00082     const std::string HITBOXRENDERCOMPONENT = "HitboxRenderComponent";
00083     const std::string INTERACTIONCONFIGCOMPONENT = "InteractionConfigComponent";
00084     const std::string SCRIPTINGCOMPONENT = "ScriptingComponent";
00085     const std::string SOUNDINTENTCOMPONENT = "SoundIntentComponent";
00086     const std::string ENTITYPARTSCOMPONENT = "EntityPartsComponent";
00087     const std::string LIFETIMECOMPONENT = "LifetimeComponent";
00088     const std::string LIFESPANCOMPONENT = "LifeSpanComponent";
00089     const std::string MUSICCOMPONENT = "MusicComponent";
00090     const std::string COLLIDERCOMPONENT = "ColliderComponent";
00091     const std::string PARALLAXCOMPONENT = "ParallaxComponent";
00092     const std::string GAMEZONECOMPONENT = "GameZoneComponent";
00093     const std::string CHARGEDSHOTCOMPONENT = "ChargedShotComponent";
00094     const std::string INVULNERABLECOMPONENT = "InvulnerableComponent";
00095
00096     /* Fields */
00097     const std::string SCRIPT_PATH_FIELD = "scriptPath";
00098     const std::string ADDITIONAL_FUNCTIONS_FIELD = "additionalFunctions";
00099     const std::string SCORE_FIELD = "score";
00100     const std::string SCOREVALUE_FIELD = "scoreValue";
00101     const std::string DAMAGE_FIELD = "damage";
00102     const std::string HEALTH_FIELD = "health";
00103     const std::string COOLDOWN_FIELD = "cooldown";
00104     const std::string ACTIVE_FIELD = "active";
00105     const std::string TARGET_FIELD = "target";
```

```
00106     const std::string POSITION_FIELD = "position";
00107     const std::string OFFSET_FIELD = "offset";
00108     const std::string SCALE_FIELD = "scale";
00109     const std::string ROTATION_FIELD = "rotation";
00110     const std::string SPEED_FIELD = "speed";
00111     const std::string FILEPATH_FIELD = "filePath";
00112     const std::string ANIMATIONPATH_FIELD = "animationPath";
00113     const std::string FRAMEWIDTH_FIELD = "frameWidth";
00114     const std::string FRAMEHEIGHT_FIELD = "frameHeight";
00115     const std::string FRAMECOUNT_FIELD = "frameCount";
00116     const std::string STARTWIDTH_FIELD = "startWidth";
00117     const std::string STARTHEIGHT_FIELD = "startHeight";
00118     const std::string SIZE_FIELD = "size";
00119     const std::string FIRERATE_FIELD = "fireRate";
00120     const std::string SHOTCOUNT_FIELD = "shotCount";
00121     const std::string ANGLEOFFSET_FIELD = "angleOffset";
00122     const std::string SPREADANGLE_FIELD = "spreadAngle";
00123     const std::string OFFSETDISTANCE_FIELD = "offsetDistance";
00124     const std::string DEFAULTBEHAVIOR_FIELD = "defaultBehavior";
00125     const std::string ZIGZAGAMPLITUDE_FIELD = "zigzagAmplitude";
00126     const std::string ZIGZAGFREQUENCY_FIELD = "zigzagFrequency";
00127     const std::string DETECTIONRANGE_FIELD = "detectionRange";
00128     const std::string VERTICALDEADZONE_FIELD = "verticalDeadzone";
00129     const std::string WIDTH_FIELD = "width";
00130     const std::string HEIGHT_FIELD = "height";
00131     const std::string COLOR_FIELD = "color";
00132     const std::string R_FIELD = "r";
00133     const std::string G_FIELD = "g";
00134     const std::string B_FIELD = "b";
00135     const std::string FORCE_TYPE = "force";
00136     const std::string FORCE_TYPE_FIELD = "forceType";
00137     const std::string STATES_FIELD = "states";
00138     const std::string INITIALSTATE_FIELD = "initialState";
00139     const std::string TRANSITIONS_FIELD = "transitions";
00140     const std::string CONDITIONS_FIELD = "conditions";
00141     const std::string CONDITION_FIELD = "condition";
00142     const std::string PARAM_FIELD = "param";
00143     const std::string EQUALS_FIELD = "equals";
00144     const std::string FROM_FIELD = "from";
00145     const std::string BASESCROLLSPEED_FIELD = "baseScrollSpeed";
00146     const std::string DIRECTION_FIELD = "direction";
00147     const std::string LAYERS_FIELD = "layers";
00148     const std::string ZONERECT_FIELD = "zoneRect";
00149     const std::string NAME_FIELD = "name";
00150     const std::string INDEX_FIELD = "index";
00151     const std::string SPEEDMULTIPLIER_FIELD = "speedMultiplier";
00152     const std::string SCALEMODE_FIELD = "scaleMode";
00153     const std::string SOURCESIZE_FIELD = "sourceSize";
00154     const std::string REPEAT_FIELD = "repeat";
00155     const std::string ZINDEX_FIELD = "zIndex";
00156     const std::string TO_FIELD = "to";
00157     const std::string REWIND_FIELD = "rewind";
00158     const std::string TEXTUREPATH_FIELD = "texturePath";
00159     const std::string LOOP_FIELD = "loop";
00160     const std::string X_FIELD = "x";
00161     const std::string Y_FIELD = "y";
00162     const std::string ANIMATIONSTATECOMPONENT = "AnimationStateComponent";
00163     const std::string PREFABNAME_FIELD = "prefabName";
00164     const std::string LIFETIME_FIELD = "lifetime";
00165     const std::string LIFESPAN_FIELD = "lifespan";
00166     const std::string TEXT_FIELD = "text";
00167     const std::string FONTPATH_FIELD = "fontPath";
00168     const std::string SOUND_FILE_FIELD = "soundFile";
00169     const std::string MAPPINGS_FIELD = "mappings";
00170     const std::string TAGS_FIELD = "tags";
00171     const std::string TOENTITY_FIELD = "toEntity";
00172     const std::string TOSELF_FIELD = "toSelf";
00173
00174     const std::string MUSICFILE_FIELD = "musicFile";
00175     const std::string VOLUME_FIELD = "volume";
00176     const std::string INITIALSTATEMUSIC_FIELD = "initialState";
00177     const std::string PLAYING_FIELD = "PLAYING";
00178     const std::string PAUSED_FIELD = "PAUSED";
00179     const std::string STOPPED_FIELD = "STOPPED";
00180     const std::string CHANGING_FIELD = "CHANGING";
00181
00182     const std::string MAXCHARGE_FIELD = "maxCharge";
00183     const std::string CHARGERELOADTIME_FIELD = "chargeReloadTime";
00184
00185     const std::string STRAIGHT_LINE_VALUE = "StraightLine";
00186     const std::string ZIGZAG_VALUE = "Zigzag";
00187     const std::string VERTICAL_MIRROR_VALUE = "VerticalMirror";
00188     const std::string FOLLOW_RIGHT_VALUE = "FollowRight";
00189     const std::string SCALEMODE_FITSCREEN = "FIT_SCREEN";
00190     const std::string SCALEMODE_STRETCH = "STRETCH";
00191     const std::string SCALEMODE_MANUAL = "MANUAL";
00192     const std::string COLLISION_TYPE_SOLID = "Solid";
```

```
00193     const std::string COLLISION_TYPE_TRIGGER = "Trigger";
00194     const std::string COLLISION_TYPE_PUSH = "Push";
00195     const std::string COLLISION_TYPE_NONE = "None";
00196
00197     const float MAX_HEIGHT = 1080.0f;
00198     const float MAX_WIDTH = 1920.0f;
00199     const float GAME_ZONE_BOUNDARY_THICKNESS = 100.0f;
00200
00201     const float SPATIAL_GRID_CELL_SIZE = 128.0f;
00202     const float SPATIAL_GRID_PADDING = 200.0f;
00203     const float OUT_OF_BOUNDS_MARGIN = 200.0f;
00204
00205     /* Map parsing constants */
00206     const std::string COMPONENTS_FIELD = "components";
00207     const std::string BACKGROUND_FIELD = "background";
00208     const std::string BACKGROUND_SCROLL_SPEED_FIELD = "scrollSpeed";
00209     const std::string MUSIC_FIELD = "music";
00210     const std::string POWERUPS_FIELD = "powerUps";
00211     const std::string OBSTACLES_FIELD = "obstacles";
00212     const std::string WAVES_FIELD = "waves";
00213     const std::string MAP_LENGTH_FIELD = "mapLength";
00214     const std::string POSITIONS_FIELD = "positions";
00215     const std::string TYPE_FIELD = "type";
00216     const std::string FROMX_FIELD = "fromX";
00217     const std::string FROMY_FIELD = "fromY";
00218     const std::string POSX_FIELD = "posX";
00219     const std::string POSY_FIELD = "posY";
00220     const std::string COUNT_FIELD = "count";
00221     const std::string GAMEXTRIGGER_FIELD = "gameXTrigger";
00222     const std::string GAME_ZONE_STOP_AT_X_FIELD = "gameZoneStopAtX";
00223     const std::string DISTRIBUTIONX_FIELD = "distributionX";
00224     const std::string DISTRIBUTIONY_FIELD = "distributionY";
00225     const std::string ENEMIES_FIELD = "enemies";
00226     const std::string MIN_FIELD = "min";
00227     const std::string MAX_FIELD = "max";
00228     const math::Vector2f BACKGROUND_POSITION = math::Vector2f(0.0f, 0.0f);
00229     const float BACKGROUND_PARALLAX_DIRECTION_X = -1.0f;
00230     const float BACKGROUND_PARALLAX_DIRECTION_Y = 0.0f;
00231     const std::string EMPTY_PREFAB = "empty";
00232
00233     const std::string HORIZONTAL_LINE_TYPE = "horizontalLine";
00234     const std::string VERTICAL_LINE_TYPE = "verticalLine";
00235     const std::string UNIQUE_TYPE = "unique";
00236     const std::string RANDOM_TYPE = "random";
00237     const std::string UNIFORM_TYPE = "uniform";
00238
00239     /* Animation conditions */
00240     const std::string VELOCITY_UP_CONDITION = "isVelocityUp";
00241     const std::string VELOCITY_DOWN_CONDITION = "isVelocityDown";
00242     const std::string ANIMATION_END_CONDITION = "onAnimationEnd";
00243
00244     /* Tags */
00245     const std::string CONTROLLABLETAG = "ControllableTag";
00246     const std::string PLAYERTAG = "PlayerTag";
00247     const std::string MOBTAG = "MobTag";
00248     const std::string SHOOTERTAG = "ShooterTag";
00249     const std::string PLAYERPROJECTILETAG = "PlayerProjectileTag";
00250     const std::string ENNEMYPROJECTILETAG = "EnnemyProjectileTag";
00251     const std::string PROJECTILEPASSTHROUGHTAG = "ProjectilePassThroughTag";
00252     const std::string GAMEZONECOLLIDERTAG = "GameZoneColliderTag";
00253     const std::string GAME_ZONE_STOP_TAG = "GameZoneStopTag";
00254     const std::string OBSTACLETAG = "ObstacleTag";
00255     const std::string PLAYEROBSTACLETAG = "PlayerObstacleTag";
00256
00257     /* Difficulty Multipliers */
00258     constexpr float DIFFICULTY_EASY_MULTIPLIER = 1.3f;
00259     constexpr float DIFFICULTY_NORMAL_MULTIPLIER = 1.0f;
00260     constexpr float DIFFICULTY_HARD_MULTIPLIER = 0.7f;
00261     const std::string CLIENTEFFECTTAG = "ClientEffectTag";
00262     const std::string BACKGROUNDMUSICTAG = "BackGroundMusicTag";
00263     const std::string POWERUP_TAG = "PowerUpTag";
00264     const std::string FORCE_TAG = "ForceTag";
00265
00266     /* Action constants */
00267     const std::string DEALDEATH_ACTION = "DealDeath";
00268     const std::string TAKEDEATH_ACTION = "TakeDeath";
00269     const std::string DEALDAMAGE_ACTION = "DealDamage";
00270     const std::string TAKEDAMAGE_ACTION = "TakeDamage";
00271     const std::string ADDLIFE_ACTION = "AddLife";
00272     const std::string INTERACTION_CALL_SCRIPTING_ACTION = "InteractionCallScripting";
00273
00274     /* Cooldown constants */
00275     const float TRIGGER_DAMAGE_COOLDOWN = 0.1f;
00276
00277     /* Prefabs */
00278     const std::string PLAYER_PREFAB_NAME = "player";
00279     const std::string GAME_ZONE_PREFAB = "gamezone";
```

```
00280     const std::string SMALL_EXPLOSION = "small_explosion";
00281     const std::string BIG_EXPLOSION = "big_explosion";
00282     const std::string OBSTACLE_1 = "obstacle1";
00283     const std::string ENEMY_1 = "bat";
00284     const std::string ENEMY_2 = "canon";
00285     const std::string POWERUP_ADD_LIFE = "powerUpAddLife";
00286     const std::string POWERUP_FORCE = "force";
00287     const std::string POWERUP_FLYING_FORCE = "flyingForce";
00288     const std::string POWERUP_SPEED = "powerUpSpeed";
00289     const std::string POWERUP_CHARGE_TIME = "powerUpChargeTime";
00290
00291     constexpr float DEFAULT_TIMER = 0.0f;
00292
00293     /* Packet constants */
00294     constexpr std::uint8_t PACKET_NO_OP = 0x00;
00295     constexpr std::uint8_t PACKET_CONNECTION = 0x01;
00296     constexpr std::uint8_t PACKET_ACCEPT = 0x02;
00297     constexpr std::uint8_t PACKET_DISC = 0x03;
00298     constexpr std::uint8_t PACKET_EVENT = 0x04;
00299     constexpr std::uint8_t PACKET_END_GAME = 0x05;
00300     constexpr std::uint8_t PACKET_CAN_START = 0x06;
00301     constexpr std::uint8_t PACKET_CLIENT_READY = 0x07;
00302     constexpr std::uint8_t PACKET_SPAWN = 0x08;
00303     constexpr std::uint8_t PACKET_DEATH = 0x09;
00304     constexpr std::uint8_t PACKET_WHOAMI = 0x0A;
00305     constexpr std::uint8_t PACKET_SERVER_STATUS = 0x0B;
00306     constexpr std::uint8_t PACKET_REQUEST_LOBBY = 0x0C;
00307     constexpr std::uint8_t PACKET_SEND_LOBBY_CODE = 0x0D;
00308     constexpr std::uint8_t PACKET_CONNECT_TO_LOBBY = 0x0E;
00309     constexpr std::uint8_t PACKET_LOBBY_MASTER_REQUEST_START = 0x0F;
00310     constexpr std::uint8_t PACKET_LOBBY_CONNECT_VALUE = 0x10;
00311     constexpr std::uint8_t PACKET_LEVEL_COMPLETE = 0x11;
00312     constexpr std::uint8_t PACKET_NEXT_LEVEL = 0x12;
00313     constexpr std::uint8_t PACKET_REGISTER = 0x13;
00314     constexpr std::uint8_t PACKET_CONNECT_USER = 0x14;
00315     constexpr std::uint8_t PACKET_LOGIN = 0x15;
00316     constexpr std::uint8_t PACKET_GAME_STATE_BATCH = 0x16;
00317     constexpr std::uint8_t PACKET_GAME_STATE_BATCH_COMPRESSED = 0x17;
00318     constexpr std::uint8_t PACKET_GAME_STATE_COMPRESSED = 0x18;
00319     constexpr std::uint8_t PACKET_REQUEST_LEADERBOARD = 0x19;
00320     constexpr std::uint8_t PACKET_LEADERBOARD = 0x1A;
00321     constexpr std::uint8_t PACKET_REGISTER_FAIL = 0x1B;
00322     constexpr std::uint8_t PACKET_REQUEST_PROFILE = 0x1C;
00323     constexpr std::uint8_t PACKET_PROFILE = 0x1D;
00324     constexpr std::uint8_t PACKET_GAME_RULES = 0x1E;
00325     constexpr std::uint8_t PACKET_REQUEST_GAME_RULES_UPDATE = 0x1F;
00326     constexpr std::uint8_t PACKET_NEW_CHAT = 0x20;
00327     constexpr std::uint8_t PACKET_BROADCASTED_CHAT = 0x21;
00328     constexpr std::uint8_t PACKET_FORCE_LEAVE = 0x22;
00329     constexpr std::uint8_t PACKET_LEAVE_LOBBY = 0x23;
00330     constexpr std::uint8_t PACKET_ACK_LEAVE_LOBBY = 0x24;
00331
00332     constexpr std::uint8_t MAX_INDEX_PACKET_TYPE = 37;
00333     const int MAX_CLIENT_PER_LOBBY = 4;
00334
00335     /* Lobby connection codes */
00336     const std::string LOBBY_LEAVE_MARKER = "__LEAVE__";
00337     const std::string LOBBY_LEAVE_KEYWORD = "LEAVE";
00338
00339     enum class ForceLeaveType {
00340         CLOSED = 0,
00341         KICKED = 1,
00342         BANNED = 2,
00343     };
00344
00345     /* Scripting APU constant */
00346     const std::string INIT_FUNCTION = "init";
00347     const std::string UPDATE_FUNCTION = "update";
00348     const std::string DEATH_FUNCTION = "death";
00349     const std::string ONINTERACT_FUNCTION = "OnInteract";
00350     const std::string ACTIVATE_OR_DEACTIVATE_FORCE_FUNCTION = "ActivateOrDeactivateForce";
00351     const std::string ADD_FORCE_LEVEL_FUNCTION = "addForceLevel";
00352
00353     /* Constants for Scripting API */
00354     const std::string PRINT_FUNCTION = "print";
00355     const std::string CREATE_MOVE_INTENT_FUNCTION = "createMoveIntent";
00356     const std::string GET_ENTITY_POSITION_FUNCTION = "getEntityPosition";
00357     const std::string GET_NEAREST_PLAYER_POSITION_FUNCTION = "getNearestPlayerPosition";
00358     const std::string GET_ENTITY_SPEED_FUNCTION = "getEntitySpeed";
00359     const std::string SET_ENTITY_SPEED_FUNCTION = "setEntitySpeed";
00360     const std::string GET_MAX_CHARGE_FUNCTION = "getMaxCharge";
00361     const std::string SET_MAX_CHARGE_FUNCTION = "setMaxCharge";
00362     const std::string GET_CHARGE_RELOAD_TIME_FUNCTION = "getChargeReloadTime";
00363     const std::string SET_CHARGE_RELOAD_TIME_FUNCTION = "setChargeReloadTime";
00364     const std::string CREATE_SHOOT_INTENT_FUNCTION = "createShootIntent";
00365     const std::string SET_ANIMATION_STATE_FUNCTION = "setAnimationState";
00366     const std::string SPAWN_ENTITY_FUNCTION = "spawnEntity";
```

```
00367     const std::string GET_ENTITY_ID_FUNCTION = "getEntityId";
00368     const std::string ADD_PART_ID_FUNCTION = "addPartId";
00369     const std::string SET_PARENT_ID_FUNCTION = "setParentId";
00370     const std::string SET_ENTITY_ROTATION_FUNCTION = "setEntityRotation";
00371     const std::string GET_ENTITY_PARTS_FUNCTION = "getEntityParts";
00372     const std::string CREATE_DEATH_INTENT_FUNCTION = "createDeathIntent";
00373     const std::string IS_ENTITY_ALIVE_FUNCTION = "isEntityAlive";
00374     const std::string GET_PARENT_ID_FUNCTION = "getParentId";
00375     const std::string GET_OWNER_FUNCTION = "getOwner";
00376     const std::string SET_FIRE_RATE_FUNCTION = "setFireRate";
00377     const std::string GET_FIRE_RATE_FUNCTION = "getFireRate";
00378     const std::string REMOVE_PART_ID_FUNCTION = "removePartId";
00379     const std::string IS_ENTITY_PLAYER_FUNCTION = "isEntityPlayer";
00380     const std::string GET_ENTITY_SIZE_FUNCTION = "getEntitySize";
00381     const std::string SET_POSITION_FUNCTION = "setPosition";
00382     const std::string SET_PROJECTILE_PREFAB_FUNCTION = "setProjectilePrefab";
00383     const std::string COUNT_FORCES_BY_TYPE_FUNCTION = "countForcesByType";
00384     const std::string GET_FORCE_POSITION_BY_TYPE_FUNCTION = "getForcePositionByType";
00385     const std::string GET_GAME_ZONE_POSITION_FUNCTION = "getGameZonePosition";
00386     const std::string GET_GAME_ZONE_SIZE_FUNCTION = "getGameZoneSize";
00387     const std::string GET_GAME_ZONE_VELOCITY_FUNCTION = "getGameZoneVelocity";
00388     const std::string REVERSE_SHOOT_ORIENTATION_FUNCTION = "reverseShootOrientation";
00389     const std::string SET_GAME_ZONE_VELOCITY_FUNCTION = "setGameZoneVelocity";
00390     const std::string GET_ENTITY_VELOCITY_FUNCTION = "getEntityVelocity";
00391     const std::string SET_INVULNERABLE_FUNCTION = "setInvulnerable";
00392     const std::string GET_NEAREST_ENEMY_POSITION_FUNCTION = "getNearestEnemyPosition";
00393     const std::string GET_ENTITY_OWNER_FUNCTION = "getEntityOwner";
00394     const std::string SET_ENTITY_OWNER_FUNCTION = "setEntityOwner";
00395 }
00396
00397 #endif /* !CONSTANTS_HPP_ */
```

5.17 constants.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** R-Type
00004 ** File description:
00005 ** Header
00006 */
00007
00008 #ifndef SERVER_CONSTANTS_HPP_
00009 #define SERVER_CONSTANTS_HPP_
00010
00011 #include <cstdint>
00012 #include "../common/constants.hpp"
00013
00014 namespace constants {
00015     /* TPS */
00016     constexpr int64_t TPS = 60;
00017
00018     /* Core */
00019     constexpr long SERVER_THREAD_SLEEP_MS = 10;
00020     constexpr int SERVER_UP = 1;
00021
00022     /* Server */
00023     constexpr uint8_t ID_SERVER = 0;
00024     constexpr uint8_t BITMASK_INT = 32;
00025     constexpr int MAX_CLIENT = 4;
00026     constexpr int CLIENT_TIMEOUT_SECONDS = 15;
00027
00028     /* HTTP Server */
00029     const std::string DEFAULT_HTTP_PASSWORD = "defaultpassword";
00030     const std::string HTTP_ENV_FILE_PATH = "./server/http/.env";
00031     constexpr int HTTP_SERVER_PORT = 5173;
00032
00033     constexpr float SHOOT_INPUT_COOLDOWN = 0.2f;
00034
00035     const std::string USERS_JSON_PATH = "saves/users.json";
00036     const std::string SCORES_JSON_PATH = "saves/scores.json";
00037     const std::string USERNAME_JSON_WARD = "username";
00038     const std::string PASSWORD_JSON_WARD = "password";
00039     const std::string GAMES_PLAYED_JSON_WARD = "games_played";
00040     const std::string WINS_JSON_WARD = "wins";
00041     const std::string HIGH_SCORE_JSON_WARD = "high_score";
00042     const std::string TIME_SPENT_JSON_WARD = "time_spent";
00043     const std::string BANNED_JSON_WARD = "banned";
00044     const std::string SCORE_JSON_WARD = "scores";
00045     const std::string GODMOD_JSON_WARD = "godmod";
00046 }
00047
00048 #endif /* !SERVER_CONSTANTS */
```

5.18 Core.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** Core.hpp
00006 */
00007
00008 #ifndef CORE_HPP_
00009 #define CORE_HPP_
00010
00011 #include <memory>
00012 #include <thread>
00013 #include <chrono>
00014 #include "../common/resourceManager/ResourceManager.hpp"
00015 #include "ClientNetwork.hpp"
00016 #include "../common/interfaces/IWindow.hpp"
00017 #include "../common/interfaces/IEvent.hpp"
00018 #include "../common/interfaces/IAudio.hpp"
00019 #include "gsm/machine/GameStateMachine.hpp"
00020 #include "../common/DLLoader/DLLoader.hpp"
00021 #include "../common/Parser/Parser.hpp"
00022
00023 class Core
00024 {
00025     public:
00026         Core();
00027         ~Core();
00028
00029         void initFirstScene();
00030         void run();
00031         void startNetwork();
00032
00033         std::shared_ptr<ClientNetwork> getNetwork();
00034
00035     private:
00036         std::shared_ptr<DLLoader<gfx::createWindow_t» _windowLoader;
00037         std::shared_ptr<DLLoader<gfx::createEvent_t» _eventLoader;
00038         std::shared_ptr<DLLoader<gfx::createAudio_t» _audioLoader;
00039
00040         std::shared_ptr<ResourceManager> _resourceManager;
00041         std::shared_ptr<gsm::GameStateMachine> _gsm;
00042         std::shared_ptr<ecs::Registry> _registry;
00043         std::shared_ptr<ClientNetwork> _clientNetwork;
00044         std::shared_ptr<Parser> _parser;
00045         std::thread _networkThread;
00046
00047         std::chrono::steady_clock::time_point _lastHealthcheckTime;
00048         float _healthcheckInterval = constants::HEALTHCHECK_INTERVAL;
00049
00050         void initNetwork();
00051         void initLibraries();
00052         void networkLoop();
00053 };
00054
00055 #endif /* !CORE_HPP_ */
```

5.19 AGameStateMachine.hpp

```
00001 #pragma once
00002
00003 #include "../../../common/gsm/IGameStateMachine.hpp"
00004 #include "../../../common/gsm/IGameState.hpp"
00005
00006 namespace gsm {
00007
00008 class AGameStateMachine : public IGameStateMachine {
00009 public:
00010     AGameStateMachine();
00011     ~AGameStateMachine() override = default;
00012
00013     void changeState(std::shared_ptr<IGameState> newState) override;
00014     void pushState(std::shared_ptr<IGameState> newState) override;
00015     void popState() override;
00016     void requestStateChange(std::shared_ptr<IGameState> newState) override;
00017     void requestStatePush(std::shared_ptr<IGameState> newState) override;
00018     void requestStatePop() override;
00019
00020     void update(float deltaTime) override;
00021
00022 protected:
```

```
00023     std::stack<std::shared_ptr<IGameState» _states;
00024     std::shared_ptr<IGameState> _pendingChangeState;
00025     std::shared_ptr<IGameState> _pendingPushState;
00026     bool _pendingPopState = false;
00027 };
00028
00029 }  // namespace gsm
```

5.20 AGameStateMachine.hpp

```
00001 #pragma once
00002
00003 #include "../../../common/gsm/IGameStateMachine.hpp"
00004 #include "../../../common/gsm/IGameState.hpp"
00005
00006 namespace gsm {
00007
00008 class AGameStateMachine : public IGameStateMachine {
00009 public:
00010     AGameStateMachine();
00011     ~AGameStateMachine() override;
00012
00013     void changeState(std::shared_ptr<IGameState> newState) override;
00014     void pushState(std::shared_ptr<IGameState> newState) override;
00015     void popState() override;
00016
00017     void update(float deltaTime) override;
00018
00019     void requestStateChange(std::shared_ptr<IGameState> newState) override;
00020
00021     std::string getCurrentStateName() const;
00022
00023 protected:
00024     std::stack<std::shared_ptr<IGameState» _states;
00025 };
00026
00027 } // namespace gsm
```

5.21 GameStateMachine.hpp

```
00001 #pragma once
00002
00003 #include "AGameStateMachine.hpp"
00004
00005 namespace gsm {
00006
00007 class GameStateMachine : public AGameStateMachine {
00008 public:
00009     GameStateMachine();
00010     ~GameStateMachine() override = default;
00011 };
00012
00013 }  // namespace gsm
```

5.22 GameStateMachine.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** GameStateMachine
00006 */
00007
00008 #pragma once
00009
00010 #include "AGameStateMachine.hpp"
00011
00012 namespace gsm {
00013
00014 class GameStateMachine : public AGameStateMachine {
00015 public:
00016     GameStateMachine();
00017     ~GameStateMachine() override = default;
```

```
00018
00019     void requestStateChange(std::shared_ptr<IGameState> newState) override;
00020     void requestStatePush(std::shared_ptr<IGameState> newState) override;
00021     void requestStatePop() override;
00022 };
00023
00024 }  // namespace gsm
```

5.23 AGameState.hpp

```
00001 #pragma once
00002
00003 #include "../../../../common/gsm/IGameState.hpp"
00004 #include "../../../../common/resourceManager/ResourceManager.hpp"
00005
00006 namespace gsm {
00007
00008 class AGameState : public IGameState {
00009     public:
00010         AGameState(std::shared_ptr<IGameStateMachine> gsm, std::shared_ptr<ResourceManager>
      resourceManager);
00011         ~AGameState() override = default;
00012
00013         void enter() override;
00014         void update(float deltaTime) override;
00015         void exit() override;
00016         std::vector<std::shared_ptr<ecs::ISystem» getSystems() const override;
00017
00018     protected:
00019         void addSystem(std::shared_ptr<ecs::ISystem> system) override;
00020         std::weak_ptr<IGameStateMachine> _gsm;
00021         std::shared_ptr<ResourceManager> _resourceManager;
00022         std::vector<std::shared_ptr<ecs::ISystem» _systems;
00023 };
00024
00025 } // namespace gsm
```

5.24 AGameState.hpp

```
00001 #pragma once
00002
00003 #include "../../../common/gsm/IGameState.hpp"
00004 #include "resourceManager/ResourceManager.hpp"
00005
00006 namespace gsm {
00007
00008 class AGameState : public IGameState {
00009     public:
00010         AGameState(std::shared_ptr<IGameStateMachine> gsm, std::shared_ptr<ResourceManager>
      resourceManager);
00011         ~AGameState() override;
00012
00013         void enter() override;
00014         void update(float deltaTime) override;
00015         void exit() override;
00016         std::vector<std::shared_ptr<ecs::ISystem» getSystems() const override;
00017         std::string getStateName() const override = 0;
00018
00019     protected:
00020         void addSystem(std::shared_ptr<ecs::ISystem> system) override;
00021         std::weak_ptr<IGameStateMachine> _gsm;
00022         std::shared_ptr<ResourceManager> _resourceManager;
00023         std::vector<std::shared_ptr<ecs::ISystem» _systems;
00024 };
00025
00026 } // namespace gsm
```

5.25 ChatState.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2026
00003 ** R-Type
00004 ** File description:
```

```
00005 ** Header
00006 */
00007
00008 #ifndef CHAT_STATE_HPP_
00009 #define CHAT_STATE_HPP_
00010
00011 #include "../../base/AGameState.hpp"
00012 #include "resourceManager/ResourceManager.hpp"
00013 #include "../../../../input/MouseInputHandler.hpp"
00014 #include "../../../../ui/elements/focusable/Button.hpp"
00015 #include "../../../../ui/elements/focusable/TextInput.hpp"
00016 #include "../../../../ui/manager/UIManager.hpp"
00017 #include "../../../../ui/core/UILayout.hpp"
00018 #include "../../../../ui/elements/Background.hpp"
00019 #include "../../../../ui/elements/Text.hpp"
00020 #include "../../../../ClientNetwork.hpp"
00021
00022 namespace gsm {
00023
00024 class ChatState : public AGameState {
00025 public:
00026     ChatState(std::shared_ptr<IGameStateMachine> gsm, std::shared_ptr<ResourceManager>
      resourceManager);
00027     ~ChatState() override = default;
00028
00029     void enter() override;
00030     void update(float deltaTime) override;
00031     void exit() override;
00032     std::string getStateName() const override { return "Chat"; }
00033
00034 private:
00035     void renderUI();
00036     void onBackButtonClicked();
00037     void onSendMessage(const std::string& text);
00038
00039 private:
00040     std::unique_ptr<MouseInputHandler> _mouseHandler;
00041     std::unique_ptr<ui::UIManager> _uiManager;
00042
00043     std::shared_ptr<ui::Background> _background;
00044     std::shared_ptr<ui::UILayout> _mainLayout;
00045
00046     std::shared_ptr<ui::Button> _backButton;
00047     std::shared_ptr<ui::Text> _titleText;
00048     std::shared_ptr<ui::TextInput> _messageInput;
00049     std::shared_ptr<ui::Button> _sendButton;
00050     std::shared_ptr<ui::UILayout> _messagesContainer;
00051 };
00052
00053 }  // namespace gsm
00054
00055 #endif /* !CHAT_STATE_HPP_ */
```

5.26 ConnectionState.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2026
00003 ** R-Type
00004 ** File description:
00005 ** Header
00006 */
00007
00008 #ifndef CONNECTION_STATE_HPP_
00009 #define CONNECTION_STATE_HPP_
00010
00011 #include "../../base/AGameState.hpp"
00012 #include "../../../../ui/manager/UIManager.hpp"
00013 #include "../../../../input/MouseInputHandler.hpp"
00014 #include "../../../../ui/elements/focusable/Button.hpp"
00015 #include "../../../../ui/core/UILayout.hpp"
00016 #include "../../../../ui/elements/Background.hpp"
00017 #include "../../../../ui/elements/focusable/TextInput.hpp"
00018 #include "../../../../ui/elements/Text.hpp"
00019 #include <memory>
00020 #include "../../../../ui/elements/SpritePreview.hpp"
00021
00022
00023 namespace gsm {
00024
00025 class ConnectionState : public AGameState {
00026     public:
00027         ConnectionState(std::shared_ptr<IGameStateMachine> gsm, std::shared_ptr<ResourceManager>
      resourceManager);
```

```
00028         ~ConnectionState() override = default;
00029
00030         void enter() override;
00031         void update(float deltaTime) override;
00032         void exit() override;
00033         std::string getStateName() const override { return "Connecting"; }
00034
00035     private:
00036         void renderUI();
00037         void updateUIStatus();
00038
00039         std::unique_ptr<ui::UIManager> _uiManager;
00040         std::unique_ptr<MouseInputHandler> _mouseHandler;
00041         std::shared_ptr<ui::Background> _background;
00042         std::shared_ptr<ui::TextInput> _ipInput;
00043         std::shared_ptr<ui::TextInput> _portInput;
00044         std::shared_ptr<ui::Button> _connectButton;
00045         std::shared_ptr<ui::Button> _levelEditorButton;
00046         std::shared_ptr<ui::Button> _quitButton;
00047         std::shared_ptr<ui::Text> _spacer;
00048         std::shared_ptr<ui::UILayout> _layout;
00049         std::shared_ptr<ui::SpritePreview> _loadingAnimation;
00050         std::shared_ptr<ui::UILayout> _loadingLayout;
00051         bool _wasConnected = false;
00052 };
00053
00054 } // namespace gsm
00055
00056 #endif /* !CONNECTION_STATE_HPP_ */
```

5.27 ForceLeaveState.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ForceLeaveState
00006 */
00007
00008 #ifndef FORCELEAVESTATE_HPP_
00009 #define FORCELEAVESTATE_HPP_
00010
00011 #include "../../base/AGameState.hpp"
00012 #include "resourceManager/ResourceManager.hpp"
00013 #include "../../../../../common/interfaces/IWindow.hpp"
00014 #include "ui/manager/UIManager.hpp"
00015 #include "ui/elements/Text.hpp"
00016 #include "ui/elements/focusable/Button.hpp"
00017 #include "ui/core/UILayout.hpp"
00018 #include "../../../../input/MouseInputHandler.hpp"
00019 #include "../../../../colors.hpp"
00020 #include "../../../../../common/constants.hpp"
00021
00022 namespace gsm {
00023
00024 class ForceLeaveState : public AGameState {
00025     public:
00026         ForceLeaveState(std::shared_ptr<IGameStateMachine> gsm, std::shared_ptr<ResourceManager>
      resourceManager, constants::ForceLeaveType leaveType);
00027         ~ForceLeaveState() override = default;
00028
00029         void enter() override;
00030         void update(float deltaTime) override;
00031         void exit() override;
00032         std::string getStateName() const override { return "ForceLeave"; }
00033
00034     private:
00035         void renderUI();
00036
00037         constants::ForceLeaveType _leaveType;
00038         std::unique_ptr<ui::UIManager> _uiManager;
00039         std::unique_ptr<MouseInputHandler> _mouseHandler;
00040         std::shared_ptr<ui::Text> _reasonText;
00041         std::shared_ptr<ui::Button> _leaveButton;
00042         std::shared_ptr<ui::UILayout> _bottomRightLayout;
00043 };
00044
00045 }  // namespace gsm
00046
00047 #endif  // FORCELEAVESTATE_HPP_
```

5.28 HowToPlayState.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2026
00003 ** R-Type
00004 ** File description:
00005 ** How to Play State
00006 */
00007
00008 #ifndef HOW_TO_PLAY_HPP_
00009 #define HOW_TO_PLAY_HPP_
00010
00011 #include "../../base/AGameState.hpp"
00012 #include "resourceManager/ResourceManager.hpp"
00013 #include "../../../../input/MouseInputHandler.hpp"
00014 #include "../../../../ui/elements/focusable/Button.hpp"
00015 #include "../../../../ui/elements/Text.hpp"
00016 #include "../../../../ui/manager/UIManager.hpp"
00017 #include "../../../../ui/core/UILayout.hpp"
00018 #include "../../../../ui/elements/Background.hpp"
00019 #include "../../../../ui/elements/Box.hpp"
00020
00021 namespace gsm {
00022
00023 class HowToPlayState : public AGameState {
00024 public:
00025     HowToPlayState(std::shared_ptr<IGameStateMachine> gsm, std::shared_ptr<ResourceManager>
      resourceManager);
00026     ~HowToPlayState() override = default;
00027
00028     void enter() override;
00029     void update(float deltaTime) override;
00030     void exit() override;
00031     std::string getStateName() const override { return "How to Play"; }
00032
00033 private:
00034     void renderUI();
00035
00036 private:
00037     std::unique_ptr<MouseInputHandler> _mouseHandler;
00038     std::unique_ptr<ui::UIManager> _uiManager;
00039
00040     std::shared_ptr<ui::Background> _background;
00041     std::shared_ptr<ui::Text> _titleText;
00042     std::shared_ptr<ui::Button> _backButton;
00043     std::vector<std::shared_ptr<ui::Text» _controlTexts;
00044     std::vector<std::shared_ptr<ui::Text» _objectiveTexts;
00045 };
00046
00047 }  // namespace gsm
00048
00049 #endif /* !HOW_TO_PLAY_HPP_ */
```

5.29 InGameState.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** InGameState
00006 */
00007
00008 #ifndef INGAMESTATE_HPP_
00009 #define INGAMESTATE_HPP_
00010
00011 #include "../../base/AGameState.hpp"
00012 #include "resourceManager/ResourceManager.hpp"
00013 #include "../../../../../common/Prefab/entityPrefabManager/EntityPrefabManager.hpp"
00014 #include "../../../../../common/Parser/Parser.hpp"
00015 #include "../../../../../common/interfaces/IWindow.hpp"
00016 #include <vector>
00017 #include <string>
00018 #include <memory>
00019
00020 #ifdef _WIN32
00021 #include <windows.h>
00022 #include <psapi.h>
00023 #else
00024 #include <unistd.h>
00025 #endif
00026
00027 #ifndef _WIN32
```

```
00028 class SystemConfig {
00029 public:
00030     static long getPageSize() {
00031         return sysconf(_SC_PAGESIZE);
00032     }
00033 };
00034 #endif
00035
00036 namespace gsm {
00037
00038 struct ScoreFeedback {
00039     std::string text;
00040     float lifetime;
00041     float maxLifetime;
00042 };
00043
00044 class InGameState : public AGameState {
00045     public:
00046         InGameState(std::shared_ptr<IGameStateMachine> gsm, std::shared_ptr<ResourceManager>
      resourceManager);
00047         ~InGameState() override = default;
00048
00049         void enter() override;
00050         void update(float deltaTime) override;
00051         void exit() override;
00052         std::string getStateName() const override { return "In Game"; }
00053
00054     private:
00055         void renderHUD(float deltaTime);
00056         void drawHealthHUD(std::shared_ptr<gfx::IWindow> window, float health, float maxHealth);
00057         void drawScoreHUD(std::shared_ptr<gfx::IWindow> window, int score);
00058         void drawShotChargeHUD(std::shared_ptr<gfx::IWindow> window, float shotCharge, float
      maxShotCharge);
00059         void drawInGameMetrics(std::shared_ptr<gfx::IWindow> window, float deltaTime);
00060
00061     private:
00062         std::shared_ptr<ecs::Registry> _registry;
00063         std::shared_ptr<EntityPrefabManager> _prefabManager;
00064         std::shared_ptr<Parser> _parser;
00065         int _previousScore;
00066         int _previousHealth;
00067         std::vector<ScoreFeedback> _scoreFeedbacks;
00068         std::vector<ScoreFeedback> _healthFeedbacks;
00069         float _whoAmITimer = 0.0f;
00070         bool _localPlayerFound = false;
00071 };
00072
00073 }  // namespace gsm
00074
00075 #endif  // INGAMESTATE_HPP_
```

5.30 InGameState.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** InGameState
00006 */
00007
00008 #ifndef SERVER_INGAMESTATE_HPP_
00009 #define SERVER_INGAMESTATE_HPP_
00010
00011 #include "../../AGameState.hpp"
00012 #include "resourceManager/ResourceManager.hpp"
00013 #include "systems/base/ISystem.hpp"
00014
00015 namespace gsm {
00016
00017 class InGameState : public AGameState {
00018 public:
00019     InGameState(std::shared_ptr<IGameStateMachine> gsm, std::shared_ptr<ResourceManager>
      resourceManager);
00020     ~InGameState() override = default;
00021
00022     void enter() override;
00023     void exit() override;
00024
00025     void update(float deltaTime) override;
00026     std::string getStateName() const override { return "Classic mode"; }
00027     private:
00028
00029 };
```

```
00030
00031 }  // namespace gsm
00032
00033 #endif  // SERVER_INGAMESTATE_HPP_
```

5.31 LeaderboardState.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2026
00003 ** R-Type
00004 ** File description:
00005 ** Leaderboard State
00006 */
00007
00008 #ifndef LEADERBOARDSTATE_HPP_
00009 #define LEADERBOARDSTATE_HPP_
00010
00011 #include "../../base/AGameState.hpp"
00012 #include "../../../../../common/resourceManager/ResourceManager.hpp"
00013 #include "../../../../input/MouseInputHandler.hpp"
00014 #include "../../../../ui/manager/UIManager.hpp"
00015 #include "../../../../ui/elements/Background.hpp"
00016 #include "../../../../ui/elements/Text.hpp"
00017 #include "../../../../ui/elements/focusable/Button.hpp"
00018 #include "../../../../ui/core/UILayout.hpp"
00019
00020 namespace gsm {
00021
00022 class LeaderboardState : public AGameState {
00023 private:
00024     std::unique_ptr<MouseInputHandler> _mouseHandler;
00025     std::unique_ptr<ui::UIManager> _uiManager;
00026     std::shared_ptr<ui::Background> _background;
00027     std::shared_ptr<ui::UILayout> _mainLayout;
00028     std::shared_ptr<ui::Text> _titleText;
00029     std::vector<std::shared_ptr<ui::Text» _leaderTexts;
00030     std::shared_ptr<ui::Button> _backButton;
00031
00032 public:
00033     LeaderboardState(
00034         std::shared_ptr<IGameStateMachine> gsm,
00035         std::shared_ptr<ResourceManager> resourceManager
00036     );
00037
00038     void enter() override;
00039     void update(float deltaTime) override;
00040     void exit() override;
00041     std::string getStateName() const override { return "Leaderboard"; }
00042
00043 private:
00044     void loadLeaderboardData();
00045     void renderUI();
00046 };
00047
00048 }  // namespace gsm
00049
00050 #endif /* !LEADERBOARDSTATE_HPP_ */
```

5.32 LevelCompleteState.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** LevelCompleteState
00006 */
00007
00008 #ifndef CLIENT_LEVELCOMPLETESTATE_HPP_
00009 #define CLIENT_LEVELCOMPLETESTATE_HPP_
00010
00011 #include "../../base/AGameState.hpp"
00012 #include "resourceManager/ResourceManager.hpp"
00013 #include "../../../../../common/interfaces/IWindow.hpp"
00014 #include "ui/manager/UIManager.hpp"
00015 #include "ui/elements/Text.hpp"
00016 #include "../../../../colors.hpp"
00017 #include "../../../../ui/elements/SpritePreview.hpp"
00018
```

```
00019 namespace gsm {
00020
00021 class LevelCompleteState : public AGameState {
00022     public:
00023         LevelCompleteState(std::shared_ptr<IGameStateMachine> gsm,
00024             std::shared_ptr<ResourceManager> resourceManager);
00025         ~LevelCompleteState() override = default;
00026
00027         void enter() override;
00028         void update(float deltaTime) override;
00029         void exit() override;
00030         std::string getStateName() const override { return "Level Complete"; }
00031
00032         void onNextLevel();
00033
00034     private:
00035         std::unique_ptr<ui::UIManager> _uiManager;
00036         std::shared_ptr<ui::Text> _subtitleText;
00037         bool _waitingForNextLevel;
00038         std::shared_ptr<ui::SpritePreview> _victoryAnimation;
00039
00040 };
00041
00042 }  // namespace gsm
00043
00044 #endif  // CLIENT_LEVELCOMPLETESTATE_HPP_
```

