Indie Studio

1.0

Generated by Doxygen 1.8.15

Chapter 1

Welcome to the Indie Studio project documentation

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

Class Documentation

4.1 Indie::Components::AIComponent Class Reference

AI Component.

`#include <AIComponent.hpp>`

Public Member Functions

- bool hasMoved (irr::core::vector3df position, irr::core::vector3df nextPosition, AIComponent ∗ai) const

  Checks if the ai has moved to the next position (in center of the case)
- void setNextDirection (std::vector< std::vector< OBJECT >> &map, irr::core::vector2di acPos)

  Sets the nextDirection.
- void setDirection (DIRECTION direction)

  Sets the direction of AI.
- void setAction (ACTION action)

  Sets the action of AI.
- void setBehavior (unsigned int value)

  Sets behavior of AI, the behvior depends ont the number of enemies in the same area with the AI.
- void setDodge (bool value)

  Sets the dodge value, if is on player range.
- bool getDodge () const

  Gets the dodge value of AI.
- unsigned int getBehavior () const

  Gets the behavior value of AI.
- DIRECTION getDirection () const

  Gets the actual direction of the AI.
- ACTION getAction () const

  Gets the actual action of AI.
- irr::core::vector2di getNextPosition () const

  Gets the next postion of AI.

4.1.1 Detailed Description

AI Component.

4.1.2 Member Function Documentation

4.1.2.1 getAction()

```
ACTION Indie::Components::AIComponent::getAction ( ) const
```

Gets the actual action of AI.

Returns

The action value

4.1.2.2 getBehavior()

```
unsigned int Indie::Components::AIComponent::getBehavior ( ) const
```

Gets the behavior value of AI.

Returns

The behavior value

4.1.2.3 getDirection()

```
DIRECTION Indie::Components::AIComponent::getDirection ( ) const
```

Gets the actual direction of the AI.

Returns

The direction value

4.1.2.4 getDodge()

```
bool Indie::Components::AIComponent::getDodge ( ) const
```

Gets the dodge value of AI.

Returns

The dodge value

4.1.2.5 getNextPosition()

```
irr::core::vector2di Indie::Components::AIComponent::getNextPosition ( ) const
```

Gets the next postion of AI.

Returns

The vector of the nextPosition

4.1.2.6 hasMoved()

```
bool Indie::Components::AIComponent::hasMoved (
            irr::core::vector3df position,
            irr::core::vector3df nextPosition,
            AIComponent * ai ) const
```

Checks if the ai has moved to the next position (in center of the case)

Returns

true Ai has move
false Ai hasn't move yet

4.1.2.7 setAction()

```
void Indie::Components::AIComponent::setAction (
            ACTION action )
```

Sets the action of AI.

Parameters

action	Enum Action: All possible actions for the AI

4.1.2.8 setBehavior()

```
void Indie::Components::AIComponent::setBehavior (
            unsigned int value )
```

Sets behavior of AI, the behvior depends ont the number of enemies in the same area with the AI.

Parameters

value	The behavior value

4.1.2.9 setDirection()

```
void Indie::Components::AIComponent::setDirection (
            DIRECTION direction )
```

Sets the direction of AI.

Parameters

direction	Enum Direction: All direction actions for the AI

4.1.2.10 setDodge()

```
void Indie::Components::AIComponent::setDodge (
            bool value )
```

Sets the dodge value, if is on player range.

Parameters

value	The dodge value

4.1.2.11 setNextDirection()

```
void Indie::Components::AIComponent::setNextDirection (
            std::vector< std::vector< OBJECT >> & map,
            irr::core::vector2di acPos )
```

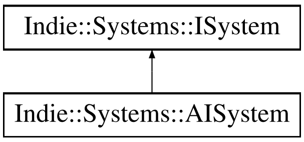
Sets the nextDirection.

Parameters

map	The pathfinding map with the shortless path
acPos	Actual position of AI

4.2 Indie::Systems::AISystem Class Reference

Inheritance diagram for Indie::Systems::AISystem:



Public Member Functions

- void onUpdate (irr::f32 deltaTime, EntityManager &entityManager) const final

  Updates things according to system type.

4.2.1 Member Function Documentation

4.2.1.1 onUpdate()

```
void Indie::Systems::AISystem::onUpdate (
            irr::f32 deltaTime,
            EntityManager & entityManager ) const  [final], [virtual]
```

Updates things according to system type.

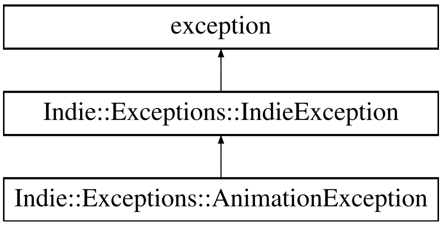
Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.
entityManager	The entity manager

Implements Indie::Systems::ISystem.