5.33 LevelCompleteState.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** LevelCompleteState
00006 */
00007
00008 #ifndef SERVER_LEVELCOMPLETESTATE_HPP_
00009 #define SERVER_LEVELCOMPLETESTATE_HPP_
00010
00011 #include "../../AGameState.hpp"
00012 #include "resourceManager/ResourceManager.hpp"
00013 #include <vector>
00014
00015 namespace gsm {
00016
00017 class LevelCompleteState : public AGameState {
00018 public:
00019     LevelCompleteState(std::shared_ptr<IGameStateMachine> gsm,
00020         std::shared_ptr<ResourceManager> resourceManager);
00021     ~LevelCompleteState() override = default;
00022
00023     void enter() override;
00024     void update(float deltaTime) override;
00025     std::string getStateName() const override { return "Level Complete"; }
00026
00027 private:
00028     float _transitionTimer;
00029     std::vector<int> _savedPlayerScores;
00030     static constexpr float TRANSITION_DELAY = 2.0f;
00031 };
00032
00033 }  // namespace gsm
00034
00035 #endif  // SERVER_LEVELCOMPLETESTATE_HPP_
```

5.34 LevelEditorState.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** LevelEditorState
00006 */
00007
00008 #ifndef LEVELEDITORSTATE_HPP_
00009 #define LEVELEDITORSTATE_HPP_
00010
00011 #include <filesystem>  // NOLINT(build/c++17)
```

```
00012 #include <optional>
00013 #include <map>
00014 #include <nlohmann/json.hpp>
00015 #include "../../base/AGameState.hpp"
00016 #include "../../../../constants.hpp"
00017 #include "../../../../../common/gsm/IGameStateMachine.hpp"
00018 #include "../../../../input/MouseInputHandler.hpp"
00019 #include "../../../../ui/elements/Background.hpp"
00020 #include "../../../../ui/elements/Panel.hpp"
00021 #include "../../../../ui/elements/Text.hpp"
00022 #include "../../../../../libs/Multimedia/EventTypes.hpp"
00023 #include "../../../../ui/elements/focusable/Button.hpp"
00024 #include "../../../../ui/elements/focusable/TextInput.hpp"
00025 #include "../../../../ui/elements/focusable/Dropdown.hpp"
00026 #include "../../../../ui/manager/UIManager.hpp"
00027 #include "../../../../ui/core/UILayout.hpp"
00028 #include "../../../../ui/elements/SpritePreview.hpp"
00029 #include "../../../../SettingsConfig.hpp"
00030
00031 namespace gsm {
00032
00033 struct HorizontalLineObstacle {
00034     float fromX;
00035     float posY;
00036     int count;
00037 };
00038
00039 struct VerticalLineObstacle {
00040     float fromY;
00041     float posX;
00042     int count;
00043 };
00044
00045 struct UniqueObstacle {
00046     float posX;
00047     float posY;
00048 };
00049
00050 struct ObstacleSelection {
00051     std::string prefabName;
00052     std::string type;
00053     int index;
00054 };
00055
00056 struct PowerUpData {
00057     float posX;
00058     float posY;
00059 };
00060
00061 struct PowerUpSelection {
00062     std::string prefabName;
00063     int index;
00064 };
00065
00066 struct WaveDistribution {
00067     float min;
00068     float max;
00069     std::string type;
00070 };
00071
00072 struct WaveEnemy {
00073     std::string type;
00074     WaveDistribution distributionX;
00075     WaveDistribution distributionY;
00076     int count;
00077 };
00078
00079 struct Wave {
00080     float gameXTrigger;
00081     std::vector<WaveEnemy> enemies;
00082 };
00083
00084 struct WaveSelection {
00085     int waveIndex;
00086     int enemyIndex;
00087 };
00088
00089 struct ObstacleGroup {
00090     std::vector<HorizontalLineObstacle> horizontalLines;
00091     std::vector<VerticalLineObstacle> verticalLines;
00092     std::vector<UniqueObstacle> uniques;
00093 };
00094
00095 struct LevelPreviewSprite {
00096     std::string texturePath;
00097     float width;
00098     float height;
```

```
00099     float posX;
00100     float posY;
00101     float scale;
00102     float rotation;
00103
00104     bool isAnimation = false;
00105     float frameCount = 0.0f;
00106     float frameWidth = 0.0f;
00107     float frameHeight = 0.0f;
00108     float animationSpeed = 0.1f;
00109     bool animationLoop = true;
00110 };
00111
00112 class LevelEditorState : public AGameState {
00113 public:
00114     LevelEditorState(std::shared_ptr<IGameStateMachine> gsm, std::shared_ptr<ResourceManager>
      resourceManager, std::optional<std::filesystem::path> levelPath = std::nullopt);
00115     ~LevelEditorState() override = default;
00116
00117     void enter() override;
00118     void update(float deltaTime) override;
00119     void exit() override;
00120     std::string getStateName() const override { return "Level Editor"; }
00121
00122 private:
00123     void renderUI();
00124     void createUI();
00125     void createBottomPanel();
00126     void createBottomPanelBase();
00127     void createEditorModeDropdown();
00128     void createObstacleUI();
00129     void createPowerUpUI();
00130     void createWaveUI();
00131     void createEnemyUI();
00132     void updateSaveButtonText();
00133     bool validateFields();
00134     std::vector<std::string> loadAvailableMusics();
00135     std::vector<std::string> loadAvailableBackgrounds();
00136     std::vector<std::string> loadAvailableObstacles();
00137     std::vector<std::string> loadAvailablePowerUps();
00138     std::vector<std::string> loadAvailableEnemies();
00139     void saveToHistory();
00140     void loadFromHistory(size_t index);
00141     void updateHistoryButtons();
00142     void initializeViewport();
00143     void handleZoom(float deltaTime, gfx::EventType eventResult);
00144     void handleCanvasDrag(float deltaTime);
00145     void renderLevelPreview();
00146     void renderSpriteInLevelPreview(const LevelPreviewSprite& spriteData, const std::string&
      prefabName, float screenX, float screenY, float canvasLeft, float canvasRight, float canvasTop, float
      canvasBottom);
00147     LevelPreviewSprite extractSpriteDataFromPrefab(const std::string& prefabPath);
00148
00149     void hideAllUIElements();
00150     void hideObstacleUI();
00151     void hidePowerUpUI();
00152     void hideWaveUI();
00153     void hideEnemyUI();
00154     void showObstacleUI(bool showCount);
00155     void showPowerUpUI();
00156     void showWaveUI();
00157     struct WorldCoordinates {
00158         float worldX;
00159         float worldY;
00160         float levelX;
00161         float levelY;
00162     };
00163     WorldCoordinates extractWorldCoordinates(const math::Vector2f& mousePos, float sidePanelWidth)
      const;
00164
00165     /* Obstacles methods */
00166     void parseObstacles();
00167     void renderAllObstacles(float levelX, float levelY, float canvasLeft, float canvasRight, float
      canvasTop, float canvasBottom);
00168     void saveObstacles();
00169     void handleObstacleClick(float mouseX, float mouseY, float levelX, float levelY);
00170     void startObstacleDrag(math::Vector2f mousePos, float viewportZoom, float sidePanelWidth);
00171     void handleObstacleDrag(math::Vector2f mousePos, float viewportZoom, float sidePanelWidth);
00172     std::optional<ObstacleSelection> getObstacleAtPosition(float mouseX, float mouseY, float levelX,
      float levelY);
00173
00174     /* PowerUps methods */
00175     void parsePowerUps();
00176     void renderAllPowerUps(float levelX, float levelY, float canvasLeft, float canvasRight, float
      canvasTop, float canvasBottom);
00177     void savePowerUps();
00178     void handlePowerUpClick(float mouseX, float mouseY, float levelX, float levelY);
```

```
00179     void startPowerUpDrag(math::Vector2f mousePos, float viewportZoom, float sidePanelWidth);
00180     void handlePowerUpDrag(math::Vector2f mousePos, float viewportZoom, float sidePanelWidth);
00181     std::optional<PowerUpSelection> getPowerUpAtPosition(float mouseX, float mouseY, float levelX,
      float levelY);
00182
00183     /* Waves methods */
00184     void parseWaves();
00185     void renderAllWaves(float levelX, float levelY, float canvasLeft, float canvasRight, float
      canvasTop, float canvasBottom);
00186     void saveWaves();
00187     void handleWaveClick(float mouseX, float mouseY, float levelX, float levelY);
00188     void startWaveDrag(math::Vector2f mousePos, float viewportZoom, float sidePanelWidth);
00189     void handleWaveDrag(math::Vector2f mousePos, float viewportZoom, float sidePanelWidth);
00190     std::optional<WaveSelection> getWaveAtPosition(float mouseX, float mouseY, float levelX, float
      levelY);
00191     void updateEnemyUI();
00192
00193     std::unique_ptr<MouseInputHandler> _mouseHandler;
00194     std::unique_ptr<ui::UIManager> _uiManager;
00195
00196     std::shared_ptr<ui::Background> _background;
00197     std::shared_ptr<ui::Panel> _sidePanel;
00198     std::shared_ptr<ui::Panel> _bottomPanel;
00199     std::shared_ptr<ui::Panel> _canvasPanel;
00200     std::shared_ptr<ui::Dropdown> _editorModeDropdown;
00201     std::shared_ptr<ui::Text> _obstaclePrefabLabel;
00202     std::shared_ptr<ui::Dropdown> _obstaclePrefabDropdown;
00203     std::shared_ptr<ui::Text> _powerUpPrefabLabel;
00204     std::shared_ptr<ui::Dropdown> _powerUpPrefabDropdown;
00205     std::shared_ptr<ui::Panel> _spritePreviewPanel;
00206     std::shared_ptr<ui::SpritePreview> _spritePreview;
00207     std::shared_ptr<ui::Text> _spriteWidthLabel;
00208     std::shared_ptr<ui::Text> _spriteHeightLabel;
00209     std::shared_ptr<ui::Button> _saveButton;
00210     std::shared_ptr<ui::Button> _backButton;
00211     std::shared_ptr<ui::Text> _nameLabel;
00212     std::shared_ptr<ui::TextInput> _levelNameInput;
00213     std::shared_ptr<ui::Text> _mapLengthLabel;
00214     std::shared_ptr<ui::TextInput> _mapLengthInput;
00215     std::shared_ptr<ui::Text> _scrollSpeedLabel;
00216     std::shared_ptr<ui::TextInput> _scrollSpeedInput;
00217     std::shared_ptr<ui::Text> _musicLabel;
00218     std::shared_ptr<ui::Dropdown> _musicDropdown;
00219     std::shared_ptr<ui::Text> _backgroundLabel;
00220     std::shared_ptr<ui::Dropdown> _backgroundDropdown;
00221     std::shared_ptr<ui::Button> _undoButton;
00222     std::shared_ptr<ui::Button> _redoButton;
00223     std::shared_ptr<ui::Text> _cursorPosLabel;
00224     std::shared_ptr<ui::Text> _cursorPosYLabel;
00225     std::shared_ptr<ui::Button> _resetViewButton;
00226     std::shared_ptr<ui::Button> _filterButton;
00227     std::shared_ptr<ui::Button> _showHitboxesButton;
00228     std::shared_ptr<ui::Text> _obstacleTypeLabel;
00229     std::shared_ptr<ui::Dropdown> _obstacleTypeDropdown;
00230     std::shared_ptr<ui::Text> _obstaclePosXLabel;
00231     std::shared_ptr<ui::TextInput> _obstaclePosXInput;
00232     std::shared_ptr<ui::Text> _obstaclePosYLabel;
00233     std::shared_ptr<ui::TextInput> _obstaclePosYInput;
00234     std::shared_ptr<ui::Text> _obstacleCountLabel;
00235     std::shared_ptr<ui::TextInput> _obstacleCountInput;
00236     std::shared_ptr<ui::Button> _obstacleDeleteButton;
00237     std::shared_ptr<ui::Button> _obstacleDuplicateButton;
00238     std::shared_ptr<ui::Text> _powerUpPosXLabel;
00239     std::shared_ptr<ui::TextInput> _powerUpPosXInput;
00240     std::shared_ptr<ui::Text> _powerUpPosYLabel;
00241     std::shared_ptr<ui::TextInput> _powerUpPosYInput;
00242     std::shared_ptr<ui::Button> _powerUpDeleteButton;
00243     std::shared_ptr<ui::Button> _powerUpDuplicateButton;
00244     std::shared_ptr<ui::Text> _waveLabel;
00245     std::shared_ptr<ui::Text> _waveIndexLabel;
00246     std::shared_ptr<ui::Button> _wavePrevButton;
00247     std::shared_ptr<ui::Button> _waveNextButton;
00248     std::shared_ptr<ui::Button> _waveDeleteButton;
00249     std::shared_ptr<ui::Button> _waveDuplicateButton;
00250     std::shared_ptr<ui::Text> _waveTriggerXLabel;
00251     std::shared_ptr<ui::TextInput> _waveTriggerXInput;
00252
00253     std::shared_ptr<ui::Text> _enemyLabel;
00254     std::shared_ptr<ui::Text> _enemyIndexLabel;
00255     std::shared_ptr<ui::Button> _enemyPrevButton;
00256     std::shared_ptr<ui::Button> _enemyNextButton;
00257     std::shared_ptr<ui::Button> _enemyAddButton;
00258     std::shared_ptr<ui::Button> _enemyDeleteButton;
00259     std::shared_ptr<ui::Text> _enemyTypeLabel;
00260     std::shared_ptr<ui::TextInput> _enemyTypeInput;
00261     std::shared_ptr<ui::Button> _enemyApplyTypeButton;
00262     std::shared_ptr<ui::Text> _enemyAppliedTypeLabel;
```

```
00263     std::shared_ptr<ui::Text> _enemyCountLabel;
00264     std::shared_ptr<ui::TextInput> _enemyCountInput;
00265     std::shared_ptr<ui::Text> _enemyDistXMinLabel;
00266     std::shared_ptr<ui::TextInput> _enemyDistXMinInput;
00267     std::shared_ptr<ui::Text> _enemyDistXMaxLabel;
00268     std::shared_ptr<ui::TextInput> _enemyDistXMaxInput;
00269     std::shared_ptr<ui::Text> _enemyDistYMinLabel;
00270     std::shared_ptr<ui::TextInput> _enemyDistYMinInput;
00271     std::shared_ptr<ui::Text> _enemyDistYMaxLabel;
00272     std::shared_ptr<ui::TextInput> _enemyDistYMaxInput;
00273     std::shared_ptr<ui::Text> _enemyDistXTypeLabel;
00274     std::shared_ptr<ui::Dropdown> _enemyDistXTypeDropdown;
00275     std::shared_ptr<ui::Text> _enemyDistYTypeLabel;
00276     std::shared_ptr<ui::Dropdown> _enemyDistYTypeDropdown;
00277
00278     bool _hasUnsavedChanges = false;
00279     bool _showHitboxes = false;
00280     std::string _displayFilter = "All"; // "All", "Obstacles", "PowerUps", "Waves"
00281
00282     std::optional<std::filesystem::path> _levelPath;
00283     nlohmann::json _levelData;
00284
00285     std::vector<nlohmann::json> _history;
00286     size_t _currentHistoryIndex = 0;
00287     float _lastChangeTime = constants::CHANGE_DEBOUNCE_TIME + 1.0f;
00288     float _currentDebounceTime = constants::CHANGE_DEBOUNCE_TIME;
00289     bool _hasPendingChange = false;
00290     bool _isLoadingFromHistory = false;
00291     bool _isSelectingObject = false;
00292     bool _undoPressedLastFrame = false;
00293     bool _redoPressedLastFrame = false;
00294     bool _leftMousePressedLastFrame = false;
00295
00296     math::Vector2f _viewportOffset;
00297     float _viewportZoom = 1.0f;
00298     float _minZoom = 0.1f;
00299     float _maxZoom = 2.0f;
00300
00301     bool _isDragging = false;
00302     math::Vector2f _lastMousePos;
00303     math::Vector2f _dragStartPos;
00304     bool _isDraggingObstacle = false;
00305     math::Vector2f _dragObstacleOffset;
00306     bool _isDraggingPowerUp = false;
00307     math::Vector2f _dragPowerUpOffset;
00308     bool _isDraggingWave = false;
00309     float _dragWaveOffsetX = 0.0f;
00310
00311     std::map<std::string, ObstacleGroup> _obstaclesByName;
00312     std::optional<ObstacleSelection> _selectedObstacle;
00313     std::map<std::string, std::vector<PowerUpData» _powerUpsByName;
00314     std::optional<PowerUpSelection> _selectedPowerUp;
00315     std::vector<Wave> _waves;
00316     std::optional<WaveSelection> _selectedWave;
00317     int _currentWaveIndex = 0;
00318     int _currentEnemyIndex = 0;
00319     std::vector<std::string> _availableEnemies;
00320
00321     std::map<std::string, LevelPreviewSprite> _obstacleAnimationData;
00322     std::map<std::string, float> _obstacleAnimationFrames;
00323     std::map<std::string, float> _obstacleAnimationTimes;
00324     std::map<std::string, LevelPreviewSprite> _powerUpAnimationData;
00325     std::map<std::string, float> _powerUpAnimationFrames;
00326     std::map<std::string, float> _powerUpAnimationTimes;
00327
00328     std::optional<ObstacleSelection> _clipboardObstacle;
00329     std::optional<PowerUpSelection> _clipboardPowerUp;
00330     std::optional<WaveSelection> _clipboardWave;
00331     bool _copyPressedLastFrame = false;
00332     bool _pastePressedLastFrame = false;
00333
00334 };
00335
00336 }  // namespace gsm
00337
00338
00339 #endif /* !LEVELEDITORSTATE_HPP_ */
```

5.35 LevelEditorSelectorState.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
```

```
00004 ** File description:
00005 ** LevelEditorSelectorState
00006 */
00007
00008 #ifndef LEVELEDITORSELECTORSTATE_HPP_
00009 #define LEVELEDITORSELECTORSTATE_HPP_
00010
00011 #include "../../base/AGameState.hpp"
00012 #include <filesystem>  // NOLINT(build/c++17)
00013 #include <nlohmann/json.hpp>
00014 #include "../../../../constants.hpp"
00015 #include "../../../../../common/gsm/IGameStateMachine.hpp"
00016 #include "../../../../input/MouseInputHandler.hpp"
00017 #include "../../../../ui/elements/focusable/Button.hpp"
00018 #include "../../../../ui/elements/Text.hpp"
00019 #include "../../../../ui/elements/Background.hpp"
00020 #include "../../../../ui/manager/UIManager.hpp"
00021 #include "../../../../ui/core/UILayout.hpp"
00022 #include "../../../../SettingsConfig.hpp"
00023 #include "../LevelEditor/LevelEditorState.hpp"
00024
00025 namespace gsm {
00026
00027 class LevelEditorSelectorState : public AGameState {
00028 public:
00029     LevelEditorSelectorState(std::shared_ptr<IGameStateMachine> gsm, std::shared_ptr<ResourceManager>
      resourceManager);
00030     ~LevelEditorSelectorState() override = default;
00031
00032     void enter() override;
00033     void update(float deltaTime) override;
00034     void exit() override;
00035     std::string getStateName() const override { return "Level Editor Selector"; }
00036
00037 private:
00038     void renderUI();
00039     void createLevelSelectionUI();
00040     std::vector<std::pair<std::filesystem::path, int» getAvailableLevels();
00041     std::optional<std::filesystem::path> createNewLevel();
00042     void swapLevels(const std::filesystem::path& path1, const std::filesystem::path& path2);
00043     void showDeleteConfirmation(const std::filesystem::path& levelPath, const std::string& levelName);
00044     void showDeleteConfirmationPopup(const std::filesystem::path& levelPath, const std::string&
      levelName);
00045     void hideDeleteConfirmationPopup();
00046     void confirmDelete();
00047     void showDuplicateConfirmation(const std::filesystem::path& levelPath, const std::string&
      levelName);
00048     void showDuplicateConfirmationPopup(const std::filesystem::path& levelPath, const std::string&
      levelName);
00049     void hideDuplicateConfirmationPopup();
00050     void confirmDuplicate();
00051     void setMainButtonsEnabled(bool enabled);
00052
00053     std::unique_ptr<MouseInputHandler> _mouseHandler;
00054     std::unique_ptr<ui::UIManager> _uiManager;
00055
00056     std::shared_ptr<ui::Background> _background;
00057     std::shared_ptr<ui::Button> _backButton;
00058     std::vector<std::shared_ptr<ui::Button» _levelButtons;
00059     std::vector<std::shared_ptr<ui::Text» _indexLabels;
00060     std::shared_ptr<ui::Button> _addLevelButton;
00061     std::vector<std::shared_ptr<ui::Button» _upButtons;
00062     std::vector<std::shared_ptr<ui::Button» _downButtons;
00063     std::vector<std::shared_ptr<ui::Button» _duplicateButtons;
00064     std::vector<std::shared_ptr<ui::Button» _deleteButtons;
00065     std::shared_ptr<ui::Button> _prevButton;
00066     std::shared_ptr<ui::Button> _nextButton;
00067     bool _shouldUpdateUI;
00068     size_t _lastLevelCount;
00069     int _currentPage;
00070     static constexpr int _levelsPerPage = 8;
00071
00072     std::shared_ptr<ui::UILayout> _deletePopupOverlay;
00073     std::shared_ptr<ui::UILayout> _deletePopupLayout;
00074     std::shared_ptr<ui::Text> _deletePopupText;
00075     std::shared_ptr<ui::Button> _deleteCancelButton;
00076     std::shared_ptr<ui::Button> _deleteConfirmButton;
00077     std::filesystem::path _pendingDeletePath;
00078     bool _shouldHideDeletePopup;
00079
00080     std::shared_ptr<ui::UILayout> _duplicatePopupOverlay;
00081     std::shared_ptr<ui::UILayout> _duplicatePopupLayout;
00082     std::shared_ptr<ui::Text> _duplicatePopupText;
00083     std::shared_ptr<ui::Button> _duplicateCancelButton;
00084     std::shared_ptr<ui::Button> _duplicateConfirmButton;
00085     std::filesystem::path _pendingDuplicatePath;
00086     std::string _pendingDuplicateName;
```

```
00087     bool _shouldHideDuplicatePopup;
00088 };
00089
00090 }  // namespace gsm
00091
00092 #endif /* !LEVELEDITORSELECTORSTATE_HPP_ */
```

5.36 LobbyWaitingState.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** LobbyWaitingState
00006 */
00007
00008 #ifndef LOBBYWAITINGSTATE_HPP_
00009 #define LOBBYWAITINGSTATE_HPP_
00010
00011 #include "../../base/AGameState.hpp"
00012 #include "resourceManager/ResourceManager.hpp"
00013 #include "../../../../input/MouseInputHandler.hpp"
00014 #include "../../../../ui/elements/focusable/Button.hpp"
00015 #include "../../../../ui/manager/UIManager.hpp"
00016 #include "../../../../ui/core/UILayout.hpp"
00017 #include "../../../../ui/elements/Text.hpp"
00018 #include "../../../../ui/elements/SpritePreview.hpp"
00019
00020 namespace gsm {
00021
00022 class LobbyWaitingState : public AGameState {
00023 public:
00024     LobbyWaitingState(std::shared_ptr<IGameStateMachine> gsm, std::shared_ptr<ResourceManager>
      resourceManager, bool isLobbyMaster);
00025     ~LobbyWaitingState() override = default;
00026
00027     void enter() override;
00028     void update(float deltaTime) override;
00029     void exit() override;
00030     std::string getStateName() const override { return "Waiting to start"; }
00031
00032 private:
00033     void renderUI();
00034     void updateUIStatus();
00035     void setupLobbyMasterUI();
00036     void setupPlayerUI();
00037
00038 private:
00039     std::unique_ptr<MouseInputHandler> _mouseHandler;
00040     std::unique_ptr<ui::UIManager> _uiManager;
00041     std::shared_ptr<ui::UILayout> _centerLayout;
00042     std::shared_ptr<ui::Text> _lobbyCodeText;
00043     std::shared_ptr<ui::Text> _statusText;
00044     std::shared_ptr<ui::Button> _startGameButton;
00045
00046     std::shared_ptr<ui::UILayout> _topLeftLayout;
00047     std::shared_ptr<ui::Text> _gamemodeLabel;
00048     std::shared_ptr<ui::Button> _gamemodeButton;
00049     std::shared_ptr<ui::Text> _difficultyLabel;
00050     std::shared_ptr<ui::Button> _difficultyButton;
00051     std::shared_ptr<ui::Text> _crossfireLabel;
00052     std::shared_ptr<ui::Button> _crossfireButton;
00053     std::shared_ptr<ui::Button> _leaveButton;
00054
00055     std::shared_ptr<ui::UILayout> _topRightLayout;
00056     std::shared_ptr<ui::Button> _chatButton;
00057     std::shared_ptr<ui::SpritePreview> _loadingAnimation;
00058     std::shared_ptr<ui::UILayout> _loadingLayout;
00059
00060     std::shared_ptr<ui::UILayout> _bottomLeftLayout;
00061     std::shared_ptr<ui::Button> _copyCodeButton;
00062
00063     bool _isLobbyMaster;
00064 };
00065
00066 }  // namespace gsm
00067
00068 #endif  // LOBBYWAITINGSTATE_HPP_
```

5.37 LoginState.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2026
00003 ** R-Type
00004 ** File description:
00005 ** Header
00006 */
00007
00008 #ifndef LOGIN_STATE_HPP_
00009 #define LOGIN_STATE_HPP_
00010
00011 #include "../../base/AGameState.hpp"
00012 #include "resourceManager/ResourceManager.hpp"
00013 #include "../../../../input/MouseInputHandler.hpp"
00014 #include "../../../../ui/elements/focusable/Button.hpp"
00015 #include "../../../../ui/elements/focusable/TextInput.hpp"
00016 #include "../../../../ui/manager/UIManager.hpp"
00017 #include "../../../../ui/core/UILayout.hpp"
00018 #include "../../../../ui/elements/Background.hpp"
00019
00020 namespace gsm {
00021
00022 class LoginState : public AGameState {
00023 public:
00024     LoginState(std::shared_ptr<IGameStateMachine> gsm, std::shared_ptr<ResourceManager>
      resourceManager);
00025     ~LoginState() override = default;
00026
00027     void enter() override;
00028     void update(float deltaTime) override;
00029     void exit() override;
00030     std::string getStateName() const override { return "Login"; }
00031
00032 private:
00033     void renderUI();
00034
00035 private:
00036     std::unique_ptr<MouseInputHandler> _mouseHandler;
00037     std::unique_ptr<ui::UIManager> _uiManager;
00038
00039     std::shared_ptr<ui::Background> _background;
00040     std::shared_ptr<ui::UILayout> _mainLayout;
00041
00042     std::shared_ptr<ui::TextInput> _usernameInput;
00043     std::shared_ptr<ui::TextInput> _passwordInput;
00044
00045     std::shared_ptr<ui::Button> _loginButton;
00046     std::shared_ptr<ui::Button> _backButton;
00047 };
00048
00049 }  // namespace gsm
00050
00051 #endif /* !LOGIN_STATE_HPP_ */
```

5.38 MainMenuState.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** MainMenuState
00006 */
00007
00008 #ifndef MAINMENUSTATE_HPP_
00009 #define MAINMENUSTATE_HPP_
00010
00011 #include "../../base/AGameState.hpp"
00012 #include "resourceManager/ResourceManager.hpp"
00013 #include "../../../../input/MouseInputHandler.hpp"
00014 #include "../../../../ui/elements/focusable/Button.hpp"
00015 #include "../../../../ui/manager/UIManager.hpp"
00016 #include "../../../../ui/core/UILayout.hpp"
00017 #include "../../../../ui/elements/Background.hpp"
00018 #include "../../../../ui/elements/focusable/TextInput.hpp"
00019 #include "../../../../ui/elements/Text.hpp"
00020 #include "../Leaderboard/LeaderboardState.hpp"
00021
00022
00023 namespace gsm {
00024
00025 class MainMenuState : public AGameState {
```

```
00026 public:
00027     MainMenuState(std::shared_ptr<IGameStateMachine> gsm, std::shared_ptr<ResourceManager>
      resourceManager);
00028     ~MainMenuState() override = default;
00029
00030     void enter() override;
00031     void update(float deltaTime) override;
00032     void exit() override;
00033     std::string getStateName() const override { return "Main Menu"; }
00034
00035 private:
00036     void renderUI();
00037     void updateUIStatus();
00038     void checkLobbyConnectionTransition();
00039
00040 private:
00041     std::unique_ptr<MouseInputHandler> _mouseHandler;
00042     std::shared_ptr<ui::Button> _usernameButton;
00043     std::shared_ptr<ui::Button> _settingsButton;
00044     std::shared_ptr<ui::Button> _quitButton;
00045     std::shared_ptr<ui::Button> _requestCodeButton;
00046     std::shared_ptr<ui::Button> _lobbyConnectButton;
00047     std::unique_ptr<ui::UIManager> _uiManager;
00048     std::shared_ptr<ui::UILayout> _mainMenuLayout;
00049     std::shared_ptr<ui::UILayout> _headerLayout;
00050     std::shared_ptr<ui::UILayout> _topLeftLayout;
00051     std::shared_ptr<ui::UILayout> _rightLayout;
00052     std::shared_ptr<ui::Button> _devButton;
00053     std::shared_ptr<ui::Button> _howToPlayButton;
00054     std::shared_ptr<ui::Button> _leaderboardButton;
00055     std::shared_ptr<ui::Button> _registerButton;
00056     std::shared_ptr<ui::Button> _loginButton;
00057     std::shared_ptr<ui::Button> _disconnectButton;
00058     std::shared_ptr<ui::Button> _chatButton;
00059
00060     std::shared_ptr<ui::TextInput> _lobbyCodeInput;
00061
00062     std::shared_ptr<ui::Background> _background;
00063
00064     bool _previousLobbyConnectedState;
00065     bool _previousLobbyMasterState;
00066 };
00067
00068 }  // namespace gsm
00069
00070 #endif  // MAINMENUSTATE_HPP_
```

5.39 PauseState.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** PauseState
00006 */
00007
00008 #ifndef PAUSESTATE_HPP_
00009 #define PAUSESTATE_HPP_
00010
00011 #include "../../base/AGameState.hpp"
00012 #include "../../../../input/MouseInputHandler.hpp"
00013 #include "../../../../../common/interfaces/IWindow.hpp"
00014 #include "../../../../../../common/interfaces/IEvent.hpp"
00015 #include "../../../../../../common/interfaces/IAudio.hpp"
00016 #include "../../../../../../common/InputMapping/IInputProvider.hpp"
00017
00018 namespace gsm {
00019
00020 class PauseState : public AGameState {
00021 public:
00022     PauseState(std::shared_ptr<IGameStateMachine> gsm, std::shared_ptr<ResourceManager>
      resourceManager);
00023     ~PauseState() override = default;
00024
00025     void enter() override;
00026     void update(float deltaTime) override;
00027     void exit() override;
00028     std::string getStateName() const override { return "Paused"; }
00029
00030 private:
00031     void renderUI();
00032
00033 private:
```

```
00034     std::unique_ptr<MouseInputHandler> _mouseHandler;
00035     std::unique_ptr<ui::UIManager> _uiManager;
00036
00037     std::shared_ptr<ui::UILayout> _menuLayout;
00038     std::shared_ptr<ui::Button> _resumeButton;
00039     std::shared_ptr<ui::Button> _settingsButton;
00040     std::shared_ptr<ui::Button> _leaveButton;
00041 };
00042
00043 }  // namespace gsm
00044
00045 #endif  // PAUSESTATE_HPP_
```

5.40 ProfileState.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2026
00003 ** R-Type
00004 ** File description:
00005 ** Header
00006 */
00007
00008 #ifndef PROFILE_STATE_HPP_
00009 #define PROFILE_STATE_HPP_
00010
00011 #include "../../base/AGameState.hpp"
00012 #include "resourceManager/ResourceManager.hpp"
00013 #include "../../../../input/MouseInputHandler.hpp"
00014 #include "../../../../ui/elements/focusable/Button.hpp"
00015 #include "../../../../ui/elements/Text.hpp"
00016 #include "../../../../ui/manager/UIManager.hpp"
00017 #include "../../../../ui/core/UILayout.hpp"
00018 #include "../../../../ui/elements/Background.hpp"
00019 #include "../../../../ClientNetwork.hpp"
00020
00021 namespace gsm {
00022
00023 class ReplayState;
00024
00025 class ProfileState : public AGameState {
00026 public:
00027     ProfileState(std::shared_ptr<IGameStateMachine> gsm, std::shared_ptr<ResourceManager>
      resourceManager);
00028     ~ProfileState() override = default;
00029
00030     void enter() override;
00031     void update(float deltaTime) override;
00032     void exit() override;
00033     std::string getStateName() const override { return "Profile"; }
00034
00035 private:
00036     void renderUI();
00037     void loadUserData();
00038
00039 private:
00040     std::unique_ptr<MouseInputHandler> _mouseHandler;
00041     std::unique_ptr<ui::UIManager> _uiManager;
00042
00043     std::shared_ptr<ui::Background> _background;
00044     std::shared_ptr<ui::UILayout> _mainLayout;
00045
00046     std::shared_ptr<ui::Text> _titleText;
00047     std::shared_ptr<ui::Text> _usernameText;
00048     std::shared_ptr<ui::Text> _gamesPlayedText;
00049     std::shared_ptr<ui::Text> _winsText;
00050     std::shared_ptr<ui::Text> _highScoreText;
00051
00052     std::shared_ptr<ui::Button> _button1;
00053     std::shared_ptr<ui::Button> _button2;
00054     std::shared_ptr<ui::Button> _backButton;
00055 };
00056
00057 }  // namespace gsm
00058
00059 #endif /* !PROFILE_STATE_HPP_ */
```

5.41 RegisterState.hpp

```
00001 /*
```

```
00002 ** EPITECH PROJECT, 2026
00003 ** R-Type
00004 ** File description:
00005 ** Header
00006 */
00007
00008 #ifndef REGISTER_STATE_HPP_
00009 #define REGISTER_STATE_HPP_
00010
00011 #include "../../base/AGameState.hpp"
00012 #include "resourceManager/ResourceManager.hpp"
00013 #include "../../../../input/MouseInputHandler.hpp"
00014 #include "../../../../ui/elements/focusable/Button.hpp"
00015 #include "../../../../ui/elements/focusable/TextInput.hpp"
00016 #include "../../../../ui/elements/Text.hpp"
00017 #include "../../../../ui/manager/UIManager.hpp"
00018 #include "../../../../ui/core/UILayout.hpp"
00019 #include "../../../../ui/elements/Background.hpp"
00020
00021 namespace gsm {
00022
00023 class RegisterState : public AGameState {
00024 public:
00025     RegisterState(std::shared_ptr<IGameStateMachine> gsm, std::shared_ptr<ResourceManager>
      resourceManager);
00026     ~RegisterState() override = default;
00027
00028     void enter() override;
00029     void update(float deltaTime) override;
00030     void exit() override;
00031     std::string getStateName() const override { return "Register"; }
00032
00033 private:
00034     void renderUI();
00035
00036 private:
00037     std::unique_ptr<MouseInputHandler> _mouseHandler;
00038     std::unique_ptr<ui::UIManager> _uiManager;
00039
00040     std::shared_ptr<ui::Background> _background;
00041     std::shared_ptr<ui::UILayout> _mainLayout;
00042
00043     std::shared_ptr<ui::TextInput> _usernameInput;
00044     std::shared_ptr<ui::TextInput> _passwordInput;
00045     std::shared_ptr<ui::TextInput> _confirmPasswordInput;
00046
00047     std::shared_ptr<ui::Text> _errorMessage;
00048
00049     std::shared_ptr<ui::Button> _registerButton;
00050     std::shared_ptr<ui::Button> _backButton;
00051 };
00052
00053 }  // namespace gsm
00054
00055 #endif /* !REGISTER_STATE_HPP_ */
```

5.42 ReplayState.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ReplayState
00006 */
00007
00008 #ifndef REPLAYSTATE_HPP_
00009 #define REPLAYSTATE_HPP_
00010
00011 #include "../../base/AGameState.hpp"
00012 #include "../../../../../common/gsm/IGameStateMachine.hpp"
00013 #include "../../../../input/MouseInputHandler.hpp"
00014 #include "../../../../ui/elements/focusable/Button.hpp"
00015 #include "../../../../ui/elements/Text.hpp"
00016 #include "../../../../ui/elements/Background.hpp"
00017 #include "../../../../ui/manager/UIManager.hpp"
00018 #include "../../../../ui/core/UILayout.hpp"
00019 #include "../../../../ui/elements/focusable/Slider.hpp"
00020 #include "../../../../../common/types/Vector2f.hpp"
00021 #include "../../../../../common/interfaces/IAudio.hpp"
00022 #include "../../../../components/temporary/MusicIntentComponent.hpp"
00023 #include "../../../../SettingsConfig.hpp"
00024 #include <nlohmann/json.hpp>
00025 #include <filesystem>
```

```
00026
00027 namespace gsm {
00028
00029 class ReplayState : public AGameState {
00030 public:
00031     ReplayState(std::shared_ptr<IGameStateMachine> gsm, std::shared_ptr<ResourceManager>
      resourceManager);
00032     ~ReplayState() override = default;
00033
00034     void enter() override;
00035     void update(float deltaTime) override;
00036     void exit() override;
00037     std::string getStateName() const override { return "Replay"; }
00038
00039 private:
00040     void renderUI();
00041     void renderReplaySprites();
00042     void renderParallaxBackground(const nlohmann::json& parallaxData, std::shared_ptr<gfx::IWindow>
      window);
00043     void renderHealthBar(const nlohmann::json& healthBarData, std::shared_ptr<gfx::IWindow> window);
00044     void renderText(const nlohmann::json& textData, std::shared_ptr<gfx::IWindow> window);
00045     void renderRectangle(const nlohmann::json& rectangleData, std::shared_ptr<gfx::IWindow> window);
00046     void renderHitbox(const nlohmann::json& hitboxData, std::shared_ptr<gfx::IWindow> window);
00047     void updateViewForFrame(const nlohmann::json& frame);
00048     void loadReplay(const std::filesystem::path& replayFile);
00049     void playReplay(float deltaTime);
00050     void processAudioForFrame(const nlohmann::json& frame);
00051     std::vector<std::filesystem::path> getAvailableReplays();
00052     void createReplaySelectionUI();
00053     void createPlaybackControlsUI();
00054     void updatePlaybackControls();
00055
00056     std::unique_ptr<MouseInputHandler> _mouseHandler;
00057     std::unique_ptr<ui::UIManager> _uiManager;
00058     std::unique_ptr<ui::UIManager> _playbackUIManager;
00059
00060     std::shared_ptr<ui::Background> _background;
00061     std::shared_ptr<ui::Button> _backButton;
00062     std::vector<std::shared_ptr<ui::Button» _replayButtons;
00063
00064     std::shared_ptr<ui::Button> _playPauseButton;
00065     std::shared_ptr<ui::Button> _replayBackButton;
00066     std::shared_ptr<ui::Button> _increaseSpeedButton;
00067     std::shared_ptr<ui::Button> _decreaseSpeedButton;
00068     std::shared_ptr<ui::Slider> _progressSlider;
00069     std::shared_ptr<ui::Text> _timeText;
00070     std::shared_ptr<ui::Text> _speedText;
00071     std::shared_ptr<ui::UILayout> _playbackLayout;
00072
00073     std::vector<nlohmann::json> _frames;
00074     float _replayTime;
00075     float _totalReplayTime;
00076     size_t _currentFrameIndex;
00077     bool _isPlaying;
00078     bool _isPaused;
00079     bool _shouldSwitch;
00080     float _spacePressCooldown;
00081     float _playbackSpeed;
00082
00083     float _renderOffsetX;
00084     float _renderOffsetY;
00085 };
00086
00087 }  // namespace gsm
00088
00089 #endif /* !REPLAYSTATE_HPP_ */
```

5.43 ResultsState.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ResultsState
00006 */
00007
00008 #ifndef RESULTSSTATE_HPP_
00009 #define RESULTSSTATE_HPP_
00010
00011 #include "../../base/AGameState.hpp"
00012 #include "resourceManager/ResourceManager.hpp"
00013 #include "../../../../../common/interfaces/IWindow.hpp"
00014 #include "ui/manager/UIManager.hpp"
```

```
00015 #include "ui/elements/Text.hpp"
00016 #include "ui/elements/focusable/Button.hpp"
00017 #include "ui/core/UILayout.hpp"
00018 #include "../../../../input/MouseInputHandler.hpp"
00019 #include "../../../../colors.hpp"
00020 #include "../../../../ui/elements/SpritePreview.hpp"
00021
00022 namespace gsm {
00023
00024 class ResultsState : public AGameState {
00025     public:
00026         ResultsState(std::shared_ptr<IGameStateMachine> gsm, std::shared_ptr<ResourceManager>
      resourceManager, bool isWin);
00027         ~ResultsState() override = default;
00028
00029         void enter() override;
00030         void update(float deltaTime) override;
00031         void exit() override;
00032         std::string getStateName() const override { return "Results"; }
00033
00034     private:
00035         void updateUserStats();
00036         void renderUI();
00037
00038         bool _isWin;
00039         std::unique_ptr<ui::UIManager> _uiManager;
00040         std::unique_ptr<MouseInputHandler> _mouseHandler;
00041         std::shared_ptr<ui::Text> _resultText;
00042         std::shared_ptr<ui::SpritePreview> _victoryAnimation;
00043         std::shared_ptr<ui::SpritePreview> _youDiedAnimation;
00044
00045         std::shared_ptr<ui::Button> _leaveButton;
00046         std::shared_ptr<ui::UILayout> _bottomRightLayout;
00047 };
00048
00049 }  // namespace gsm
00050
00051 #endif  // RESULTSSTATE_HPP_
```

5.44 SettingsState.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** SettingsState
00006 */
00007
00008 #ifndef SETTINGSSTATE_HPP_
00009 #define SETTINGSSTATE_HPP_
00010
00011 #include "../../base/AGameState.hpp"
00012 #include "resourceManager/ResourceManager.hpp"
00013 #include "../../../../input/MouseInputHandler.hpp"
00014 #include "../../../../ui/elements/focusable/Button.hpp"
00015 #include "../../../../ui/elements/focusable/Slider.hpp"
00016 #include "../../../../ui/elements/focusable/ToggleSwitch.hpp"
00017 #include "../../../../ui/elements/Text.hpp"
00018 #include "../../../../ui/elements/Background.hpp"
00019 #include "../../../../ui/manager/UIManager.hpp"
00020 #include "../../../../ui/core/UILayout.hpp"
00021 #include "../../../../../common/types/Vector2f.hpp"
00022 #include "../../../../../../common/InputMapping/InputAction.hpp"
00023 #include "../../../../../libs/Multimedia/EventTypes.hpp"
00024 #include <optional>
00025 #include "../../../../SettingsManager.hpp"
00026
00027 namespace gsm {
00028
00029 class SettingsState : public AGameState {
00030 public:
00031     SettingsState(std::shared_ptr<IGameStateMachine> gsm, std::shared_ptr<ResourceManager>
      resourceManager);
00032     ~SettingsState() override = default;
00033
00034     void enter() override;
00035     void update(float deltaTime) override;
00036     void exit() override;
00037     std::string getStateName() const override { return "Settings"; }
00038
00039 private:
00040     void renderUI();
00041     void cycleColorBlindnessFilter();
```

```
00042     void toggleHighContrastFilter();
00043     void updateBrightnessFilter(float value);
00044     void applyColorBlindnessFilter(int state);
00045     void applyHighContrastFilter(bool enabled);
00046     void cycleUIScale();
00047     void updateMusicVolume(float value);
00048     void updateSoundVolume(float value);
00049     void updateToggleValue(bool value);
00050     void cycleScreenResolution();
00051     void updateTargetFPS(int fps);
00052     void updateRenderQuality(float quality);
00053     void updateInGameMetrics(bool enabled);
00054     void setScreenResolution(SettingsConfig::ScreenResolution resolution);
00055     void updateResolutionButtonColors(SettingsConfig::ScreenResolution current);
00056
00057     void startKeyRebind(ecs::RemappableAction action, bool rebindPrimary, std::shared_ptr<ui::Button>
      button);
00058     void handleKeyRebind(gfx::EventType newKey);
00059     void updateKeyBindingButtonText(std::shared_ptr<ui::Button> button, ecs::RemappableAction action,
      bool isPrimary);
00060     std::string getRemappableActionName(ecs::RemappableAction action) const;
00061
00062     std::string getScreenResolutionText(SettingsConfig::ScreenResolution resolution);
00063
00064 private:
00065     std::unique_ptr<MouseInputHandler> _mouseHandler;
00066     std::shared_ptr<ui::Button> _backButton;
00067     std::shared_ptr<ui::Button> _highContrastButton;
00068     std::shared_ptr<ui::Button> _colorBlindnessButton;
00069     std::shared_ptr<ui::Slider> _brightnessSlider;
00070     std::shared_ptr<ui::Slider> _musicVolumeSlider;
00071     std::shared_ptr<ui::Slider> _soundVolumeSlider;
00072     std::shared_ptr<ui::ToggleSwitch> _toggleSwitch;
00073     std::shared_ptr<ui::Text> _toggleLabel;
00074     std::shared_ptr<ui::UILayout> _toggleLayout;
00075     std::vector<std::shared_ptr<ui::Button>> _resolutionButtons;
00076     std::shared_ptr<ui::Slider> _fpsSlider;
00077     std::shared_ptr<ui::Slider> _renderQualitySlider;
00078     std::shared_ptr<ui::Button> _scaleButton;
00079     std::unique_ptr<ui::UIManager> _uiManager;
00080     std::shared_ptr<ui::UILayout> _settingsLayout;
00081     std::shared_ptr<ui::UILayout> _leftColumnLayout;
00082     std::shared_ptr<ui::UILayout> _rightColumnLayout;
00083     std::shared_ptr<ui::UILayout> _centerColumnLayout;
00084     std::shared_ptr<ui::UILayout> _titleLayout;
00085     std::shared_ptr<ui::Background> _background;
00086     math::Vector2f _savedViewCenter;
00087
00088     std::shared_ptr<SettingsManager> _settingsManager;
00089
00090     std::shared_ptr<ui::UILayout> _moveUpLayout;
00091     std::shared_ptr<ui::Text> _moveUpLabel;
00092     std::shared_ptr<ui::Button> _moveUpPrimaryButton;
00093     std::shared_ptr<ui::Button> _moveUpSecondaryButton;
00094
00095     std::shared_ptr<ui::UILayout> _moveDownLayout;
00096     std::shared_ptr<ui::Text> _moveDownLabel;
00097     std::shared_ptr<ui::Button> _moveDownPrimaryButton;
00098     std::shared_ptr<ui::Button> _moveDownSecondaryButton;
00099
00100     std::shared_ptr<ui::UILayout> _moveLeftLayout;
00101     std::shared_ptr<ui::Text> _moveLeftLabel;
00102     std::shared_ptr<ui::Button> _moveLeftPrimaryButton;
00103     std::shared_ptr<ui::Button> _moveLeftSecondaryButton;
00104
00105     std::shared_ptr<ui::UILayout> _moveRightLayout;
00106     std::shared_ptr<ui::Text> _moveRightLabel;
00107     std::shared_ptr<ui::Button> _moveRightPrimaryButton;
00108     std::shared_ptr<ui::Button> _moveRightSecondaryButton;
00109
00110     std::shared_ptr<ui::UILayout> _shootLayout;
00111     std::shared_ptr<ui::Text> _shootLabel;
00112     std::shared_ptr<ui::Button> _shootPrimaryButton;
00113     std::shared_ptr<ui::Button> _shootSecondaryButton;
00114
00115     std::shared_ptr<ui::UILayout> _forceLayout;
00116     std::shared_ptr<ui::Text> _forceLabel;
00117     std::shared_ptr<ui::Button> _forcePrimaryButton;
00118     std::shared_ptr<ui::Button> _forceSecondaryButton;
00119
00120     bool _isWaitingForKey = false;
00121     std::optional<ecs::RemappableAction> _actionToRebind;
00122     bool _rebindingPrimary = true;
00123     std::string _rebindLabel;
00124     std::shared_ptr<ui::Button> _buttonToUpdate;
00125     gfx::EventType _originalKey = gfx::EventType::NOTHING;
00126
```

```
00127     std::shared_ptr<ui::ToggleSwitch> _inGameMetricsToggle;
00128     std::shared_ptr<ui::Text> _inGameMetricsLabel;
00129
00130     std::string getColorBlindnessText(int state);
00131     std::string getUIScaleText(ui::UIScale scale);
00132 };
00133
00134 }  // namespace gsm
00135
00136 #endif  // SETTINGSSTATE_HPP_
```

5.45 GraphicalInputProvider.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** GraphicalInputProvider
00006 */
00007
00008 #pragma once
00009
00010 #include "../../common/InputMapping/IInputProvider.hpp"
00011 #include "../../common/interfaces/IEvent.hpp"
00012 #include "../../common/InputMapping/InputMappingManager.hpp"
00013 #include <memory>
00014
00015 namespace ecs {
00016
00017 class GraphicalInputProvider : public IInputProvider {
00018     public:
00019         GraphicalInputProvider(std::shared_ptr<gfx::IEvent> eventSystem,
00020             std::shared_ptr<InputMappingManager> mappingManager);
00021         ~GraphicalInputProvider() override = default;
00022
00023         float getAxisValue(event_t axis, size_t clientID = 0) override;
00024
00025         bool isActionPressed(InputAction action, size_t clientID = 0) override;
00026         float getActionAxis(InputAction action, size_t clientID = 0) override;
00027         InputMapping getInputMapping(size_t clientID = 0) const override;
00028
00029         void setToggleMode(bool enabled);
00030         bool isToggleMode() const;
00031
00032     private:
00033         std::shared_ptr<gfx::IEvent> _eventSystem;
00034         std::shared_ptr<InputMappingManager> _mappingManager;
00035         bool _toggleMode;
00036         std::map<InputAction, bool> _toggledStates;
00037         std::map<InputAction, bool> _lastKeyState;
00038         std::map<std::pair<InputAction, gfx::EventType>, bool> _keyPressedState;
00039         std::map<std::pair<InputAction, gfx::EventType>, bool> _toggledKeyStates;
00040         std::map<std::pair<InputAction, gfx::EventType>, int> _lastToggleFrame;
00041         int _currentFrame;
00042 };
00043
00044 } // namespace ecs
```

5.46 initResourcesManager.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** initResourcesManager
00006 */
00007
00008 #ifndef initResourcesManager_HPP_
00009 #define initResourcesManager_HPP_
00010
00011 #include "../../common/resourceManager/ResourceManager.hpp"
00012 #include <memory>
00013 #include "../../common/Parser/Parser.hpp"
00014 #include "../../common/DLLoader/DLLoader.hpp"
00015 #include "../../common/interfaces/IWindow.hpp"
00016 #include "../../common/interfaces/IEvent.hpp"
00017 #include "../../common/interfaces/IAudio.hpp"
00018
```

```
00019 std::shared_ptr<ResourceManager> initResourcesManager(
00020     std::shared_ptr<DLLoader<gfx::createWindow_t>>,
00021     std::shared_ptr<DLLoader<gfx::createEvent_t>>,
00022     std::shared_ptr<DLLoader<gfx::createAudio_t>>,
00023     std::shared_ptr<ClientNetwork>,
00024     std::shared_ptr<Parser> parser
00025 );
00026
00027 #endif /* !initResourcesManager_HPP_ */
```

5.47 initResourcesManager.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** initResourcesManager
00006 */
00007
00008 #ifndef initResourcesManager_HPP_
00009 #define initResourcesManager_HPP_
00010
00011 #include "../../common/resourceManager/ResourceManager.hpp"
00012 #include "../Server.hpp"
00013 #include "../Lobby.hpp"
00014 #include "../../common/ECS/entity/registry/Registry.hpp"
00015 #include "../../common/Parser/Parser.hpp"
00016 #include "../../common/systems/systemManager/ISystemManager.hpp"
00017 #include "../../common/InputMapping/IInputProvider.hpp"
00018 #include "../gsm/machine/GameStateMachine.hpp"
00019 #include <memory>
00020
00021 std::shared_ptr<ResourceManager> initResourcesManager(
00022     std::shared_ptr<rserv::Server> server,
00023     std::shared_ptr<rserv::Lobby> lobby
00024 );
00025
00026 #endif /* !initResourcesManager_HPP_ */
```

5.48 MouseInputHandler.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** MouseInputHandler
00006 */
00007
00008 #ifndef MOUSEINPUTHANDLER_HPP_
00009 #define MOUSEINPUTHANDLER_HPP_
00010
00011 #include <memory>
00012 #include <optional>
00013 #include "../../common/resourceManager/ResourceManager.hpp"
00014 #include "../../common/types/Vector2f.hpp"
00015 #include "../constants.hpp"
00016
00017 struct MouseClickInfo {
00018     math::Vector2f position;
00019     constants::MouseButton button;
00020 };
00021
00022 class MouseInputHandler {
00023     public:
00024         MouseInputHandler(std::shared_ptr<ResourceManager> resourceManager);
00025         ~MouseInputHandler() = default;
00026
00027         std::optional<MouseClickInfo> pollMouseClick();
00028         math::Vector2f getMousePosition() const;
00029         math::Vector2f getWorldMousePosition() const;
00030         math::Vector2f getNormalizedMousePosition() const;
00031         bool isMouseButtonPressed(int button) const;
00032
00033     private:
00034         std::weak_ptr<ResourceManager> _resourceManager;
00035 };
00036
00037 #endif /* !MOUSEINPUTHANDLER_HPP_ */
```

5.49 DefaultPacketHandlers.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** Default packet handlers
00006 */
00007
00008 #ifndef CLIENT_DEFAULT_PACKET_HANDLERS_HPP_
00009 #define CLIENT_DEFAULT_PACKET_HANDLERS_HPP_
00010
00011 #include <memory>
00012 #include "../../common/interfaces/IPacketManager.hpp"
00013
00014 namespace rcli::packet {
00015     bool registerDefaultPacketHandlers(std::shared_ptr<pm::IPacketManager> packet);
00016 }
00017
00018 #endif // CLIENT_DEFAULT_PACKET_HANDLERS_HPP_
```

5.50 DefaultPacketHandlers.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** Default packet handlers
00006 */
00007
00008 #ifndef COMMON_DEFAULT_PACKET_HANDLERS_HPP_
00009 #define COMMON_DEFAULT_PACKET_HANDLERS_HPP_
00010
00011 #include <memory>
00012 #include "../interfaces/IPacketManager.hpp"
00013 #include "../../libs/Packet/serializer/BigEndianSerialization.hpp"
00014
00015 namespace common::packet {
00016
00017 using SerializerPtr = std::shared_ptr<pm::ISerializer>;
00018
00019 inline SerializerPtr makeSerializer() {
00020     return std::make_shared<pm::BigEndianSerialization>();
00021 }
00022
00023 bool registerDefaultPacketHandlers(std::shared_ptr<pm::IPacketManager> packet);
00024
00025 }
00026
00027 #endif // COMMON_DEFAULT_PACKET_HANDLERS_HPP_
```

5.51 DefaultPacketHandlers.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** Default packet handlers
00006 */
00007
00008 #ifndef DEFAULT_PACKET_HANDLERS_HPP_
00009 #define DEFAULT_PACKET_HANDLERS_HPP_
00010
00011 #include <memory>
00012 #include "../../common/interfaces/IPacketManager.hpp"
00013
00014 namespace rserv::packet {
00015     bool registerDefaultPacketHandlers(std::shared_ptr<pm::IPacketManager> packet);
00016 }
00017
00018 #endif // DEFAULT_PACKET_HANDLERS_HPP_
```

5.52 SettingsConfig.hpp

```
00001 /*
```

```
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** SettingsConfig
00006 */
00007
00008 #ifndef SETTINGSCONFIG_HPP_
00009 #define SETTINGSCONFIG_HPP_
00010
00011 #include "ui/elements/base/UIElement.hpp"
00012 #include "constants.hpp"
00013
00014 class SettingsConfig {
00015 public:
00016     SettingsConfig() = default;
00017     ~SettingsConfig() = default;
00018
00019     int getColorBlindnessState() const { return _colorBlindnessState; }
00020     void setColorBlindnessState(int state) { _colorBlindnessState = state; }
00021
00022     float getBrightnessValue() const { return _brightnessValue; }
00023     void setBrightnessValue(float value) { _brightnessValue = value; }
00024
00025     bool isHighContrastEnabled() const { return _highContrastEnabled; }
00026     void setHighContrastEnabled(bool enabled) { _highContrastEnabled = enabled; }
00027
00028     ui::UIScale getUIScale() const { return _uiScale; }
00029     void setUIScale(ui::UIScale scale) { _uiScale = scale; }
00030
00031     float getMusicVolume() const { return _musicVolume; }
00032     void setMusicVolume(float volume) { _musicVolume = volume; }
00033
00034     float getSoundVolume() const { return _soundVolume; }
00035     void setSoundVolume(float volume) { _soundVolume = volume; }
00036
00037     enum class ScreenResolution {
00038         RES_800x600 = 0,
00039         RES_1024x768 = 1,
00040         RES_1280x720 = 2,
00041         RES_1920x1080 = 3,
00042         FULLSCREEN = 4
00043     };
00044
00045     ScreenResolution getScreenResolution() const { return _screenResolution; }
00046     void setScreenResolution(ScreenResolution resolution) { _screenResolution = resolution; }
00047
00048     int getTargetFPS() const { return _targetFPS; }
00049     void setTargetFPS(int fps) { _targetFPS = fps; }
00050
00051     float getRenderQuality() const { return _renderQuality; }
00052     void setRenderQuality(float quality) { _renderQuality = quality; }
00053
00054     bool isInGameMetricsEnabled() const { return _inGameMetricsEnabled; }
00055     void setInGameMetricsEnabled(bool enabled) { _inGameMetricsEnabled = enabled; }
00056
00057     std::string getUsername() const { return _username; }
00058     void setUsername(const std::string& username) { _username = username; }
00059
00060     std::string getScreenResolutionName(ScreenResolution resolution) const;
00061     std::pair<int, int> getScreenResolutionSize(ScreenResolution resolution) const;
00062     bool isFullscreen(ScreenResolution resolution) const;
00063
00064     void saveAccessibility(const std::string& filepath = constants::ACCESSIBILITY_FILE_PATH);
00065     void loadAccessibility(const std::string& filepath = constants::ACCESSIBILITY_FILE_PATH);
00066
00067     void saveSettings(const std::string& filepath = constants::SETTINGS_FILE_PATH);
00068     void loadSettings(const std::string& filepath = constants::SETTINGS_FILE_PATH);
00069
00070 private:
00071     int _colorBlindnessState = 0;
00072     float _brightnessValue = 1.0f;
00073     bool _highContrastEnabled = false;
00074     ui::UIScale _uiScale = ui::UIScale::Normal;
00075     float _musicVolume = 100.0f;
00076     float _soundVolume = 100.0f;
00077     ScreenResolution _screenResolution = ScreenResolution::RES_1920x1080;
00078     int _targetFPS = 60;
00079     float _renderQuality = 1.0f;
00080     bool _inGameMetricsEnabled = false;
00081     std::string _username = "";
00082 };
00083
00084 #endif  // SETTINGSCONFIG_HPP_
```

5.53 SettingsManager.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** SettingsManager
00006 */
00007
00008 #ifndef SETTINGSMANAGER_HPP_
00009 #define SETTINGSMANAGER_HPP_
00010
00011 #include <memory>
00012 #include "../../common/InputMapping/InputMappingManager.hpp"
00013 #include "../../common/InputMapping/IInputProvider.hpp"
00014 #include "SettingsConfig.hpp"
00015 #include "../../common/interfaces/IWindow.hpp"
00016
00017 class SettingsManager {
00018 public:
00019     SettingsManager(std::shared_ptr<ecs::InputMappingManager> mappingManager,
00020                     std::shared_ptr<ecs::IInputProvider> inputProvider,
00021                     std::shared_ptr<SettingsConfig> settingsConfig);
00022     ~SettingsManager() = default;
00023
00024     void loadAll();
00025     void saveAll();
00026
00027     void saveKeybinds();
00028     void loadKeybinds();
00029
00030     void saveAccessibility();
00031     void loadAccessibility();
00032
00033     void saveSettings();
00034     void loadSettings();
00035
00036     void applyAccessibilityToWindow(std::shared_ptr<gfx::IWindow> window);
00037
00038 private:
00039     std::shared_ptr<ecs::InputMappingManager> _mappingManager;
00040     std::shared_ptr<ecs::IInputProvider> _inputProvider;
00041     std::shared_ptr<SettingsConfig> _settingsConfig;
00042 };
00043
00044 #endif /* !SETTINGSMANAGER_HPP_ */
```

5.54 AnimationStateSyncSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** AnimationStateSyncSystem
00006 */
00007
00008 #ifndef ANIMATIONSTATESYNCSYSTEM_HPP_
00009 #define ANIMATIONSTATESYNCSYSTEM_HPP_
00010
00011 #include "../../../common/systems/base/ASystem.hpp"
00012 #include <unordered_map>
00013 #include <string>
00014
00015 namespace ecs {
00016
00017 class AnimationStateSyncSystem : public ASystem {
00018     public:
00019         AnimationStateSyncSystem() = default;
00020         ~AnimationStateSyncSystem() override = default;
00021
00022     protected:
00023         void update(std::shared_ptr<ResourceManager> resourceManager,
00024                     std::shared_ptr<Registry> registry,
00025                     float deltaTime) override;
00026 };
00027
00028 }  // namespace ecs
00029
00030 #endif  // ANIMATIONSTATESYNCSYSTEM_HPP_
```

5.55 MusicSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** MusicSystem
00006 */
00007
00008 #ifndef MUSICSYSTEM_HPP_
00009 #define MUSICSYSTEM_HPP_
00010
00011 #include "../../../common/systems/base/ASystem.hpp"
00012 #include <memory>
00013
00014 namespace ecs {
00015
00016 class MusicSystem : public ASystem {
00017     public:
00018         MusicSystem();
00019         ~MusicSystem() override = default;
00020
00021     protected:
00022         void update(
00023             std::shared_ptr<ResourceManager> resourceManager,
00024             std::shared_ptr<Registry> registry,
00025             float deltaTime
00026         ) override;
00027 };
00028
00029 } // namespace ecs
00030
00031 #endif /* !MUSICSYSTEM_HPP_ */
```

5.56 SoundSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** SoundSystem
00006 */
00007
00008 #ifndef SOUNDSYSTEM_HPP_
00009 #define SOUNDSYSTEM_HPP_
00010
00011 #include "../../../common/systems/base/ASystem.hpp"
00012 #include <memory>
00013
00014 namespace ecs {
00015
00016 class SoundSystem : public ASystem {
00017     public:
00018         SoundSystem();
00019         ~SoundSystem() override = default;
00020
00021     protected:
00022         void update(
00023             std::shared_ptr<ResourceManager> resourceManager,
00024             std::shared_ptr<Registry> registry,
00025             float deltaTime
00026         ) override;
00027 };
00028
00029 } // namespace ecs
00030
00031 #endif /* !SOUNDSYSTEM_HPP_ */
```

5.57 ClientEffectCleanupSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ClientEffectCleanupSystem
00006 */
00007
```

```
00008 #ifndef CLIENTEFFECTCLEANUPSYSTEM_HPP_
00009 #define CLIENTEFFECTCLEANUPSYSTEM_HPP_
00010
00011 #include "../../common/systems/base/ASystem.hpp"
00012
00013 namespace ecs {
00014
00015 class ClientEffectCleanupSystem : public ASystem {
00016     public:
00017         ClientEffectCleanupSystem();
00018         ~ClientEffectCleanupSystem() = default;
00019
00020         void update(
00021             std::shared_ptr<ResourceManager> resourceManager,
00022             std::shared_ptr<Registry> registry,
00023             float deltaTime
00024         ) override;
00025 };
00026
00027 }
00028
00029 #endif /* !CLIENTEFFECTCLEANUPSYSTEM_HPP_ */
```

5.58 HideLifetimeSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** HideLifetimeSystem
00006 */
00007
00008 #ifndef HIDELIFETIMESYSTEM_HPP_
00009 #define HIDELIFETIMESYSTEM_HPP_
00010
00011 #include "../../common/systems/base/ASystem.hpp"
00012
00013 namespace ecs {
00014
00015 class HideLifetimeSystem : public ASystem {
00016     public:
00017         HideLifetimeSystem();
00018         ~HideLifetimeSystem() = default;
00019
00020         void update(
00021             std::shared_ptr<ResourceManager> resourceManager,
00022             std::shared_ptr<Registry> registry,
00023             float deltaTime
00024         ) override;
00025 };
00026
00027 }
00028
00029 #endif /* !HIDELIFETIMESYSTEM_HPP_ */
```

5.59 ForceInputSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2026
00003 ** ryanR-type
00004 ** File description:
00005 ** ForceInputSystem
00006 */
00007
00008 #ifndef FORCEINPUTSYSTEM_HPP_
00009 #define FORCEINPUTSYSTEM_HPP_
00010
00011 #include "../../../common/systems/base/ASystem.hpp"
00012 #include "../../../common/ECS/entity/registry/Registry.hpp"
00013 #include "resourceManager/ResourceManager.hpp"
00014
00015 namespace ecs {
00016
00017 class ForceInputSystem : public ASystem {
00018     public:
00019         ForceInputSystem();
00020         ~ForceInputSystem() = default;
00021
```

```
00022         void update(
00023             std::shared_ptr<ResourceManager> resourceManager,
00024             std::shared_ptr<Registry> registry,
00025             float deltaTime
00026         ) override;
00027
00028     private:
00029         bool isPlayerAlive(
00030             std::shared_ptr<Registry> registry,
00031             Entity entityId
00032         ) const;
00033         float _lastForceTime;
00034 };
00035
00036 }  // namespace ecs
00037
00038 #endif /* !FORCEINPUTSYSTEM_HPP_ */
```

5.60 MovementInputSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** MovementInputSystem
00006 */
00007
00008 #ifndef MOVEMENTINPUTSYSTEM_HPP_
00009 #define MOVEMENTINPUTSYSTEM_HPP_
00010
00011 #include "../../../common/systems/base/ASystem.hpp"
00012 #include "../../../common/components/temporary/InputIntentComponent.hpp"
00013 #include "../../../common/InputMapping/IInputProvider.hpp"
00014 #include "../../../common/InputMapping/InputAction.hpp"
00015 #include <memory>
00016
00017 namespace gfx {
00018     class IEvent;
00019 }
00020
00021 namespace ecs {
00022
00023 class MovementInputSystem : public ASystem {
00024     public:
00025         MovementInputSystem();
00026         ~MovementInputSystem() = default;
00027
00028         void update(std::shared_ptr<ResourceManager> resourceManager, std::shared_ptr<Registry>
      registry, float deltaTime) override;
00029
00030     private:
00031         math::Vector2f getMovementDirection(std::shared_ptr<ResourceManager> resourceManager) const;
00032         void updateInputIntent(std::shared_ptr<Registry> registry, Entity entityId, const
      math::Vector2f &direction);
00033         math::Vector2f getAnalogStickInput(std::shared_ptr<IInputProvider> inputProvider) const;
00034         void sendAxisEvents(std::shared_ptr<ResourceManager> resourceManager, const math::Vector2f
      &direction);
00035         bool isPlayerAlive(std::shared_ptr<Registry> registry, Entity entityId) const;
00036         bool _wasMovingLastFrame = false;
00037 };
00038
00039 } // namespace ecs
00040
00041 #endif /* !MOVEMENTINPUTSYSTEM_HPP_ */
```

5.61 ShootInputSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ShootInputSystem
00006 */
00007
00008 #ifndef SHOOTINPUTSYSTEM_HPP_
00009     #define SHOOTINPUTSYSTEM_HPP_
00010
00011 #include <memory>
```

```
00012 #include "../../../common/systems/base/ASystem.hpp"
00013
00014 namespace ecs {
00015
00016 class ShootInputSystem : public ASystem {
00017     public:
00018         ShootInputSystem();
00019         ~ShootInputSystem() = default;
00020
00021         void update(
00022             std::shared_ptr<ResourceManager> resourceManager,
00023             std::shared_ptr<Registry> registry,
00024             float deltaTime
00025         ) override;
00026
00027     private:
00028         bool isPlayerAlive(std::shared_ptr<Registry> registry, Entity entityId) const;
00029 };
00030
00031 }
00032
00033 #endif /* !SHOOTINPUTSYSTEM_HPP_ */
```

5.62 NetworkInterpolationSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** NetworkInterpolationSystem
00006 */
00007
00008 #ifndef NETWORKINTERPOLATIONSYSTEM_HPP_
00009 #define NETWORKINTERPOLATIONSYSTEM_HPP_
00010
00011 #include "../components/permanent/NetworkStateComponent.hpp"
00012 #include "../../common/systems/base/ASystem.hpp"
00013 #include "../../common/ECS/entity/registry/Registry.hpp"
00014 #include "../../common/components/permanent/TransformComponent.hpp"
00015 #include "../../common/components/permanent/HealthComponent.hpp"
00016
00017 namespace ecs {
00018
00019 class NetworkInterpolationSystem : public ASystem {
00020     public:
00021         NetworkInterpolationSystem() = default;
00022         ~NetworkInterpolationSystem() override = default;
00023
00024         void update(
00025             std::shared_ptr<ResourceManager> resourceManager,
00026             std::shared_ptr<Registry> registry,
00027             float deltaTime
00028         ) override;
00029
00030     private:
00031         void interpolateTransform(
00032             std::shared_ptr<NetworkStateComponent> networkState,
00033             std::shared_ptr<TransformComponent> transform
00034         );
00035         float getTransformInterpolationFactor(std::shared_ptr<NetworkStateComponent> networkState)
      const;
00036 };
00037
00038 }  // namespace ecs
00039
00040 #endif /* !NETWORKINTERPOLATIONSYSTEM_HPP_ */
```

5.63 AnimationRenderingSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** AnimationRenderingSystem
00006 */
00007
00008 #ifndef ANIMATIONRENDERINGSYSTEM_HPP_
00009 #define ANIMATIONRENDERINGSYSTEM_HPP_
```

```
00010
00011
00012 #include "../../../common/systems/base/ASystem.hpp"
00013 #include <memory>
00014 #include "../../components/rendering/AnimationComponent.hpp"
00015 #include "../../../common/ECS/entity/Entity.hpp"
00016 #include "../../../common/ECS/entity/registry/Registry.hpp"
00017 #include "../../../common/Parser/Animation/AnimationConditionFactory.hpp"
00018 namespace ecs {
00019
00020 class AnimationRenderingSystem : public ASystem {
00021     public:
00022         AnimationRenderingSystem();
00023         ~AnimationRenderingSystem() override = default;
00024
00025     protected:
00026         void update(std::shared_ptr<ResourceManager> resourceManager,
00027         std::shared_ptr<Registry> registry, float deltaTime) override;
00028
00029     private:
00030         std::unordered_map<Entity, float> _waitTimers;
00031 };
00032
00033 }  // namespace ecs
00034
00035 #endif /* !ANIMATIONRENDERINGSYSTEM_HPP_ */
```

5.64 GameZoneRenderingSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** GameZoneRenderingSystem
00006 */
00007
00008 #ifndef GAMEZONERENDERINGSYSTEM_HPP_
00009 #define GAMEZONERENDERINGSYSTEM_HPP_
00010
00011 #include "../../../common/systems/base/ASystem.hpp"
00012 #include "../../../common/components/permanent/GameZoneComponent.hpp"
00013 #include <memory>
00014
00015 namespace ecs {
00016
00017 class GameZoneRenderingSystem : public ASystem {
00018     public:
00019         GameZoneRenderingSystem();
00020         ~GameZoneRenderingSystem() override = default;
00021
00022     protected:
00023         void update(std::shared_ptr<ResourceManager> resourceManager,
00024             std::shared_ptr<Registry> registry, float deltaTime) override;
00025 };
00026
00027 }  // namespace ecs
00028
00029 #endif /* !GAMEZONERENDERINGSYSTEM_HPP_ */
```

5.65 GameZoneViewSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** GameZoneViewSystem
00006 */
00007
00008 #ifndef GAMEZONEVIEWSYSTEM_HPP_
00009 #define GAMEZONEVIEWSYSTEM_HPP_
00010
00011 #include "../../../common/systems/base/ASystem.hpp"
00012 #include "../../../common/components/permanent/GameZoneComponent.hpp"
00013 #include <memory>
00014
00015 namespace ecs {
00016
00017 class GameZoneViewSystem : public ASystem {
```

```
00018     public:
00019         GameZoneViewSystem();
00020         ~GameZoneViewSystem() override = default;
00021
00022     protected:
00023         void update(std::shared_ptr<ResourceManager> resourceManager,
00024             std::shared_ptr<Registry> registry, float deltaTime) override;
00025 };
00026
00027 }  // namespace ecs
00028
00029 #endif /* !GAMEZONEVIEWSYSTEM_HPP_ */
```

5.66 HealthBarRenderingSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** HealthBarRenderingSystem
00006 */
00007
00008 #ifndef HEALTHBARRENDERINGSYSTEM_HPP_
00009 #define HEALTHBARRENDERINGSYSTEM_HPP_
00010
00011 #include "../../../common/systems/base/ASystem.hpp"
00012 #include <memory>
00013
00014 namespace ecs {
00015
00016 class HealthBarRenderingSystem : public ASystem {
00017     public:
00018         HealthBarRenderingSystem();
00019         ~HealthBarRenderingSystem() override = default;
00020
00021     protected:
00022         void update(std::shared_ptr<ResourceManager> resourceManager,
00023             std::shared_ptr<Registry> registry, float deltaTime) override;
00024 };
00025
00026 }  // namespace ecs
00027
00028 #endif /* !HEALTHBARRENDERINGSYSTEM_HPP_ */
```

5.67 HitboxRenderingSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** HitboxRenderingSystem
00006 */
00007
00008 #ifndef HITBOXRENDERINGSYSTEM_HPP_
00009 #define HITBOXRENDERINGSYSTEM_HPP_
00010
00011 #include "../../../common/systems/base/ASystem.hpp"
00012 #include <memory>
00013
00014 namespace ecs {
00015
00016 class HitboxRenderingSystem : public ASystem {
00017     public:
00018         HitboxRenderingSystem();
00019         ~HitboxRenderingSystem() override = default;
00020
00021     protected:
00022         void update(std::shared_ptr<ResourceManager> resourceManager,
00023             std::shared_ptr<Registry> registry, float deltaTime) override;
00024 };
00025
00026 } // namespace ecs
00027
00028 #endif /* !HITBOXRENDERINGSYSTEM_HPP_ */
```

5.68 ParallaxRenderingSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ParallaxRenderingSystem
00006 */
00007
00008 #ifndef PARALLAXRENDERINGSYSTEM_HPP_
00009 #define PARALLAXRENDERINGSYSTEM_HPP_
00010
00011 #include "../../../common/systems/base/ASystem.hpp"
00012 #include "../../components/rendering/ParallaxComponent.hpp"
00013 #include "../../../common/types/Vector2f.hpp"
00014 #include <memory>
00015
00016 namespace ecs {
00017
00018 class ParallaxRenderingSystem : public ASystem {
00019     public:
00020         ParallaxRenderingSystem();
00021         ~ParallaxRenderingSystem() override = default;
00022
00023     protected:
00024         void update(std::shared_ptr<ResourceManager> resourceManager,
00025             std::shared_ptr<Registry> registry, float deltaTime) override;
00026
00027     private:
00028         math::Vector2f calculateScale(
00029             std::shared_ptr<ParallaxLayer> layer,
00030             float screenWidth,
00031             float screenHeight
00032         );
00033
00034         void renderLayer(
00035             std::shared_ptr<ParallaxLayer> layer,
00036             std::shared_ptr<ResourceManager> resourceManager,
00037             const math::Vector2f& basePosition,
00038             float screenWidth, float screenHeight);
00039 };
00040
00041 }  // namespace ecs
00042
00043 #endif /* !PARALLAXRENDERINGSYSTEM_HPP_ */
```

5.69 RectangleRenderingSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** RectangleRenderingSystem
00006 */
00007
00008 #ifndef RECTANGLERENDERINGSYSTEM_HPP_
00009 #define RECTANGLERENDERINGSYSTEM_HPP_
00010
00011 #include "../../../common/systems/base/ASystem.hpp"
00012 #include <memory>
00013
00014 namespace ecs {
00015
00016 class RectangleRenderingSystem : public ASystem {
00017     public:
00018         RectangleRenderingSystem();
00019         ~RectangleRenderingSystem() override = default;
00020
00021     protected:
00022         void update(std::shared_ptr<ResourceManager> resourceManager,
00023             std::shared_ptr<Registry> registry, float deltaTime) override;
00024 };
00025
00026 } // namespace ecs
00027
00028 #endif /* !RECTANGLERENDERINGSYSTEM_HPP_ */
```

5.70 SpriteRenderingSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** SpriteRenderingSystem
00006 */
00007
00008 #ifndef SPRITERENDERINGSYSTEM_HPP_
00009 #define SPRITERENDERINGSYSTEM_HPP_
00010
00011 #include "../../../common/systems/base/ASystem.hpp"
00012 #include <memory>
00013
00014 namespace ecs {
00015
00016 class SpriteRenderingSystem : public ASystem {
00017     public:
00018         SpriteRenderingSystem();
00019         ~SpriteRenderingSystem() override = default;
00020
00021     protected:
00022         void update(std::shared_ptr<ResourceManager> resourceManager,
00023             std::shared_ptr<Registry> registry, float deltaTime) override;
00024 };
00025
00026 } // namespace ecs
00027
00028 #endif /* !SPRITERENDERINGSYSTEM_HPP_ */
```

5.71 TextRenderingSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** TextRenderingSystem
00006 */
00007
00008 #ifndef TEXTRENDERINGSYSTEM_HPP_
00009 #define TEXTRENDERINGSYSTEM_HPP_
00010
00011 #include "../../../common/systems/base/ASystem.hpp"
00012 #include <memory>
00013
00014 namespace ecs {
00015
00016 class TextRenderingSystem : public ASystem {
00017     public:
00018         TextRenderingSystem() ;
00019         ~TextRenderingSystem() override = default;
00020
00021     protected:
00022         void update(std::shared_ptr<ResourceManager> resourceManager,
00023             std::shared_ptr<Registry> registry, float deltaTime) override;
00024 };
00025
00026 } // namespace ecs
00027
00028 #endif /* !TEXTRENDERINGSYSTEM_HPP_ */
```

5.72 ReplaySystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ReplaySystem
00006 */
00007
00008 #ifndef REPLAYSYSTEM_HPP_
00009 #define REPLAYSYSTEM_HPP_
00010
00011 #include <memory>
00012 #include <string>
00013 #include <unordered_map>
```

```
00014 #include <unordered_set>
00015 #include <nlohmann/json.hpp>
00016 #include "../../../common/systems/base/ASystem.hpp"
00017 #include "../../../common/ECS/entity/registry/Registry.hpp"
00018 #include "../../../common/resourceManager/ResourceManager.hpp"
00019 #include "../components/rendering/SpriteComponent.hpp"
00020 #include "../components/rendering/AnimationComponent.hpp"
00021 #include "../components/rendering/ParallaxComponent.hpp"
00022 #include "../components/rendering/HealthBarComponent.hpp"
00023 #include "../components/rendering/TextComponent.hpp"
00024 #include "../components/rendering/RectangleRenderComponent.hpp"
00025 #include "../components/rendering/HitboxRenderComponent.hpp"
00026 #include "../components/temporary/SoundIntentComponent.hpp"
00027 #include "../../../common/components/permanent/GameZoneComponent.hpp"
00028 #include "../../../common/components/permanent/ColliderComponent.hpp"
00029 #include "../../../common/components/permanent/TransformComponent.hpp"
00030 #include "../../../common/components/permanent/HealthComponent.hpp"
00031
00032 namespace ecs {
00033
00034 class ReplaySystem : public ASystem {
00035     public:
00036         ReplaySystem();
00037         ~ReplaySystem() = default;
00038
00039         void update(std::shared_ptr<ResourceManager> resourceManager, std::shared_ptr<Registry>
      registry, float deltaTime) override;
00040
00041     private:
00042         void saveReplayToFile(const nlohmann::json& frameData);
00043
00044         std::string getSpriteId(const std::string& texturePath);
00045         std::filesystem::path getNextReplayFile();
00046
00047         float _totalElapsedTime;
00048         std::filesystem::path _currentReplayFile;
00049 };
00050
00051 } // namespace ecs
00052
00053 #endif /* !REPLAYSYSTEM_HPP_ */
```

5.73 AFocusableElement.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** AFocusableElement
00006 */
00007
00008 #ifndef AFOCUSABLEELEMENT_HPP_
00009 #define AFOCUSABLEELEMENT_HPP_
00010
00011 #include "../elements/base/UIElement.hpp"
00012 #include "IFocusable.hpp"
00013 #include <functional>
00014
00015 namespace ui {
00016
00017 class AFocusableElement : public UIElement, public IFocusable {
00018     public:
00019         explicit AFocusableElement(std::shared_ptr<ResourceManager> resourceManager);
00020         virtual ~AFocusableElement() = default;
00021
00022         virtual void setFocused(bool focused) override;
00023         virtual bool isFocused() const override;
00024         virtual bool canBeFocused() const override;
00025         virtual void onFocusGained() override;
00026         virtual void onFocusLost() override;
00027         virtual void onActivated() override;
00028
00029         void setOnFocusGained(std::function<void()> callback);
00030         void setOnFocusLost(std::function<void()> callback);
00031         void setOnActivated(std::function<void()> callback);
00032
00033         virtual void handleInput(const math::Vector2f& mousePos, bool mousePressed) override;
00034
00035     protected:
00036         bool _focused = false;
00037         bool _pressedInside = false;
00038         bool _wasPressed = false;
00039         std::function<void()> _onFocusGained;
```

```
00040         std::function<void()> _onFocusLost;
00041         std::function<void()> _onActivated;
00042
00043         virtual void onFocusStateChanged(bool focused);
00044 };
00045
00046 } // namespace ui
00047
00048 #endif /* !AFOCUSABLEELEMENT_HPP_ */
```

5.74 IFocusable.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** IFocusable
00006 */
00007
00008 #ifndef IFOCUSABLE_HPP_
00009 #define IFOCUSABLE_HPP_
00010
00011 #include <memory>
00012
00013 namespace ui {
00014
00015 class IFocusable {
00016     public:
00017         virtual ~IFocusable() = default;
00018
00019         virtual void setFocused(bool focused) = 0;
00020         virtual bool isFocused() const = 0;
00021         virtual bool canBeFocused() const = 0;
00022
00023         virtual void onFocusGained() = 0;
00024         virtual void onFocusLost() = 0;
00025         virtual void onActivated() = 0;
00026
00027         virtual bool onNavigateLeft() { return false; }
00028         virtual bool onNavigateRight() { return false; }
00029 };
00030
00031 }  // namespace ui
00032
00033 #endif /* !IFOCUSABLE_HPP_ */
```

5.75 UILayout.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** UILayout
00006 */
00007
00008 #ifndef UILAYOUT_HPP_
00009 #define UILAYOUT_HPP_
00010
00011 #include <memory>
00012 #include <vector>
00013 #include "../elements/base/UIElement.hpp"
00014 #include "../../../common/types/Vector2f.hpp"
00015 #include "../../../common/interfaces/IWindow.hpp"
00016
00017 using namespace ::gfx;
00018
00019 namespace ui {
00020
00021 enum class LayoutDirection {
00022     Horizontal,
00023     Vertical
00024 };
00025
00026 enum class LayoutAlignment {
00027     Start,
00028     Center,
00029     End
00030 };
```

```
00031
00032 enum class AnchorX {
00033     None,
00034     Left,
00035     Center,
00036     Right
00037 };
00038
00039 enum class AnchorY {
00040     None,
00041     Top,
00042     Center,
00043     Bottom
00044 };
00045
00046 struct LayoutConfig {
00047     LayoutDirection direction = LayoutDirection::Vertical;
00048     LayoutAlignment alignment = LayoutAlignment::Start;
00049     float spacing = 0.0f;
00050     math::Vector2f padding = math::Vector2f(0.0f, 0.0f);
00051     math::Vector2f offset = math::Vector2f(0.0f, 0.0f);
00052     bool autoResize = false;
00053     AnchorX anchorX = AnchorX::None;
00054     AnchorY anchorY = AnchorY::None;
00055
00056     struct BackgroundConfig {
00057         bool enabled = false;
00058         color_t fillColor = {0, 0, 0, 0};
00059         color_t outlineColor = {0, 0, 0, 0};
00060         float cornerRadius = 0.0f;
00061     } background;
00062 };
00063
00064 class UILayout : public UIElement {
00065     public:
00066         UILayout(std::shared_ptr<ResourceManager> resourceManager, const LayoutConfig& config =
      LayoutConfig());
00067         ~UILayout() override = default;
00068
00069         void addElement(std::shared_ptr<UIElement> element);
00070         void removeElement(std::shared_ptr<UIElement> element);
00071         void clearElements();
00072
00073         void setDirection(LayoutDirection direction);
00074         void setAlignment(LayoutAlignment alignment);
00075         void setSpacing(float spacing);
00076         void setPadding(const math::Vector2f& padding);
00077         void setOffset(const math::Vector2f& offset);
00078         void setAutoResize(bool autoResize);
00079         void setAnchor(AnchorX anchorX, AnchorY anchorY);
00080
00081         void setBackgroundEnabled(bool enabled);
00082         void setBackgroundFillColor(const color_t& color);
00083         void setBackgroundOutlineColor(const color_t& color);
00084         void setBackgroundCornerRadius(float radius);
00085
00086         LayoutDirection getDirection() const;
00087         LayoutAlignment getAlignment() const;
00088         float getSpacing() const;
00089         math::Vector2f getPadding() const;
00090         bool isAutoResize() const;
00091
00092         void updateLayout();
00093
00094         void setScale(UIScale scale) override;
00095
00096         void render() override;
00097         void update(float deltaTime) override;
00098
00099         float getScaledSpacing() const;
00100         void applyAnchor();    private:
00101         LayoutConfig _config;
00102         std::vector<std::shared_ptr<UIElement» _layoutElements;
00103
00104         void calculatePositions();
00105         float getTotalSize() const;
00106         math::Vector2f calculateElementPosition(size_t index, float totalSize) const;
00107 };
00108
00109 }  // namespace ui
00110
00111 #endif /* !UILAYOUT_HPP_ */
```

5.76 Background.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** Background
00006 */
00007
00008 #ifndef BACKGROUND_HPP_
00009 #define BACKGROUND_HPP_
00010
00011 #include "base/UIElement.hpp"
00012 #include "../../../common/constants.hpp"
00013 #include <string>
00014 #include <vector>
00015
00016 namespace ui {
00017
00018 class Background : public UIElement {
00019 public:
00020     Background(std::shared_ptr<ResourceManager> resourceManager);
00021     ~Background() override = default;
00022
00023     void render() override;
00024     void update(float deltaTime) override;
00025
00026     void addLayer(const std::string& texturePath, float speedX, float speedY = 0.0f,
00027         const math::Vector2f& sourceSize = math::Vector2f(constants::MAX_WIDTH,
      constants::MAX_HEIGHT));
00028
00029 private:
00030     struct Layer {
00031         std::string texturePath;
00032         float speedX;
00033         float speedY;
00034         math::Vector2f sourceSize;
00035         float offsetX = 0.0f;
00036         float offsetY = 0.0f;
00037     };
00038
00039     float calculateScale(const Layer& layer, float screenWidth);
00040
00041     std::vector<Layer> _layers;
00042 };
00043
00044 }  // namespace ui
00045
00046 #endif  // BACKGROUND_HPP_
```

5.77 UIElement.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** UIElement
00006 */
00007
00008 #ifndef UIELEMENT_HPP_
00009 #define UIELEMENT_HPP_
00010
00011 #include <memory>
00012 #include <vector>
00013 #include <functional>
00014 #include "../../../../common/types/Vector2f.hpp"
00015 #include "../../../../common/resourceManager/ResourceManager.hpp"
00016
00017 namespace ui {
00018
00019 enum class UIState {
00020     Normal,
00021     Hovered,
00022     Pressed,
00023     Disabled,
00024     Focused
00025 };
00026
00027 enum class UIScale {
00028     Small,
00029     Normal,
00030     Large
```

```
00031 };
00032
00033 class UIElement : public std::enable_shared_from_this<UIElement> {
00034     public:
00035         UIElement(std::shared_ptr<ResourceManager> resourceManager);
00036         virtual ~UIElement() = default;
00037
00038         void setPosition(const math::Vector2f& position);
00039         void setSize(const math::Vector2f& size);
00040         math::Vector2f getPosition() const;
00041         math::Vector2f getSize() const;
00042
00043         math::Vector2f getAbsolutePosition() const;
00044         math::Vector2f getAbsoluteSize() const;
00045
00046         void setVisible(bool visible);
00047         bool isVisible() const;
00048
00049         void setScalingEnabled(bool enabled);
00050         bool isScalingEnabled() const;
00051
00052         void setFocusEnabled(bool enabled);
00053         bool isFocusEnabled() const;
00054
00055         void setState(UIState state);
00056         UIState getState() const;
00057
00058         virtual void setScale(UIScale scale);
00059         UIScale getScale() const;
00060
00061         void setParent(std::weak_ptr<UIElement> parent);
00062         std::shared_ptr<UIElement> getParent() const;
00063         void addChild(std::shared_ptr<UIElement> child);
00064         void removeChild(std::shared_ptr<UIElement> child);
00065         const std::vector<std::shared_ptr<UIElement»& getChildren() const;
00066
00067         virtual void handleInput(const math::Vector2f& mousePos, bool mousePressed);
00068         virtual bool containsPoint(const math::Vector2f& point) const;
00069
00070         void setOnClick(std::function<void()> callback);
00071         void setOnHover(std::function<void()> callback);
00072         void setOnRelease(std::function<void()> callback);
00073
00074         virtual void render();
00075
00076         virtual void update(float deltaTime);
00077
00078     protected:
00079         std::weak_ptr<ResourceManager> _resourceManager;
00080         math::Vector2f _position;
00081         math::Vector2f _size;
00082         bool _visible = true;
00083         bool _scalingEnabled = true;
00084         bool _focusEnabled = true;
00085         UIState _state = UIState::Normal;
00086         UIScale _scale = UIScale::Normal;
00087         std::weak_ptr<UIElement> _parent;
00088         std::vector<std::shared_ptr<UIElement» _children;
00089
00090         bool _pressedInside = false;
00091         bool _wasPressed = false;
00092
00093         std::function<void()> _onClick;
00094         std::function<void()> _onHover;
00095         std::function<void()> _onRelease;
00096
00097         std::pair<int, int> getWindowSize() const;
00098         std::pair<int, int> getLogicalSize() const;
00099
00100         float getScaleFactor() const;
00101 };
00102
00103 } // namespace ui
00104
00105 #endif /* !UIELEMENT_HPP_ */
```

5.78 Box.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2026
00003 ** R-Type
00004 ** File description:
00005 ** Box
```

```
00006 */
00007
00008 #ifndef BOX_HPP_
00009 #define BOX_HPP_
00010
00011 #include "base/UIElement.hpp"
00012 #include "../../../common/interfaces/IWindow.hpp"
00013
00014 namespace ui {
00015
00016 class Box : public UIElement {
00017 public:
00018     explicit Box(std::shared_ptr<ResourceManager> resourceManager);
00019     ~Box() override = default;
00020
00021     void setBackgroundColor(const gfx::color_t& color);
00022     void setBorderColor(const gfx::color_t& color);
00023     void setBorderThickness(float thickness);
00024     void setCornerRadius(float radius);
00025
00026     void render() override;
00027
00028 private:
00029     gfx::color_t _backgroundColor = gfx::color_t{40, 40, 60, 200};
00030     gfx::color_t _borderColor = gfx::color_t{80, 80, 120, 255};
00031     float _borderThickness = 2.0f;
00032     float _cornerRadius = 8.0f;
00033 };
00034
00035 }  // namespace ui
00036
00037 #endif /* !BOX_HPP_ */
```

5.79 Button.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** Button
00006 */
00007
00008 #ifndef BUTTON_HPP_
00009 #define BUTTON_HPP_
00010
00011 #include "../../core/AFocusableElement.hpp"
00012 #include <string>
00013 #include "../../../../common/interfaces/IWindow.hpp"
00014 #include "../../../constants.hpp"
00015 #include "../../../colors.hpp"
00016
00017 namespace ui {
00018
00019 class Button : public AFocusableElement {
00020     public:
00021         explicit Button(std::shared_ptr<ResourceManager> resourceManager);
00022         virtual ~Button() = default;
00023
00024         void setText(const std::string& text);
00025         const std::string& getText() const;
00026         void setTextColor(const gfx::color_t& color);
00027         void setFontPath(const std::string& fontPath);
00028
00029         void setNormalColor(const gfx::color_t& color);
00030         void setHoveredColor(const gfx::color_t& color);
00031         void setPressedColor(const gfx::color_t& color);
00032         void setDisabledColor(const gfx::color_t& color);
00033         void setFocusedColor(const gfx::color_t& color);
00034         void setBaseFontSize(size_t fontSize);
00035         size_t getBaseFontSize() const;
00036
00037         void setIconPath(const std::string& iconPath);
00038         void setIconSize(const math::Vector2f& size);
00039
00040         virtual void render() override;
00041
00042     private:
00043         std::string _text;
00044         gfx::color_t _textColor = colors::UI_TEXT;
00045         std::string _fontPath = constants::MAIN_FONT;
00046
00047         gfx::color_t _normalColor = colors::BUTTON_PRIMARY;
00048         gfx::color_t _hoveredColor = colors::BUTTON_PRIMARY_HOVER;
```