4.3 Indie::Exceptions::AnimationException Class Reference

Inheritance diagram for Indie::Exceptions::AnimationException:


exception
Indie::Exceptions::IndieException
Indie::Exceptions::AnimationException

Public Member Functions

- **AnimationException** (const std::string &component, const std::string &message)

4.4 Indie::Bar Class Reference

Public Member Functions

- **Bar** (ContextManager &context)
- void **init** (const std::string &filepath, int min, int max, int current)
- void **update** ()
- void **setSize** (int min, int max)
- void **setValue** (int value)
- void **setLevel** (int level)
- void **draw** (irr::core::position2d< irr::s32 > pos) const

4.5 Indie::Components::BombComponent Class Reference

Bomb component.

```
#include <BombComponent.hpp>
```

Public Member Functions

- **BombComponent** (int idOwner, unsigned int range)
- unsigned int getRange () const

  Gets the range of the bomb.
- int getIdOwner () const

  Gets the Id of the player who droped the bomb.
- bool hasExploded () const

  Checks if the bomb has exploded.
- void setRange (unsigned int range)

  Sets the Range of the bomb.
- void setExploded (bool exploded)

  Sets bomb status.

4.5.1 Detailed Description

Bomb component.

4.5.2 Member Function Documentation

4.5.2.1 getIdOwner()

```
int Indie::Components::BombComponent::getIdOwner ( ) const
```

Gets the Id of the player who droped the bomb.

Returns

The player's Id

4.5.2.2 getRange()

```
unsigned int Indie::Components::BombComponent::getRange ( ) const
```

Gets the range of the bomb.

Returns

The bomb range

4.5.2.3 hasExploded()

```
bool Indie::Components::BombComponent::hasExploded ( ) const
```

Checks if the bomb has exploded.

Returns

true Bomb has exploded
false Bomb hasn't exploded yet

4.5.2.4 setExploded()

```
void Indie::Components::BombComponent::setExploded (
            bool exploded )
```

Sets bomb status.

Parameters

exploded	True if the bomb has exploded, false otherwise

4.5.2.5 setRange()

```
void Indie::Components::BombComponent::setRange (
             unsigned int range )
```

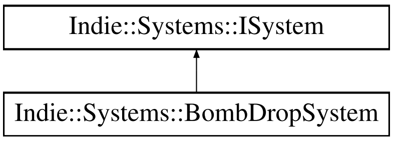
Sets the Range of the bomb.

Parameters

range	The range

4.6 Indie::Systems::BombDropSystem Class Reference

Inheritance diagram for Indie::Systems::BombDropSystem:



Public Member Functions

- void onUpdate (irr::f32 deltaTime, EntityManager &entityManager) const final

    Updates things according to system type.

4.6.1 Member Function Documentation

4.6.1.1 onUpdate()

```
void Indie::Systems::BombDropSystem::onUpdate (
             irr::f32 deltaTime,
             EntityManager & entityManager ) const  [final], [virtual]
```

Updates things according to system type.

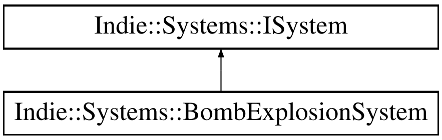
Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.
entityManager	The entity manager

Implements Indie::Systems::ISystem.

4.7 Indie::Systems::BombExplosionSystem Class Reference

Inheritance diagram for Indie::Systems::BombExplosionSystem:



Public Member Functions

- void onUpdate (irr::f32 deltaTime, EntityManager &entityManager) const final

  Updates things according to system type.

4.7.1 Member Function Documentation

4.7.1.1 onUpdate()

```
void Indie::Systems::BombExplosionSystem::onUpdate (
            irr::f32 deltaTime,
            EntityManager & entityManager ) const  [final], [virtual]
```

Updates things according to system type.

Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.
entityManager	The entity manager

Implements Indie::Systems::ISystem.

4.8 Indie::Button Class Reference

Public Types

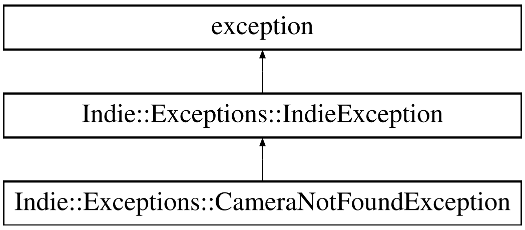
- enum **Status** : int { **Idle**, **Selected**, **Pressed** }