```
00049         gfx::color_t _pressedColor = colors::BUTTON_PRIMARY_PRESSED;
00050         gfx::color_t _disabledColor = colors::UI_DISABLED;
00051         gfx::color_t _focusedColor = colors::UI_FOCUSED;
00052         size_t _baseFontSize = constants::BUTTON_FONT_SIZE_BASE;
00053
00054         std::string _iconPath;
00055         math::Vector2f _iconSize = math::Vector2f(0.f, 0.f);
00056
00057         gfx::color_t getCurrentColor() const;
00058         size_t getFontSize() const;
00059 };
00060
00061 }  // namespace ui
00062
00063 #endif /* !BUTTON_HPP_ */
```

5.80 Dropdown.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** Dropdown
00006 */
00007
00008 #ifndef DROPDOWN_HPP_
00009 #define DROPDOWN_HPP_
00010
00011 #include "../../core/AFocusableElement.hpp"
00012 #include <string>
00013 #include <vector>
00014 #include <functional>
00015 #include "../../../../common/interfaces/IWindow.hpp"
00016 #include "../../../colors.hpp"
00017 #include "../../../constants.hpp"
00018
00019 namespace ui {
00020
00021 enum class DropdownDirection {
00022     Down,
00023     Up,
00024     Left,
00025     Right
00026 };
00027
00028 class Dropdown : public AFocusableElement {
00029     public:
00030         Dropdown(std::shared_ptr<ResourceManager> resourceManager);
00031         ~Dropdown();
00032
00033         virtual void render() override;
00034         virtual void handleInput(const math::Vector2f& mousePos, bool mousePressed) override;
00035         virtual void update(float deltaTime) override;
00036         virtual bool containsPoint(const math::Vector2f& point) const override;
00037
00038         void setOptions(const std::vector<std::string>& options);
00039         void addOption(const std::string& option);
00040         void clearOptions();
00041
00042         const std::vector<std::string>& getOptions() const;
00043         const std::string& getSelectedOption() const;
00044
00045         void setSelectedIndex(size_t index);
00046         size_t getSelectedIndex() const;
00047         const std::string& getSelectedValue() const;
00048
00049         void setDirection(DropdownDirection direction);
00050         DropdownDirection getDirection() const;
00051
00052         void setPlaceholder(const std::string& placeholder);
00053         const std::string& getPlaceholder() const;
00054
00055         void setTextColor(const gfx::color_t& color);
00056         void setPlaceholderColor(const gfx::color_t& color);
00057         void setFontPath(const std::string& fontPath);
00058         void setBaseFontSize(size_t fontSize);
00059         size_t getBaseFontSize() const;
00060
00061         void setOnSelectionChanged(std::function<void(const std::string&, size_t)> callback);
00062
00063         void open();
00064         void close();
00065         bool isOpen() const;
```

```
00066
00067     private:
00068         std::vector<std::string> _options;
00069         size_t _selectedIndex = 0;
00070         bool _hasSelection = false;
00071
00072         DropdownDirection _direction = DropdownDirection::Down;
00073         bool _isOpen = false;
00074         int _hoveredOptionIndex = -1;
00075
00076         std::string _placeholder = "Select...";
00077
00078         std::function<void(const std::string&, size_t)> _onSelectionChanged;
00079
00080         gfx::color_t _textColor = colors::UI_TEXT;
00081         gfx::color_t _placeholderColor = colors::UI_DISABLED;
00082         std::string _fontPath = constants::MAIN_FONT;
00083         size_t _baseFontSize = constants::BUTTON_FONT_SIZE_BASE;
00084
00085         gfx::color_t _normalColor = colors::BUTTON_PRIMARY;
00086         gfx::color_t _hoveredColor = colors::BUTTON_PRIMARY_HOVER;
00087         gfx::color_t _pressedColor = colors::BUTTON_PRIMARY_PRESSED;
00088         gfx::color_t _disabledColor = colors::UI_DISABLED;
00089         gfx::color_t _focusedColor = colors::UI_FOCUSED;
00090
00091         void renderClosed();
00092         void renderOpen();
00093         void drawArrow(float arrowX, float arrowY, float arrowSize);
00094         bool isMouseOverOption(const math::Vector2f& mousePos, size_t optionIndex) const;
00095         math::Vector2f getOptionPosition(size_t optionIndex) const;
00096         math::Vector2f getDropdownSize() const;
00097         size_t getFontSize() const;
00098         gfx::color_t _getCurrentColor() const;
00099
00100         bool _dropdownWasPressed = false;
00101 };
00102
00103 }  // namespace ui
00104
00105 #endif /* !DROPDOWN_HPP_ */
```

5.81 Slider.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** Slider
00006 */
00007
00008 #ifndef SLIDER_HPP_
00009 #define SLIDER_HPP_
00010
00011 #include "../../core/AFocusableElement.hpp"
00012 #include <string>
00013 #include <functional>
00014 #include "../../../../common/interfaces/IWindow.hpp"
00015 #include "../../../constants.hpp"
00016 #include "../../../colors.hpp"
00017
00018 namespace ui {
00019
00020 class Slider : public AFocusableElement {
00021     public:
00022         explicit Slider(std::shared_ptr<ResourceManager> resourceManager);
00023         virtual ~Slider() = default;
00024
00025         void setMinValue(float minValue);
00026         void setMaxValue(float maxValue);
00027         void setValue(float value);
00028         float getValue() const;
00029         float getMinValue() const;
00030         float getMaxValue() const;
00031
00032         void setStep(float step);
00033         float getStep() const;
00034
00035         void setLabel(const std::string& label);
00036         const std::string& getLabel() const;
00037         void setLabelColor(const gfx::color_t& color);
00038         void setFontPath(const std::string& fontPath);
00039         void setBaseFontSize(size_t fontSize);
00040         size_t getBaseFontSize() const;
```

```
00041         void setShowPercentage(bool show);
00042
00043         void setTrackColor(const gfx::color_t& color);
00044         void setFillColor(const gfx::color_t& color);
00045         void setHandleColor(const gfx::color_t& color);
00046         void setHandleHoveredColor(const gfx::color_t& color);
00047         void setHandleFocusedColor(const gfx::color_t& color);
00048
00049         void setOnValueChanged(std::function<void(float)> callback);
00050
00051         virtual void render() override;
00052         virtual void handleInput(const math::Vector2f& mousePos, bool mousePressed) override;
00053         virtual void onActivated() override;
00054         virtual bool onNavigateLeft() override;
00055         virtual bool onNavigateRight() override;
00056
00057         void incrementValue();
00058         void decrementValue();
00059
00060     private:
00061         float _minValue = 0.0f;
00062         float _maxValue = 1.0f;
00063         float _value = 0.5f;
00064         float _step = 0.1f;
00065         float _visualNormalizedValue = 0.5f;
00066
00067         std::string _label;
00068         gfx::color_t _labelColor = colors::SLIDER_LABEL;
00069         std::string _fontPath = constants::MAIN_FONT;
00070         size_t _baseFontSize = constants::BUTTON_FONT_SIZE_BASE;
00071         float _outlineThickness = 2.0f;
00072         bool _showPercentage = true;
00073
00074         gfx::color_t _trackColor = colors::SLIDER_TRACK;
00075         gfx::color_t _fillColor = colors::SLIDER_FILL;
00076         gfx::color_t _handleColor = colors::SLIDER_HANDLE;
00077         gfx::color_t _handleHoveredColor = colors::SLIDER_HANDLE_HOVER;
00078         gfx::color_t _handleFocusedColor = colors::SLIDER_HANDLE_FOCUSED;
00079
00080         std::function<void(float)> _onValueChanged;
00081
00082         bool _isDragging = false;
00083         bool _wasMousePressed = false;
00084
00085         float getNormalizedValue() const;
00086         void setNormalizedValue(float normalized);
00087         gfx::color_t getCurrentHandleColor() const;
00088         size_t getFontSize() const;
00089         float getHandleRadius() const;
00090         float getTrackHeight() const;
00091         float getLabelHeight() const;
00092 };
00093
00094 }  // namespace ui
00095
00096 #endif /* !SLIDER_HPP_ */
```

5.82 TextInput.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** TextInput
00006 */
00007
00008 #ifndef TEXTINPUT_HPP_
00009 #define TEXTINPUT_HPP_
00010
00011 #include "../../core/AFocusableElement.hpp"
00012 #include <string>
00013 #include <functional>
00014 #include "../../../../libs/Multimedia/EventTypes.hpp"
00015 #include "../../../../common/interfaces/IWindow.hpp"
00016 #include "../../../colors.hpp"
00017 #include "../../../constants.hpp"
00018
00019 namespace ui {
00020
00021 class TextInput : public AFocusableElement {
00022     public:
00023         TextInput(std::shared_ptr<ResourceManager> resourceManager);
00024         ~TextInput();
```

```
00025
00026         virtual void render() override;
00027
00028         void setText(const std::string& text);
00029         const std::string& getText() const;
00030         void setPlaceholder(const std::string& placeholder);
00031         const std::string& getPlaceholder() const;
00032         void setTextColor(const gfx::color_t& color);
00033         void setPlaceholderColor(const gfx::color_t& color);
00034         void setFontPath(const std::string& fontPath);
00035         void setBaseFontSize(size_t fontSize);
00036         size_t getBaseFontSize() const;
00037
00038         void setOnTextChanged(std::function<void(const std::string&)> callback);
00039         void setOnSubmit(std::function<void(const std::string&)> callback);
00040         void setOnFocusLost(std::function<void()> callback);
00041         void setOnNavigate(std::function<void(bool up)> callback);
00042         void setMaxLength(size_t maxLength);
00043
00044         virtual void handleInput(const math::Vector2f& mousePos, bool mousePressed) override;
00045         void handleKeyboardInput(gfx::EventType event);
00046         void handleTextInput(const std::string& text);
00047
00048         virtual void update(float deltaTime) override;
00049
00050     private:
00051         std::string _text;
00052         std::string _placeholder;
00053         size_t _cursorPosition = 0;
00054         float _cursorBlinkTimer = 0.0f;
00055         bool _showCursor = true;
00056         size_t _maxLength = 0;
00057
00058         gfx::color_t _textColor = {0, 0, 0};
00059         gfx::color_t _placeholderColor = {128, 128, 128};
00060         std::string _fontPath = constants::MAIN_FONT;
00061         size_t _baseFontSize = 24;
00062
00063         std::function<void(const std::string&)> _onTextChanged;
00064         std::function<void(const std::string&)> _onSubmit;
00065         std::function<void()> _onFocusLost;
00066         std::function<void(bool)> _onNavigate;
00067
00068         void insertChar(char c);
00069         void deleteChar();
00070         void moveCursorLeft();
00071         void moveCursorRight();
00072         size_t getFontSize() const;
00073         void updateCursorBlink(float deltaTime);
00074
00075         gfx::color_t _normalColor = colors::WHITE;
00076         gfx::color_t _hoveredColor = colors::LIGHT_GRAY;
00077         gfx::color_t _pressedColor = colors::DARK_GRAY;
00078         gfx::color_t _disabledColor = colors::UI_DISABLED;
00079         gfx::color_t _focusedColor = colors::UI_FOCUSED;
00080         gfx::color_t _getCurrentColor() const;
00081 };
00082
00083 }
00084
00085 #endif /* !TEXTINPUT_HPP_ */
```

5.83 ToggleSwitch.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ToggleSwitch
00006 */
00007
00008 #ifndef TOGGLESWITCH_HPP_
00009 #define TOGGLESWITCH_HPP_
00010
00011 #include "../../core/AFocusableElement.hpp"
00012 #include <string>
00013 #include <functional>
00014 #include "../../../../common/interfaces/IWindow.hpp"
00015 #include "../../../constants.hpp"
00016 #include "../../../colors.hpp"
00017
00018 namespace ui {
00019
```

```
00020 class ToggleSwitch : public AFocusableElement {
00021     public:
00022         explicit ToggleSwitch(std::shared_ptr<ResourceManager> resourceManager);
00023         virtual ~ToggleSwitch() = default;
00024
00025         void setValue(bool value);
00026         bool getValue() const;
00027
00028         void setFontPath(const std::string& fontPath);
00029         void setBaseFontSize(size_t fontSize);
00030         size_t getBaseFontSize() const;
00031
00032         void setOnText(const std::string& text);
00033         void setOffText(const std::string& text);
00034
00035         void setTrackColor(const gfx::color_t& color);
00036         void setHandleColor(const gfx::color_t& color);
00037         void setHandleHoveredColor(const gfx::color_t& color);
00038         void setHandleFocusedColor(const gfx::color_t& color);
00039         void setOnColor(const gfx::color_t& color);
00040         void setOffColor(const gfx::color_t& color);
00041
00042         void setOnValueChanged(std::function<void(bool)> callback);
00043
00044         virtual void render() override;
00045         virtual void handleInput(const math::Vector2f& mousePos, bool mousePressed) override;
00046         virtual bool containsPoint(const math::Vector2f& point) const override;
00047
00048     private:
00049         bool _value = false;
00050         std::string _fontPath = constants::MAIN_FONT;
00051         size_t _baseFontSize = constants::BUTTON_FONT_SIZE_BASE;
00052         std::string _onText = "ON";
00053         std::string _offText = "OFF";
00054
00055         gfx::color_t _trackColor = colors::TOGGLE_TRACK;
00056         gfx::color_t _handleColor = colors::TOGGLE_HANDLE;
00057         gfx::color_t _handleHoveredColor = colors::TOGGLE_HANDLE_HOVER;
00058         gfx::color_t _handleFocusedColor = colors::TOGGLE_HANDLE_FOCUSED;
00059         gfx::color_t _onColor = colors::TOGGLE_ON;
00060         gfx::color_t _offColor = colors::TOGGLE_OFF;
00061
00062         std::function<void(bool)> _onValueChanged;
00063
00064         bool _isHovered = false;
00065 };
00066
00067 }  // namespace ui
00068
00069 #endif  // TOGGLESWITCH_HPP_
```

5.84 Image.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2026
00003 ** R-Type
00004 ** File description:
00005 ** Image UI Element
00006 */
00007
00008 #ifndef IMAGE_HPP_
00009 #define IMAGE_HPP_
00010
00011 #include "base/UIElement.hpp"
00012 #include <string>
00013
00014 namespace ui {
00015
00016 class Image : public UIElement {
00017 public:
00018     Image(std::shared_ptr<ResourceManager> resourceManager);
00019     ~Image() override = default;
00020
00021     void setTexturePath(const std::string& path);
00022
00023     void render() override;
00024     void update(float deltaTime) override;
00025
00026 private:
00027     std::string _texturePath;
00028 };
00029
00030 }  // namespace ui
```

```
00031
00032 #endif  // IMAGE_HPP_
```

5.85 Panel.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** Panel
00006 */
00007
00008 #ifndef PANEL_HPP_
00009 #define PANEL_HPP_
00010
00011 #include "base/UIElement.hpp"
00012 #include "../../../common/interfaces/IWindow.hpp"
00013
00014 namespace ui {
00015
00016 class Panel : public UIElement {
00017     public:
00018         Panel(std::shared_ptr<ResourceManager> resourceManager);
00019         virtual ~Panel() = default;
00020
00021         void setScale(UIScale scale) override;
00022
00023         void setBackgroundColor(gfx::color_t color);
00024         void setBorderColor(gfx::color_t color);
00025         void setBorderThickness(float thickness);
00026
00027         void render() override;
00028         void update(float deltaTime) override;
00029
00030     private:
00031         gfx::color_t _backgroundColor = {50, 50, 50, 255};
00032         gfx::color_t _borderColor = {100, 100, 100, 255};
00033         float _borderThickness = 2.0f;
00034 };
00035
00036 }  // namespace ui
00037
00038 #endif /* !PANEL_HPP_ */
```

5.86 SpritePreview.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** SpritePreview
00006 */
00007
00008 #ifndef SPRITEPREVIEW_HPP_
00009 #define SPRITEPREVIEW_HPP_
00010
00011 #include <memory>
00012 #include <filesystem>
00013 #include <nlohmann/json.hpp>
00014 #include "base/UIElement.hpp"
00015 #include "../../../common/types/Vector2f.hpp"
00016 #include "../../../common/resourceManager/ResourceManager.hpp"
00017
00018 namespace ui {
00019
00020 class SpritePreview : public UIElement {
00021     public:
00022         SpritePreview(std::shared_ptr<ResourceManager> resourceManager);
00023         ~SpritePreview() = default;
00024
00025         bool loadPrefab(const std::filesystem::path& prefabPath);
00026
00027         void setDisplayBounds(const math::Vector2f& position, const math::Vector2f& size);
00028         void setTransform(float scale, float rotation);
00029
00030         void render() override;
00031         void update(float deltaTime) override;
00032
```

```
00033         void clear();
00034
00035     private:
00036         struct SpriteData {
00037             std::string texturePath;
00038             math::Vector2f frameSize;
00039             math::Vector2f position;
00040             math::Vector2f scale;
00041             float rotation = 0.0f;
00042         };
00043
00044         struct AnimationData {
00045             std::string texturePath;
00046             math::Vector2f frameSize;
00047             float frameCount = 0.0f;
00048             float startWidth = 0.0f;
00049             float startHeight = 0.0f;
00050             float speed = 0.1f;
00051             bool loop = true;
00052         };
00053
00054         enum class PreviewType {
00055             None,
00056             Sprite,
00057             Animation
00058         };
00059
00060         PreviewType _type = PreviewType::None;
00061         SpriteData _spriteData;
00062         AnimationData _animationData;
00063         float _currentFrame = 0.0f;
00064         float _animationTime = 0.0f;
00065         bool _loaded = false;
00066
00067         bool extractSpriteFromPrefab(const nlohmann::json& prefab);
00068         bool extractAnimationFromPrefab(const nlohmann::json& prefab);
00069
00070         void renderSprite();
00071         void renderAnimation();
00072 };
00073
00074 }  // namespace ui
00075
00076 #endif /* !SPRITEPREVIEW_HPP_ */
```

5.87 Text.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** Text
00006 */
00007
00008 #ifndef TEXT_HPP_
00009 #define TEXT_HPP_
00010
00011 #include "base/UIElement.hpp"
00012 #include "../../../common/types/Vector2f.hpp"
00013 #include "../../../common/interfaces/IWindow.hpp"
00014 #include "resourceManager/ResourceManager.hpp"
00015 #include <memory>
00016 #include <string>
00017 #include "../../colors.hpp"
00018
00019 namespace ui {
00020
00021 class Text : public UIElement {
00022 public:
00023     Text(std::shared_ptr<ResourceManager> resourceManager);
00024     ~Text() override = default;
00025
00026     void render() override;
00027     void update(float deltaTime) override;
00028     void setScale(UIScale scale) override;
00029
00030     void setText(const std::string& text);
00031     std::string getText() const;
00032
00033     void setTextColor(const gfx::color_t& color);
00034     void setFontSize(unsigned int size);
00035     void setFontPath(const std::string& path);
00036     void setOutlineColor(const gfx::color_t& color);
```

```
00037     void setOutlineThickness(float thickness);
00038
00039 private:
00040     std::string _text;
00041     gfx::color_t _textColor;
00042     unsigned int _fontSize;
00043     unsigned int _baseFontSize;
00044     std::string _fontPath;
00045     gfx::color_t _outlineColor;
00046     float _outlineThickness;
00047 };
00048
00049 }  // namespace ui
00050
00051 #endif  // TEXT_HPP_
```

5.88 UIManager.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** UIManager
00006 */
00007
00008 #ifndef UIMANAGER_HPP_
00009 #define UIMANAGER_HPP_
00010
00011 #include <memory>
00012 #include <vector>
00013 #include <set>
00014 #include "../elements/base/UIElement.hpp"
00015 #include "../navigation/UINavigationManager.hpp"
00016 #include "../../../common/InputMapping/InputAction.hpp"
00017 #include "../../../common/InputMapping/IInputProvider.hpp"
00018 #include "../../../common/types/Vector2f.hpp"
00019 #include "../../../client/constants.hpp"
00020 #include "../../../common/interfaces/IEvent.hpp"
00021
00022 namespace ui {
00023
00024 class UIManager {
00025     public:
00026         UIManager();
00027         ~UIManager() = default;
00028
00029         void addElement(std::shared_ptr<UIElement> element);
00030         void removeElement(std::shared_ptr<UIElement> element);
00031         void clearElements();
00032
00033         void update(float deltaTime);
00034
00035         void render();
00036
00037         void handleMouseInput(const math::Vector2f& mousePos, bool mousePressed);
00038         bool handleNavigationInput(ecs::InputAction action);
00039         bool handleNavigationInputs(std::shared_ptr<ecs::IInputProvider> inputProvider, float
      deltaTime);
00040         void handleKeyboardInput(gfx::EventType event);
00041         void handleTextInput(const std::string& text);
00042
00043         std::shared_ptr<UINavigationManager> getNavigationManager();
00044
00045         void setNavigationEnabled(bool enabled);
00046         bool isNavigationEnabled() const;
00047
00048         bool focusFirstElement();
00049         void clearFocus();
00050
00051         std::shared_ptr<IFocusable> getFocusedElement() const;
00052
00053         void setGlobalScale(UIScale scale);
00054         void cycleGlobalScale();
00055         UIScale getGlobalScale() const;
00056
00057         void setOnBack(std::function<void()> callback);
00058
00059         void setCursorCallback(std::function<void(bool)> callback);
00060
00061         bool isMouseHoveringAnyElement(const math::Vector2f& mousePos) const;
00062
00063         void setResourceManager(std::shared_ptr<ResourceManager> resourceManager);
00064
```

```
00065     private:
00066         std::vector<std::shared_ptr<UIElement» _elements;
00067         std::shared_ptr<UINavigationManager> _navigationManager;
00068         math::Vector2f _lastMousePos;
00069         bool _mouseMovementDetected;
00070
00071         float _navigationCooldown = 0.0f;
00072         UIScale _globalScale = UIScale::Normal;
00073         std::function<void()> _onBack;
00074         std::function<void(bool)> _cursorCallback;
00075         std::set<gfx::EventType> _consumedTextKeys;
00076         std::set<ecs::InputAction> _blockedActions;
00077         std::shared_ptr<ResourceManager> _resourceManager;
00078
00079         bool hasMouseMoved(const math::Vector2f& mousePos);
00080
00081         void refreshNavigationElements();
00082 };
00083
00084 } // namespace ui
00085
00086 #endif /* !UIMANAGER_HPP_ */
```

5.89 UINavigationManager.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** UINavigationManager
00006 */
00007
00008 #ifndef UINAVIGATIONMANAGER_HPP_
00009 #define UINAVIGATIONMANAGER_HPP_
00010
00011 #include <vector>
00012 #include <memory>
00013 #include <functional>
00014 #include "../core/IFocusable.hpp"
00015 #include "../../../common/InputMapping/InputAction.hpp"
00016 #include "../../../common/types/Vector2f.hpp"
00017
00018 namespace ui {
00019
00020 enum class NavigationDirection {
00021     Up,
00022     Down,
00023     Left,
00024     Right
00025 };
00026
00027 class UINavigationManager {
00028     public:
00029         UINavigationManager();
00030         ~UINavigationManager() = default;
00031
00032         void addFocusableElement(std::shared_ptr<IFocusable> element);
00033         void removeFocusableElement(std::shared_ptr<IFocusable> element);
00034         void clearFocusableElements();
00035
00036         bool handleNavigationInput(ecs::InputAction action);
00037
00038         bool setFocus(std::shared_ptr<IFocusable> element);
00039         std::shared_ptr<IFocusable> getFocusedElement() const;
00040         void clearFocus();
00041
00042         bool focusFirstElement();
00043         bool focusNextElement();
00044         bool focusPreviousElement();
00045
00046         void setNavigationEnabled(bool enabled);
00047         bool isNavigationEnabled() const;
00048
00049         void setOnFocusChanged(std::function<void(std::shared_ptr<IFocusable>)> callback);
00050
00051         void onMouseMovement();
00052
00053         void enableFocus();
00054
00055         bool isFocusDisabled() const;
00056
00057     private:
00058         std::vector<std::weak_ptr<IFocusable» _focusableElements;
```

```
00059         std::weak_ptr<IFocusable> _currentFocused;
00060         bool _navigationEnabled;
00061         bool _focusDisabled;
00062         std::function<void(std::shared_ptr<IFocusable>)> _onFocusChanged;
00063
00064         void cleanupExpiredElements();
00065         int getCurrentFocusedIndex() const;
00066         bool navigateInDirection(NavigationDirection direction);
00067 };
00068
00069 }  // namespace ui
00070
00071 #endif /* !UINAVIGATIONMANAGER_HPP_ */
```

5.90 Utils.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** Utils
00006 */
00007 #include "ClientNetwork.hpp"
00008
00009 #ifndef UTILS_HPP_
00010 #define UTILS_HPP_
00011
00012 class Utils {
00013     public:
00014         Utils();
00015         ~Utils();
00016
00017         void helper();
00018         void parseCli(int ac, char **av, std::shared_ptr<ClientNetwork> clientNetwork);
00019
00020     protected:
00021     private:
00022 };
00023
00024 #endif /* !UTILS_HPP_ */
```

5.91 Utils.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** Utils
00006 */
00007
00008 #include <vector>
00009 #include <memory>
00010
00011 #include "ServerConfig.hpp"
00012
00013 #ifndef UTILS_HPP_
00014 #define UTILS_HPP_
00015
00016 class Utils {
00017     public:
00018         Utils();
00019         ~Utils();
00020
00021         void helper();
00022         void parsCli(int ac, char **av, std::shared_ptr<rserv::ServerConfig> config);
00023         static std::string createAlphaNumericCode();
00024     protected:
00025     private:
00026 };
00027
00028 #endif /* !UTILS_HPP_ */
```

5.92 CollisionRules.hpp

```
00001 #ifndef COLLISIONRULES_HPP_
```

```
00002 #define COLLISIONRULES_HPP_
00003
00004 #include <vector>
00005 #include <string>
00006 #include "CollisionRulesData.hpp"
00007 #include "../components/permanent/ColliderComponent.hpp"
00008
00009 namespace ecs {
00010
00011 class CollisionRules {
00012     public:
00013         static const CollisionRules& getInstance();
00014
00015         static void initWithData(const CollisionRulesData& data);
00016
00017         bool canCollide(
00018             CollisionType type,
00019             const std::vector<std::string>& tagsA,
00020             const std::vector<std::string>& tagsB
00021         ) const;
00022
00023     private:
00024         CollisionRules();
00025         ~CollisionRules() = default;
00026         CollisionRules(const CollisionRules&) = delete;
00027         CollisionRules& operator=(const CollisionRules&) = delete;
00028
00029         const std::vector<CollisionRule>& getAllowRules(CollisionType type) const;
00030
00031         std::shared_ptr<std::vector<CollisionRule» _solidAllowRules;
00032         std::shared_ptr<std::vector<CollisionRule» _triggerAllowRules;
00033         std::shared_ptr<std::vector<CollisionRule» _pushAllowRules;
00034
00035         bool entityMatchesGroup(
00036             const std::vector<std::string>& entityTags,
00037             const std::vector<std::string>& group
00038         ) const;
00039         bool ruleMatches(
00040             const CollisionRule& rule,
00041             const std::vector<std::string>& tagsA,
00042             const std::vector<std::string>& tagsB
00043         ) const;
00044 };
00045
00046 }  // namespace ecs
00047
00048 #endif // COLLISIONRULES_HPP_
```

5.93 CollisionRulesData.hpp

```
00001 #ifndef COLLISION_RULES_DATA_HPP_
00002 #define COLLISION_RULES_DATA_HPP_
00003
00004 #include <vector>
00005 #include <string>
00006 #include <memory>
00007
00008 namespace ecs {
00009
00010 struct CollisionRule {
00011     std::vector<std::string> groupA;
00012     std::vector<std::string> groupB;
00013 };
00014
00015 struct CollisionRulesData {
00016     std::shared_ptr<std::vector<CollisionRule» solidAllowRules;
00017     std::shared_ptr<std::vector<CollisionRule» triggerAllowRules;
00018     std::shared_ptr<std::vector<CollisionRule» pushAllowRules;
00019 };
00020
00021 }  // namespace ecs
00022
00023 #endif // COLLISION_RULES_DATA_HPP_
```

5.94 IComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
```

```
00003 ** ryanR-type
00004 ** File description:
00005 ** IComponent
00006 */
00007
00008 #ifndef ICOMPONENT_HPP_
00009 #define ICOMPONENT_HPP_
00010
00011 namespace ecs {
00012
00013 class IComponent {
00014     public:
00015         IComponent() = default;
00016         ~IComponent() = default;
00017 };
00018
00019 } // namespace ecs
00020
00021 #endif /* !ICOMPONENT_HPP_ */
```

5.95 AnimationStateComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2026
00003 ** ryanR-type
00004 ** File description:
00005 ** AnimationStateComponent
00006 */
00007
00008 #ifndef ANIMATIONSTATECOMPONENT_HPP_
00009 #define ANIMATIONSTATECOMPONENT_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012 #include <cstddef>
00013 #include <string>
00014
00015 namespace ecs {
00016
00017 class AnimationStateComponent : public IComponent {
00018     public:
00019         AnimationStateComponent(const std::string& initialState = "")
00020             : _currentState(initialState) {};
00021         ~AnimationStateComponent() = default;
00022
00023         void setCurrentState(const std::string& state) { _currentState = state; }
00024         std::string getCurrentState() const { return _currentState; }
00025
00026     private:
00027         std::string _currentState;
00028 };
00029
00030 }  // namespace ecs
00031
00032
00033 #endif /* !ANIMATIONSTATECOMPONENT_HPP_ */
```

5.96 ChargedShotComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2026
00003 ** ryanR-type
00004 ** File description:
00005 ** ChargedShotComponent
00006 */
00007
00008 #ifndef CHARGEDSHOTCOMPONENT_HPP_
00009 #define CHARGEDSHOTCOMPONENT_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012 #include <algorithm>
00013
00014 namespace ecs {
00015
00016 class ChargedShotComponent : public IComponent {
00017     public:
00018         ChargedShotComponent(
00019             const float &charge = 0,
00020             const float &maxCharge = 0,
```

```
00021             const float &chargeReloadTime = 0
00022         ) : _charge(charge), _maxCharge(maxCharge), _chargeReloadTime(chargeReloadTime) {};
00023         ~ChargedShotComponent() = default;
00024
00025         float getCharge() const { return _charge; };
00026         void setCharge(const float &charge) { _charge = charge; };
00027
00028         float getMaxCharge() const { return _maxCharge; };
00029         void setMaxCharge(const float &maxCharge) { _maxCharge = maxCharge; };
00030
00031         float getReloadTime() const { return _chargeReloadTime; };
00032         void setReloadTime(const float &reloadTime) { _chargeReloadTime = reloadTime; };
00033
00034     private:
00035         float _charge;
00036         float _maxCharge;
00037         float _chargeReloadTime;
00038 };
00039
00040 }
00041
00042 #endif /* !CHARGEDSHOTCOMPONENT_HPP_ */
```

5.97 ColliderComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ColliderComponent
00006 */
00007
00008 #ifndef COLLIDERCOMPONENT_HPP_
00009 #define COLLIDERCOMPONENT_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012 #include "../../types/FRect.hpp"
00013 #include "../../types/Vector2f.hpp"
00014 #include "../../types/OrientedRect.hpp"
00015 #include <cmath>
00016 #include <algorithm>
00017 #include <limits>
00018 #include <vector>
00019
00020 namespace ecs {
00021
00022 enum class CollisionType {
00023     None = 0,
00024     Solid = 1,
00025     Trigger = 2,
00026     Push = 3
00027 };
00028
00029 class ColliderComponent : public IComponent {
00030     public:
00031         ColliderComponent(
00032             math::Vector2f offset = math::Vector2f(0.0f, 0.0f),
00033             math::Vector2f size = math::Vector2f(0.0f, 0.0f),
00034             CollisionType type = CollisionType::Solid
00035         ) : _offset(offset), _size(size), _type(type) {};
00036         ~ColliderComponent() = default;
00037
00038         math::Vector2f getOffset() const;
00039         void setOffset(math::Vector2f offset);
00040
00041         math::Vector2f getSize() const;
00042         void setSize(math::Vector2f size);
00043
00044         CollisionType getType() const;
00045         void setType(CollisionType type);
00046
00047         math::FRect getHitbox(math::Vector2f entityPosition, math::Vector2f scale =
      math::Vector2f(1.0f, 1.0f)) const;
00048
00049         math::FRect getScaledHitbox(math::Vector2f entityPosition, math::Vector2f scale) const;
00050
00051         math::OrientedRect getOrientedHitbox(math::Vector2f entityPosition, math::Vector2f scale,
      float rotation) const;
00052
00053         math::FRect getHitbox(math::Vector2f entityPosition, math::Vector2f scale, float rotation)
      const;
00054
00055     private:
```

```
00056         math::Vector2f _offset;
00057         math::Vector2f _size;
00058         CollisionType _type;
00059 };
00060
00061 }  // namespace ecs
00062
00063 #endif /* !COLLIDERCOMPONENT_HPP_ */
```

5.98 CompositeEntityComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** CompositeEntityComponent
00006 */
00007
00008 #ifndef COMPOSITEENTITYCOMPONENT_HPP_
00009 #define COMPOSITEENTITYCOMPONENT_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012 #include <cstddef>
00013
00014 namespace ecs {
00015
00016 class CompositeEntityComponent : public IComponent {
00017     public:
00018         CompositeEntityComponent(size_t parent_id) : parentId(parent_id) {};
00019         ~CompositeEntityComponent() = default;
00020
00021         size_t getParentId() const { return parentId; }
00022         void setParentId(size_t id) { parentId = id; }
00023
00024     private:
00025         size_t parentId;
00026 };
00027
00028 }  // namespace ecs
00029
00030 #endif /* !COMPOSITEENTITYCOMPONENT_HPP_ */
```

5.99 DamageComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** DamageComponent
00006 */
00007
00008 #ifndef DAMAGECOMPONENT_HPP_
00009 #define DAMAGECOMPONENT_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012
00013 namespace ecs {
00014
00015 class DamageComponent : public IComponent {
00016     public:
00017         DamageComponent(float damage = 0.0f) : _damage(damage) {};
00018         ~DamageComponent() = default;
00019
00020         float getDamage() const { return _damage; }
00021         void setDamage(float damage) { _damage = damage; }
00022
00023     private:
00024         float _damage;
00025 };
00026
00027 }  // namespace ecs
00028
00029 #endif /* !DAMAGECOMPONENT_HPP_ */
```

5.100 DamageCooldownComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** DamageCooldownComponent
00006 */
00007
00008 #ifndef DAMAGECOOLDOWNCOMPONENT_HPP_
00009 #define DAMAGECOOLDOWNCOMPONENT_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012 #include "../../constants.hpp"
00013
00014 namespace ecs {
00015
00016 class DamageCooldownComponent : public IComponent {
00017     public:
00018         DamageCooldownComponent(float cooldown = constants::TRIGGER_DAMAGE_COOLDOWN)
00019             : _cooldown(cooldown), _lastDamageTime(0.0f) {};
00020         ~DamageCooldownComponent() = default;
00021
00022         float getCooldown() const { return _cooldown; }
00023         void setCooldown(float cooldown) { _cooldown = cooldown; }
00024
00025         float getLastDamageTime() const { return _lastDamageTime; }
00026         void setLastDamageTime(float time) { _lastDamageTime = time; }
00027
00028     private:
00029         float _cooldown;
00030         float _lastDamageTime;
00031 };
00032
00033 }
00034
00035 #endif /* !DAMAGECOOLDOWNCOMPONENT_HPP_ */
```

5.101 EntityPartsComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** EntityPartsComponent
00006 */
00007
00008 #ifndef ENTITYPARTSCOMPONENT_HPP_
00009 #define ENTITYPARTSCOMPONENT_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012 #include <vector>
00013
00014 namespace ecs {
00015
00016 class EntityPartsComponent : public IComponent {
00017     public:
00018         EntityPartsComponent() = default;
00019         ~EntityPartsComponent() = default;
00020
00021         std::vector<size_t> partIds;
00022 };
00023
00024 }  // namespace ecs
00025
00026 #endif /* !ENTITYPARTSCOMPONENT_HPP_ */
```

5.102 GameZoneComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** GameZoneComponent
00006 */
00007
00008 #ifndef GAMEZONECOMPONENT_HPP_
```

```
00009 #define GAMEZONECOMPONENT_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012 #include "../../types/FRect.hpp"
00013 #include "../../constants.hpp"
00014
00015 namespace ecs {
00016
00017 class GameZoneComponent : public IComponent {
00018     public:
00019         GameZoneComponent(math::FRect zone = math::FRect(0.0f, 0.0f, constants::MAX_WIDTH,
      constants::MAX_HEIGHT))
00020             : _zone(zone) {};
00021         ~GameZoneComponent() = default;
00022
00023         math::FRect getZone() const { return _zone; };
00024         void setZone(math::FRect zone) { _zone = zone; };
00025
00026     private:
00027         math::FRect _zone;
00028 };
00029
00030 }  // namespace ecs
00031
00032 #endif /* !GAMEZONECOMPONENT_HPP_ */
```

5.103 HealthComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** HealthComponent
00006 */
00007
00008 #ifndef HEALTHCOMPONENT_HPP_
00009 #define HEALTHCOMPONENT_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012 #include "../../ECS/entity/Entity.hpp"
00013
00014 namespace ecs {
00015
00016 class HealthComponent : public IComponent {
00017     public:
00018         HealthComponent(float health = 100) : _health(health), _baseHealth(health),
      _lastDamageSource(0) {};
00019         ~HealthComponent() = default;
00020
00021         float getHealth() const { return _health; }
00022         void setHealth(float health) { _health = health; };
00023
00024         float getBaseHealth() const { return _baseHealth; };
00025         void setBaseHealth(float health) { _baseHealth = health; };
00026
00027         ecs::Entity getLastDamageSource() const { return _lastDamageSource; }
00028         void setLastDamageSource(ecs::Entity source) { _lastDamageSource = source; }
00029
00030     private:
00031         float _health;
00032         float _baseHealth;
00033         ecs::Entity _lastDamageSource;
00034 };
00035
00036 }
00037
00038 #endif /* !HEALTHCOMPONENT_HPP_ */
```

5.104 InteractionConfigComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** InteractionConfigComponent
00006 */
00007
00008 #ifndef INTERACTIONCONFIGCOMPONENT_HPP_
```

```
00009 #define INTERACTIONCONFIGCOMPONENT_HPP_
00010
00011 #include <vector>
00012 #include <string>
00013 #include "../base/IComponent.hpp"
00014
00015 namespace ecs {
00016
00017 struct InteractionMapping {
00018     std::vector<std::string> targetTags;
00019     std::vector<std::string> actionsToOther;
00020     std::vector<std::string> actionsToSelf;
00021 };
00022
00023 class InteractionConfigComponent : public IComponent {
00024     public:
00025         InteractionConfigComponent(){
00026             _mappings = std::vector<InteractionMapping>();
00027         };
00028         InteractionConfigComponent(const std::vector<InteractionMapping>& mappings)
00029             : _mappings(mappings) {}
00030         ~InteractionConfigComponent() = default;
00031
00032         const std::vector<InteractionMapping>& getMappings() const { return _mappings; }
00033         void setMappings(const std::vector<InteractionMapping>& mappings) { _mappings = mappings; }
00034         void addMapping(const InteractionMapping& mapping) { _mappings.push_back(mapping); }
00035
00036     private:
00037         std::vector<InteractionMapping> _mappings;
00038 };
00039
00040 } // namespace ecs
00041
00042 #endif /* !INTERACTIONCONFIGCOMPONENT_HPP_ */
```

5.105 InvulnerableComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2026
00003 ** ryanR-type
00004 ** File description:
00005 ** InvulnerableComponent
00006 */
00007
00008 #ifndef INVULNERABLECOMPONENT_HPP_
00009 #define INVULNERABLECOMPONENT_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012
00013 namespace ecs {
00014
00015 class InvulnerableComponent : public IComponent {
00016     public:
00017         InvulnerableComponent(bool active = false) : _active(active) {};
00018         ~InvulnerableComponent()  = default;
00019
00020         bool isActive() const { return _active; };
00021         void setActive(bool active) { _active = active; };
00022
00023     private:
00024         bool _active;
00025 };
00026
00027 }  // namespace ecs
00028
00029 #endif /* !INVULNERABLECOMPONENT_HPP_ */
```

5.106 LifetimeComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** LifetimeComponent
00006 */
00007
00008 #ifndef LIFETIMECOMPONENT_HPP_
00009 #define LIFETIMECOMPONENT_HPP_
```

```
00010
00011 #include "../base/IComponent.hpp"
00012
00013 namespace ecs {
00014
00015 class LifetimeComponent : public IComponent {
00016     public:
00017         LifetimeComponent(float lifetime = 0.0f) : _lifetime(lifetime) {};
00018         ~LifetimeComponent() = default;
00019
00020         float getLifetime() const { return _lifetime; };
00021         void setLifetime(float lifetime) { _lifetime = lifetime; };
00022
00023     private:
00024         float _lifetime;
00025 };
00026
00027 } // ecs
00028
00029
00030 #endif /* !LIFETIMECOMPONENT_HPP_ */
```

5.107 OwnerComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** OwnerComponent
00006 */
00007
00008 #ifndef OWNERCOMPONENT_HPP_
00009 #define OWNERCOMPONENT_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012 #include "../ECS/entity/Entity.hpp"
00013
00014 namespace ecs {
00015
00016 class OwnerComponent : public IComponent {
00017     public:
00018         OwnerComponent(Entity owner = 0) : _owner(owner) {};
00019         ~OwnerComponent() = default;
00020
00021         Entity getOwner() const { return _owner; }
00022         void setOwner(Entity owner) { _owner = owner; }
00023
00024     private:
00025         Entity _owner;
00026 };
00027
00028 }  // namespace ecs
00029
00030 #endif /* !OWNERCOMPONENT_HPP_ */
```

5.108 ProjectilePrefabComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ProjectilePrefabComponent
00006 */
00007
00008 #ifndef PROJECTILEPREFABCOMPONENT_HPP_
00009 #define PROJECTILEPREFABCOMPONENT_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012 #include <string>
00013
00014 namespace ecs {
00015
00016 class ProjectilePrefabComponent : public IComponent {
00017     public:
00018         ProjectilePrefabComponent(const std::string &prefabName = "")
00019             : _prefabName(prefabName) {};
00020         ~ProjectilePrefabComponent() = default;
00021
```

```
00022         std::string getPrefabName() const { return _prefabName; };
00023         void setPrefabName(const std::string &prefabName) { _prefabName = prefabName; };
00024
00025     private:
00026         std::string _prefabName;
00027 };
00028
00029 }
00030
00031 #endif /* !PROJECTILEPREFABCOMPONENT_HPP_ */
```

5.109 ScoreComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ScoreComponent
00006 */
00007
00008 #ifndef SCORECOMPONENT_HPP_
00009 #define SCORECOMPONENT_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012
00013 namespace ecs {
00014
00015 class ScoreComponent : public IComponent {
00016     public:
00017         ScoreComponent(int score = 0) : _score(score) {};
00018         ~ScoreComponent() = default;
00019
00020         int getScore() const { return _score; }
00021         void setScore(int score) { _score = score; }
00022
00023     private:
00024         int _score;
00025 };
00026
00027 }  // namespace ecs
00028
00029 #endif /* !SCORECOMPONENT_HPP_ */
```

5.110 ScoreValueComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ScoreValueComponent
00006 */
00007
00008 #ifndef SCOREVALUECOMPONENT_HPP_
00009 #define SCOREVALUECOMPONENT_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012
00013 namespace ecs {
00014
00015 class ScoreValueComponent : public IComponent {
00016     public:
00017         ScoreValueComponent(int scoreValue = 0) : _scoreValue(scoreValue) {};
00018         ~ScoreValueComponent() = default;
00019
00020         int getScoreValue() const { return _scoreValue; }
00021         void setScoreValue(int scoreValue) { _scoreValue = scoreValue; }
00022
00023     private:
00024         int _scoreValue;
00025 };
00026
00027 }  // namespace ecs
00028
00029 #endif /* !SCOREVALUECOMPONENT_HPP_ */
```

5.111 ScriptingComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2026
00003 ** ryanR-type
00004 ** File description:
00005 ** ScriptingComponent
00006 */
00007
00008 #ifndef SCRIPTINGCOMPONENT_HPP_
00009 #define SCRIPTINGCOMPONENT_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012 #include <string>
00013 #include <map>
00014 #include "../../Error/ScriptingError.hpp"
00015 #include "../../constants.hpp"
00016
00017 // To suppress warnings from sol2 includes
00018 #ifdef __GNUC__
00019 #pragma GCC diagnostic push
00020 #pragma GCC diagnostic ignored "-Wsign-conversion"
00021 #pragma GCC diagnostic ignored "-Wconversion"
00022 #pragma GCC diagnostic ignored "-Wfloat-equal"
00023 #endif
00024 #ifdef _MSC_VER
00025 #pragma warning(push)
00026 #pragma warning(disable: 5321)
00027 #endif
00028 #include <sol/sol.hpp>
00029 #ifdef _MSC_VER
00030 #pragma warning(pop)
00031 #endif
00032 #ifdef __GNUC__
00033 #pragma GCC diagnostic pop
00034 #endif
00035
00036 namespace ecs {
00037
00038 class ScriptingComponent : public IComponent {
00039     public:
00040         ScriptingComponent(
00041             std::string script_name = "",
00042             std::vector<std::string> additionalFunctions = std::vector<std::string>(),
00043             std::shared_ptr<sol::state> lua = nullptr,
00044             size_t entityId = 0
00045         ) : _scriptName(script_name), _additionalFunctions(additionalFunctions), _initialized(false) {
00046             if (lua != nullptr) {
00047                 init(*lua, entityId);
00048             }
00049         };
00050         ~ScriptingComponent() = default;
00051
00052         void init(sol::state& lua, size_t entityId);
00053
00054         const std::string& getScriptName() const;
00055
00056         void setEnvironment(const sol::table& table) { _env = table; };
00057         sol::table getEnvironment() const { return _env; };
00058
00059         bool hasFunction(const std::string& name) const { return _functions.find(name) !=
    _functions.end(); };
00060         sol::function getFunction(const std::string& name) const { return _functions.at(name); };
00061         void addFunction(const std::string& name, const sol::function& function) { _functions[name] =
    function; };
00062         void removeFunction(const std::string& name) { _functions.erase(name); };
00063
00064         std::vector<std::string> getFunctionNames() const {
00065             std::vector<std::string> names;
00066             for (const auto& pair : _functions) {
00067                 names.push_back(pair.first);
00068             }
00069             return names;
00070         }
00071
00072         bool isInitialized() const { return _initialized; };
00073         void setInitialized(bool value) { _initialized = value; };
00074
00075         void clearLuaReferences() {
00076             _functions.clear();
00077             _env = sol::lua_nil;
00078             _initialized = false;
00079         }
00080
00081     protected:
00082     private:
```

```
00083         std::string _scriptName;
00084         std::vector<std::string> _additionalFunctions;
00085         sol::table _env;
00086         std::map<std::string, sol::function> _functions;
00087         bool _initialized = false;
00088 };
00089
00090 } // namespace ecs
00091
00092 #endif /* !SCRIPTINGCOMPONENT_HPP_ */
```

5.112 ShootingStatsComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ShootingStatsComponent
00006 */
00007
00008 #ifndef SHOOTINGSTATSCOMPONENT_HPP_
00009 #define SHOOTINGSTATSCOMPONENT_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012 #include "../../Prefab/IPrefab.hpp"
00013 #include <memory>
00014 #include <vector>
00015
00016 namespace ecs {
00017
00018 struct MultiShotPattern {
00019     int shotCount = 1;
00020     float angleSpread = 0.0f;
00021     float offsetDistance = 0.0f;
00022     float angleOffset = 0.0f;
00023 };
00024
00025 class ShootingStatsComponent : public IComponent {
00026     public:
00027         ShootingStatsComponent(
00028             float fireRate = 1.0f,
00029             const MultiShotPattern &pattern = MultiShotPattern()
00030         ) : _fireRate(fireRate),
00031             _multiShotPattern(pattern),
00032             _cooldownTimer(0.0f) {};
00033         ~ShootingStatsComponent() = default;
00034
00035         float getFireRate() const { return _fireRate; };
00036         void setFireRate(float fireRate) { _fireRate = fireRate; };
00037
00038         MultiShotPattern getMultiShotPattern() const { return _multiShotPattern; };
00039         void setMultiShotPattern(const MultiShotPattern &pattern) { _multiShotPattern = pattern; };
00040
00041         float getCooldownTimer() const { return _cooldownTimer; };
00042         void setCooldownTimer(float timer) { _cooldownTimer = timer; };
00043
00044     private:
00045         float _fireRate;
00046         MultiShotPattern _multiShotPattern;
00047         float _cooldownTimer;
00048 };
00049
00050 }  // namespace ecs
00051
00052 #endif /* !SHOOTINGSTATSCOMPONENT_HPP_ */
```

5.113 SpeedComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** SpeedComponent
00006 */
00007
00008 #ifndef SPEEDCOMPONENT_HPP_
00009 #define SPEEDCOMPONENT_HPP_
00010
```

```
00011 #include "../base/IComponent.hpp"
00012 #include "constants.hpp"
00013
00014 namespace ecs {
00015
00016 class SpeedComponent : public IComponent {
00017     public:
00018         SpeedComponent(float speed = constants::BASE_SPEED) : _speed(speed) {};
00019         ~SpeedComponent() = default;
00020
00021         float getSpeed() const { return _speed; };
00022         void setSpeed(float speed) { _speed = speed; };
00023
00024     private:
00025         float _speed;
00026 };
00027
00028 }  // namespace ecs
00029
00030 #endif /* !SPEEDCOMPONENT_HPP_ */
```

5.114 TransformComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** TransformComponent
00006 */
00007
00008 #ifndef TRANSFORMCOMPONENT_HPP_
00009 #define TRANSFORMCOMPONENT_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012 #include "../../types/Vector2f.hpp"
00013
00014 namespace ecs {
00015
00016 class TransformComponent : public IComponent {
00017     public:
00018         TransformComponent(math::Vector2f position = math::Vector2f(0.0f, 0.0f), float rotation =
      0.0f, math::Vector2f scale = math::Vector2f(1.0f, 1.0f))
00019             : _position(position), _rotation(rotation), _scale(scale) {};
00020         ~TransformComponent() = default;
00021
00022         math::Vector2f getPosition() const { return _position; };
00023         void setPosition(math::Vector2f position) { _position = position; };
00024
00025         float getRotation() const { return _rotation; };
00026         void setRotation(float rotation) { _rotation = rotation; };
00027
00028         math::Vector2f getScale() const { return _scale; };
00029         void setScale(math::Vector2f scale) { _scale = scale; };
00030
00031     private:
00032         math::Vector2f _position;
00033         float _rotation;
00034         math::Vector2f _scale;
00035 };
00036
00037 } // namespace ecs
00038
00039 #endif /* !TRANSFORMCOMPONENT_HPP_ */
```

5.115 VelocityComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** VelocityComponent
00006 */
00007
00008 #ifndef VELOCITYCOMPONENT_HPP_
00009 #define VELOCITYCOMPONENT_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012 #include "../../types/Vector2f.hpp"
```

```
00013
00014 namespace ecs {
00015
00016 class VelocityComponent : public IComponent {
00017     public:
00018         VelocityComponent(math::Vector2f velocity = math::Vector2f(0.0f, 0.0f)) : _velocity(velocity)
      {};
00019         ~VelocityComponent() = default;
00020
00021         math::Vector2f getVelocity() const { return _velocity; };
00022         void setVelocity(math::Vector2f velocity) { _velocity = velocity; };
00023
00024     private:
00025         math::Vector2f _velocity;
00026 };
00027
00028 } // namespace ecs
00029
00030 #endif /* !VELOCITYCOMPONENT_HPP_ */
```

5.116 ClientEffectTag.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ClientEffectTag
00006 */
00007
00008 #ifndef CLIENTEFFECTTAG_HPP_
00009 #define CLIENTEFFECTTAG_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012
00013 namespace ecs {
00014
00015 class ClientEffectTag : public IComponent {
00016     public:
00017         ClientEffectTag() = default;
00018         ~ClientEffectTag() = default;
00019 };
00020
00021 }
00022
00023 #endif /* !CLIENTEFFECTTAG_HPP_ */
```

5.117 ControllableTag.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ControllableTag
00006 */
00007
00008 #ifndef CONTROLLABLETAG_HPP_
00009 #define CONTROLLABLETAG_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012
00013 namespace ecs {
00014
00015 class ControllableTag : public IComponent {
00016     public:
00017         ControllableTag() = default;
00018         ~ControllableTag() = default;
00019 };
00020
00021 } // namespace ecs
00022
00023 #endif /* !CONTROLLABLETAG_HPP_ */
```

5.118 EnnemyProjectileTag.hpp

```
00001 /*
```

```
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** EnnemyProjectileTag
00006 */
00007
00008 #ifndef ENNEMYPROJECTILETAG_HPP_
00009 #define ENNEMYPROJECTILETAG_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012
00013 namespace ecs {
00014
00015 class EnnemyProjectileTag : public IComponent {
00016     public:
00017         EnnemyProjectileTag() = default;
00018         ~EnnemyProjectileTag() = default;
00019 };
00020
00021 } // namespace ecs
00022
00023 #endif /* !ENNEMYPROJECTILETAG_HPP_ */
```

5.119 ForceTag.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2026
00003 ** ryanR-type
00004 ** File description:
00005 ** ForceTag
00006 */
00007
00008 #ifndef FORCETAG_HPP_
00009 #define FORCETAG_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012 #include <string>
00013
00014 namespace ecs {
00015
00016 class ForceTag : public IComponent {
00017     public:
00018         ForceTag(const std::string& type = "") : forceType(type) {}
00019         ~ForceTag() = default;
00020
00021         std::string getForceType() const { return forceType; }
00022         void setForceType(const std::string& type) { forceType = type; }
00023
00024     private:
00025         std::string forceType;
00026 };
00027
00028 }  // namespace ecs
00029
00030 #endif /* !FORCETAG_HPP_ */
```

5.120 GameEndTag.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** GameEndTag
00006 */
00007
00008 #ifndef GAMEENDTAG_HPP_
00009 #define GAMEENDTAG_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012
00013 namespace ecs {
00014
00015 class GameEndTag : public IComponent {
00016     public:
00017         GameEndTag() = default;
00018         ~GameEndTag() = default;
00019 };
00020
```

```
00021 }  // namespace ecs
00022
00023 #endif /* !GAMEENDTAG_HPP_ */
```

5.121 GameZoneColliderTag.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** GameZoneColliderTag
00006 */
00007
00008 #ifndef GAMEZONECOLLIDERTAG_HPP_
00009 #define GAMEZONECOLLIDERTAG_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012
00013 namespace ecs {
00014
00015 class GameZoneColliderTag : public IComponent {
00016     public:
00017         GameZoneColliderTag() = default;
00018         ~GameZoneColliderTag() = default;
00019 };
00020
00021 } // namespace ecs
00022
00023 #endif /* !GAMEZONECOLLIDERTAG_HPP_ */
```

5.122 GameZoneStopTag.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** GameZoneStopTag
00006 */
00007
00008 #ifndef GAMEZONESTOPTAG_HPP_
00009 #define GAMEZONESTOPTAG_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012
00013 namespace ecs {
00014
00015 class GameZoneStopTag : public IComponent {
00016     public:
00017         GameZoneStopTag() = default;
00018         ~GameZoneStopTag() = default;
00019 };
00020
00021 } // namespace ecs
00022
00023 #endif /* !GAMEZONESTOPTAG_HPP_ */
```

5.123 LocalPlayerTag.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** LocalPlayerTag
00006 */
00007
00008 #ifndef LOCALPLAYERTAG_HPP_
00009 #define LOCALPLAYERTAG_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012
00013 namespace ecs {
00014
00015 class LocalPlayerTag : public IComponent {
```

```
00016     public:
00017         LocalPlayerTag() = default;
00018         ~LocalPlayerTag() = default;
00019 };
00020
00021 }  // namespace ecs
00022
00023 #endif /* !LOCALPLAYERTAG_HPP_ */
```

5.124 MobTag.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** MobTag
00006 */
00007
00008 #ifndef MOBTAG_HPP_
00009 #define MOBTAG_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012
00013 namespace ecs {
00014
00015 class MobTag : public IComponent {
00016     public:
00017         MobTag() = default;
00018         ~MobTag() = default;
00019 };
00020
00021 } // namespace ecs
00022
00023 #endif /* !MOBTAG_HPP_ */
```

5.125 ObstacleTag.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ObstacleTag
00006 */
00007
00008 #ifndef OBSTACLETAG_HPP_
00009 #define OBSTACLETAG_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012
00013 namespace ecs {
00014
00015 class ObstacleTag : public IComponent {
00016     public:
00017         ObstacleTag() = default;
00018         ~ObstacleTag() = default;
00019 };
00020
00021 } // namespace ecs
00022
00023 #endif /* !OBSTACLETAG_HPP_ */
```

5.126 PlayerObstacleTag.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2026
00003 ** ryanR-type
00004 ** File description:
00005 ** PlayerObstacleTag
00006 */
00007
00008 #ifndef PLAYEROBSTACLETAG_HPP_
00009 #define PLAYEROBSTACLETAG_HPP_
00010
```

```
00011 #include "../base/IComponent.hpp"
00012
00013 namespace ecs {
00014
00015 class PlayerObstacleTag : public IComponent {
00016     public:
00017         PlayerObstacleTag() = default;
00018         ~PlayerObstacleTag() = default;
00019 };
00020
00021 } // namespace ecs
00022
00023 #endif /* !PLAYEROBSTACLETAG_HPP_ */
```

5.127 PlayerProjectileTag.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** PlayerProjectileTag
00006 */
00007
00008 #ifndef PLAYERPROJECTILETAG_HPP_
00009 #define PLAYERPROJECTILETAG_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012
00013 namespace ecs {
00014
00015 class PlayerProjectileTag : public IComponent {
00016     public:
00017         PlayerProjectileTag() = default;
00018         ~PlayerProjectileTag() = default;
00019 };
00020
00021 } // namespace ecs
00022
00023 #endif /* !PLAYERPROJECTILETAG_HPP_ */
```

5.128 PlayerTag.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** PlayerTag
00006 */
00007
00008 #ifndef PLAYERTAG_HPP_
00009 #define PLAYERTAG_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012
00013 namespace ecs {
00014
00015 class PlayerTag : public IComponent {
00016     public:
00017         PlayerTag() = default;
00018         ~PlayerTag() = default;
00019 };
00020
00021 } // namespace ecs
00022
00023 #endif /* !PLAYERTAG_HPP_ */
```

5.129 PowerUpTag.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** PowerUpTag
```

```
00006 */
00007
00008 #ifndef POWERUPTAG_HPP_
00009 #define POWERUPTAG_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012
00013 namespace ecs {
00014
00015 class PowerUpTag : public IComponent {
00016     public:
00017         PowerUpTag() = default;
00018         ~PowerUpTag() = default;
00019 };
00020
00021 }  // namespace ecs
00022
00023 #endif /* !POWERUPTAG_HPP_ */
```

5.130 ProjectilePassThroughTag.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ProjectilePassThroughTag
00006 */
00007
00008 #ifndef PROJECTILEPASSTHROUGHTAG_HPP_
00009 #define PROJECTILEPASSTHROUGHTAG_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012
00013 namespace ecs {
00014
00015 class ProjectilePassThroughTag : public IComponent {
00016     public:
00017         ProjectilePassThroughTag() = default;
00018         ~ProjectilePassThroughTag() = default;
00019 };
00020
00021 } // namespace ecs
00022
00023 #endif /* !PROJECTILEPASSTHROUGHTAG_HPP_ */
```

5.131 ShooterTag.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ShooterTag
00006 */
00007
00008 #ifndef SHOOTERTAG_HPP_
00009 #define SHOOTERTAG_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012
00013 namespace ecs {
00014
00015 class ShooterTag : public IComponent {
00016     public:
00017         ShooterTag() = default;
00018         ~ShooterTag() = default;
00019 };
00020
00021 }
00022
00023 #endif /* !SHOOTERTAG_HPP_ */
```

5.132 DamageIntentComponent.hpp

```
00001 /*
```

```
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** DamageIntentComponent
00006 */
00007
00008 #ifndef DAMAGEINTENTCOMPONENT_HPP_
00009 #define DAMAGEINTENTCOMPONENT_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012 #include "../../ECS/entity/Entity.hpp"
00013
00014 namespace ecs {
00015
00016 class DamageIntentComponent : public IComponent {
00017     public:
00018         DamageIntentComponent(float damages = 0.0f, ecs::Entity source = 0) : _damages(damages),
      _source(source) {};
00019         ~DamageIntentComponent() = default;
00020
00021         float getDamages() { return _damages; };
00022         void setDamages(float damages) { _damages = damages; };
00023
00024         ecs::Entity getSource() const { return _source; };
00025         void setSource(ecs::Entity source) { _source = source; };
00026
00027     private:
00028         float _damages;
00029         ecs::Entity _source;
00030 };
00031
00032 }  // namespace ecs
00033
00034 #endif /* !DAMAGEINTENTCOMPONENT_HPP_ */
```

5.133 DeathIntentComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** DeathIntentComponent
00006 */
00007
00008 #ifndef DEATHINTENTCOMPONENT_HPP_
00009 #define DEATHINTENTCOMPONENT_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012 #include "../../ECS/entity/Entity.hpp"
00013
00014 namespace ecs {
00015
00016 class DeathIntentComponent : public IComponent {
00017     public:
00018         DeathIntentComponent(ecs::Entity source = 0) : _source(source) {};
00019         ~DeathIntentComponent() = default;
00020
00021         ecs::Entity getSource() const { return _source; }
00022         void setSource(ecs::Entity source) { _source = source; }
00023
00024     private:
00025         ecs::Entity _source;
00026 };
00027
00028 }
00029
00030 #endif /* !DEATHINTENTCOMPONENT_HPP_ */
```

5.134 InputIntentComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** InputIntentComponent
00006 */
00007
00008 #ifndef INPUTINTENTCOMPONENT_HPP_
```

```
00009 #define INPUTINTENTCOMPONENT_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012 #include "../../types/Vector2f.hpp"
00013
00014 namespace ecs {
00015
00016 class InputIntentComponent : public IComponent {
00017     public:
00018         InputIntentComponent(const math::Vector2f &direction = math::Vector2f(0.0f, 0.0f))
00019             : _direction(direction) {
00020         };
00021         ~InputIntentComponent() = default;
00022
00023         math::Vector2f getDirection() const { return _direction; };
00024         void setDirection(const math::Vector2f &direction) { _direction = direction; };
00025
00026     private:
00027         math::Vector2f _direction;
00028 };
00029
00030 }  // namespace ecs
00031
00032 #endif /* !INPUTINTENTCOMPONENT_HPP_ */
```

5.135 MovementIntentComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** MovementIntentComponent
00006 */
00007
00008 #ifndef MOVEMENTINTENTCOMPONENT_HPP_
00009 #define MOVEMENTINTENTCOMPONENT_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012 #include "../../types/Vector2f.hpp"
00013
00014 namespace ecs {
00015
00016 class MovementIntentComponent : public IComponent {
00017     public:
00018         MovementIntentComponent(const math::Vector2f &direction = math::Vector2f(0.0f, 0.0f), bool
      active = false)
00019             : _direction(direction), _active(active) {
00020         };
00021         ~MovementIntentComponent() = default;
00022
00023         math::Vector2f getDirection() const { return _direction; };
00024         void setDirection(const math::Vector2f &direction) { _direction = direction; };
00025
00026         bool isActive() const { return _active; };
00027         void setActive(bool active) { _active = active; };
00028
00029     private:
00030         math::Vector2f _direction;
00031         bool _active;
00032 };
00033
00034 }  // namespace ecs
00035
00036 #endif /* !MOVEMENTINTENTCOMPONENT_HPP_ */
```

5.136 ScoreIntentComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ScoreIntentComponent
00006 */
00007
00008 #ifndef SCOREINTENTCOMPONENT_HPP_
00009 #define SCOREINTENTCOMPONENT_HPP_
00010
00011 class ScoreIntentComponent {
```

```
00012     public:
00013         ScoreIntentComponent(int score = 0) : _score(score) {};
00014         ~ScoreIntentComponent() = default;
00015
00016         int getScore() const { return _score; }
00017         void setScore(int newScore) { _score = newScore; }
00018     protected:
00019     private:
00020         int _score;
00021 };
00022
00023 #endif /* !SCOREINTENTCOMPONENT_HPP_ */
```

5.137 ShootIntentComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ShootIntentComponent
00006 */
00007
00008 #ifndef SHOOTINTENTCOMPONENT_HPP_
00009 #define SHOOTINTENTCOMPONENT_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012 #include "../../types/Vector2f.hpp"
00013
00014 namespace ecs {
00015
00016 class ShootIntentComponent : public IComponent {
00017     public:
00018         ShootIntentComponent(float angle = 0.0f) : _angle(angle) {
00019             _position = math::Vector2f(0.0f, 0.0f);
00020         }
00021         ~ShootIntentComponent() = default;
00022
00023         void setAngle(float angle) { _angle = angle; }
00024         float getAngle() const { return _angle; }
00025
00026     private:
00027         float _angle;
00028         math::Vector2f _position;
00029 };
00030
00031 }  // namespace ecs
00032
00033 #endif /* !SHOOTINTENTCOMPONENT_HPP_ */
```

5.138 SpawnIntentComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** SpawnIntentComponent
00006 */
00007
00008 #ifndef SPAWNINTENTCOMPONENT_HPP_
00009 #define SPAWNINTENTCOMPONENT_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012 #include <string>
00013 #include <optional>
00014 #include "../../types/Vector2f.hpp"
00015 #include "../../ECS/entity/EntityCreationContext.hpp"
00016
00017 namespace ecs {
00018
00019 class SpawnIntentComponent : public IComponent {
00020     public:
00021         SpawnIntentComponent(
00022             const std::string &prefabName,
00023             const math::Vector2f &position,
00024             float gameViewXTrigger = 0.0f
00025         ) : _prefabName(prefabName),
00026             _position(position),
00027             _creationContext(EntityCreationContext::forLocalClient()),
```

```
00028             _gameViewXTrigger(gameViewXTrigger) {}
00029
00030         SpawnIntentComponent(
00031             const std::string &prefabName,
00032             const math::Vector2f &position,
00033             const EntityCreationContext &context,
00034             float gameViewXTrigger = 0.0f
00035         ) : _prefabName(prefabName),
00036             _position(position),
00037             _creationContext(context),
00038             _gameViewXTrigger(gameViewXTrigger) {}
00039
00040         ~SpawnIntentComponent() = default;
00041
00042         void setPrefabName(const std::string &prefabName) { _prefabName = prefabName; }
00043         std::string getPrefabName() const { return _prefabName; }
00044
00045         void setPosition(const math::Vector2f &position) { _position = position; }
00046         math::Vector2f getPosition() const { return _position; }
00047
00048         void setCreationContext(const EntityCreationContext &context) {
00049             _creationContext = context;
00050         }
00051         EntityCreationContext getCreationContext() const { return _creationContext; }
00052
00053         void setGameViewXTrigger(const float &gameViewXTrigger) { _gameViewXTrigger =
      gameViewXTrigger; };
00054         float getGameViewXTrigger() const { return _gameViewXTrigger; };
00055
00056     private:
00057         std::string _prefabName;
00058         math::Vector2f _position;
00059         EntityCreationContext _creationContext;
00060         float _gameViewXTrigger;
00061 };
00062
00063 }
00064
00065 #endif /* !SPAWNINTENTCOMPONENT_HPP_ */
```

5.139 TriggerIntentComponent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** TriggerIntentComponent
00006 */
00007
00008 #ifndef TRIGGERINTENTCOMPONENT_HPP_
00009 #define TRIGGERINTENTCOMPONENT_HPP_
00010
00011 #include "../base/IComponent.hpp"
00012 #include "../../ECS/entity/Entity.hpp"
00013
00014 namespace ecs {
00015
00016 class TriggerIntentComponent : public IComponent {
00017     public:
00018         TriggerIntentComponent(Entity self = 0, Entity other = 0) : _self(self), _other(other) {};
00019         ~TriggerIntentComponent() = default;
00020
00021         Entity getSelf() const { return _self; }
00022         void setSelf(Entity self) { _self = self; }
00023
00024         Entity getOther() const { return _other; }
00025         void setOther(Entity other) { _other = other; }
00026
00027     private:
00028         Entity _self;
00029         Entity _other;
00030 };
00031
00032 }  // namespace ecs
00033
00034 #endif /* !TRIGGERINTENTCOMPONENT_HPP_ */
```

5.140 debug.hpp

```
00001 /*
```

```
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** debug
00006 */
00007
00008 #ifndef DEBUG_HPP_
00009 #define DEBUG_HPP_
00010
00011 #ifdef _WIN32
00012     #ifndef WIN32_LEAN_AND_MEAN
00013         #define WIN32_LEAN_AND_MEAN
00014     #endif
00015     #include <windows.h>
00016     #ifdef ERROR
00017         #undef ERROR
00018     #endif
00019     #ifdef INFO
00020         #undef INFO
00021     #endif
00022     #ifdef WARNING
00023         #undef WARNING
00024     #endif
00025 #endif
00026
00027
00028 #include <string>
00029
00030 namespace debug {
00031
00032 enum debugType {
00033     NETWORK = 0,
00034     ECS = 1,
00035     CORE = 2
00036 };
00037
00038 enum debugLevel {
00039     INFO = 0,
00040     WARNING = 1,
00041     ERROR = 2
00042 };
00043
00044 class Debug {
00045     public:
00046         ~Debug() = default;
00047         static void printDebug(const bool isDebug, const std::string &message, debugType type,
      debugLevel level);
00048 };
00049
00050 } // namespace debug
00051
00052 #endif /* !DEBUG_HPP_ */
```

5.141 DLLoader.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** r-type
00004 ** File description:
00005 ** DLLoader
00006 */
00007
00008 #ifndef DLLOADER_HPP_
00009 #define DLLOADER_HPP_
00010
00011 #ifdef _WIN32
00012     #include <windows.h>
00013     #define RTLD_LAZY 0
00014 #else
00015     #include <dlfcn.h>
00016 #endif
00017
00018 #include <iostream>
00019 #include <ostream>
00020 #include <memory>
00021 #include "ILoader.hpp"
00022
00023 template <typename T>
00024
00025 class DLLoader : public ILoader {
00026     private:
00027 #ifdef _WIN32
00028         HMODULE _handler = nullptr;
```

```
00029         mutable std::string _lastError;
00030 #else
00031         void *_handler = nullptr;
00032 #endif
00033
00034     public:
00035         ~DLLoader() override {
00036             if (_handler != nullptr) {
00037                 Close();
00038             }
00039         }
00040
00041         void *getHandler() const override {
00042             return _handler;
00043         };
00044
00045         void *Open(const char *path, int flag = RTLD_LAZY) override {
00046 #ifdef _WIN32
00047             (void)flag;
00048             _handler = LoadLibraryA(path);
00049             if (!_handler) {
00050                 _lastError = "Failed to load library: " + std::string(path);
00051             }
00052 #else
00053             _handler = dlopen(path, flag);
00054 #endif
00055             return _handler;
00056         };
00057
00058         void *Symbol(const char *symbolName) override {
00059 #ifdef _WIN32
00060             void *symbol = (void*)GetProcAddress(_handler, symbolName);
00061             if (!symbol) {
00062                 _lastError = "Failed to get symbol: " + std::string(symbolName);
00063                 std::cerr « "GetProcAddress error: " « _lastError « std::endl;
00064                 return nullptr;
00065             }
00066             return symbol;
00067 #else
00068             void *symbol = dlsym(_handler, symbolName);
00069             const char *error = dlerror();
00070             if (error) {
00071                 std::cerr « "dlerror: " « error « std::endl;
00072                 return nullptr;
00073             }
00074             return symbol;
00075 #endif
00076         };
00077
00078         T getSymbol(const char *symbolName) {
00079 #ifdef _WIN32
00080             return reinterpret_cast<T>(GetProcAddress(_handler, symbolName));
00081 #else
00082             return reinterpret_cast<T>(dlsym(_handler, symbolName));
00083 #endif
00084         };
00085
00086         int Close() override{
00087             if (_handler == nullptr)
00088                 return -1;
00089 #ifdef _WIN32
00090             int result = FreeLibrary(_handler) ? 0 : -1;
00091 #else
00092             int result = dlclose(_handler);
00093 #endif
00094             _handler = nullptr;
00095             return result;
00096         };
00097
00098         const char *Error() override {
00099 #ifdef _WIN32
00100             return _lastError.c_str();
00101 #else
00102             return dlerror();
00103 #endif
00104         };
00105 };
00106
00107 #endif /* !DLLOADER_HPP_ */
```

5.142 ILoader.hpp

```
00001 /*
```

```
00002 ** EPITECH PROJECT, 2025
00003 ** r-type
00004 ** File description:
00005 ** ILoader
00006 */
00007
00008 #ifndef ILoader_HPP_
00009 #define ILoader_HPP_
00010
00011
00012 class ILoader {
00013     public:
00014         virtual ~ILoader() = default;
00015
00016         virtual void *Open(const char *path, int flag) = 0;
00017         virtual void *Symbol(const char *symbolName) = 0;
00018         virtual int Close() = 0;
00019         virtual const char *Error() = 0;
00020         virtual void *getHandler() const = 0;
00021
00022     protected:
00023     private:
00024 };
00025
00026 #endif /* !ILoader_HPP_ */
```

5.143 LoaderType.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** r-type
00004 ** File description:
00005 ** LoaderType
00006 */
00007
00008 #ifndef LOADERTYPE_HPP_
00009 #define LOADERTYPE_HPP_
00010
00011 enum ModuleType_t{
00012     MULTIMEDIA_MODULE = 0,
00013     NETWORK_SERVER_MODULE = 1,
00014     NETWORK_CLIENT_MODULE = 2,
00015     PACKET_MODULE = 3,
00016     BUFFER_MODULE = 4,
00017     UNKNOWN_MODULE
00018 };
00019
00020 typedef ModuleType_t (*getTypeFunc_t)();
00021
00022 typedef void *(*createNetworkLib_t)();
00023 typedef void *(*createBuffer_t)();
00024 typedef void *(*createPacket_t)();
00025
00026 #define pathLoad "./libraries"
00027
00028 #ifdef _WIN32
00029     #define multimediaLib "Multimedia"
00030     #define networkServerLib "NetworkServer"
00031     #define networkClientLib "NetworkClient"
00032     #define bufferLib "Buffer"
00033     #define packetLib "Packet"
00034     #define sharedLibExt ".dll"
00035 #elif __APPLE__
00036     #define multimediaLib "libMultimedia"
00037     #define networkServerLib "libNetworkServer"
00038     #define networkClientLib "libNetworkClient"
00039     #define bufferLib "libBuffer"
00040     #define packetLib "libPacket"
00041     #define sharedLibExt ".dylib"
00042 #else
00043     #define multimediaLib "libMultimedia"
00044     #define networkServerLib "libNetworkServer"
00045     #define networkClientLib "libNetworkClient"
00046     #define bufferLib "libBuffer"
00047     #define packetLib "libPacket"
00048     #define sharedLibExt ".so"
00049 #endif
00050
00051 #endif /* !LOADERTYPE_HPP_ */
```

5.144 AComponentArray.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** AComposantType
00006 */
00007
00008 #ifndef AComposantType_HPP_
00009 #define AComposantType_HPP_
00010
00011 #include "IComponentArray.hpp"
00012 #include <vector>
00013 #include <memory>
00014
00015 namespace ecs {
00016
00017 template <typename T>
00018 class AComponentArray : public IComponentArray {
00019     public:
00020         AComponentArray();
00021         ~AComponentArray() override;
00022
00023         void add(Entity entityId, std::shared_ptr<T> component);
00024         std::shared_ptr<T> get(Entity entityId) const;
00025         std::vector<std::shared_ptr<T» getAll(Entity entityId) const;
00026         void removeComponents(Entity entityId) override;
00027         void removeOneComponent(Entity entityId) override;
00028         bool has(Entity entityId) const;
00029
00030         Entity getMaxEntityId() const override;
00031         void clear() override;
00032
00033     private:
00034         std::vector<std::vector<std::shared_ptr<T»> _components;
00035 };
00036
00037 } // namespace ecs
00038
00039 #include "AComponentArray.tpp"
00040
00041 #endif /* !AComposantType_HPP_ */
```

5.145 IComponentArray.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** IComponentArray
00006 */
00007
00008 #ifndef ICOMPONENTARRAY_HPP_
00009 #define ICOMPONENTARRAY_HPP_
00010
00011 #include "../Entity.hpp"
00012
00013 namespace ecs {
00014
00015 class IComponentArray {
00016     public:
00017         virtual ~IComponentArray() = default;
00018         virtual Entity getMaxEntityId() const = 0;
00019         virtual void removeComponents(Entity entityId) = 0;
00020         virtual void removeOneComponent(Entity entityId) = 0;
00021         virtual void clear() = 0;
00022 };
00023
00024 } // namespace ecs
00025
00026 #endif /* !ICOMPONENTARRAY_HPP_ */
```

5.146 Entity.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
```

```
00003 ** ryanR-type
00004 ** File description:
00005 ** Entity
00006 */
00007
00008 #ifndef ENTITY_HPP_
00009 #define ENTITY_HPP_
00010
00011 #include <cstddef>
00012
00013 namespace ecs {
00014
00015 using Entity = size_t;
00016
00017 }  // namespace ecs
00018
00019 #endif /* !ENTITY_HPP_ */
```

5.147 EntityCreationContext.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** EntityCreationContext
00006 */
00007
00008 #ifndef ENTITYCREATIONCONTEXT_HPP_
00009 #define ENTITYCREATIONCONTEXT_HPP_
00010
00011 #include <cstddef>
00012 #include <optional>
00013
00014 namespace ecs {
00015
00016 enum class EntityCreationOrigin {
00017     SERVER,
00018     CLIENT_LOCAL
00019 };
00020
00021 struct EntityCreationContext {
00022     EntityCreationOrigin origin = EntityCreationOrigin::CLIENT_LOCAL;
00023
00024     static EntityCreationContext forServer() {
00025         return {EntityCreationOrigin::SERVER};
00026     }
00027
00028     static EntityCreationContext forLocalClient() {
00029         return {EntityCreationOrigin::CLIENT_LOCAL};
00030     }
00031 };
00032
00033 } // namespace ecs
00034
00035 #endif /* !ENTITYCREATIONCONTEXT_HPP_ */
```

5.148 EntityFactory.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** EntityFactory
00006 */
00007
00008 #ifndef ENTITYFACTORY_HPP_
00009 #define ENTITYFACTORY_HPP_
00010
00011 #include "IEntityFactory.hpp"
00012 #include <atomic>
00013
00014 namespace ecs {
00015
00016 class EntityFactory : public IEntityFactory {
00017     public:
00018         explicit EntityFactory();
00019         ~EntityFactory() override;
00020
```

```
00021         Entity createEntity(
00022             const std::shared_ptr<Registry>& registry,
00023             const EntityCreationContext& context = EntityCreationContext::forLocalClient()
00024         ) override;
00025
00026     private:
00027         std::atomic<size_t> _nextLocalId;
00028 };
00029
00030 }  // namespace ecs
00031
00032 #endif /* !ENTITYFACTORY_HPP_ */
```

5.149 IEntityFactory.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** IEntityFactory
00006 */
00007
00008 #ifndef IENTITYFACTORY_HPP_
00009 #define IENTITYFACTORY_HPP_
00010
00011 #include <memory>
00012 #include <string>
00013 #include "../Entity.hpp"
00014 #include "../EntityCreationContext.hpp"
00015 #include "../registry/Registry.hpp"
00016
00017 namespace ecs {
00018
00019 class IEntityFactory {
00020     public:
00021         virtual ~IEntityFactory() = default;
00022
00023         virtual Entity createEntity(
00024             const std::shared_ptr<Registry>& registry,
00025             const EntityCreationContext& context = EntityCreationContext::forLocalClient()
00026         ) = 0;
00027 };
00028
00029 } // namespace ecs
00030
00031 #endif /* !IENTITYFACTORY_HPP_ */
```

5.150 Registry.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** Registry
00006 */
00007
00008 #ifndef REGISTRY_HPP_
00009 #define REGISTRY_HPP_
00010
00011 #include "../componentArray/IComponentArray.hpp"
00012 #include "../componentArray/AComponentArray.hpp"
00013 #include <memory>
00014 #include <unordered_map>
00015 #include <string>
00016 #include <functional>
00017 #include <mutex>
00018
00019 namespace ecs {
00020
00021 template <typename... Components> class View;
00022 template <typename... Components> class Group;
00023
00024 class Registry : public std::enable_shared_from_this<Registry> {
00025     public:
00026         Registry();
00027         explicit Registry(Entity nextEntityId);
00028         ~Registry();
00029
```

```
00030         template <typename T>
00031         void registerComponent();
00032
00033         template <typename T>
00034         void addComponent(Entity entityId, std::shared_ptr<T> component);
00035
00036         template <typename T>
00037         std::shared_ptr<T> getComponent(Entity entityId) const;
00038
00039         template <typename T>
00040         std::vector<std::shared_ptr<T» getComponents(Entity entityId) const;
00041
00042         template <typename T>
00043         void removeAllComponents(Entity entityId);
00044
00045         template <typename T>
00046         void removeOneComponent(Entity entityId);
00047
00048         template <typename T>
00049         bool hasComponent(Entity entityId) const;
00050
00051         template <typename... Components>
00052         View<Components...> view();
00053
00054         Entity getMaxEntityId() const;
00055
00056         Entity createEntity();
00057         void destroyEntity(Entity entityId);
00058         void clearAllEntities();
00059
00060         void setOnEntityDestroyed(std::function<void(Entity)> callback);
00061     protected:
00062     private:
00063         Entity _nextEntityId;
00064         std::unordered_map<std::string, std::shared_ptr<IComponentArray» _components;
00065         std::function<void(Entity)> _onEntityDestroyed;
00066         mutable std::recursive_mutex _mutex;
00067 };
00068
00069 } // namespace ecs
00070
00071 #include "Registry.tpp"
00072
00073 #endif /* !REGISTRY_HPP_ */
```

5.151 View.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** View
00006 */
00007
00008 #ifndef VIEW_HPP_
00009 #define VIEW_HPP_
00010
00011 #include <vector>
00012 #include <memory>
00013 #include <type_traits>
00014
00015 namespace ecs {
00016
00017 template <typename... Components>
00018 class View {
00019     public:
00020         View(std::shared_ptr<Registry> registry);
00021
00022         class Iterator;
00023
00024         Iterator begin();
00025         Iterator end();
00026
00027         class Iterator {
00028             public:
00029                 Iterator(std::shared_ptr<Registry> registry, size_t entityId, size_t maxEntityId);
00030                 bool operator!=(const Iterator &other) const;
00031                 Iterator& operator++();
00032                 size_t operator*() const;
00033                 bool operator==(const Iterator &other) const;
00034
00035             private:
00036                 bool hasAllComponents() const;
```

```
00037                 std::shared_ptr<Registry> _registry;
00038                 size_t _entityId;
00039                 size_t _maxEntityId;
00040         };
00041
00042     private:
00043         std::shared_ptr<Registry> _registry;
00044 };
00045
00046
00047 }  // namespace ecs
00048
00049 #include "View.tpp"
00050
00051 #endif /* !VIEW_HPP_ */
```

5.152 AError.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** AError
00006 */
00007
00008 #ifndef AERROR_HPP_
00009 #define AERROR_HPP_
00010
00011 #include <string>
00012 #include "IError.hpp"
00013
00014 namespace err {
00015
00016 class AError : public IError {
00017     public:
00018         AError(const std::string &message, int code = 0);
00019
00020         virtual ~AError() noexcept override = default;
00021         const char *what() const noexcept override;
00022         int getCode() const noexcept override;
00023         std::string getDetails() const noexcept override;
00024
00025         virtual std::string getType() const noexcept override = 0;
00026
00027     protected:
00028         std::string m_message;
00029         int m_code;
00030 };
00031
00032 }
00033
00034 #endif /* !AERROR_HPP_ */
```

5.153 ClientError.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ClientError
00006 */
00007
00008 #ifndef CLIENTERROR_HPP_
00009 #define CLIENTERROR_HPP_
00010
00011 #include "AError.hpp"
00012
00013 namespace err {
00014
00015 class ClientError : public AError {
00016     public:
00017         enum ErrorCode {
00018             UNKNOWN = 2000,
00019             CONNECTION_FAILED = 2001,
00020             DISCONNECTED = 2002,
00021             TIMEOUT = 2003,
00022             NOT_INITIALIZED = 2004,
00023             CAN_NOT_OPEN_FILE = 2005
```

```
00024         };
00025
00026         ClientError(const std::string &message, ErrorCode code = UNKNOWN);
00027         ~ClientError() override;
00028
00029         std::string getType() const noexcept override;
00030     protected:
00031     private:
00032 };
00033
00034 }  // namespace err
00035
00036 #endif /* !CLIENTERROR_HPP_ */
```

5.154 ClientNetworkError.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ClientNetworkError
00006 */
00007
00008 #ifndef CLIENTNETWORKERROR_HPP_
00009 #define CLIENTNETWORKERROR_HPP_
00010
00011 #include "AError.hpp"
00012
00013 namespace err {
00014
00015 class ClientNetworkError : public AError {
00016     public:
00017         enum ErrorCode {
00018             UNKNOWN = 1000,
00019             CONNECTION_FAILED = 1001,
00020             TIMEOUT = 1002,
00021             INVALID_REQUEST = 1003,
00022             INTERNAL_ERROR = 1004,
00023             LIBRARY_LOAD_FAILED = 1005,
00024             CONFIG_ERROR = 1006
00025         };
00026
00027         ClientNetworkError(const std::string &message, ErrorCode code = UNKNOWN);
00028         virtual ~ClientNetworkError() noexcept = default;
00029         std::string getType() const noexcept override;
00030
00031     private:
00032 };
00033
00034 }
00035
00036 #endif /* !CLIENTNETWORKERROR_HPP_ */
```

5.155 IError.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** IError
00006 */
00007
00008 #ifndef IERROR_HPP_
00009 #define IERROR_HPP_
00010
00011 #include <string>
00012 #include <exception>
00013
00014 namespace err {
00015
00016 class IError : public std::exception {
00017     public:
00018
00019         virtual ~IError() noexcept = default;
00020         virtual const char *what() const noexcept override = 0;
00021         virtual int getCode() const noexcept = 0;
00022         virtual std::string getType() const noexcept = 0;
00023         virtual std::string getDetails() const noexcept = 0;
```

```
00024
00025     protected:
00026     private:
00027 };
00028
00029 }
00030
00031 #endif /* !IERROR_HPP_ */
00032
```

5.156 LibrairiesLoadError.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** LibrairiesLoadError
00006 */
00007
00008 #ifndef LIBRAIRIESLOADERROR_HPP_
00009 #define LIBRAIRIESLOADERROR_HPP_
00010
00011 #include "AError.hpp"
00012
00013 namespace err {
00014
00015 class LibrairiesLoadError : public AError  {
00016     public:
00017         enum ErrorCode {
00018             UNKNOWN = 1000,
00019             LIBRARY_NOT_FOUND = 1001,
00020             SYMBOL_NOT_FOUND = 1002
00021         };
00022
00023         LibrairiesLoadError(const std::string &message, ErrorCode code = UNKNOWN);
00024         ~LibrairiesLoadError() override;
00025         std::string getType() const noexcept override;
00026
00027     protected:
00028     private:
00029 };
00030
00031 }
00032
00033 #endif /* !LIBRAIRIESLOADERROR_HPP_ */
```

5.157 PacketError.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** R-Type
00004 ** File description:
00005 ** Header
00006 */
00007
00008 #ifndef PACKET_ERROR_HPP
00009     #define PACKET_ERROR_HPP
00010
00011 #include "AError.hpp"
00012
00013 namespace err {
00014
00015 class PacketError : public AError {
00016     public:
00017         enum ErrorCode {
00018             UNKNOWN = 1000,
00019             SERIALIZER_ATTRIBUTION_FAILED = 1001,
00020             STRING_FORMATTING_ERROR = 1002
00021         };
00022
00023         PacketError(const std::string &message, ErrorCode code = UNKNOWN);
00024         ~PacketError() override;
00025         std::string getType() const noexcept override;
00026
00027     protected:
00028     private:
00029 };
00030
```

```
00031 }
00032
00033 #endif /* !PACKET_ERROR_HPP */
```

5.158 ParserError.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ParserError
00006 */
00007
00008 #ifndef PARSERERROR_HPP_
00009 #define PARSERERROR_HPP_
00010
00011 #include "AError.hpp"
00012 namespace err {
00013
00014 class ParserError : public AError {
00015     public:
00016         enum ErrorCode {
00017             UNKNOWN = 1000,
00018             FILE_NOT_FOUND = 1001,
00019             INVALID_FORMAT = 1002,
00020             MISSING_FIELD = 1003,
00021             TYPE_MISMATCH = 1004
00022         };
00023
00024         ParserError(const std::string &message, ErrorCode code = UNKNOWN);
00025         virtual ~ParserError() noexcept = default;
00026         std::string getType() const noexcept override;
00027 };
00028
00029 }  // namespace err
00030
00031 #endif /* !PARSERERROR_HPP_ */
```

5.159 ScriptingError.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2026
00003 ** ryanR-type
00004 ** File description:
00005 ** ScriptingError
00006 */
00007
00008 #ifndef SCRIPTINGERROR_HPP_
00009 #define SCRIPTINGERROR_HPP_
00010
00011
00012 #include "AError.hpp"
00013
00014 namespace err {
00015
00016 class ScriptingError : public AError {
00017     public:
00018         enum ErrorCode {
00019             UNKNOWN = 1000,
00020             LOAD_FAILED = 1001,
00021             RUN_FAILED = 1002
00022         };
00023
00024         ScriptingError(const std::string &message, ErrorCode code = UNKNOWN);
00025         ~ScriptingError() noexcept = default;
00026         std::string getType() const noexcept override;
00027
00028     protected:
00029     private:
00030 };
00031
00032 }  // namespace err
00033
00034 #endif /* !SCRIPTINGERROR_HPP_ */
```

5.160 ServerErrror.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ServerErrror
00006 */
00007
00008 #ifndef SERVER_ERROR_HPP
00009     #define SERVER_ERROR_HPP
00010
00011 #include "AError.hpp"
00012
00013 namespace err {
00014
00015 class ServerError : public AError {
00016     public:
00017         enum ErrorCode {
00018             UNKNOWN = 1000,
00019             CONNECTION_FAILED = 1001,
00020             TIMEOUT = 1002,
00021             INVALID_REQUEST = 1003,
00022             INTERNAL_ERROR = 1004,
00023             LIBRARY_LOAD_FAILED = 1005,
00024             CONFIG_ERROR = 1006
00025         };
00026
00027         ServerError(const std::string &message, ErrorCode code = UNKNOWN);
00028         virtual ~ServerError() noexcept = default;
00029         std::string getType() const noexcept override;
00030
00031     private:
00032 };
00033
00034 }
00035
00036 #endif /* !SERVER_ERROR_HPP */
```

5.161 FloatQuantization.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2026
00003 ** ryanR-type
00004 ** File description:
00005 ** Float Quantization utilities for network optimization
00006 */
00007
00008 #ifndef FLOAT_QUANTIZATION_HPP_
00009 #define FLOAT_QUANTIZATION_HPP_
00010
00011 #include <cstdint>
00012 #include <cmath>
00013
00014 namespace quantization {
00015
00016 constexpr float POSITION_PRECISION = 100.0f;    // 0.01 units precision
00017 constexpr float VELOCITY_PRECISION = 10.0f;     // 0.1 units/s precision
00018 constexpr float ROTATION_PRECISION = 10.0f;     // 0.1 degrees precision
00019 constexpr float SCALE_PRECISION = 100.0f;       // 0.01 scale precision
00020 constexpr float SIZE_PRECISION = 100.0f;        // 0.01 size precision
00021 constexpr float HEALTH_PRECISION = 10.0f;       // 0.1 health precision
00022 constexpr float DAMAGE_PRECISION = 10.0f;       // 0.1 damage precision
00023 constexpr float SPEED_PRECISION = 10.0f;        // 0.1 speed precision
00024 constexpr float TIME_PRECISION = 100.0f;        // 0.01 seconds precision
00025 constexpr float ANGLE_PRECISION = 100.0f;       // 0.01 radians precision
00026
00027 inline int32_t quantizeFloat(float value, float precision) {
00028     return static_cast<int32_t>(std::round(value * precision));
00029 }
00030
00031 inline float dequantizeFloat(int32_t quantized, float precision) {
00032     return static_cast<float>(quantized) / precision;
00033 }
00034
00035 inline uint64_t packQuantizedFloat(float value, float precision) {
00036     int32_t quantized = quantizeFloat(value, precision);
00037     uint64_t result;
00038     std::memcpy(&result, &quantized, sizeof(int32_t));
00039     return result;
00040 }
00041
```

```
00042 inline float dequantizeFloat(uint64_t packed, float precision) {
00043     int32_t quantized;
00044     std::memcpy(&quantized, &packed, sizeof(int32_t));
00045     return dequantizeFloat(quantized, precision);
00046 }
00047
00048 inline uint64_t packPosition(float value) {
00049     return packQuantizedFloat(value, POSITION_PRECISION);
00050 }
00051
00052 inline float unpackPosition(uint64_t packed) {
00053     return dequantizeFloat(packed, POSITION_PRECISION);
00054 }
00055
00056 inline uint64_t packVelocity(float value) {
00057     return packQuantizedFloat(value, VELOCITY_PRECISION);
00058 }
00059
00060 inline float unpackVelocity(uint64_t packed) {
00061     return dequantizeFloat(packed, VELOCITY_PRECISION);
00062 }
00063
00064 inline uint64_t packRotation(float value) {
00065     return packQuantizedFloat(value, ROTATION_PRECISION);
00066 }
00067
00068 inline float unpackRotation(uint64_t packed) {
00069     return dequantizeFloat(packed, ROTATION_PRECISION);
00070 }
00071
00072 inline uint64_t packScale(float value) {
00073     return packQuantizedFloat(value, SCALE_PRECISION);
00074 }
00075
00076 inline float unpackScale(uint64_t packed) {
00077     return dequantizeFloat(packed, SCALE_PRECISION);
00078 }
00079
00080 inline uint64_t packSize(float value) {
00081     return packQuantizedFloat(value, SIZE_PRECISION);
00082 }
00083
00084 inline float unpackSize(uint64_t packed) {
00085     return dequantizeFloat(packed, SIZE_PRECISION);
00086 }
00087
00088 inline uint64_t packHealth(float value) {
00089     return packQuantizedFloat(value, HEALTH_PRECISION);
00090 }
00091
00092 inline float unpackHealth(uint64_t packed) {
00093     return dequantizeFloat(packed, HEALTH_PRECISION);
00094 }
00095
00096 inline uint64_t packDamage(float value) {
00097     return packQuantizedFloat(value, DAMAGE_PRECISION);
00098 }
00099
00100 inline float unpackDamage(uint64_t packed) {
00101     return dequantizeFloat(packed, DAMAGE_PRECISION);
00102 }
00103
00104 inline uint64_t packSpeed(float value) {
00105     return packQuantizedFloat(value, SPEED_PRECISION);
00106 }
00107
00108 inline float unpackSpeed(uint64_t packed) {
00109     return dequantizeFloat(packed, SPEED_PRECISION);
00110 }
00111
00112 inline uint64_t packTime(float value) {
00113     return packQuantizedFloat(value, TIME_PRECISION);
00114 }
00115
00116 inline float unpackTime(uint64_t packed) {
00117     return dequantizeFloat(packed, TIME_PRECISION);
00118 }
00119
00120 inline uint64_t packAngle(float value) {
00121     return packQuantizedFloat(value, ANGLE_PRECISION);
00122 }
00123
00124 inline float unpackAngle(uint64_t packed) {
00125     return dequantizeFloat(packed, ANGLE_PRECISION);
00126 }
00127
00128 }  // namespace quantization
```

```
00129
00130 #endif /* !FLOAT_QUANTIZATION_HPP_ */
```

5.162 GameRules.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** GameRules
00006 */
00007
00008 #ifndef GAMERULES_HPP_
00009 #define GAMERULES_HPP_
00010
00011 namespace GameRulesNS {
00012
00013 enum class Difficulty {
00014     EASY,
00015     NORMAL,
00016     HARD
00017 };
00018
00019 enum class Gamemode {
00020     CLASSIC,
00021     INFINITE_MODE
00022 };
00023
00024 }  // namespace GameRulesNS
00025
00026 class GameRules {
00027     public:
00028         GameRules();
00029         ~GameRules() = default;
00030
00031         void setGamemode(GameRulesNS::Gamemode gamemode);
00032         GameRulesNS::Gamemode getGamemode() const;
00033
00034         void setDifficulty(GameRulesNS::Difficulty difficulty);
00035         GameRulesNS::Difficulty getDifficulty() const;
00036
00037         void setCrossfire(bool crossfire);
00038         bool getCrossfire() const;
00039
00040     private:
00041         GameRulesNS::Gamemode _gamemode;
00042         GameRulesNS::Difficulty _difficulty;
00043         bool _crossfire;
00044 };
00045
00046 #endif /* !GAMERULES_HPP_ */
```

5.163 IGameState.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** IGameState
00006 */
00007
00008 #pragma once
00009
00010 #include <memory>
00011 #include <vector>
00012 #include "../systems/base/ISystem.hpp"
00013
00014 namespace gsm {
00015
00016 class IGameStateMachine;
00017
00018 class IGameState {
00019 public:
00020     virtual ~IGameState() = default;
00021
00022     virtual void enter() = 0;
00023     virtual void update(float deltaTime) = 0;
00024     virtual void exit() = 0;
```

```
00025     virtual void addSystem(std::shared_ptr<ecs::ISystem> system) = 0;
00026     virtual std::vector<std::shared_ptr<ecs::ISystem» getSystems() const = 0;
00027     virtual std::string getStateName() const = 0;
00028 };
00029
00030 } // namespace gsm
```

5.164 IGameStateMachine.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** IGameStateMachine
00006 */
00007
00008 #pragma once
00009
00010 #include <memory>
00011 #include <stack>
00012
00013 namespace gsm {
00014
00015 class IGameState;
00016
00017 class IGameStateMachine {
00018 public:
00019     virtual ~IGameStateMachine() = default;
00020
00021     virtual void changeState(std::shared_ptr<IGameState> newState) = 0;
00022     virtual void pushState(std::shared_ptr<IGameState> newState) = 0;
00023     virtual void popState() = 0;
00024     virtual void requestStateChange(std::shared_ptr<IGameState> newState) = 0;
00025     virtual void requestStatePush(std::shared_ptr<IGameState> newState) = 0;
00026     virtual void requestStatePop() = 0;
00027
00028     virtual void update(float deltaTime) = 0;
00029 };
00030
00031 } // namespace gsm
```

5.165 IInputProvider.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** IInputProvider
00006 */
00007
00008 #ifndef IINPUTPROVIDER_HPP_
00009 #define IINPUTPROVIDER_HPP_
00010
00011 #include <utility>
00012 #include "../../libs/Multimedia/EventTypes.hpp"
00013 #include "InputAction.hpp"
00014 #include "InputMapping.hpp"
00015
00016 namespace ecs {
00017
00018 class IInputProvider {
00019     public:
00020         using event_t = gfx::EventType;
00021         virtual ~IInputProvider() = default;
00022
00023         virtual float getAxisValue(event_t axis, size_t clientID = 0) = 0;
00024         virtual bool isActionPressed(InputAction action, size_t clientID = 0) = 0;
00025         virtual float getActionAxis(InputAction action, size_t clientID = 0) = 0;
00026         virtual InputMapping getInputMapping(size_t clientID = 0) const = 0;
00027 };
00028
00029 }  // namespace ecs
00030
00031 #endif /* !IINPUTPROVIDER_HPP_ */
```

5.166 InputAction.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** InputAction
00006 */
00007
00008 #ifndef INPUTACTION_HPP_
00009 #define INPUTACTION_HPP_
00010
00011 namespace ecs {
00012
00013 enum class InputAction {
00014     MOVE_X,
00015     MOVE_Y,
00016     SHOOT,
00017     FORCE,
00018     PAUSE,
00019     MENU_UP,
00020     MENU_DOWN,
00021     MENU_LEFT,
00022     MENU_RIGHT,
00023     MENU_SELECT,
00024     MENU_BACK,
00025 };
00026
00027 enum class RemappableAction {
00028     MOVE_LEFT,
00029     MOVE_RIGHT,
00030     MOVE_UP,
00031     MOVE_DOWN,
00032     SHOOT,
00033     FORCE,
00034 };
00035
00036 }  // namespace ecs
00037
00038 #endif /* !INPUTACTION_HPP_ */
```

5.167 InputMapping.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** InputMapping
00006 */
00007
00008 #ifndef INPUTMAPPING_HPP_
00009 #define INPUTMAPPING_HPP_
00010
00011 #include <map>
00012 #include <vector>
00013 #include "../../libs/Multimedia/EventTypes.hpp"
00014 #include "InputAction.hpp"
00015
00016 namespace ecs {
00017
00018 struct RemappableKeyBinding {
00019     gfx::EventType primary;
00020     gfx::EventType secondary;
00021
00022     RemappableKeyBinding()
00023         : primary(gfx::EventType::NOTHING), secondary(gfx::EventType::NOTHING) {}
00024     RemappableKeyBinding(gfx::EventType p, gfx::EventType s)
00025         : primary(p), secondary(s) {}
00026 };
00027
00028 struct InputMapping {
00029     std::map<RemappableAction, RemappableKeyBinding> remappableKeys;
00030     std::map<InputAction, std::map<gfx::EventType, float» fixedMappings;
00031
00032     std::map<InputAction, std::map<gfx::EventType, float» getAllMappings() const {
00033         std::map<InputAction, std::map<gfx::EventType, float» all = fixedMappings;
00034
00035         for (const auto& [action, binding] : remappableKeys) {
00036             InputAction inputAction;
00037             switch (action) {
00038                 case RemappableAction::MOVE_LEFT: inputAction = InputAction::MOVE_X; break;
00039                 case RemappableAction::MOVE_RIGHT: inputAction = InputAction::MOVE_X; break;
```

```
00040                 case RemappableAction::MOVE_UP: inputAction = InputAction::MOVE_Y; break;
00041                 case RemappableAction::MOVE_DOWN: inputAction = InputAction::MOVE_Y; break;
00042                 case RemappableAction::SHOOT: inputAction = InputAction::SHOOT; break;
00043                 case RemappableAction::FORCE: inputAction = InputAction::FORCE; break;
00044             }
00045
00046             float value = (action == RemappableAction::MOVE_LEFT ||
00047                 action == RemappableAction::MOVE_UP) ? -1.0f : 1.0f;
00048
00049             if (binding.primary != gfx::EventType::NOTHING) {
00050                 all[inputAction][binding.primary] = value;
00051             }
00052             if (binding.secondary != gfx::EventType::NOTHING) {
00053                 all[inputAction][binding.secondary] = value;
00054             }
00055         }
00056
00057         return all;
00058     }
00059 };
00060
00061 }  // namespace ecs
00062
00063 #endif /* !INPUTMAPPING_HPP_ */
```

5.168 InputMappingManager.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** InputMappingManager
00006 */
00007
00008 #ifndef INPUTMAPPINGMANAGER_HPP_
00009 #define INPUTMAPPINGMANAGER_HPP_
00010
00011 #include <string>
00012 #include <vector>
00013 #include "InputMapping.hpp"
00014
00015 namespace ecs {
00016
00017 class InputMappingManager {
00018 public:
00019     InputMappingManager();
00020     ~InputMappingManager() = default;
00021
00022     void loadDefault();
00023
00024     void setMapping(const InputMapping& mapping);
00025     const InputMapping& getMapping() const;
00026     InputMapping& getMutableMapping();
00027
00028     gfx::EventType getKeyForRemappableAction(RemappableAction action, bool getPrimary = true) const;
00029     void remapKey(RemappableAction action, gfx::EventType newKey, bool setPrimary);
00030
00031     static std::string eventTypeToString(gfx::EventType eventType);
00032     bool isKeyboardKey(gfx::EventType eventType);
00033     static gfx::EventType stringToEventType(const std::string& str);
00034     static std::string remappableActionToString(RemappableAction action);
00035     static RemappableAction stringToRemappableAction(const std::string& str);
00036
00037 private:
00038     InputMapping _mapping;
00039 };
00040
00041 }  // namespace ecs
00042
00043 #endif /* !INPUTMAPPINGMANAGER_HPP_ */
```

5.169 IAudio.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** IAudio
```

```
00006 */
00007
00008 #ifndef IAUDIO_HPP_
00009     #define IAUDIO_HPP_
00010
00011 #include <string>
00012
00013 namespace gfx
00014 {
00015
00016 class IAudio
00017 {
00018     public:
00019         virtual ~IAudio() = default;
00020
00021         virtual void playMusic(const std::string& musicPath, bool loop = true) = 0;
00022         virtual void stopMusic() = 0;
00023         virtual void pauseMusic() = 0;
00024         virtual void resumeMusic() = 0;
00025         virtual void setMusicVolume(float volume) = 0;
00026         virtual float getMusicVolume() const = 0;
00027         virtual bool isMusicPlaying() const = 0;
00028
00029         virtual void playSound(const std::string& soundPath, float volume = 100.0f, float pitch =
      1.0f) = 0;
00030         virtual void setSoundVolume(float volume) = 0;
00031         virtual float getSoundVolume() const = 0;
00032         virtual void stopAllSounds() = 0;
00033 };
00034
00035 typedef IAudio *(*createAudio_t)();
00036
00037 }
00038
00039 #endif /* !IAUDIO_HPP_ */
```

5.170 IBuffer.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** IBuffer
00006 */
00007
00008 #ifndef IBUFFER_HPP_
00009 #define IBUFFER_HPP_
00010
00011 #include <memory>
00012 #include <vector>
00013 #include <cstdint>
00014
00015
00016 class IBuffer {
00017     public:
00018         virtual ~IBuffer() = default;
00019
00020
00021         virtual void createBuffer(size_t size) = 0;
00022         virtual void deleteBuffer() = 0;
00023         virtual void clear() = 0;
00024
00025         virtual bool writeBuffer(const std::vector<uint64_t> &data, size_t size) = 0;
00026         virtual std::shared_ptr<std::vector<uint64_t» readBuffer(size_t size) = 0;
00027
00028         virtual size_t getCapacity() const = 0;
00029         virtual size_t getUsedSize() const = 0;
00030         virtual size_t getAvailableSize() const = 0;
00031         virtual bool isEmpty() const = 0;
00032         virtual bool isFull() const = 0;
00033
00034         virtual std::vector<uint64_t> getBuffer() const = 0;
00035     protected:
00036     private:
00037 };
00038
00039 #endif /* !IBUFFER_HPP_ */
```

5.171 IEvent.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** r-type
00004 ** File description:
00005 ** IEvent
00006 */
00007
00008 #ifndef IEVENT_HPP_
00009 #define IEVENT_HPP_
00010
00011 #include <utility>
00012 #include <memory>
00013 #include <string>
00014 #include "EventTypes.hpp"
00015
00016 namespace gfx {
00017
00018 class IEvent {
00019     public:
00020         using event_t = EventType;
00021         virtual ~IEvent() = default;
00022         virtual void init() = 0;
00023         virtual event_t pollEvents() = 0;
00024         virtual std::string getLastTextInput() = 0;
00025         virtual void cleanup() = 0;
00026         virtual std::pair<int, int> getMousePos() = 0;
00027         virtual bool isKeyPressed(event_t key) = 0;
00028         virtual bool isMouseButtonPressed(int button) = 0;
00029         virtual float getAxisValue(event_t axis) = 0;
00030
00031 };
00032
00033 typedef IEvent *(*createEvent_t)(void*, void*);
00034
00035 } // namespace gfx
00036
00037 #endif /* !IEVENT_HPP_ */
```

5.172 IEventLoop.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** IEventLoop
00006 */
00007
00008 #ifndef IEVENTLOOP_HPP_
00009 #define IEVENTLOOP_HPP_
00010
00011 #include <functional>
00012 #include <memory>
00013
00014 namespace net {
00015
00016 class IEventLoop {
00017     public:
00018         virtual ~IEventLoop() = default;
00019         virtual void run() = 0;
00020         virtual void runOne() = 0;
00021         virtual void stop() = 0;
00022         virtual bool stopped() const = 0;
00023         virtual void post(std::function<void()> task) = 0;
00024         virtual void restart() = 0;
00025     };
00026
00027 } // namespace net
00028
00029 #endif /* !IEVENTLOOP_HPP_ */
```

5.173 INetwork.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
```

```
00004 ** File description:
00005 ** INetwork
00006 */
00007
00008 #ifndef INETWORK_HPP_
00009 #define INETWORK_HPP_
00010
00011 #include <vector>
00012 #include <functional>
00013 #include <memory>
00014 #include "INetworkEndpoint.hpp"
00015 #include "IPacketManager.hpp"
00016 #include "IBuffer.hpp"
00017
00018 namespace net {
00019
00020 enum class ConnectionState {
00021     DISCONNECTED,
00022     CONNECTING,
00023     CONNECTED,
00024     RECONNECTING,
00025     ERROR_STATE
00026 };
00027
00028
00029 class INetwork {
00030     public:
00031
00032         virtual ~INetwork() = default;
00033
00034         virtual void init(uint16_t port, const std::string host) = 0;
00035         virtual void stop() = 0;
00036
00037         virtual bool sendTo(const INetworkEndpoint& endpoint, std::vector<uint8_t> packet) = 0;
00038         virtual bool broadcast(const std::vector<std::shared_ptr<INetworkEndpoint>& endpoints, const
      std::vector<uint8_t>& data) = 0;
00039         virtual bool hasIncomingData() const = 0;
00040         virtual std::vector<uint8_t> receiveFrom(const uint8_t &connectionId) = 0;
00041         virtual std::pair<std::shared_ptr<INetworkEndpoint>, std::vector<uint8_t>> receiveAny() = 0;
00042
00043         virtual void setConnectionCallback(std::function<void(int)> onConnect) = 0;
00044         virtual void setDisconnectionCallback(std::function<void(int)> onDisconnect) = 0;
00045         virtual ConnectionState getConnectionState() const = 0;
00046         virtual void setConnectionState(ConnectionState state) = 0;
00047 };
00048
00049 } // namespace net
00050
00051 #endif /* !INETWORK_HPP_ */
```

5.174 INetworkAddress.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** R-Type
00004 ** File description:
00005 ** INetworkAddress - Interface for IP address representation
00006 */
00007
00008 #ifndef INETWORK_ADDRESS_HPP
00009 #define INETWORK_ADDRESS_HPP
00010
00011 #include <string>
00012 #include <memory>
00013
00014 namespace net {
00015
00016 class INetworkAddress {
00017     public:
00018         virtual bool isV4() const = 0;
00019         virtual bool isV6() const = 0;
00020         virtual std::string toString() const = 0;
00021
00022         virtual ~INetworkAddress() = default;
00023         virtual std::shared_ptr<INetworkAddress> operator=(const INetworkAddress& other) = 0;
00024         virtual std::shared_ptr<void> getInternalAddress() = 0;
00025         virtual std::shared_ptr<const void> getInternalAddress() const = 0;
00026         virtual void setFromInternal(std::shared_ptr<void> internalAddr) = 0;
00027 };
00028
00029 } // namespace net
00030
00031 #endif // INETWORK_ADDRESS_HPP
```

5.175 INetworkEndpoint.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** NetworkEndpoint
00006 */
00007
00008 #ifndef INETWORKENDPOINT_HPP_
00009 #define INETWORKENDPOINT_HPP_
00010
00011 #include <string>
00012 #include <cstdint>
00013
00014 namespace net {
00015
00016 class INetworkEndpoint {
00017     public:
00018         virtual ~INetworkEndpoint() noexcept = default;
00019
00020         virtual const std::string& getAddress() const = 0;
00021         virtual uint16_t getPort() const = 0;
00022
00023         virtual void setAddress(const std::string& address) = 0;
00024         virtual void setPort(uint16_t port) = 0;
00025         virtual bool operator==(const INetworkEndpoint& other) const = 0;
00026         virtual bool operator!=(const INetworkEndpoint& other) const = 0;
00027         virtual bool operator<(const INetworkEndpoint& other) const = 0;
00028 };
00029
00030 } // namespace net
00031
00032 #endif /* !INETWORKENDPOINT_HPP_ */
```

5.176 INetworkErrorCode.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** R-Type
00004 ** File description:
00005 ** NetworkErrorCode
00006 */
00007
00008 #ifndef INETWORK_ERROR_CODE_HPP
00009 #define INETWORK_ERROR_CODE_HPP
00010
00011 #include <string>
00012 #include <memory>
00013
00014 namespace net {
00015
00016 enum class NetworkError {
00017     SUCCESS = 0,
00018     WOULD_BLOCK = 1,
00019     AGAIN = 2,
00020     CONNECTION_REFUSED = 3,
00021     NETWORK_UNREACHABLE = 4,
00022     TIMED_OUT = 5,
00023     OTHER = 6
00024 };
00025
00026 class INetworkErrorCode {
00027     public:
00028         virtual ~INetworkErrorCode() noexcept = default;
00029
00030         virtual void clear() = 0;
00031         virtual bool hasError() const = 0;
00032         virtual explicit operator bool() const = 0;
00033         virtual std::string message() const = 0;
00034         virtual NetworkError getError() const = 0;
00035         virtual void setError(NetworkError error, const std::string& msg = "") = 0;
00036         virtual bool operator==(NetworkError error) const = 0;
00037         virtual bool operator!=(NetworkError error) const = 0;
00038
00039         virtual std::shared_ptr<void> getInternalErrorCode() = 0;
00040         virtual std::shared_ptr<const void> getInternalErrorCode() const = 0;
00041         virtual void setFromInternal(std::shared_ptr<void> internalEc) = 0;
00042 };
00043
00044 } // namespace net
00045
00046 #endif // INETWORK_ERROR_CODE_HPP
```

5.177 INetworkFactory.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2026
00003 ** R-Type
00004 ** File description:
00005 ** INetworkFactory - Factory interface for creating network objects
00006 */
00007
00008 #ifndef INETWORK_FACTORY_HPP_
00009     #define INETWORK_FACTORY_HPP_
00010
00011 #include <memory>
00012 #include "INetworkSocket.hpp"
00013 #include "INetworkResolver.hpp"
00014 #include "IEventLoop.hpp"
00015
00016 namespace net {
00017
00018 class INetworkFactory {
00019     public:
00020         virtual ~INetworkFactory() = default;
00021         virtual std::shared_ptr<IEventLoop> createEventLoop() = 0;
00022         virtual std::shared_ptr<INetworkSocket> createSocket(std::shared_ptr<IEventLoop> eventLoop) =
      0;
00023         virtual std::shared_ptr<INetworkResolver> createResolver(std::shared_ptr<IEventLoop>
      eventLoop) = 0;
00024 };
00025
00026 } // namespace net
00027
00028 #endif /* !INETWORK_FACTORY_HPP_ */
```

5.178 INetworkResolver.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** R-Type
00004 ** File description:
00005 ** NetworkResolver
00006 */
00007
00008 #ifndef INETWORK_RESOLVER_HPP
00009 #define INETWORK_RESOLVER_HPP
00010
00011 #include <memory>
00012 #include <string>
00013 #include <vector>
00014
00015 namespace net {
00016
00017 class INetworkEndpoint;
00018 class INetworkErrorCode;
00019
00020 class INetworkResolver {
00021     public:
00022         virtual ~INetworkResolver() = default;
00023         virtual std::vector<std::shared_ptr<INetworkEndpoint» resolve(const std::string& host,
00024             const std::string& port, std::shared_ptr<INetworkErrorCode> ec) = 0;
00025 };
00026
00027 } // namespace net
00028
00029 #endif // INETWORK_RESOLVER_HPP
```

5.179 INetworkSocket.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** R-Type
00004 ** File description:
00005 ** Header
00006 */
00007
00008 #ifndef INETWORK_SOCKET_HPP
00009     #define INETWORK_SOCKET_HPP
00010
```

```
00011 #include <memory>
00012 #include <string>
00013 #include <vector>
00014 #include <cstdint>
00015
00016 namespace net {
00017
00018 class INetworkEndpoint;
00019 class INetworkErrorCode;
00020
00021 class INetworkSocket {
00022     public:
00023         virtual ~INetworkSocket() = default;
00024         virtual bool open(std::shared_ptr<INetworkErrorCode> ec) = 0;
00025         virtual bool bind(const INetworkEndpoint& endpoint, std::shared_ptr<INetworkErrorCode> ec) =
      0;
00026         virtual std::size_t sendTo(const std::vector<uint8_t>& data, const INetworkEndpoint& endpoint,
      int flags, std::shared_ptr<INetworkErrorCode> ec) = 0;
00027         virtual std::size_t receiveFrom(std::shared_ptr<std::vector<uint8_t» buffer,
      std::shared_ptr<INetworkEndpoint> sender, int flags, std::shared_ptr<INetworkErrorCode> ec) = 0;
00028         virtual bool setNonBlocking(bool nonBlocking, std::shared_ptr<INetworkErrorCode> ec) = 0;
00029         virtual bool close(std::shared_ptr<INetworkErrorCode> ec) = 0;
00030         virtual bool isOpen() const = 0;
00031 };
00032
00033 } // namespace net
00034
00035 #endif // INETWORK_SOCKET_HPP
```

5.180 IPacketManager.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** IPacketManager
00006 */
00007
00008 #ifndef IPacketManager_HPP_
00009 #define IPacketManager_HPP_
00010
00011 #include <vector>
00012 #include <cstdint>
00013 #include <functional>
00014 #include <string>
00015 #include "IBuffer.hpp"
00016
00017 #define MAGIC_NUMBER 0x93
00018 #define HEADER_SIZE 11
00019
00020 #define LENGTH_CONNECTION_PACKET 0
00021 #define LENGTH_ACCEPTATION_PACKET 1
00022 #define LENGTH_DISCONNECTION_PACKET 1
00023 #define LENGTH_EVENT_PACKET 9
00024 #define LENGTH_END_GAME_PACKET 0
00025 #define LENGTH_DEATH_PACKET 8
00026 #define LENGTH_WHOAMI_PACKET 0
00027 #define LENGTH_SERVER_STATUS_PACKET 32
00028 #define LENGTH_LOBBY_CODE_PACKET 8
00029 #define LENGTH_REQUEST_LOBBY_PACKET 0
00030 #define LENGTH_CONNECT_TO_LOBBY_PACKET 1
00031 #define LENGTH_REGISTER_PACKET 16
00032 #define LENGTH_CONNECT_USER_PACKET 8
00033 #define LENGTH_LOGIN_PACKET 16
00034 #define LENGTH_REQUEST_LEADERBOARD_PACKET 0
00035 #define LENGTH_LEADERBOARD_PACKET 160
00036 #define LENGTH_FAIL_REGISTER_PACKET 0
00037 #define LENGTH_REQUEST_PROFILE_PACKET 0
00038 #define LENGTH_PROFILE_PACKET 40
00039 #define LENGTH_GAME_RULES_PACKET 3
00040 #define LENGTH_REQUEST_GAME_RULES_UPDATE_PACKET 2
00041 #define LENGTH_FORCE_LEAVE_PACKET 1
00042
00043 #define NO_OP_PACKET 0x00
00044 #define CONNECTION_CLIENT_PACKET 0x01
00045 #define ACCEPTATION_PACKET 0x02
00046 #define DISCONNECTION_PACKET 0x03
00047 #define EVENT_PACKET 0x04
00048 #define END_GAME_PACKET 0x05
00049 #define CAN_START_PACKET 0x06
00050 #define CLIENT_READY_PACKET 0x07
00051 #define SPAWN_PLAYER_PACKET 0x08
00052 #define DEATH_PLAYER_PACKET 0x09
```

```
00053 #define WHOAMI_PACKET 0x0A
00054 #define SERVER_STATUS_PACKET 0x0B
00055 #define REQUEST_LOBBY_PACKET 0x0C
00056 #define SEND_LOBBY_CODE_PACKET 0x0D
00057 #define CONNECT_TO_LOBBY 0x0E
00058 #define LOBBY_MASTER_REQUEST_START 0x0F
00059 #define LOBBY_CONNECT_VALUE 0x10
00060 #define LEVEL_COMPLETE_PACKET 0x11
00061 #define NEXT_LEVEL_PACKET 0x12
00062 #define REGISTER_PACKET 0x13
00063 #define CONNECT_USER_PACKET 0x14
00064 #define LOGIN_PACKET 0x15
00065 #define GAME_STATE_BATCH_PACKET 0x16
00066 #define GAME_STATE_BATCH_COMPRESSED_PACKET 0x17
00067 #define GAME_STATE_COMPRESSED_PACKET 0x18
00068 #define REQUEST_LEADERBOARD_PACKET 0x19
00069 #define LEADERBOARD_PACKET 0x1A
00070 #define REGISTER_FAIL_PACKET 0x1B
00071 #define REQUEST_PROFILE_PACKET 0x1C
00072 #define PROFILE_PACKET 0x1D
00073 #define GAME_RULES_PACKET 0x1E
00074 #define REQUEST_GAME_RULES_UPDATE_PACKET 0x1F
00075 #define NEW_CHAT_PACKET 0x20
00076 #define BROADCASTED_CHAT_PACKET 0x21
00077 #define FORCE_LEAVE_PACKET 0x22
00078 #define LEAVE_LOBBY_PACKET 0x23
00079 #define ACK_LEAVE_LOBBY 0x24
00080
00081 namespace pm {
00082
00083     class IPacketManager {
00084     public:
00085         virtual ~IPacketManager() = default;
00086
00087         virtual uint32_t getLength() const = 0;
00088         virtual uint32_t getSequenceNumber() const = 0;
00089         virtual uint8_t getType() const = 0;
00090         virtual std::vector<uint64_t> getPayload() const = 0;
00091         virtual std::vector<std::vector<uint64_t» getBatchedPayloads() const = 0;
00092         virtual uint8_t getIdClient() const = 0;
00093
00094         virtual void setType(uint8_t type) = 0;
00095         virtual void setLength(uint32_t length) = 0;
00096         virtual void setSequenceNumber(uint32_t sequenceNumber) = 0;
00097         virtual void setPayload(std::vector<uint64_t> payload) = 0;
00098         virtual void setIdClient(uint8_t idClient) = 0;
00099
00100         virtual std::vector<uint64_t> formatString(const std::string str) = 0;
00101         virtual std::vector<uint8_t> pack(uint8_t idClient, uint32_t sequenceNumber, uint8_t type,
      std::vector<uint64_t> payload) = 0;
00102         virtual std::vector<uint8_t> packBatchedGameState(uint8_t idClient, uint32_t sequenceNumber,
      const std::vector<std::vector<uint64_t»& entities) = 0;
00103         virtual bool unpack(std::vector<uint8_t> data) = 0;
00104
00105         virtual void reset() = 0;
00106
00107         virtual void registerBuilder(uint8_t type,
      std::function<std::vector<uint8_t>(std::vector<uint64_t>)> builder) = 0;
00108         virtual void registerParser(uint8_t type, std::function<bool(const std::vector<uint8_t>)>
      parser) = 0;
00109         virtual void registerLength(uint8_t type, uint32_t length) = 0;
00110         virtual void
      registerGameStatePackFunction(std::function<std::vector<uint8_t>(std::vector<uint64_t>,
      std::shared_ptr<unsigned int>)> func) = 0;
00111         virtual void registerGameStateUnpackFunction(std::function<unsigned int(const
      std::vector<uint8_t>, unsigned int)> func) = 0;
00112         virtual void clearAllHandlers() = 0;
00113 };
00114 } // namespace pm
00115
00116 #endif /* !IPacketManager_HPP_ */
```

5.181 IWindow.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** IWindow
00006 */
00007
00008 #ifndef IWINDOW_HPP_
00009 #define IWINDOW_HPP_
```

```
00010
00011 #include <string>
00012 #include <utility>
00013 #include <cstdint>
00014 #include "../../common/types/FRect.hpp"
00015 #include "../../common/types/Vector2f.hpp"
00016
00017 namespace gfx {
00018
00019 struct color_t {
00020     uint8_t r;
00021     uint8_t g;
00022     uint8_t b;
00023     uint8_t a = 255;
00024 };
00025
00026 class IWindow {
00027     public:
00028         virtual ~IWindow() = default;
00029
00030         virtual void init() = 0;
00031         virtual void display() = 0;
00032         virtual void closeWindow() = 0;
00033         virtual bool isOpen() = 0;
00034         virtual void clear() = 0;
00035         virtual void resizeWindow(size_t x, size_t y) = 0;
00036
00037         virtual void drawSprite(std::string asset, color_t color, std::pair<size_t, size_t> position)
      = 0;
00038         virtual void drawText(std::string text, color_t color, std::pair<size_t, size_t> position,
      const std::string& fontPath, size_t fontSize = 24, color_t outlineColor = {0, 0, 0}, float
      outlineThickness = 0.0f) = 0;
00039         virtual std::pair<size_t, size_t> getTextSize(const std::string& text, const std::string&
      fontPath, size_t fontSize = 24) = 0;
00040         virtual void drawRectangleOutline(color_t color, std::pair<size_t, size_t> position,
      std::pair<size_t, size_t> size, size_t borderWidth = 5) = 0;
00041         virtual void drawFilledRectangle(color_t color, std::pair<size_t, size_t> position,
      std::pair<size_t, size_t> size) = 0;
00042         virtual void drawRoundedRectangleFilled(color_t color, std::pair<size_t, size_t> position,
      std::pair<size_t, size_t> size, float radius) = 0;
00043         virtual void drawRoundedRectangleOutline(color_t color, std::pair<size_t, size_t> position,
      std::pair<size_t, size_t> size, float radius) = 0;
00044
00045         virtual bool isMouseOver(std::pair<size_t, size_t> position, std::pair<size_t, size_t> size) =
      0;
00046         virtual std::pair<int, int> getWindowSize() = 0;
00047
00048         virtual void drawSprite(const std::string& texturePath, float x, float y, float scaleX = 1.0f,
      float scaleY = 1.0f, float rotation = 0.0f) = 0;
00049         virtual void drawSprite(const std::string& texturePath, float x, float y, const math::FRect
      frameRect, float scaleX = 1.0f, float scaleY = 1.0f, float rotation = 0.0f) = 0;
00050         virtual void drawSprite(const std::string& texturePath, float x, float y, float scaleX, float
      scaleY, float rotation, color_t color) = 0;
00051         virtual void drawSprite(const std::string& texturePath, float x, float y, const math::FRect
      frameRect, float scaleX, float scaleY, float rotation, color_t color) = 0;
00052
00053         virtual void updateView() = 0;
00054         virtual void setViewCenter(float x, float y) = 0;
00055         virtual math::Vector2f getViewCenter() = 0;
00056         virtual math::Vector2f mapPixelToCoords(int x, int y) = 0;
00057
00058         virtual std::pair<int, int> getLogicalSize() const = 0;
00059         virtual float getScaleFactor() const = 0;
00060
00061         virtual void addShaderFilter(const std::string& path) = 0;
00062         virtual void removeShaderFilter(const std::string& path) = 0;
00063         virtual void setShaderUniform(const std::string& filterPath, const std::string& name, float
      value) = 0;
00064         virtual void setFramerateLimit(unsigned int fps) = 0;
00065         virtual void setFullscreen(bool fullscreen) = 0;
00066         virtual void setRenderQuality(float quality) = 0;
00067         virtual void setCursor(bool isHand) = 0;
00068
00069         virtual std::string getClipboardText() = 0;
00070         virtual void setClipboardText(const std::string& text) = 0;
00071 };
00072
00073 typedef IWindow *(*createWindow_t)();
00074
00075 } // namespace gfx
00076
00077 #endif /* !IWINDOW_HPP_ */
```

5.182 GameStateHandlersOptimized.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** R-Type
00004 ** File description:
00005 ** Optimized Game state handlers using varint encoding for bandwidth reduction
00006 */
00007
00008 #ifndef GAME_STATE_HANDLERS_OPTIMIZED_HPP_
00009 #define GAME_STATE_HANDLERS_OPTIMIZED_HPP_
00010
00011 #include <memory>
00012 #include "../../libs/Packet/PacketManager.hpp"
00013
00014 namespace common::packet {
00015
00016 bool registerOptimizedGameStateHandlers(std::shared_ptr<pm::IPacketManager> packet);
00017 bool isOptimizedHandlersAvailable();
00018
00019 } // namespace common::packet
00020
00021 #endif /* !GAME_STATE_HANDLERS_OPTIMIZED_HPP_ */
```

5.183 AnimationConditionFactory.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** AnimationConditionFactory
00006 */
00007
00008 #ifndef ANIMATIONCONDITIONFACTORY_HPP_
00009 #define ANIMATIONCONDITIONFACTORY_HPP_
00010
00011 #include <functional>
00012 #include <string>
00013 #include <unordered_map>
00014 #include <memory>
00015 #include "../../ECS/entity/Entity.hpp"
00016 #include "../../ECS/entity/registry/Registry.hpp"
00017
00018 namespace ecs {
00019
00020 class AnimationConditionFactory {
00021     public:
00022         using ConditionFunction = std::function<bool(std::shared_ptr<Registry>, Entity)>;
00023
00024         static const AnimationConditionFactory& getInstance();
00025
00026         void registerCondition(const std::string& name, ConditionFunction condition);
00027         bool evaluateCondition(const std::string& name, std::shared_ptr<Registry> registry, Entity
      entity) const;
00028         bool hasCondition(const std::string& name) const;
00029         void unregisterCondition(const std::string& name);
00030         void clearConditions();
00031
00032         static bool getConditionValue(const std::string& param, std::shared_ptr<Registry> registry,
      Entity entity);
00033
00034     private:
00035         AnimationConditionFactory();
00036         void initializeConditions();
00037
00038         AnimationConditionFactory(const AnimationConditionFactory&) = delete;
00039         AnimationConditionFactory& operator=(const AnimationConditionFactory&) = delete;
00040
00041         std::unordered_map<std::string, ConditionFunction> _conditions;
00042 };
00043
00044 }  // namespace ecs
00045
00046 #endif /* !ANIMATIONCONDITIONFACTORY_HPP_ */
```

5.184 CollisionRulesParser.hpp

```
00001 #ifndef COLLISION_RULES_PARSER_HPP_
```

```
00002 #define COLLISION_RULES_PARSER_HPP_
00003
00004 #include <string>
00005 #include <map>
00006 #include <vector>
00007 #include <nlohmann/json.hpp>
00008 #include "../CollisionRules/CollisionRulesData.hpp"
00009
00010 namespace ecs {
00011
00012 class CollisionRulesParser {
00013     public:
00014         static CollisionRulesData parseFromFile(const std::string& filePath);
00015         static CollisionRulesData parseFromJsonString(const std::string& jsonString);
00016         static CollisionRulesData parseFromJson(const nlohmann::json& json);
00017
00018     private:
00019         static void parseRulesForType(
00020             const nlohmann::json& typeJson,
00021             std::shared_ptr<std::vector<CollisionRule» allowRules
00022         );
00023 };
00024
00025 }  // namespace ecs
00026
00027 #endif // COLLISION_RULES_PARSER_HPP_
```

5.185 ComponentMetadata.hpp

```
00001 #ifndef COMPONENTMETADATA_HPP_
00002 #define COMPONENTMETADATA_HPP_
00003
00004 #include <string>
00005 #include <vector>
00006 #include <map>
00007 #include <memory>
00008 #include <functional>
00009 #include <typeindex>
00010 #include "../ParserParam.hpp"
00011 #include "../../components/base/IComponent.hpp"
00012 #include "../../ECS/entity/Entity.hpp"
00013 #include "../../ECS/entity/registry/Registry.hpp"
00014
00015 namespace parser {
00016
00017 using ComponentCreator = std::function<std::shared_ptr<ecs::IComponent>(
00018     const std::map<std::string, std::shared_ptr<FieldValue» &
00019 )>;
00020
00021 using ComponentAdder = std::function<void(
00022     std::shared_ptr<ecs::Registry>, ecs::Entity, std::shared_ptr<ecs::IComponent>
00023 )>;
00024
00025 struct ComponentMetadata {
00026     std::string name;                   // Name of JSON component (eg: "TransformComponent")
00027     std::type_index typeIndex;          // typeid of component
00028     std::vector<Field> fields;          // List of parsable fields
00029     ComponentCreator creator;         // Creation lambda
00030     ComponentAdder adder;             // Adder to registry lambda
00031 };
00032
00033 }  // namespace parser
00034
00035 #endif  // COMPONENTMETADATA_HPP_
```

5.186 ComponentRegistrar.hpp

```
00001 #ifndef COMPONENTREGISTRAR_HPP_
00002 #define COMPONENTREGISTRAR_HPP_
00003
00004 #include "ComponentRegistry.hpp"
00005 #include "ComponentMetadata.hpp"
00006
00007 namespace parser {
00008
00009 template<typename T>
00010 class ComponentRegistrar {
00011     public:
```

```
00012         ComponentRegistrar(
00013             const std::string& name,
00014             const std::vector<Field>& fields,
00015             const ComponentCreator& creator
00016         ) {
00017             ComponentMetadata metadata{
00018                 name,
00019                 std::type_index(typeid(T)),
00020                 fields,
00021                 creator,
00022                 [](
00023                     std::shared_ptr<ecs::Registry> registry,
00024                     ecs::Entity entity,
00025                     std::shared_ptr<ecs::IComponent> component
00026                 ) {
00027                     auto typedComponent = std::static_pointer_cast<T>(component);
00028                     auto cloned = std::make_shared<T>(*typedComponent);
00029                     registry->addComponent(entity, cloned);
00030                 }
00031             };
00032
00033             ComponentRegistry::getInstance().registerComponent(metadata);
00034         }
00035 };
00036
00037 template<typename T>
00038 class TagComponentRegistrar {
00039     public:
00040         explicit TagComponentRegistrar(const std::string& name) {
00041             ComponentMetadata metadata{
00042                 name,
00043                 std::type_index(typeid(T)),
00044                 {{"target", FieldType::STRING, false, nullptr}},
00045                 []([[maybe_unused]] const auto& fields) {
00046                     return std::make_shared<T>();
00047                 },
00048                 [](
00049                     std::shared_ptr<ecs::Registry> registry,
00050                     ecs::Entity entity,
00051                     std::shared_ptr<ecs::IComponent> component
00052                 ) {
00053                     registry->addComponent(entity, std::static_pointer_cast<T>(component));
00054                 }
00055             };
00056
00057             ComponentRegistry::getInstance().registerComponent(metadata);
00058         }
00059 };
00060
00061 }  // namespace parser
00062
00063 #define REGISTER_COMPONENT(Type, Name, ...) \
00064     namespace { \
00065         static parser::ComponentRegistrar<Type> CONCAT(_registrar_, __COUNTER__)(Name, __VA_ARGS__); \
00066     }
00067
00068 #define REGISTER_TAG_COMPONENT(Type, Name) \
00069     namespace { \
00070         static parser::TagComponentRegistrar<Type> CONCAT(_tag_registrar_, __COUNTER__)(Name); \
00071     }
00072
00073 #define CONCAT_IMPL(x, y) x##y
00074 #define CONCAT(x, y) CONCAT_IMPL(x, y)
00075
00076 #endif  // COMPONENTREGISTRAR_HPP_
```

5.187 ComponentRegistry.hpp

```
00001 #ifndef COMPONENTREGISTRY_HPP_
00002 #define COMPONENTREGISTRY_HPP_
00003
00004 #include <map>
00005 #include <string>
00006 #include <memory>
00007 #include <typeindex>
00008 #include "ComponentMetadata.hpp"
00009
00010 namespace parser {
00011
00012 class ComponentRegistry {
00013 public:
00014     static ComponentRegistry& getInstance();
00015
```

```
00016     void registerComponent(const ComponentMetadata &metadata);
00017
00018     bool hasComponent(const std::string& name) const;
00019
00020     const std::map<std::string, ComponentMetadata> &getAllComponents() const;
00021
00022     std::shared_ptr<std::map<std::string, std::pair<std::type_index, std::vector<Field>>>
      getComponentDefinitions() const;
00023     std::map<std::type_index, ComponentCreator> getComponentCreators() const;
00024     std::map<std::type_index, ComponentAdder> getComponentAdders() const;
00025
00026 private:
00027     ComponentRegistry() = default;
00028     ComponentRegistry(const ComponentRegistry&) = delete;
00029     ComponentRegistry& operator=(const ComponentRegistry&) = delete;
00030
00031     std::map<std::string, ComponentMetadata> _components;
00032 };
00033
00034 }  // namespace parser
00035
00036 #endif  // COMPONENTREGISTRY_HPP_
```

5.188 ComposantParser.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ComposantParser
00006 */
00007
00008 #ifndef COMPOSANTPARSER_HPP_
00009 #define COMPOSANTPARSER_HPP_
00010
00011 #include <string>
00012 #include <memory>
00013 #include <map>
00014 #include <typeindex>
00015 #include "../../components/base/IComponent.hpp"
00016 #include "../ParserParam.hpp"
00017 #include <nlohmann/json.hpp>
00018 #include <functional>
00019
00020 class ComposantParser {
00021     public:
00022         using ShouldParseComponentCallback = std::function<bool(const std::map<std::string,
      std::shared_ptr<FieldValue>&)>;
00023
00024         ComposantParser(
00025             std::shared_ptr<const std::map<std::string, std::pair<std::type_index, std::vector<Field>>>
      componentDefinitions,
00026             const std::map<std::type_index, ComponentCreator> &componentCreators,
00027             const ShouldParseComponentCallback &shouldParseCallback = nullptr
00028         );
00029         ~ComposantParser();
00030
00031         std::pair<std::shared_ptr<ecs::IComponent>, std::type_index> parseComponent(const std::string
      &componentName, const nlohmann::json &componentData);
00032
00033     protected:
00034     private:
00035         std::shared_ptr<FieldValue> parseFieldValue(const nlohmann::json &jsonValue, FieldType type);
00036         std::shared_ptr<const std::map<std::string, std::pair<std::type_index, std::vector<Field>>>
      _componentDefinitions;
00037         const std::map<std::type_index, ComponentCreator> &_componentCreators;
00038         ShouldParseComponentCallback _shouldParseCallback;
00039 };
00040
00041 #endif /* !COMPOSANTPARSER_HPP_ */
```

5.189 EntityParser.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** EntityParser
```

```
00006 */
00007
00008 #ifndef ENTITYPARSER_HPP_
00009 #define ENTITYPARSER_HPP_
00010
00011 #include <string>
00012 #include <vector>
00013 #include <memory>
00014 #include <map>
00015 #include "../ParserParam.hpp"
00016 #include "../ComposantParser/ComposantParser.hpp"
00017 #include "../../Prefab/IPrefab.hpp"
00018 #include "../../Prefab/ParsedEntityPrefab.hpp"
00019 #include <nlohmann/json.hpp>
00020
00021 class EntityParser {
00022     public:
00023         using ShouldParseComponentCallback = ComposantParser::ShouldParseComponentCallback;
00024         EntityParser(
00025             std::shared_ptr<const std::map<std::string, std::pair<std::type_index, std::vector<Field>>>
      componentDefinitions,
00026             const std::map<std::type_index, ComponentCreator> &componentCreators,
00027             const std::map<std::type_index, ComponentAdder> &componentAdders,
00028             const ShouldParseComponentCallback &shouldParseCallback = nullptr
00029         );
00030         ~EntityParser();
00031
00032         std::shared_ptr<IPrefab> parseEntity(const std::string &filePath);
00033
00034     protected:
00035     private:
00036         ComposantParser _composantParser;
00037         const std::map<std::type_index, ComponentAdder>& _componentAdders;
00038         ShouldParseComponentCallback _shouldParseCallback;
00039 };
00040
00041 #endif /* !ENTITYPARSER_HPP_ */
```

5.190 MapHandler.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** MapHandler
00006 */
00007
00008 #ifndef MAPHANDLER_HPP_
00009 #define MAPHANDLER_HPP_
00010
00011 #include <string>
00012 #include <vector>
00013 #include <memory>
00014 #include <nlohmann/json.hpp>
00015
00016 struct MapData {
00017     int index;
00018     std::string name;
00019     std::string filePath;
00020     nlohmann::json jsonData;
00021 };
00022
00023 class MapHandler {
00024     public:
00025         MapHandler();
00026         ~MapHandler();
00027
00028         void parseAllLevels(const std::string& directoryPath);
00029
00030         const MapData& getCurrentMap() const;
00031
00032         nlohmann::json getCurrentMapJson() const;
00033
00034         size_t getCurrentMapIndex() const;
00035
00036         size_t getTotalMaps() const;
00037
00038         bool hasMaps() const;
00039
00040         bool isLastMap() const;
00041
00042         bool advanceToNextMap();
00043
```

```
00044         bool hasCompletedAllMaps() const;
00045
00046         void resetToFirstMap();
00047
00048         const std::vector<MapData>& getAllMaps() const;
00049
00050     private:
00051         std::vector<MapData> _maps;
00052         size_t _currentMapIndex;
00053         bool _completedAllMaps;
00054
00055         void sortMapsByIndex();
00056 };
00057
00058 #endif /* !MAPHANDLER_HPP_ */
```

5.191 MapParser.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** MapParser
00006 */
00007
00008 #ifndef MAPPARSER_HPP_
00009 #define MAPPARSER_HPP_
00010
00011 #include <string>
00012 #include <memory>
00013 #include <vector>
00014 #include <nlohmann/json.hpp>
00015 #include "../../ECS/entity/registry/Registry.hpp"
00016 #include "../../ECS/entity/EntityCreationContext.hpp"
00017 #include "../../Prefab/entityPrefabManager/EntityPrefabManager.hpp"
00018 #include "../../constants.hpp"
00019 #include "../../types/Vector2f.hpp"
00020
00021 class MapParser {
00022     public:
00023         MapParser(
00024             std::shared_ptr<EntityPrefabManager> prefabManager,
00025             std::shared_ptr<ecs::Registry> registry
00026         );
00027         ~MapParser();
00028
00029         void parseMapFromFile(const std::string& filePath);
00030         void parseMap(const nlohmann::json& mapJson);
00031
00032         void generateMapEntities();
00033
00034         nlohmann::json getMapJson() const;
00035         void setMapJson(const nlohmann::json& mapJson);
00036
00037         void setCreationContext(const ecs::EntityCreationContext& context);
00038         ecs::EntityCreationContext getCreationContext() const;
00039
00040     private:
00041         std::shared_ptr<EntityPrefabManager> _prefabManager;
00042         std::shared_ptr<ecs::Registry> _registry;
00043         ecs::EntityCreationContext _creationContext;
00044         nlohmann::json _mapJson;
00045
00046         void createBackgroundEntity(const std::string &entityName);
00047         void createMusicEntity(const std::string &prefabName);
00048         void createGameZoneEntity(float scrollSpeed);
00049         void createGameEndEntity(float mapLength);
00050         void createGameZoneStopEntity(float stopAtX);
00051
00052         void parsePowerUps(const nlohmann::json &powerUps);
00053         void parseObstacles(const nlohmann::json &obstacles);
00054         void parseWaves(const nlohmann::json &waves);
00055
00056         std::vector<float> getPositionsFromDistrib(
00057             int count,
00058             const nlohmann::json &distribution,
00059             float limit
00060         );
00061         ecs::Entity createEntityFromPrefab(
00062             const std::string &prefabName,
00063             float x, float y
00064         );
00065 };
```

```
00066
00067 #endif /* !MAPPARSER_HPP_ */
```

5.192 Parser.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** Parser
00006 */
00007
00008 #ifndef PARSER_HPP_
00009 #define PARSER_HPP_
00010
00011 #include "../Prefab/IPrefab.hpp"
00012 #include <memory>
00013 #include "./EntityParser/EntityParser.hpp"
00014 #include "../Prefab/entityPrefabManager/EntityPrefabManager.hpp"
00015 #include "ParserParam.hpp"
00016 #include "./MapParser/MapParser.hpp"
00017 #include "../common/ECS/entity/registry/Registry.hpp"
00018
00019 typedef enum {
00020     CLIENT = 0,
00021     SERVER = 1
00022 } ParsingType;
00023
00024 class Parser {
00025     public:
00026         Parser(std::shared_ptr<EntityPrefabManager> prefab, ParsingType type,
00027             std::shared_ptr<ecs::Registry> registry);
00028         ~Parser();
00029
00030         std::shared_ptr<EntityPrefabManager> getPrefabManager() const;
00031         void setPrefabManager(std::shared_ptr<EntityPrefabManager> prefab);
00032         void parseAllEntities(std::string directoryPath);
00033         void parseEntity(std::string entityPath);
00034
00035         const std::map<std::type_index, ComponentAdder>& getComponentAdders() const;
00036         ParsingType getParsingType() const;
00037         bool isClientParsing() const;
00038         bool isServerParsing() const;
00039         bool shouldParseComponent(std::map<std::string, std::shared_ptr<FieldValue>> fields) const;
00040
00041         void parseMapFromFile(const std::string& filePath);
00042         void parseMapFromJson(const nlohmann::json& mapJson);
00043
00044         std::shared_ptr<MapParser> getMapParser() const;
00045         void setRegistry(std::shared_ptr<ecs::Registry> registry);
00046
00047     private:
00048         std::shared_ptr<EntityParser> _entityParser;
00049         std::shared_ptr<MapParser> _mapParser;
00050         std::shared_ptr<EntityPrefabManager> _prefabManager;
00051
00052         std::shared_ptr<std::map<std::string, std::pair<std::type_index, std::vector<Field>>>
      _componentDefinitions;
00053         std::map<std::type_index, ComponentCreator> _componentCreators;
00054         std::map<std::type_index, ComponentAdder> _componentAdders;
00055         ParsingType _parsingType;
00056 };
00057
00058 #endif /* !PARSER_HPP_ */
```

5.193 ParserParam.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ParserParam
00006 */
00007
00008 #ifndef PARSERPARAM_HPP_
00009 #define PARSERPARAM_HPP_
00010
00011 #include <string>
```

```
00012 #include <vector>
00013 #include <map>
00014 #include <variant>
00015 #include <functional>
00016 #include <memory>
00017 #include "../types/Vector2f.hpp"
00018 #include <nlohmann/json.hpp>
00019 #include "../components/base/IComponent.hpp"
00020 #include "../components/permanent/TransformComponent.hpp"
00021 #include "../components/permanent/VelocityComponent.hpp"
00022 #include "../components/permanent/SpeedComponent.hpp"
00023 #include "../../client/components/rendering/SpriteComponent.hpp"
00024 #include "../../client/components/rendering/AnimationComponent.hpp"
00025 #include "../components/tags/ControllableTag.hpp"
00026 #include "../components/tags/PlayerTag.hpp"
00027 #include "../components/permanent/ColliderComponent.hpp"
00028
00029 enum class ParserParam {
00030     NONE = 0,
00031     NAME = 1,
00032     COMPONENTS = 2,
00033 };
00034
00035 enum class FieldType {
00036     VECTOR2F = 0,
00037     FLOAT = 1,
00038     STRING = 2,
00039     INT = 3,
00040     BOOL = 4,
00041     OBJECT = 5,
00042     JSON = 6,
00043     UNDEFINED = 7
00044 };
00045
00046 using FieldValueMap = std::map<std::string, std::shared_ptr<struct FieldValue»;
00047 using FieldValueVariant = std::variant<math::Vector2f, float, std::string, int, bool, FieldValueMap,
      nlohmann::json>;
00048
00049 struct FieldValue : FieldValueVariant {
00050     using FieldValueVariant::FieldValueVariant;
00051     using FieldValueVariant::operator=;
00052
00053     template<typename T>
00054     FieldValue(T&& value) : FieldValueVariant(std::forward<T>(value)) {}
00055 };
00056
00057 struct Field {
00058     std::string name = "";
00059     FieldType type;
00060     bool optional = false;
00061     std::shared_ptr<FieldValue> defaultValue = nullptr;
00062
00063     Field(std::string n, FieldType t, bool opt = false, std::shared_ptr<FieldValue> def = nullptr)
00064         : name(std::move(n)), type(t), optional(opt), defaultValue(std::move(def)) {}
00065 };
00066
00067 #include <typeindex>
00068 #include "../ECS/entity/registry/Registry.hpp"
00069 #include "../ECS/entity/Entity.hpp"
00070
00071 using ComponentCreator = std::function<std::shared_ptr<ecs::IComponent>(const std::map<std::string,
      std::shared_ptr<FieldValue»&)>;
00072 using ComponentAdder = std::function<void(std::shared_ptr<ecs::Registry>, ecs::Entity,
      std::shared_ptr<ecs::IComponent>)>;
00073
00074 #endif /* !PARSERPARAM_HPP_ */
```

5.194 JsonLoader.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** JsonLoader
00006 */
00007
00008 #ifndef JSONLOADER_HPP_
00009 #define JSONLOADER_HPP_
00010
00011 #include <string>
00012 #include <optional>
00013 #include <nlohmann/json.hpp>
00014
```

```
00015 namespace parser {
00016
00017 class JsonLoader {
00018     public:
00019         struct LoadResult {
00020             nlohmann::json data;
00021             bool success;
00022             std::string errorMessage;
00023         };
00024
00025         static nlohmann::json loadFromFile(const std::string& filePath);
00026
00027         static LoadResult tryLoadFromFile(const std::string& filePath) noexcept;
00028
00029         static nlohmann::json parseFromString(
00030             const std::string& jsonString,
00031             const std::string& sourceName = "string"
00032         );
00033
00034         static LoadResult tryParseFromString(const std::string& jsonString) noexcept;
00035
00036         static bool fileExists(const std::string& filePath);
00037
00038         static std::string getExtension(const std::string& filePath);
00039
00040     private:
00041         JsonLoader() = delete;
00042 };
00043
00044 }  // namespace parser
00045
00046 #endif /* !JSONLOADER_HPP_ */
```

5.195 JsonValidation.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** JsonValidation
00006 */
00007
00008 #ifndef JSONVALIDATION_HPP_
00009 #define JSONVALIDATION_HPP_
00010
00011 #include <string>
00012 #include <vector>
00013 #include <optional>
00014 #include <nlohmann/json.hpp>
00015
00016 namespace parser {
00017
00018 struct ValidationResult {
00019     bool valid;
00020     std::vector<std::string> errors;
00021     std::vector<std::string> warnings;
00022
00023     operator bool() const { return valid; }
00024 };
00025
00026 class JsonValidation {
00027 public:
00028     static ValidationResult hasRequiredFields(
00029         const nlohmann::json& json,
00030         const std::vector<std::string>& requiredFields,
00031         const std::string& contextName = ""
00032     );
00033
00034     static bool hasFieldOfType(
00035         const nlohmann::json& json,
00036         const std::string& fieldName,
00037         const std::string& expectedType
00038     );
00039
00040     template<typename T>
00041     static T getOrDefault(
00042         const nlohmann::json& json,
00043         const std::string& fieldName,
00044         const T& defaultValue
00045     ) {
00046         if (json.contains(fieldName)) {
00047             try {
00048                 return json[fieldName].get<T>();
```

```
00049             } catch (...) {
00050                 return defaultValue;
00051             }
00052         }
00053         return defaultValue;
00054     }
00055
00056     static std::optional<nlohmann::json> getNestedField(
00057         const nlohmann::json& json,
00058         const std::string& path
00059     );
00060
00061 private:
00062     JsonValidation() = delete;
00063 };
00064
00065 }  // namespace parser
00066
00067 #endif /* !JSONVALIDATION_HPP_ */
```

5.196 pch.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2026
00003 ** ryanR-type
00004 ** File description:
00005 ** Precompiled Header
00006 */
00007
00008 #ifndef PCH_HPP_
00009 #define PCH_HPP_
00010
00011 // ============================================================================
00012 // C++ Standard Library Headers
00013 // ============================================================================
00014
00015 // Containers
00016 #include <array>
00017 #include <deque>
00018 #include <forward_list>
00019 #include <list>
00020 #include <map>
00021 #include <queue>
00022 #include <set>
00023 #include <stack>
00024 #include <unordered_map>
00025 #include <unordered_set>
00026 #include <vector>
00027
00028 // Strings
00029 #include <string>
00030 #include <string_view>
00031
00032 // Memory
00033 #include <memory>
00034 #include <memory_resource>
00035
00036 // Utilities
00037 #include <algorithm>
00038 #include <functional>
00039 #include <optional>
00040 #include <tuple>
00041 #include <utility>
00042 #include <variant>
00043
00044 // Type traits and concepts
00045 #include <concepts>
00046 #include <type_traits>
00047
00048 // Time
00049 #include <chrono>
00050
00051 // I/O
00052 #include <fstream>
00053 #include <iostream>
00054 #include <sstream>
00055
00056 // Numeric
00057 #include <cmath>
00058 #include <limits>
00059 #include <numeric>
00060
00061 // Threading (if used)
```

```
00062 #include <atomic>
00063 #include <mutex>
00064 #include <thread>
00065
00066 // Error handling
00067 #include <exception>
00068 #include <stdexcept>
00069
00070 // C compatibility
00071 #include <cassert>
00072 #include <cstddef>
00073 #include <cstdint>
00074 #include <cstdlib>
00075 #include <cstring>
00076
00077 // ============================================================================
00078 // Third-party Libraries (stable headers)
00079 // ============================================================================
00080
00081 // nlohmann/json
00082 #include <nlohmann/json.hpp>
00083
00084 #endif  // PCH_HPP_
```

5.197 APrefab.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** APrefab
00006 */
00007
00008 #ifndef APREFAB_HPP_
00009 #define APREFAB_HPP_
00010
00011 #include "IPrefab.hpp"
00012 #include "../ECS/entity/registry/Registry.hpp"
00013 #include "../ECS/entity/factory/EntityFactory.hpp"
00014
00015 class APrefab : public IPrefab {
00016     public:
00017         APrefab() = default;
00018         virtual ~APrefab() = default;
00019
00020         ecs::Entity instantiate(
00021             const std::shared_ptr<ecs::Registry>& registry,
00022             const std::shared_ptr<ecs::IEntityFactory>& factory,
00023             const ecs::EntityCreationContext& context = ecs::EntityCreationContext::forLocalClient()
00024         ) override;
00025
00026         ecs::Entity instantiate(const std::shared_ptr<ecs::Registry>& registry) override;
00027 };
00028
00029 #endif /* !APREFAB_HPP_ */
```

5.198 EntityPrefabManager.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** EntityPrefabManager
00006 */
00007
00008 #ifndef ENTITYPREFABMANAGER_HPP_
00009 #define ENTITYPREFABMANAGER_HPP_
00010
00011 #include <vector>
00012 #include <string>
00013 #include <map>
00014 #include <memory>
00015 #include <functional>
00016 #include "../../ECS/entity/Entity.hpp"
00017 #include "../../ECS/entity/EntityCreationContext.hpp"
00018 #include "../../ECS/entity/factory/IEntityFactory.hpp"
00019 #include "../../ECS/entity/factory/EntityFactory.hpp"
00020 #include "../IPrefab.hpp"
```

```
00021
00022 class EntityPrefabManager
00023 {
00024     public:
00025         EntityPrefabManager();
00026         ~EntityPrefabManager();
00027
00028         void registerPrefab(const std::string &name, const std::shared_ptr<IPrefab> &prefab);
00029         std::shared_ptr<IPrefab> getPrefab(const std::string &name) const;
00030
00031         ecs::Entity createEntityFromPrefab(
00032             const std::string &prefabName,
00033             const std::shared_ptr<ecs::Registry> &registry,
00034             const ecs::EntityCreationContext &context
00035         );
00036
00037         ecs::Entity createEntityFromPrefab(
00038             const std::string &prefabName,
00039             const std::shared_ptr<ecs::Registry> &registry
00040         );
00041
00042         bool hasPrefab(const std::string &name) const;
00043         void deletePrefab(const std::string &name);
00044         void clearPrefabs();
00045
00046         std::shared_ptr<ecs::IEntityFactory> getEntityFactory() const;
00047         void setEntityFactory(std::shared_ptr<ecs::IEntityFactory> factory);
00048
00049         void setOnEntityCreated(std::function<void(ecs::Entity, const std::string&)> callback);
00050
00051     private:
00052         std::map<std::string, std::shared_ptr<IPrefab» _prefabs;
00053         std::shared_ptr<ecs::IEntityFactory> _entityFactory;
00054         std::function<void(ecs::Entity, const std::string&)> _onEntityCreated;
00055 };
00056
00057 #endif /* !ENTITYPREFABMANAGER_HPP_ */
```

5.199 IPrefab.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** IPrefab
00006 */
00007
00008 #ifndef IPREFAB_HPP_
00009 #define IPREFAB_HPP_
00010 #include <memory>
00011 #include "../ECS/entity/registry/Registry.hpp"
00012 #include "../ECS/entity/Entity.hpp"
00013 #include "../ECS/entity/EntityCreationContext.hpp"
00014 #include "../ECS/entity/factory/IEntityFactory.hpp"
00015
00016 class IPrefab {
00017     public:
00018         virtual ~IPrefab() = default;
00019
00020         virtual ecs::Entity instantiate(
00021             const std::shared_ptr<ecs::Registry>& registry,
00022             const std::shared_ptr<ecs::IEntityFactory>& factory,
00023             const ecs::EntityCreationContext& context = ecs::EntityCreationContext::forLocalClient()
00024         ) = 0;
00025
00026         virtual ecs::Entity instantiate(const std::shared_ptr<ecs::Registry>& registry) = 0;
00027 };
00028
00029 #endif /* !IPREFAB_HPP_ */
```

5.200 ParsedEntityPrefab.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ParsedEntityPrefab
00006 */
```

```
00007
00008 #ifndef PARSEDENTITYPREFAB_HPP_
00009 #define PARSEDENTITYPREFAB_HPP_
00010
00011 #include "IPrefab.hpp"
00012 #include <vector>
00013 #include <memory>
00014 #include <string>
00015 #include <typeindex>
00016 #include <map>
00017 #include <functional>
00018 #include "../components/base/IComponent.hpp"
00019 #include "../ECS/entity/registry/Registry.hpp"
00020 #include "../ECS/entity/EntityCreationContext.hpp"
00021 #include "../ECS/entity/factory/IEntityFactory.hpp"
00022 #include "../Parser/ParserParam.hpp"
00023
00024 class ParsedEntityPrefab : public IPrefab {
00025     public:
00026         ParsedEntityPrefab(const std::string& name, const std::map<std::type_index, ComponentAdder>&
      adders);
00027         ~ParsedEntityPrefab();
00028
00029         void addComponent(std::shared_ptr<ecs::IComponent> component, std::type_index typeIndex);
00030         const std::vector<std::shared_ptr<ecs::IComponent>>& getComponents() const;
00031         std::string getName() const;
00032
00033         ecs::Entity instantiate(
00034             const std::shared_ptr<ecs::Registry>& registry,
00035             const std::shared_ptr<ecs::IEntityFactory>& factory,
00036             const ecs::EntityCreationContext& context = ecs::EntityCreationContext::forLocalClient()
00037         ) override;
00038
00039         ecs::Entity instantiate(const std::shared_ptr<ecs::Registry>& registry) override;
00040
00041     private:
00042         std::string _name;
00043         std::vector<std::pair<std::shared_ptr<ecs::IComponent>, std::type_index>> _components;
00044         const std::map<std::type_index, ComponentAdder>& _componentAdders;
00045
00046         void addParsedComponents(
00047             const std::shared_ptr<ecs::Registry>& registry,
00048             ecs::Entity entity
00049         );
00050 };
00051
00052 #endif /* !PARSEDENTITYPREFAB_HPP_ */
```

5.201 ResourceManager.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ResourceManager
00006 */
00007
00008 #ifndef RESOURCEMANAGER_HPP_
00009 #define RESOURCEMANAGER_HPP_
00010
00011 #include <unordered_map>
00012 #include <memory>
00013
00014
00015 class ResourceManager {
00016     public:
00017         template<typename T>
00018         void add(std::shared_ptr<T> resource);
00019
00020         template<typename T>
00021         std::shared_ptr<T> get();
00022
00023         template<typename T>
00024         bool has();
00025
00026         void clear() {
00027             resources.clear();
00028         }
00029
00030         template<typename T>
00031         void remove() {
00032             resources.erase(typeid(T).hash_code());
00033         }
```

```
00034     private:
00035         std::unordered_map<size_t, std::shared_ptr<void» resources;
00036 };
00037
00038
00039 #include "ResourceManager.tpp"
00040
00041 #endif /* !RESOURCEMANAGER_HPP_ */
```

5.202 Signal.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** Signal
00006 */
00007
00008 #ifndef SIGNAL_HPP_
00009 #define SIGNAL_HPP_
00010
00011 #ifdef _WIN32
00012     #ifndef _WIN32_WINNT
00013         #define _WIN32_WINNT 0x0A00
00014     #endif
00015
00016     #ifndef WIN32_LEAN_AND_MEAN
00017         #define WIN32_LEAN_AND_MEAN
00018     #endif
00019 #endif
00020
00021 #include <csignal>
00022
00023 class Signal {
00024     public:
00025         Signal();
00026         ~Signal();
00027
00028         static volatile sig_atomic_t stopFlag;
00029         static void signalHandler(int signum);
00030         static void setupSignalHandlers();
00031
00032     protected:
00033     private:
00034 };
00035
00036 #endif /* !SIGNAL_HPP_ */
```

5.203 SpatialGrid.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** SpatialGrid
00006 */
00007
00008 #ifndef SPATIALGRID_HPP_
00009 #define SPATIALGRID_HPP_
00010
00011 #include <vector>
00012 #include <unordered_set>
00013 #include <memory>
00014 #include <cmath>
00015 #include <cstddef>
00016 #include "../types/FRect.hpp"
00017 #include "../types/Vector2f.hpp"
00018 #include "../constants.hpp"
00019
00020 namespace ecs {
00021
00022 using Entity = size_t;
00023
00024 class SpatialGrid {
00025     public:
00026         SpatialGrid(
00027             float worldWidth = constants::MAX_WIDTH,
00028             float worldHeight = constants::MAX_HEIGHT,
```

```
00029             float cellSize = constants::SPATIAL_GRID_CELL_SIZE,
00030             float padding = constants::SPATIAL_GRID_PADDING
00031         );
00032         ~SpatialGrid() = default;
00033
00034         void clear();
00035         void insert(Entity entityId, const math::FRect& bounds);
00036         std::vector<Entity> query(const math::FRect& bounds) const;
00037         std::vector<std::pair<Entity, Entity» getPotentialPairs() const;
00038         void setCellSize(float cellSize);
00039         void setOffset(float offsetX, float offsetY);
00040
00041         float getCellSize() const { return _cellSize; }
00042         size_t getNumCols() const { return _numCols; }
00043         size_t getNumRows() const { return _numRows; }
00044         float getOffsetX() const { return _offsetX; }
00045         float getOffsetY() const { return _offsetY; }
00046
00047     private:
00048         size_t getCellIndex(float x, float y) const;
00049         std::vector<size_t> getCellIndices(const math::FRect& bounds) const;
00050
00051         float _worldWidth;
00052         float _worldHeight;
00053         float _cellSize;
00054         float _padding;
00055         float _offsetX;
00056         float _offsetY;
00057         size_t _numCols;
00058         size_t _numRows;
00059         std::vector<std::vector<Entity» _cells;
00060 };
00061
00062 }
00063
00064 #endif /* !SPATIALGRID_HPP_ */
```

5.204 ASystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ASystem
00006 */
00007
00008 #ifndef ASystem_HPP_
00009 #define ASystem_HPP_
00010
00011 #include <memory>
00012
00013 #include "ISystem.hpp"
00014 #include "../../resourceManager/ResourceManager.hpp"
00015 #include "../../ECS/entity/registry/Registry.hpp"
00016
00017 namespace ecs {
00018
00019 class ASystem : public ISystem {
00020     public:
00021         ASystem();
00022         ~ASystem() = default;
00023         void updateSystem(std::shared_ptr<ResourceManager> resourceManager, std::shared_ptr<Registry>
      registry, float deltaTime) override;
00024
00025     protected:
00026         virtual void update(std::shared_ptr<ResourceManager> resourceManager,
      std::shared_ptr<Registry> registry, float deltaTime) = 0;
00027
00028     private:
00029 };
00030
00031 } // namespace ecs
00032
00033 #endif /* !ASystem_HPP_ */
```

5.205 ISystem.hpp

```
00001 /*
```

```
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ISystem
00006 */
00007
00008 #ifndef ISystem_HPP_
00009 #define ISystem_HPP_
00010
00011 #include "../../resourceManager/ResourceManager.hpp"
00012 #include "../../ECS/entity/registry/Registry.hpp"
00013 #include <memory>
00014
00015 namespace ecs {
00016
00017 class ISystem {
00018     public:
00019         virtual ~ISystem() = default;
00020         virtual void updateSystem(std::shared_ptr<ResourceManager> resourceManager,
      std::shared_ptr<Registry> registry, float deltaTime) = 0;
00021 };
00022
00023 } // namespace ecs
00024
00025 #endif /* !ISystem_HPP_ */
```

5.206 GameZoneStopSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** GameZoneStopSystem
00006 */
00007
00008 #ifndef GAMEZONESTOPSYSTEM_HPP_
00009 #define GAMEZONESTOPSYSTEM_HPP_
00010
00011 #include "../base/ASystem.hpp"
00012
00013 namespace ecs {
00014
00015 class GameZoneStopSystem : public ASystem {
00016     public:
00017         GameZoneStopSystem();
00018         ~GameZoneStopSystem() = default;
00019
00020         void update(
00021             std::shared_ptr<ResourceManager> resourceManager,
00022             std::shared_ptr<Registry> registry,
00023             float deltaTime
00024         ) override;
00025 };
00026
00027 }
00028
00029 #endif /* !GAMEZONESTOPSYSTEM_HPP_ */
```

5.207 OutOfBoundsSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** OutOfBoundsSystem
00006 */
00007
00008 #ifndef OUTOFBOUNDSSYSTEM_HPP_
00009 #define OUTOFBOUNDSSYSTEM_HPP_
00010
00011 #include "../base/ASystem.hpp"
00012
00013 namespace ecs {
00014
00015 class OutOfBoundsSystem : public ASystem {
00016     public:
00017         OutOfBoundsSystem();
00018         ~OutOfBoundsSystem() = default;
```

```
00019
00020         void update(
00021             std::shared_ptr<ResourceManager> resourceManager,
00022             std::shared_ptr<Registry> registry,
00023             float deltaTime
00024         ) override;
00025
00026     private:
00027         float _margin;
00028 };
00029
00030 }
00031
00032 #endif /* !OUTOFBOUNDSSYSTEM_HPP_ */
```

5.208 DeathSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** DeathSystem
00006 */
00007
00008 #ifndef DEATHSYSTEM_HPP_
00009 #define DEATHSYSTEM_HPP_
00010
00011 #include "../base/ASystem.hpp"
00012 #include <memory>
00013 #include "../../types/Vector2f.hpp"
00014
00015 namespace ecs {
00016
00017 class DeathSystem : public ASystem {
00018     public:
00019         DeathSystem();
00020         ~DeathSystem() = default;
00021
00022         void update(
00023             std::shared_ptr<ResourceManager> resourceManager,
00024             std::shared_ptr<Registry> registry,
00025             float deltaTime
00026         ) override;
00027 };
00028
00029 }
00030
00031 #endif /* !DEATHSYSTEM_HPP_ */
```

5.209 HealthSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** HealthSystem
00006 */
00007
00008 #ifndef HEALTHSYSTEM_HPP_
00009 #define HEALTHSYSTEM_HPP_
00010
00011 #include "../base/ASystem.hpp"
00012 #include <memory>
00013
00014 namespace ecs {
00015
00016 class HealthSystem : public ASystem {
00017     public:
00018         HealthSystem();
00019         ~HealthSystem() override = default;
00020
00021         void update(
00022             std::shared_ptr<ResourceManager> resourceManager,
00023             std::shared_ptr<Registry> registry,
00024             float deltaTime
00025         ) override;
00026
00027     private:
```

```
00028         void _handleDamageUpdates(std::shared_ptr<ResourceManager> resourceManager,
      std::shared_ptr<Registry> registry);
00029         void _handleHealthUpdates(std::shared_ptr<Registry> registry);
00030 };
00031
00032 }
00033
00034 #endif /* !HEALTHSYSTEM_HPP_ */
```

5.210 InputNormalizer.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** InputNormalizer
00006 */
00007
00008 #ifndef INPUTNORMALIZER_HPP_
00009 #define INPUTNORMALIZER_HPP_
00010
00011 #include "../../types/Vector2f.hpp"
00012
00013 namespace ecs {
00014
00015 class InputNormalizer {
00016     public:
00017         static math::Vector2f normalizeDirection(const math::Vector2f &direction);
00018         static math::Vector2f normalizeAnalogInput(float rawX, float rawY);
00019 };
00020
00021 } // namespace ecs
00022
00023 #endif /* !INPUTNORMALIZER_HPP_ */
```

5.211 ActionFactory.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ActionFactory
00006 */
00007
00008 #ifndef ACTIONFACTORY_HPP_
00009 #define ACTIONFACTORY_HPP_
00010
00011 #include <functional>
00012 #include <memory>
00013 #include <string>
00014 #include <unordered_map>
00015 #include "../../ECS/entity/Entity.hpp"
00016 #include "../../resourceManager/ResourceManager.hpp"
00017 #include "TagRegistry.hpp"
00018
00019 namespace ecs {
00020     class Registry;
00021 }
00022
00023 class ActionFactory {
00024     public:
00025         static const ActionFactory& getInstance();
00026
00027         using ActionFunction = std::function<void(std::shared_ptr<ecs::Registry>,
      std::shared_ptr<ResourceManager>, ecs::Entity, ecs::Entity)>;
00028
00029         void registerAction(const std::string& actionId, ActionFunction action);
00030
00031         void executeAction(
00032             const std::string& actionId,
00033             std::shared_ptr<ecs::Registry> registry,
00034             std::shared_ptr<ResourceManager> resourceManager,
00035             ecs::Entity self, ecs::Entity other) const;
00036
00037         bool hasAction(const std::string& actionId) const;
00038
00039     private:
00040         ActionFactory();
```

```
00041         ~ActionFactory() = default;
00042         ActionFactory(const ActionFactory&) = delete;
00043         ActionFactory& operator=(const ActionFactory&) = delete;
00044
00045         void initializeConditions();
00046
00047         std::unordered_map<std::string, ActionFunction> _actions;
00048 };
00049
00050 #endif /* !ACTIONFACTORY_HPP_ */
```

5.212 InteractionSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** InteractionSystem
00006 */
00007
00008 #ifndef INTERACTIONSYSTEM_HPP_
00009 #define INTERACTIONSYSTEM_HPP_
00010
00011 #include "../base/ASystem.hpp"
00012 #include <memory>
00013 #include <string>
00014 #include <vector>
00015
00016 namespace ecs {
00017     class Registry;
00018 }
00019
00020 namespace ecs {
00021
00022 class InteractionSystem : public ASystem {
00023     public:
00024         InteractionSystem();
00025         ~InteractionSystem() = default;
00026
00027         void update(
00028             std::shared_ptr<ResourceManager> resourceManager,
00029             std::shared_ptr<Registry> registry,
00030             float deltaTime
00031         ) override;
00032
00033     private:
00034         std::vector<std::string> filterDamageActions(
00035             const std::vector<std::string>& actions,
00036             std::shared_ptr<Registry> registry,
00037             Entity targetEntity
00038         );
00039 };
00040
00041 }  // namespace ecs
00042
00043
00044 #endif /* !INTERACTIONSYSTEM_HPP_ */
```

5.213 TagRegistry.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** TagRegistry
00006 */
00007
00008 #ifndef TAGREGISTRY_HPP_
00009 #define TAGREGISTRY_HPP_
00010
00011 #include <unordered_map>
00012 #include <functional>
00013 #include <memory>
00014 #include <string>
00015 #include "../../ECS/entity/Entity.hpp"
00016 #include "../../ECS/entity/registry/Registry.hpp"
00017
00018 class TagRegistry {
```

```
00019     public:
00020         static const TagRegistry& getInstance();
00021
00022         template<typename T>
00023         void registerTag(const std::string& tagName) {
00024             _tagCheckers[tagName] = [](std::shared_ptr<ecs::Registry> reg, ecs::Entity ent) {
00025                 return reg->hasComponent<T>(ent);
00026             };
00027         }
00028
00029         bool hasTag(std::shared_ptr<ecs::Registry> registry, ecs::Entity entity, const std::string&
      tagName) const;
00030         std::vector<std::string> getTags(std::shared_ptr<ecs::Registry> registry, ecs::Entity entity)
      const;
00031
00032     private:
00033         TagRegistry();
00034         ~TagRegistry() = default;
00035         TagRegistry(const TagRegistry&) = delete;
00036         TagRegistry& operator=(const TagRegistry&) = delete;
00037
00038         void initializeTags();
00039
00040         std::unordered_map<std::string,
00041             std::function<bool(std::shared_ptr<ecs::Registry>, ecs::Entity)» _tagCheckers;
00042 };
00043
00044 #endif /* !TAGREGISTRY_HPP_ */
```

5.214 TriggerSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** TriggerSystem
00006 */
00007
00008 #ifndef TRIGGERSYSTEM_HPP_
00009 #define TRIGGERSYSTEM_HPP_
00010
00011 #include "../base/ASystem.hpp"
00012 #include "../../components/temporary/TriggerIntentComponent.hpp"
00013 #include "../../components/permanent/TransformComponent.hpp"
00014 #include "../../components/permanent/ColliderComponent.hpp"
00015 #include "../../CollisionRules/CollisionRules.hpp"
00016 #include "../../SpatialGrid/SpatialGrid.hpp"
00017 #include "TagRegistry.hpp"
00018
00019 namespace ecs {
00020
00021 class TriggerSystem : public ASystem {
00022     public:
00023         TriggerSystem();
00024         ~TriggerSystem() = default;
00025
00026         void update(
00027             std::shared_ptr<ResourceManager> resourceManager,
00028             std::shared_ptr<Registry> registry,
00029             float deltaTime
00030         ) override;
00031
00032     private:
00033         void buildSpatialGrid(
00034             std::shared_ptr<Registry> registry
00035         );
00036
00037         bool checkCollision(
00038             const TransformComponent& transformA,
00039             const ColliderComponent& colliderA,
00040             const TransformComponent& transformB,
00041             const ColliderComponent& colliderB
00042         );
00043
00044         bool shouldCollide(
00045             std::shared_ptr<ResourceManager> resourceManager,
00046             std::shared_ptr<Registry> registry,
00047             size_t entityA,
00048             const ColliderComponent& colliderA,
00049             size_t entityB
00050         );
00051
00052         SpatialGrid _spatialGrid;
```

```
00053 };
00054
00055 }
00056
00057 #endif /* !TRIGGERSYSTEM_HPP_ */
```

5.215 LifetimeSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** LifetimeSystem
00006 */
00007
00008 #ifndef LIFETIMESYSTEM_HPP_
00009 #define LIFETIMESYSTEM_HPP_
00010
00011 #include "../base/ASystem.hpp"
00012 #include <memory>
00013
00014 namespace ecs {
00015
00016 class LifetimeSystem : public ASystem {
00017     public:
00018         LifetimeSystem();
00019         ~LifetimeSystem() = default;
00020
00021         void update(
00022             std::shared_ptr<ResourceManager> resourceManager,
00023             std::shared_ptr<Registry> registry,
00024             float deltaTime
00025         ) override;
00026 };
00027
00028 }
00029
00030 #endif /* !LIFETIMESYSTEM_HPP_ */
```

5.216 MapGeneratorSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** MapGeneratorSystem
00006 */
00007
00008 #ifndef MAPGENERATORSYSTEM_HPP_
00009 #define MAPGENERATORSYSTEM_HPP_
00010
00011 #include "../base/ASystem.hpp"
00012 #include <random>
00013
00014 namespace ecs {
00015
00016 class MapGeneratorSystem : public ASystem {
00017     public:
00018         MapGeneratorSystem(unsigned int seed = 42);
00019         ~MapGeneratorSystem() = default;
00020
00021         void update(
00022             std::shared_ptr<ResourceManager> resourceManager,
00023             std::shared_ptr<Registry> registry,
00024             float deltaTime
00025         ) override;
00026
00027     private:
00028         void generateObstaclesAt(
00029             float x,
00030             std::shared_ptr<ResourceManager> resourceManager,
00031             std::shared_ptr<Registry> registry
00032         );
00033         void generateRandomWave(
00034             std::shared_ptr<ResourceManager> resourceManager,
00035             std::shared_ptr<Registry> registry,
00036             float currentX
00037         );
```

```
00038         void generateRandomPowerUp(
00039             std::shared_ptr<ResourceManager> resourceManager,
00040             std::shared_ptr<Registry> registry,
00041             float currentX
00042         );
00043         float noise(float x);
00044         unsigned int _seed;
00045         std::mt19937 _rng;
00046         float _lastGeneratedX;
00047         const float _generationStep;
00048         const float _startGenerationX;
00049         float _waveTimer;
00050         const float _waveInterval;
00051         float _powerUpTimer;
00052         const float _powerUpInterval;
00053 };
00054
00055 }
00056
00057 #endif /* !MAPGENERATORSYSTEM_HPP_ */
```

5.217 InputToVelocitySystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** VelocitySystem
00006 */
00007
00008 #ifndef VELOCITYSYSTEM_HPP_
00009 #define VELOCITYSYSTEM_HPP_
00010
00011 #include "../base/ASystem.hpp"
00012
00013 namespace ecs {
00014
00015 class InputToVelocitySystem : public ASystem {
00016     public:
00017         InputToVelocitySystem();
00018         ~InputToVelocitySystem() = default;
00019
00020         void update(std::shared_ptr<ResourceManager> resourceManager, std::shared_ptr<Registry>
      registry, float deltaTime) override;
00021 };
00022
00023 } // namespace ecs
00024
00025 #endif /* !VELOCITYSYSTEM_HPP_ */
```

5.218 IntentToVelocitySystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** IntentToVelocitySystem
00006 */
00007
00008 #ifndef INTENTTOVELOCITYSYSTEM_HPP_
00009 #define INTENTTOVELOCITYSYSTEM_HPP_
00010
00011 #include "../base/ASystem.hpp"
00012
00013 namespace ecs {
00014
00015 class IntentToVelocitySystem : public ASystem {
00016     public:
00017         IntentToVelocitySystem();
00018         ~IntentToVelocitySystem() = default;
00019
00020         void update(std::shared_ptr<ResourceManager> resourceManager, std::shared_ptr<Registry>
      registry, float deltaTime) override;
00021 };
00022
00023 } // namespace ecs
00024
00025 #endif /* !INTENTTOVELOCITYSYSTEM_HPP_ */
```

5.219 MovementSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** MovementSystem
00006 */
00007
00008 #ifndef MOVEMENTSYSTEM_HPP_
00009 #define MOVEMENTSYSTEM_HPP_
00010
00011 #include "../base/ASystem.hpp"
00012 #include "../../components/temporary/MovementIntentComponent.hpp"
00013 #include "../../components/permanent/TransformComponent.hpp"
00014 #include "../../components/permanent/SpeedComponent.hpp"
00015 #include "../../components/permanent/VelocityComponent.hpp"
00016 #include "../../components/permanent/ColliderComponent.hpp"
00017 #include "../../CollisionRules/CollisionRules.hpp"
00018 #include "../../systems/interactions/TagRegistry.hpp"
00019 #include "../../SpatialGrid/SpatialGrid.hpp"
00020
00021 namespace ecs {
00022
00023 class MovementSystem : public ASystem {
00024     public:
00025         MovementSystem();
00026         ~MovementSystem() = default;
00027
00028         void update(
00029             std::shared_ptr<ResourceManager> resourceManager,
00030             std::shared_ptr<Registry> registry,
00031             float deltaTime
00032         ) override;
00033
00034     private:
00035         void buildSpatialGrid(std::shared_ptr<Registry> registry);
00036
00037         bool checkCollision(
00038             std::shared_ptr<Registry> registry,
00039             size_t entityId,
00040             math::Vector2f newPos
00041         );
00042         math::Vector2f calculateSmoothMovement(
00043             std::shared_ptr<Registry> registry,
00044             size_t entityId,
00045             math::Vector2f startPos,
00046             math::Vector2f desiredPos
00047         );
00048         math::Vector2f calculateSlidingMovement(
00049             std::shared_ptr<Registry> registry,
00050             size_t entityId,
00051             math::Vector2f basePos,
00052             math::Vector2f desiredPos
00053         );
00054         math::Vector2f calculateSmoothSlidingPosition(
00055             std::shared_ptr<Registry> registry,
00056             size_t entityId,
00057             math::Vector2f startPos,
00058             math::Vector2f desiredPos
00059         );
00060         void handlePushCollision(
00061             std::shared_ptr<Registry> registry,
00062             size_t entityId,
00063             math::Vector2f finalPos,
00064             float deltaTime
00065         );
00066         bool shouldCollide(
00067             std::shared_ptr<Registry> registry,
00068             size_t entityA,
00069             const ColliderComponent& colliderA,
00070             size_t entityB
00071         );
00072         bool checkCollisionWithBoundaries(
00073             std::shared_ptr<Registry> registry,
00074             size_t entityId,
00075             math::Vector2f newPos
00076         );
00077
00078         SpatialGrid _spatialGrid;
00079         std::vector<Entity> _boundaryEntities;
00080 };
00081
00082 }
00083
00084 #endif /* !MOVEMENTSYSTEM_HPP_ */
```

5.220 ScoreSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ScoreSystem
00006 */
00007
00008 #ifndef SCORESYSTEM_HPP_
00009 #define SCORESYSTEM_HPP_
00010
00011 #include "../base/ASystem.hpp"
00012
00013 namespace ecs {
00014
00015 class ScoreSystem : public ASystem {
00016     public:
00017         ScoreSystem();
00018         ~ScoreSystem();
00019         void update(
00020             std::shared_ptr<ResourceManager> resourceManager,
00021             std::shared_ptr<Registry> registry,
00022             float deltaTime
00023         ) override;
00024     protected:
00025     private:
00026 };
00027
00028 }  // namespace ecs
00029
00030 #endif /* !SCORESYSTEM_HPP_ */
```

5.221 ScriptingSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2026
00003 ** ryanR-type
00004 ** File description:
00005 ** ScriptingSystem
00006 */
00007
00008 #ifndef SCRIPTINGSYSTEM_HPP_
00009 #define SCRIPTINGSYSTEM_HPP_
00010
00011 #include "../base/ASystem.hpp"
00012 #include "../../components/permanent/ScriptingComponent.hpp"
00013 #include "../../components/permanent/TransformComponent.hpp"
00014 #include "../../components/permanent/SpeedComponent.hpp"
00015 #include "../../components/permanent/EntityPartsComponent.hpp"
00016 #include "../../components/permanent/CompositeEntityComponent.hpp"
00017 #include "../../components/tags/LocalPlayerTag.hpp"
00018
00019 #include <sol/sol.hpp>
00020
00021 namespace ecs {
00022
00023 class ScriptingSystem : public ASystem {
00024     public:
00025         ScriptingSystem();
00026         ~ScriptingSystem() = default;
00027
00028         void update(
00029             std::shared_ptr<ResourceManager> resourceManager,
00030             std::shared_ptr<Registry> reg,
00031             float deltaTime
00032         ) override;
00033
00034     protected:
00035     private:
00036         void bindAPI();
00037
00038         sol::state lua;
00039         std::shared_ptr<Registry> registry;
00040         std::shared_ptr<ResourceManager> resourceManager;
00041 };
00042
00043 }  // namespace ecs
00044
00045 #endif /* !SCRIPTINGSYSTEM_HPP_ */
```

5.222 ChargedShotSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2026
00003 ** ryanR-type
00004 ** File description:
00005 ** ChargedShotSystem
00006 */
00007
00008 #ifndef CHARGEDSHOTSYSTEM_HPP_
00009 #define CHARGEDSHOTSYSTEM_HPP_
00010
00011 #include "../base/ASystem.hpp"
00012
00013 namespace ecs {
00014
00015 class ChargedShotSystem : public ASystem {
00016     public:
00017         ChargedShotSystem();
00018         ~ChargedShotSystem() = default;
00019
00020         void update(
00021             std::shared_ptr<ResourceManager> resourceManager,
00022             std::shared_ptr<Registry> registry,
00023             float deltaTime
00024         ) override;
00025
00026     private:
00027 };
00028
00029 }
00030
00031 #endif /* !CHARGEDSHOTSYSTEM_HPP_ */
```

5.223 ShootingSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ShootingSystem
00006 */
00007
00008 #ifndef SHOOTINGSYSTEM_HPP_
00009 #define SHOOTINGSYSTEM_HPP_
00010
00011 #include "../base/ASystem.hpp"
00012 #include "../../components/temporary/ShootIntentComponent.hpp"
00013 #include "../../components/permanent/ShootingStatsComponent.hpp"
00014 #include "../../components/permanent/TransformComponent.hpp"
00015 #include "../../components/permanent/VelocityComponent.hpp"
00016 #include "../../types/Vector2f.hpp"
00017 #include <cmath>
00018 #include <string>
00019
00020 namespace ecs {
00021
00022 class ShootingSystem : public ASystem {
00023     public:
00024         ShootingSystem();
00025         ~ShootingSystem() = default;
00026
00027         void update(
00028             std::shared_ptr<ResourceManager> resourceManager,
00029             std::shared_ptr<Registry> registry,
00030             float deltaTime
00031         ) override;
00032
00033     private:
00034         void spawnProjectile(
00035             std::shared_ptr<Registry> registry,
00036             std::shared_ptr<ResourceManager> resourceManager,
00037             const std::string& prefabName,
00038             const math::Vector2f &position,
00039             float angle,
00040             ecs::Entity shooterEntity
00041         );
00042
00043         math::Vector2f calculateProjectileVelocity(
00044             float angle,
00045             float speed
00046         );
```

```
00047 };
00048
00049 }  // namespace ecs
00050
00051 #endif /* !SHOOTINGSYSTEM_HPP_ */
```

5.224 SpawnSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** SpawnSystem
00006 */
00007
00008 #ifndef SPAWNSYSTEM_HPP_
00009 #define SPAWNSYSTEM_HPP_
00010
00011 #include "../base/ASystem.hpp"
00012 #include "../../CollisionRules/CollisionRules.hpp"
00013 #include "../../systems/interactions/TagRegistry.hpp"
00014 #include "../../components/permanent/TransformComponent.hpp"
00015 #include "../../components/permanent/ColliderComponent.hpp"
00016
00017 namespace ecs {
00018
00019 class SpawnSystem : public ASystem {
00020     public:
00021         SpawnSystem();
00022         ~SpawnSystem() = default;
00023
00024         void update(
00025             std::shared_ptr<ResourceManager> resourceManager,
00026             std::shared_ptr<Registry> registry,
00027             float deltaTime
00028         ) override;
00029
00030     private:
00031         bool isPositionFree(
00032             Entity newEntity,
00033             const math::Vector2f& position,
00034             const std::vector<std::shared_ptr<ColliderComponent»& newColliders,
00035             std::shared_ptr<TransformComponent> newTransform,
00036             std::shared_ptr<Registry> registry
00037         );
00038
00039         math::Vector2f findNearestFreePosition(
00040             Entity newEntity,
00041             const math::Vector2f& originalPosition,
00042             std::shared_ptr<Registry> registry,
00043             float stepSize = 10.0f
00044         );
00045 };
00046
00047 }
00048
00049 #endif /* !SPAWNSYSTEM_HPP_ */
```

5.225 ASystemManager.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ASystemManager
00006 */
00007
00008 #ifndef ASystemMANAGER_HPP_
00009 #define ASystemMANAGER_HPP_
00010
00011 #include <vector>
00012 #include <memory>
00013
00014 #include "ISystemManager.hpp"
00015 #include "../../resourceManager/ResourceManager.hpp"
00016 #include "../../ECS/entity/registry/Registry.hpp"
00017 #include "../base/ISystem.hpp"
00018
```

```
00019 namespace ecs {
00020
00021 class ASystemManager : public ISystemManager {
00022     public:
00023         ASystemManager();
00024         ~ASystemManager();
00025         void updateAllSystems(std::shared_ptr<ResourceManager> resourceManager,
      std::shared_ptr<Registry> registry, float deltaTime) override;
00026         void addSystem(std::shared_ptr<ISystem> system) override;
00027         void removeSystem(std::shared_ptr<ISystem> system) override;
00028         void clearAllSystems() override;
00029
00030     private:
00031         std::vector<std::shared_ptr<ISystem» _systems;
00032 };
00033
00034 } // namespace ecs
00035
00036 #endif /* !ASystemMANAGER_HPP_ */
```

5.226 ISystemManager.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ISystemManager
00006 */
00007
00008 #ifndef ISystemMANAGER_HPP_
00009 #define ISystemMANAGER_HPP_
00010
00011 #include <memory>
00012
00013 #include "../base/ISystem.hpp"
00014 #include "../../resourceManager/ResourceManager.hpp"
00015 #include "../../ECS/entity/registry/Registry.hpp"
00016
00017 namespace ecs {
00018
00019 class ISystemManager {
00020     public:
00021         virtual ~ISystemManager() = default;
00022         virtual void updateAllSystems(std::shared_ptr<ResourceManager> resourceManager,
      std::shared_ptr<Registry> registry, float deltaTime) = 0;
00023         virtual void addSystem(std::shared_ptr<ISystem> system) = 0;
00024         virtual void removeSystem(std::shared_ptr<ISystem> system) = 0;
00025         virtual void clearAllSystems() = 0;
00026 };
00027
00028 }  // namespace ecs
00029
00030 #endif /* !ISystemMANAGER_HPP_ */
```

5.227 SystemManager.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** SystemManager
00006 */
00007
00008 #ifndef SYSTEMMANAGER_HPP_
00009 #define SYSTEMMANAGER_HPP_
00010
00011 #include "ASystemManager.hpp"
00012
00013 namespace ecs {
00014
00015 class SystemManager : public ASystemManager {
00016     public:
00017         SystemManager();
00018         ~SystemManager() = default;
00019
00020     protected:
00021     private:
00022 };
```

```
00023
00024 }  // namespace ecs
00025
00026 #endif /* !SYSTEMMANAGER_HPP_ */
```

5.228 translationToECS.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** R-Type
00004 ** File description:
00005 ** Header
00006 */
00007
00008 #ifndef TRASLATION_TO_ECS_HPP_
00009 #define TRASLATION_TO_ECS_HPP_
00010
00011 enum componentType {
00012     TRANSFORM = 0x00,
00013     HEALTH = 0x01,
00014     SCORE = 0x02,
00015     CHARGED_SHOT_COMP = 0x03,
00016     ANIMATION_STATE = 0x04,
00017 };
00018
00019 #endif /* !TRASLATION_TO_ECS_HPP_ */
```

5.229 Chrono.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** Chrono wrapper for std::chrono
00006 */
00007
00008 #ifndef CHRONO_HPP_
00009 #define CHRONO_HPP_
00010
00011 #include <chrono>
00012
00013 namespace math {
00014
00015 class Chrono {
00016     public:
00017         Chrono();
00018         ~Chrono() = default;
00019
00020         void start();
00021         void stop();
00022         void reset();
00023         float getElapsedSeconds() const;
00024         float getElapsedMilliseconds() const;
00025         bool isRunning() const;
00026
00027     private:
00028         std::chrono::high_resolution_clock::time_point _startTime;
00029         std::chrono::high_resolution_clock::time_point _stopTime;
00030         bool _isRunning;
00031 };
00032
00033 } // namespace math
00034
00035 #endif /* !CHRONO_HPP_ */
```

5.230 FRect.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** FRect
00006 */
```

```
00007
00008 #ifndef FRECT_HPP_
00009 #define FRECT_HPP_
00010
00011 namespace math {
00012
00013 class FRect {
00014     public:
00015         FRect();
00016         FRect(float left, float top, float width, float height);
00017         FRect(FRect const &other);
00018         ~FRect() = default;
00019
00020         float getLeft() const;
00021         void setLeft(float left);
00022         float getTop() const;
00023         void setTop(float top);
00024         float getWidth() const;
00025         void setWidth(float width);
00026         float getHeight() const;
00027         void setHeight(float height);
00028
00029         bool contains(float x, float y) const;
00030         bool intersects(FRect const &other) const;
00031         bool intersects(FRect const &other, FRect &intersection) const;
00032
00033         FRect &operator=(FRect const &other);
00034         bool operator==(FRect const &other) const;
00035         bool operator!=(FRect const &other) const;
00036
00037         private:
00038             float left;
00039             float top;
00040             float width;
00041             float height;
00042 };
00043
00044 } // namespace math
00045
00046 #endif /* !FRECT_HPP_ */
```

5.231 OrientedRect.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** OrientedRect
00006 */
00007
00008 #ifndef ORIENTEDRECT_HPP_
00009 #define ORIENTEDRECT_HPP_
00010
00011 #include "Vector2f.hpp"
00012 #include <vector>
00013 #include <cmath>
00014
00015 namespace math {
00016
00017 class OrientedRect {
00018     public:
00019         OrientedRect();
00020         OrientedRect(Vector2f center, Vector2f size, float rotation);
00021         OrientedRect(OrientedRect const &other);
00022         ~OrientedRect() = default;
00023
00024         Vector2f getCenter() const;
00025         void setCenter(Vector2f center);
00026         Vector2f getSize() const;
00027         void setSize(Vector2f size);
00028         float getRotation() const;
00029         void setRotation(float rotation);
00030
00031         std::vector<Vector2f> getCorners() const;
00032         Vector2f getAxisX() const;
00033         Vector2f getAxisY() const;
00034
00035         bool intersects(OrientedRect const &other) const;
00036
00037     OrientedRect &operator=(OrientedRect const &other);    private:
00038         Vector2f _center;
00039         Vector2f _size;
00040         float _rotation;
```

```
00041
00042         float projectPoint(Vector2f point, Vector2f axis) const;
00043         bool overlapOnAxis(OrientedRect const &other, Vector2f axis) const;
00044 };
00045
00046 } // namespace math
00047
00048 #endif /* !ORIENTEDRECT_HPP_ */
```

5.232 Vector2f.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** Vector2f
00006 */
00007
00008 #ifndef VECTOR2F_HPP_
00009 #define VECTOR2F_HPP_
00010
00011 namespace math {
00012
00013 class Vector2f {
00014     public:
00015         Vector2f(float x = 0.0f, float y = 0.0f);
00016         Vector2f(Vector2f const &other);
00017         ~Vector2f() = default;
00018
00019         float getX() const;
00020         void setX(float x);
00021         float getY() const;
00022         void setY(float y);
00023
00024         Vector2f getVector() const;
00025         Vector2f operator*(float scalar) const;
00026         Vector2f operator-(Vector2f const &other) const;
00027         Vector2f operator+(Vector2f const &other) const;
00028         void operator=(Vector2f const &other);
00029         void operator+=(Vector2f const &other);
00030         void operator-=(Vector2f const &other);
00031         void operator*=(float scalar);
00032         void operator/=(float scalar);
00033     private:
00034         float _x;
00035         float _y;
00036 };
00037
00038 } // namespace math
00039
00040 #endif /* !VECTOR2F_HPP_ */
```

5.233 Encryption.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2026
00003 ** R-Type
00004 ** File description:
00005 ** Encryption utilities
00006 */
00007
00008 #ifndef ENCRYPTION_HPP_
00009 #define ENCRYPTION_HPP_
00010
00011 #include <string>
00012 #include <vector>
00013
00014 namespace utils {
00015
00016 class Encryption {
00017 public:
00018     static std::string encrypt(const std::string& data);
00019     static std::string decrypt(const std::string& encryptedData);
00020     static std::string base64Encode(const std::vector<unsigned char>& data);
00021     static std::vector<unsigned char> base64Decode(const std::string& encoded);
00022
00023 private:
00024     static inline const std::string _key = "R-Type_Secure_Key_2026";
```

```
00025 };
00026
00027 }  // namespace utils
00028
00029 #endif /* !ENCRYPTION_HPP_ */
```

5.234 SecureJsonManager.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2026
00003 ** R-Type
00004 ** File description:
00005 ** Secure JSON File Manager
00006 */
00007
00008 #ifndef SECURE_JSON_MANAGER_HPP_
00009 #define SECURE_JSON_MANAGER_HPP_
00010
00011 #include <string>
00012 #include <nlohmann/json.hpp>
00013
00014 namespace utils {
00015
00016 class SecureJsonManager {
00017 public:
00018     static nlohmann::json readSecureJson(const std::string& filepath);
00019     static bool writeSecureJson(const std::string& filepath, const nlohmann::json& data);
00020 };
00021
00022 }  // namespace utils
00023
00024 #endif /* !SECURE_JSON_MANAGER_HPP_ */
```

5.235 ComponentDeltaTracker.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** R-Type
00004 ** File description:
00005 ** Component Delta Tracker - Track and send only modified components
00006 */
00007
00008 #ifndef COMPONENT_DELTA_TRACKER_HPP
00009 #define COMPONENT_DELTA_TRACKER_HPP
00010
00011 #include <unordered_map>
00012 #include <vector>
00013 #include <cstdint>
00014 #include <map>
00015 #include <set>
00016 #include "../../common/translationToECS.hpp"
00017
00018 namespace rserv {
00019
00020 struct EntitySnapshot {
00021     uint32_t entityId;
00022     uint32_t componentMask;
00023     std::map<uint8_t, std::vector<uint64_t>> components;
00024
00025     EntitySnapshot() : entityId(0), componentMask(0) {}
00026 };
00027
00028 class ComponentDeltaTracker {
00029     public:
00030         std::vector<uint64_t> createEntityDelta(uint8_t clientId, uint32_t entityId, const
      EntitySnapshot& currentSnapshot);
00031         std::vector<uint64_t> createMultiEntityDelta(uint8_t clientId, const
      std::vector<EntitySnapshot>& entities);
00032         EntitySnapshot applyDelta(uint8_t clientId, const std::vector<uint64_t>& deltaPayload);
00033         void clearClientCache(uint8_t clientId);
00034         void clearEntityCache(uint8_t clientId, uint32_t entityId);
00035         void clearAllCaches();
00036         void clearDeadEntities(const std::set<uint32_t>& aliveEntityIds);
00037
00038     private:
00039         std::unordered_map<uint8_t, std::unordered_map<uint32_t, EntitySnapshot>> _clientEntityCache;
00040
```

```
00041         std::vector<uint64_t> serializeFullSnapshot(uint32_t entityId, const EntitySnapshot&
      snapshot);
00042         std::vector<uint64_t> serializeDelta(uint32_t entityId, uint32_t changedMask, const
      std::map<uint8_t, std::vector<uint64_t>>& changedComponents);
00043 };
00044
00045 } // namespace rserv
00046
00047 #endif // COMPONENT_DELTA_TRACKER_HPP
```

5.236 ComponentSerializer.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** R-Type
00004 ** File description:
00005 ** Component Serializer - Helper for converting ECS components to/from packets
00006 */
00007
00008 #ifndef COMPONENT_SERIALIZER_HPP
00009 #define COMPONENT_SERIALIZER_HPP
00010
00011 #include "ComponentDeltaTracker.hpp"
00012 #include <cstring>
00013 #include <string>
00014
00015 namespace rserv {
00016 class ComponentSerializer {
00017     public:
00018         static EntitySnapshot createSnapshotFromComponents(uint32_t entityId, const
      std::vector<uint64_t>& componentData);
00019 };
00020 } // namespace rserv
00021
00022 #endif // COMPONENT_SERIALIZER_HPP
```

5.237 gsmStates.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** gsmStates
00006 */
00007
00008 #ifndef GSMMACHINEENUM_HPP_
00009 #define GSMMACHINEENUM_HPP_
00010
00011 namespace gsm {
00012
00013 enum class GameStateType {
00014     BOOT,
00015     LOBBY,
00016     LOADING,
00017     IN_GAME,
00018     INFINITE_MODE,
00019     LEVEL_COMPLETE,
00020     GAME_END,
00021     SHUTDOWN
00022 };
00023
00024 }  // namespace gsm
00025
00026 #endif /* !GSMMACHINEENUM_HPP_ */
```

5.238 BootState.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** BootState
00006 */
```

```
00007
00008 #ifndef SERVER_BOOTSTATE_HPP_
00009 #define SERVER_BOOTSTATE_HPP_
00010
00011 #include "../../AGameState.hpp"
00012 #include "resourceManager/ResourceManager.hpp"
00013
00014 namespace gsm {
00015
00016 class BootState : public AGameState {
00017 public:
00018     BootState(std::shared_ptr<IGameStateMachine> gsm, std::shared_ptr<ResourceManager>
      resourceManager);
00019     ~BootState() override = default;
00020
00021     void enter() override;
00022     std::string getStateName() const override { return "Booting"; }
00023 };
00024
00025 }  // namespace gsm
00026
00027 #endif  // SERVER_BOOTSTATE_HPP_
```

5.239 GameEndState.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** GameEndState
00006 */
00007
00008 #ifndef SERVER_GAMEENDSTATE_HPP_
00009 #define SERVER_GAMEENDSTATE_HPP_
00010
00011 #include "../../AGameState.hpp"
00012 #include "resourceManager/ResourceManager.hpp"
00013
00014 namespace gsm {
00015
00016 class GameEndState : public AGameState {
00017 public:
00018     GameEndState(std::shared_ptr<IGameStateMachine> gsm, std::shared_ptr<ResourceManager>
      resourceManager);
00019     ~GameEndState() override = default;
00020
00021     void update(float deltaTime) override;
00022     std::string getStateName() const override { return "Game Ended"; }
00023 };
00024
00025 }  // namespace gsm
00026
00027 #endif  // SERVER_GAMEENDSTATE_HPP_
```

5.240 InfiniteState.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** InfiniteState
00006 */
00007
00008 #ifndef SERVER_INFINITESTATE_HPP_
00009 #define SERVER_INFINITESTATE_HPP_
00010
00011 #include "../../AGameState.hpp"
00012 #include <memory>
00013 #include "resourceManager/ResourceManager.hpp"
00014
00015 namespace gsm {
00016
00017 class InfiniteState : public AGameState {
00018     public:
00019         InfiniteState(std::shared_ptr<IGameStateMachine> gsm, std::shared_ptr<ResourceManager>
      resourceManager);
00020         ~InfiniteState() override = default;
00021
```

```
00022         void enter() override;
00023         void update(float deltaTime) override;
00024         void exit() override;
00025         std::string getStateName() const override { return "Infinite Mode"; }
00026 };
00027
00028 }  // namespace gsm
00029
00030 #endif  // SERVER_INFINITESTATE_HPP_
```

5.241 LoadingState.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** LoadingState
00006 */
00007
00008 #ifndef SERVER_LOADINGSTATE_HPP_
00009 #define SERVER_LOADINGSTATE_HPP_
00010
00011 #include "../../AGameState.hpp"
00012 #include "resourceManager/ResourceManager.hpp"
00013
00014 namespace gsm {
00015
00016 class LoadingState : public AGameState {
00017 public:
00018     LoadingState(std::shared_ptr<IGameStateMachine> gsm, std::shared_ptr<ResourceManager>
      resourceManager);
00019     ~LoadingState() override = default;
00020
00021     void enter() override;
00022     std::string getStateName() const override { return "Loading"; }
00023 };
00024
00025 }  // namespace gsm
00026
00027 #endif  // SERVER_LOADINGSTATE_HPP_
```

5.242 LobbyState.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** LobbyState
00006 */
00007
00008 #ifndef SERVER_LOBBYSTATE_HPP_
00009 #define SERVER_LOBBYSTATE_HPP_
00010
00011 #include "../../AGameState.hpp"
00012 #include "resourceManager/ResourceManager.hpp"
00013
00014 namespace gsm {
00015
00016 class LobbyState : public AGameState {
00017 public:
00018     LobbyState(std::shared_ptr<IGameStateMachine> gsm, std::shared_ptr<ResourceManager>
      resourceManager);
00019     ~LobbyState() override = default;
00020
00021     void enter() override;
00022     void update(float deltaTime) override;
00023     std::string getStateName() const override { return "Waiting to start"; }
00024 };
00025
00026 }  // namespace gsm
00027
00028 #endif  // SERVER_LOBBYSTATE_HPP_
```

5.243 HttpServer.hpp

```
00001 /*
```

```
00002 ** EPITECH PROJECT, 2026
00003 ** ryanR-type
00004 ** File description:
00005 ** HttpServer
00006 */
00007
00008 #ifndef HTTPSERVER_HPP_
00009 #define HTTPSERVER_HPP_
00010
00011 #include <thread>
00012 #include <atomic>
00013 #include <memory>
00014 #include <functional>
00015 #include <string>
00016 #include <httplib.h>
00017 #include <chrono>
00018
00019 #include "../ServerConfig.hpp"
00020
00021 namespace rserv {
00022
00023 struct ServerInfo {
00024     int connectedClients;
00025     std::chrono::seconds uptime;
00026     int activeLobbies;
00027     size_t totalPlayers;
00028     std::vector<std::string> lobbyDetails;
00029     std::vector<std::string> playerDetails;
00030     std::vector<std::vector<std::string>> lobbyPlayerDetails;
00031     std::vector<std::map<std::string, int>> playerStats;
00032     std::vector<std::string> inGamePlayers;
00033     std::vector<std::string> bannedPlayers;
00034     int64_t tps;
00035 };
00036
00037 class HttpServer {
00038     public:
00039         HttpServer(
00040             std::function<bool()> statusChecker,
00041             std::function<ServerInfo()> infoGetter,
00042             std::shared_ptr<ServerConfig> serverConfig,
00043             std::function<std::string(const std::string&)> commandExecutor
00044         );
00045         ~HttpServer();
00046
00047         void start();
00048         void stop();
00049
00050         void statusEndpoint(const httplib::Request &, httplib::Response &res);
00051         void infoEndpoint(const httplib::Request &, httplib::Response &res);
00052         void configEndpoint(const httplib::Request &, httplib::Response &res);
00053         void commandsSuggestionsEndpoint(const httplib::Request &, httplib::Response &res);
00054         void commandsExecuteEndpoint(const httplib::Request &, httplib::Response &res);
00055
00056     private:
00057         void httpLoop();
00058         void loadEnv();
00059         bool checkAuth(const httplib::Request &req);
00060
00061         std::thread _httpThread;
00062         std::atomic_bool _running;
00063         std::function<bool()> _statusChecker;
00064         std::function<ServerInfo()> _infoGetter;
00065         std::shared_ptr<ServerConfig> _serverConfig;
00066         std::function<std::string(const std::string&)> _commandExecutor;
00067         std::unique_ptr<httplib::Server> _server;
00068         std::string _password;
00069 };
00070
00071 }  // namespace rserv
00072
00073 #endif /* !HTTPSERVER_HPP_ */
```

5.244 ServerInputProvider.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ServerInputProvider
00006 */
00007
00008 #ifndef SERVERINPUTPROVIDER_HPP_
```

```
00009 #define SERVERINPUTPROVIDER_HPP_
00010
00011 #include "../../common/InputMapping/IInputProvider.hpp"
00012 #include "../../common/InputMapping/InputMapping.hpp"
00013 #include "../../common/constants.hpp"
00014 #include <iostream>
00015 #include <vector>
00016 #include <algorithm>
00017 #include <chrono>
00018 #include <map>
00019 #include <set>
00020
00021 namespace ecs {
00022
00023 class ServerInputProvider : public IInputProvider {
00024     public:
00025         ServerInputProvider();
00026         ~ServerInputProvider() override = default;
00027
00028         float getAxisValue(event_t axis, size_t clientID = 0) override;
00029
00030         bool isActionPressed(InputAction action, size_t clientID = 0) override;
00031         float getActionAxis(InputAction action, size_t clientID = 0) override;
00032         InputMapping getInputMapping(size_t clientID = 0) const override;
00033
00034         void setAxisValue(ecs::InputAction action, float value, size_t clientID = 0);
00035
00036         void addClientInputMapping(size_t clientID, size_t identity, const InputMapping& mapping);
00037         void registerClient(size_t clientID);
00038         void registerEntityForClient(size_t entityId, size_t clientID);
00039         size_t getClientIdForEntity(size_t entityId) const;
00040         void updateInputFromEvent(size_t clientID, constants::EventType eventType, float value);
00041         std::vector<size_t> getConnectedClients() const;
00042
00043
00044     private:
00045         std::vector<std::tuple<size_t, size_t, InputMapping» _inputMapping;
00046         std::map<size_t, std::map<ecs::InputAction, float» _clientAxisValues;
00047         std::map<size_t, std::map<ecs::InputAction, std::chrono::steady_clock::time_point»
      _clientInputTimestamps;
00048         std::set<size_t> _registeredClients;
00049         std::map<size_t, size_t> _entityToClientId;
00050
00051         static constexpr std::chrono::milliseconds INPUT_TIMEOUT = std::chrono::milliseconds(200);
00052
00053         using InputHandler = void (ServerInputProvider::*)(size_t, float);
00054         std::vector<InputHandler> _inputHandlers;
00055
00056         void handleUp(size_t clientID, float value);
00057         void handleDown(size_t clientID, float value);
00058         void handleLeft(size_t clientID, float value);
00059         void handleRight(size_t clientID, float value);
00060         void handleStop(size_t clientID, float value);
00061         void handleShoot(size_t clientID, float value);
00062         void handleForce(size_t clientID, float value);
00063         void handleHealthCheck(size_t clientID, float value);
00064 };
00065
00066 } // namespace ecs
00067
00068 #endif /* !SERVERINPUTPROVIDER_HPP_ */
```

5.245 Lobby.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2026
00003 ** ryanR-type
00004 ** File description:
00005 ** Lobby
00006 */
00007
00008 #ifndef LOBBY_HPP_
00009 #define LOBBY_HPP_
00010 #ifdef _WIN32
00011     #ifndef _WIN32_WINNT
00012         #define _WIN32_WINNT 0x0A00
00013     #endif
00014
00015     #ifndef WIN32_LEAN_AND_MEAN
00016         #define WIN32_LEAN_AND_MEAN
00017     #endif
00018 #endif
00019
```

```
00020 #include <queue>
00021 #include <map>
00022 #include <memory>
00023 #include <mutex>
00024
00025 #include "LobbyStruct.hpp"
00026 #include "ServerConfig.hpp"
00027 #include "deltaTracker/ComponentDeltaTracker.hpp"
00028 #include "deltaTracker/ComponentSerializer.hpp"
00029 #include "../common/interfaces/INetwork.hpp"
00030 #include "../common/interfaces/IBuffer.hpp"
00031 #include "../common/DLLoader/DLLoader.hpp"
00032 #include "../common/DLLoader/LoaderType.hpp"
00033 #include "../common/constants.hpp"
00034 #include "../common/InputMapping/InputAction.hpp"
00035 #include "../common/resourceManager/ResourceManager.hpp"
00036 #include "../common/ECS/entity/registry/Registry.hpp"
00037 #include "gsm/machine/GameStateMachine.hpp"
00038 #include "Signal.hpp"
00039
00040 namespace rserv {
00041
00042 class Lobby {
00043         public:
00044             Lobby(std::shared_ptr<net::INetwork> network,
00045                 std::vector<std::tuple<uint8_t, std::shared_ptr<net::INetworkEndpoint>, std::string»
      lobbyPlayerInfo,
00046                 std::string lobbyCode, bool debug, int64_t tps);
00047             ~Lobby();
00048             void stop();
00049
00050             void startNetworkThread();
00051             void startGameThread();
00052             void networkLoop();
00053             void gameLoop();
00054
00055             void setIsDebug(bool debug);
00056             bool getIsDebug() const;
00057
00058             std::shared_ptr<ResourceManager> getResourceManager() const;
00059
00060             const std::map<uint8_t, ecs::Entity>& getClientToEntity() const;
00061
00062             std::vector<uint8_t> getConnectedClients() const;
00063             std::vector<std::tuple<uint8_t, std::string» getConnectedClientDetails() const;
00064             std::vector<std::shared_ptr<net::INetworkEndpoint» getConnectedClientEndpoints() const;
00065             size_t getClientCount() const;
00066             bool isRunning() const;
00067             void addClient(std::tuple<uint8_t, std::shared_ptr<net::INetworkEndpoint>, std::string>
      client);
00068             void removeClient(uint8_t clientId);
00069             void resetClientHeartbeats();
00070             void createPlayerEntityForClient(uint8_t clientId);
00071             void syncExistingEntitiesToClient(std::shared_ptr<net::INetworkEndpoint> clientEndpoint);
00072             std::string getLobbyCode() const;
00073             std::shared_ptr<net::INetwork> getNetwork() const;
00074             std::string getGameState() const;
00075             std::string getGameRules() const;
00076
00077             std::shared_ptr<std::queue<std::tuple<uint8_t, constants::EventType, double»>
      getEventQueue();
00078             bool hasEvents() const;
00079
00080             void enqueuePacket(std::pair<std::shared_ptr<net::INetworkEndpoint>, std::vector<uint8_t»
      packet);
00081
00082             /* Received Packet Handling */
00083             void processIncomingPackets();
00084             bool processDisconnections(uint8_t idClient);
00085             bool processEvents(uint8_t idClient);
00086             bool processWhoAmI(uint8_t idClient);
00087
00088             /* Sent Packet Handling */
00089             bool gameStatePacket();
00090             bool endGamePacket(bool isWin);
00091             std::vector<uint64_t> spawnPacket(size_t entity, const std::string prefabName);
00092             std::vector<uint64_t> deathPacket(size_t entity);
00093             bool serverStatusPacket();
00094             bool ackLeaveLobbyPacket(const net::INetworkEndpoint &endpoint, bool canDisconnect);
00095
00096             bool levelCompletePacket();
00097             bool nextLevelPacket();
00098             bool gameRulesPacket();
00099
00100             bool isGameStarted() const;
00101             bool allClientsReady() const;
00102
```

```
00103             uint32_t getSequenceNumber() const;
00104
00105             void setPacketManager(std::shared_ptr<pm::IPacketManager> packet);
00106             std::shared_ptr<pm::IPacketManager> getPacketManager() const;
00107             void incrementSequenceNumber();
00108             void setResourceManager(std::shared_ptr<ResourceManager> resourceManager);
00109             void clearEntityDeltaCache(uint8_t clientId, uint32_t entityId);
00110             void clearDeltaTrackerCaches();
00111             void createPlayerEntities();
00112             void processLobbyEvents();
00113
00114         private:
00115             bool _isDebug;
00116             int64_t _tps;
00117
00118             /* Network handling variable*/
00119             std::shared_ptr<net::INetwork> _network;
00120             std::vector<std::tuple<uint8_t, std::shared_ptr<net::INetworkEndpoint>, std::string»
      _clients;
00121             std::string _lobbyCode;
00122             std::map<uint8_t, bool> _clientsReady;
00123             std::map<uint8_t, ecs::Entity> _clientToEntity;
00124             std::shared_ptr<pm::IPacketManager> _packet;
00125             uint32_t _sequenceNumber;
00126             std::shared_ptr<std::queue<std::tuple<uint8_t, constants::EventType, double»> _eventQueue;
00127
00128             /* Packet queue for incoming packets */
00129             std::queue<std::pair<std::shared_ptr<net::INetworkEndpoint>, std::vector<uint8_t»>
      _incomingPackets;
00130             std::mutex _packetMutex;
00131             mutable std::mutex _clientsMutex;
00132
00133             /* ECS/Game handling variable */
00134             bool _gameStarted;
00135             bool _playerEntitiesCreated;
00136             std::shared_ptr<ResourceManager> _resourceManager;
00137             std::shared_ptr<gsm::GameStateMachine> _gsm;
00138             std::chrono::steady_clock::time_point _lastGameStateTime;
00139             float _statusUpdateTimer;
00140
00141             /* Threading */
00142             std::atomic_bool _running;
00143             std::thread _networkThread;
00144             std::thread _gameThread;
00145             mutable std::mutex _eventMutex;
00146
00147
00148             ComponentDeltaTracker _deltaTracker;
00149             /* Functions to build game state packets */
00150             std::vector<std::function<std::vector<uint64_t>(std::shared_ptr<ecs::Registry>,
      ecs::Entity)» _convertFunctions;
00151
00152         protected:
00153             std::vector<uint64_t> convertTransformComponent(std::shared_ptr<ecs::Registry> registry,
      ecs::Entity i);
00154             std::vector<uint64_t> convertHealthComponent(std::shared_ptr<ecs::Registry> registry,
      ecs::Entity i);
00155             std::vector<uint64_t> convertScoreComponent(std::shared_ptr<ecs::Registry> registry,
      ecs::Entity i);
00156             std::vector<uint64_t> convertAnimationStateComponent(std::shared_ptr<ecs::Registry>
      registry, ecs::Entity i);
00157             std::vector<uint64_t> convertChargedShotComponent(std::shared_ptr<ecs::Registry> registry,
      ecs::Entity i);
00158
00159     };
00160 } // namespace rserv = r-type server
00161
00162 #endif /* !LOBBY_HPP_ */
```

5.246 LobbyStruct.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2026
00003 ** ryanR-type
00004 ** File description:
00005 ** Lobby
00006 */
00007
00008 #include <string>
00009 #include <vector>
00010 #include <thread>
00011 #include "../common/interfaces/INetwork.hpp"
00012 #include <asio/ip/udp.hpp>
```

```
00013
00014 #ifndef LOBBYSTRUCT_HPP_
00015 #define LOBBYSTRUCT_HPP_
00016
00017 namespace rserv {
00018
00019 class Lobby;
00020
00021 struct LobbyStruct {
00022
00023     std::string _lobbyCode;
00024     std::vector<std::tuple<uint8_t, std::shared_ptr<net::INetworkEndpoint>, std::string» _clients;
00025     std::shared_ptr<Lobby> _lobby;
00026 };
00027
00028 }
00029
00030 #endif /* !LOBBYSTRUCT_HPP_ */
```

5.247 Server.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** R-Type
00004 ** File description:
00005 ** Header
00006 */
00007
00008 #ifndef SERVER_HPP_
00009     #define SERVER_HPP_
00010
00011 #ifdef _WIN32
00012     #ifndef _WIN32_WINNT
00013         #define _WIN32_WINNT 0x0A00
00014     #endif
00015
00016     #ifndef WIN32_LEAN_AND_MEAN
00017         #define WIN32_LEAN_AND_MEAN
00018     #endif
00019 #endif
00020
00021 #include <queue>
00022 #include <map>
00023 #include <memory>
00024 #include <thread>
00025 #include <chrono>
00026 #include <mutex>
00027
00028 #include "LobbyStruct.hpp"
00029 #include "Lobby.hpp"
00030 #include "ServerConfig.hpp"
00031 #include "deltaTracker/ComponentDeltaTracker.hpp"
00032 #include "deltaTracker/ComponentSerializer.hpp"
00033 #include "../common/interfaces/INetwork.hpp"
00034 #include "../common/interfaces/IBuffer.hpp"
00035 #include "../libs/Network/common.hpp"
00036 #include "../common/DLLoader/DLLoader.hpp"
00037 #include "../common/DLLoader/LoaderType.hpp"
00038 #include "../common/constants.hpp"
00039 #include "../common/InputMapping/InputAction.hpp"
00040 #include "../common/resourceManager/ResourceManager.hpp"
00041 #include "../common/ECS/entity/registry/Registry.hpp"
00042 #include "../common/resourceManager/ResourceManager.hpp"
00043 #include "../common/Parser/Parser.hpp"
00044 #include "../common/systems/systemManager/ISystemManager.hpp"
00045 #include "gsm/machine/GameStateMachine.hpp"
00046 #include "initResourcesManager/ServerInputProvider.hpp"
00047 #include "Signal.hpp"
00048 #include "http/HttpServer.hpp"
00049
00050 namespace rserv {
00051     class Server {
00052         public:
00053             Server();
00054             ~Server();
00055
00056             void init();
00057             void start();
00058             void stop();
00059
00060             void setConfig(std::shared_ptr<ServerConfig> config);
00061             std::shared_ptr<ServerConfig> getConfig() const;
00062             uint16_t getPort() const;
```

```
00063             void setPort(uint16_t port);
00064
00065             int getState() const;
00066             void setState(int state);
00067
00068             void initRessourceManager(std::shared_ptr<Lobby> lobby);
00069             operator int() const noexcept;
00070
00071             std::shared_ptr<net::INetwork> getNetwork() const;
00072             void setNetwork(std::shared_ptr<net::INetwork> network);
00073
00074             void onClientConnected(uint8_t idClient);
00075             void onClientDisconnected(uint8_t idClient);
00076             void onPacketReceived(uint8_t idClient, const pm::IPacketManager &packet);
00077
00078             std::vector<uint8_t> getConnectedClients() const;
00079             std::vector<std::shared_ptr<net::INetworkEndpoint» getConnectedClientEndpoints() const;
00080             size_t getClientCount() const;
00081
00082             uint8_t findClientIdByEndpoint(const net::INetworkEndpoint &endpoint) const;
00083
00084             ServerInfo getServerInfo() const;
00085             std::map<std::string, int> loadUserStats(const std::string& username) const;
00086             void saveUserBannedStatus(const std::string& username, bool banned) const;
00087
00088
00089             /* Received Packet Handling */
00090             void processIncomingPackets();
00091             bool processConnections(std::pair<std::shared_ptr<net::INetworkEndpoint>,
      std::vector<uint8_t» client);
00092             bool processDisconnections(uint8_t idClient);
00093             bool requestCode(const net::INetworkEndpoint &endpoint);
00094             bool processConnectToLobby(std::pair<std::shared_ptr<net::INetworkEndpoint>,
      std::vector<uint8_t» payload);
00095             bool processMasterStart(std::pair<std::shared_ptr<net::INetworkEndpoint>,
      std::vector<uint8_t» payload);
00096             bool processRegistration(std::pair<std::shared_ptr<net::INetworkEndpoint>,
      std::vector<uint8_t» client);
00097             bool processLogin(std::pair<std::shared_ptr<net::INetworkEndpoint>, std::vector<uint8_t»
      client);
00098             bool processLeaderboardRequest(std::shared_ptr<net::INetworkEndpoint> client);
00099             bool processProfileRequest(std::shared_ptr<net::INetworkEndpoint> client);
00100             bool processNewChatMessage(std::pair<std::shared_ptr<net::INetworkEndpoint>,
      std::vector<uint8_t» payload);
00101             bool processRequestGameRulesUpdate(std::pair<std::shared_ptr<net::INetworkEndpoint>,
      std::vector<uint8_t» payload);
00102             void cleanupClosedLobbies();
00103             void checkClientTimeouts();
00104
00105             /* Sent Packet Handling */
00106             bool connectionPacket(const net::INetworkEndpoint& endpoint);
00107             bool canStartPacket(std::vector<std::shared_ptr<net::INetworkEndpoint» endpoints);
00108             bool serverStatusPacket();
00109             bool sendCodeLobbyPacket(const net::INetworkEndpoint &endpoint);
00110             bool lobbyConnectValuePacket(const net::INetworkEndpoint &endpoint, bool canConnect);
00111             bool connectUserPacket(const net::INetworkEndpoint &endpoint, const std::string
      &username);
00112             bool leaderboardPacket(const net::INetworkEndpoint &endpoint);
00113             bool profilePacket(const net::INetworkEndpoint &endpoint);
00114             bool newChatMessagePacket(const net::INetworkEndpoint &endpoint, std::vector<uint8_t>
      message);
00115             bool forceLeavePacket(const net::INetworkEndpoint &endpoint, constants::ForceLeaveType
      leaveType);
00116
00117
00118             uint32_t getSequenceNumber() const;
00119             std::shared_ptr<pm::IPacketManager> getPacketManager() const;
00120             std::shared_ptr<pm::IPacketManager> createNewPacketManager();
00121             uint32_t getNextEntityId();
00122             void incrementSequenceNumber();
00123
00124             std::string executeCommand(const std::string& command);
00125             std::string closeLobby(const std::string& lobbyId);
00126             std::string kickPlayer(const std::string& playerId);
00127             std::string banPlayer(const std::string& playerId);
00128             std::string unbanPlayer(const std::string& playerId);
00129             std::string toggleGodmod(const std::string& playerId);
00130
00131         private:
00132             void loadNetworkLibrary();
00133             void loadBufferLibrary();
00134             void loadPacketLibrary();
00135             DLLoader<createNetworkLib_t> _networloader;
00136             DLLoader<createBuffer_t> _bufferloader;
00137             DLLoader<createPacket_t> _packetloader;
00138
00139             std::shared_ptr<ServerConfig> _config;
```

```
00140             std::shared_ptr<net::INetwork> _network;
00141             std::shared_ptr<IBuffer> _buffer;
00142             std::shared_ptr<pm::IPacketManager> _packet;
00143
00144             /* Network handling variables */
00145             uint8_t _nextClientId;
00146             uint32_t _sequenceNumber;
00147             uint32_t _nextEntityId;
00148             std::mutex _clientsMutex;
00149
00150             /* Lobby handling variables */
00151             std::vector<std::tuple<uint8_t, std::shared_ptr<net::INetworkEndpoint>, std::string»
    _clients;
00152             std::map<uint8_t, bool> _clientsReady;
00153             std::vector<std::shared_ptr<LobbyStruct» _lobbyThreads;
00154             std::vector<std::shared_ptr<Lobby» _lobbies;
00155             std::map<uint8_t, std::shared_ptr<Lobby» _clientToLobby;
00156             std::unique_ptr<HttpServer> _httpServer;
00157
00158             /* Healthcheck variables */
00159             std::map<uint8_t, std::chrono::steady_clock::time_point> _clientLastHeartbeat;
00160
00161     };
00162 }  // namespace rserv
00163
00164 #endif /* !SERVER_HPP_ */
```

5.248 ServerConfig.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** Header
00004 ** File description:
00005 ** Header
00006 */
00007
00008 #ifndef SERVER_CONFIG_HPP_
00009     #define SERVER_CONFIG_HPP_
00010
00011 #include <cstdint>
00012 #include <string>
00013
00014 namespace rserv {
00015     class ServerConfig {
00016         public:
00017             ServerConfig();
00018             ~ServerConfig();
00019
00020             int getState() const;
00021
00022             void setPort(uint16_t port);
00023             uint16_t getPort() const;
00024
00025             void setState(int state);
00026
00027             std::string getIp() const;
00028             void setIp(std::string ip);
00029
00030             void setIsDebug(bool isDebug);
00031             bool getIsDebug() const;
00032
00033             void setTps(int64_t tps);
00034             int64_t getTps() const;
00035         private:
00036             int _state;
00037             uint16_t _port;
00038             std::string _ip;
00039             bool _isDebug;
00040             int64_t _tps;
00041     };
00042 } // namespace rserv = r-type server
00043
00044 #endif /* !SERVER_CONFIG_HPP_ */
```

5.249 EndOfMapDetectionSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
```

```
00003 ** ryanR-type
00004 ** File description:
00005 ** EndOfMapDetectionSystem
00006 */
00007
00008 #ifndef ENDOFMAPDETECTIONSYSTEM_HPP_
00009 #define ENDOFMAPDETECTIONSYSTEM_HPP_
00010
00011
00012 #include "../../../common/systems/base/ASystem.hpp"
00013
00014 namespace ecs  {
00015
00016
00017 class EndOfMapDetectionSystem : public ASystem{
00018     public:
00019         EndOfMapDetectionSystem();
00020         ~EndOfMapDetectionSystem();
00021
00022         void update(std::shared_ptr<ResourceManager> resourceManager, std::shared_ptr<Registry>
      registry, float deltaTime) override;
00023
00024     protected:
00025     private:
00026 };
00027
00028 }  // namespace ecs
00029
00030 #endif /* !ENDOFMAPDETECTIONSYSTEM_HPP_ */
```

5.250 ServerForceInputSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2026
00003 ** ryanR-type
00004 ** File description:
00005 ** ServerForceInputSystem
00006 */
00007
00008 #ifndef SERVERFORCEINPUTSYSTEM_HPP_
00009 #define SERVERFORCEINPUTSYSTEM_HPP_
00010
00011 #include "../../../common/systems/base/ASystem.hpp"
00012 #include "../../../common/ECS/entity/registry/Registry.hpp"
00013 #include "resourceManager/ResourceManager.hpp"
00014
00015 namespace ecs {
00016
00017 class ServerForceInputSystem : public ASystem {
00018     public:
00019         ServerForceInputSystem();
00020         ~ServerForceInputSystem() = default;
00021
00022         void update(
00023             std::shared_ptr<ResourceManager> resourceManager,
00024             std::shared_ptr<Registry> registry,
00025             float deltaTime
00026         ) override;
00027
00028     private:
00029         void callActivateOrDeactivateForce(
00030             std::shared_ptr<Registry> registry,
00031             Entity partId,
00032             Entity entityId
00033         );
00034 };
00035
00036 }  // namespace ecs
00037
00038 #endif /* !SERVERFORCEINPUTSYSTEM_HPP_ */
```

5.251 ServerMovementInputSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ServerMovementInputSystem
```

```
00006 */
00007
00008 #ifndef SERVERMOVEMENTINPUTSYSTEM_HPP_
00009 #define SERVERMOVEMENTINPUTSYSTEM_HPP_
00010
00011 #include "../../../common/systems/base/ASystem.hpp"
00012 #include "../../../common/components/temporary/InputIntentComponent.hpp"
00013 #include "../../../common/InputMapping/IInputProvider.hpp"
00014 #include "../../../common/InputMapping/InputAction.hpp"
00015 #include "../../initResourcesManager/ServerInputProvider.hpp"
00016 #include <memory>
00017
00018 namespace ecs {
00019
00020 class ServerMovementInputSystem : public ASystem {
00021     public:
00022         ServerMovementInputSystem();
00023         ~ServerMovementInputSystem() = default;
00024
00025         void update(std::shared_ptr<ResourceManager> resourceManager, std::shared_ptr<Registry>
      registry, float deltaTime) override;
00026
00027     private:
00028         math::Vector2f getMovementDirection(std::shared_ptr<IInputProvider> inputProvider, size_t
      clientID) const;
00029         void updateInputIntent(std::shared_ptr<Registry> registry, Entity entityId, const
      math::Vector2f &direction);
00030         math::Vector2f getAnalogStickInput(std::shared_ptr<IInputProvider> inputProvider, size_t
      clientID) const;
00031         math::Vector2f normalizeDirection(const math::Vector2f &direction) const;
00032 };
00033
00034 } // namespace ecs
00035
00036 #endif /* !SERVERMOVEMENTINPUTSYSTEM_HPP_ */
```

5.252 ServerShootInputSystem.hpp

```
00001 /*
00002 ** EPITECH PROJECT, 2025
00003 ** ryanR-type
00004 ** File description:
00005 ** ServerShootInputSystem
00006 */
00007
00008 #ifndef SERVERSHOOTINPUTSYSTEM_HPP_
00009 #define SERVERSHOOTINPUTSYSTEM_HPP_
00010
00011 #include "../../../common/systems/base/ASystem.hpp"
00012 #include "../../../common/components/temporary/ShootIntentComponent.hpp"
00013 #include "../../../common/InputMapping/IInputProvider.hpp"
00014 #include "../../../common/InputMapping/InputAction.hpp"
00015 #include "../../initResourcesManager/ServerInputProvider.hpp"
00016 #include <memory>
00017 #include <map>
00018
00019 namespace ecs {
00020
00021 class ServerShootInputSystem : public ASystem {
00022     public:
00023         ServerShootInputSystem();
00024         ~ServerShootInputSystem() = default;
00025
00026         void update(std::shared_ptr<ResourceManager> resourceManager, std::shared_ptr<Registry>
      registry, float deltaTime) override;
00027
00028     private:
00029         void updateShootIntent(std::shared_ptr<Registry> registry, Entity entityId);
00030
00031         std::map<ecs::Entity, float> _cooldowns;
00032 };
00033
00034 } // namespace ecs
00035
00036 #endif /* !SERVERSHOOTINPUTSYSTEM_HPP_ */
```

Index