Public Member Functions

- **Button** (ContextManager &context, bool sound=true)
- void **init** (ContextManager &context, const std::string &filepath, int posX, int posY, irr::core::position2d< irr↩ ::s32 > pos, bool sound=true)
- void **update** (std::pair< int, int >)
- void **draw** ()
- Status **getStatus** () const
- void **setStatus** (Status)

4.9 Indie::Exceptions::CameraNotFoundException Class Reference

Inheritance diagram for Indie::Exceptions::CameraNotFoundException:


exception
Indie::Exceptions::IndieException
Indie::Exceptions::CameraNotFoundException

Public Member Functions

- **CameraNotFoundException** (const std::string &component, const std::string &message)

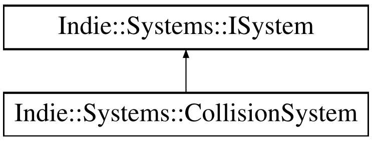
4.10 Indie::Checkbox Class Reference

Public Member Functions

- **Checkbox** (ContextManager &context)
- void **init** (std::string const &filepath, int posX, int posY, irr::core::position2d< irr::s32 > pos)
- void **update** (std::pair< int, int >)
- void **draw** ()
- bool **getStatus** () const
- void **setStatus** (bool status)

4.11 Indie::Systems::CollisionSystem Class Reference

Inheritance diagram for Indie::Systems::CollisionSystem:



Public Member Functions

- void onUpdate (irr::f32 deltaTime, EntityManager &entityManager) const final

  Updates things according to system type.

4.11.1 Member Function Documentation

4.11.1.1 onUpdate()

```
void Indie::Systems::CollisionSystem::onUpdate (
            irr::f32 deltaTime,
            EntityManager & entityManager ) const  [final], [virtual]
```

Updates things according to system type.

Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.
entityManager	The entity manager

Implements Indie::Systems::ISystem.

4.12 Indie::ContextManager Class Reference

ContextManager class.

```
#include <ContextManager.hpp>
```

Public Member Functions

- **ContextManager** (irr::core::dimension2d< irr::u32 > size=DIM(1280, 720))
- irr::IrrlichtDevice ∗ getDevice () const

  Gets the device.
- irr::video::IVideoDriver ∗ getDriver () const

  Gets the driver.
- irr::scene::ISceneManager ∗ getSceneManager () const

  Gets the Scene Manager.
- irr::gui::IGUIEnvironment ∗ getGuiEnv () const

  Gets the GUI Environment.
- void displayImage (Image ∗image, irr::core::position2d< irr::s32 > pos=irr::core::position2d< irr::s32 >(0, 0), irr::video::SColor color=irr::video::SColor(255, 255, 255, 255))

  Displays an image on the screen.
- void displayImage (Image ∗image, irr::core::rect< irr::s32 > rect, irr::core::position2d< irr::s32 > pos=irr↩ ::core::position2d< irr::s32 >(0, 0), irr::video::SColor color=irr::video::SColor(255, 255, 255, 255))

  Displays an image on the screen.

4.12.1 Detailed Description

ContextManager class.

4.12.2 Member Function Documentation

4.12.2.1 displayImage() [1/2]

```
void Indie::ContextManager::displayImage (
            Image * image,
            irr::core::position2d< irr::s32 > pos = irr::core::position2d< irr::s32 >(0, 0),
            irr::video::SColor color = irr::video::SColor(255, 255, 255, 255) )
```

Displays an image on the screen.

Parameters

image	The image to display
pos	The position of the image
color	The color of the image

4.12.2.2 displayImage() [2/2]

```
void Indie::ContextManager::displayImage (
            Image * image,
            irr::core::rect< irr::s32 > rect,
            irr::core::position2d< irr::s32 > pos = irr::core::position2d< irr::s32 >(0, 0),
            irr::video::SColor color = irr::video::SColor(255, 255, 255, 255) )
```

Displays an image on the screen.

Parameters

image	The image to display
rect	Part of the image to display
pos	The position of the image
color	The color of the image

4.12.2.3 getDevice()

```
irr::IrrlichtDevice* Indie::ContextManager::getDevice ( ) const  [inline]
```

Gets the device.

Returns

irr::IrrlichtDevice∗ The device

4.12.2.4 getDriver()

```
irr::video::IVideoDriver* Indie::ContextManager::getDriver ( ) const  [inline]
```

Gets the driver.

Returns

irr::video::IVideoDriver∗ The driver

4.12.2.5 getGuiEnv()

```
irr::gui::IGUIEnvironment* Indie::ContextManager::getGuiEnv ( ) const  [inline]
```

Gets the GUI Environment.

Returns

irr::gui::IGUIEnvironment∗ The GUI Environment

4.12.2.6 getSceneManager()

```
irr::scene::ISceneManager* Indie::ContextManager::getSceneManager ( ) const  [inline]
```

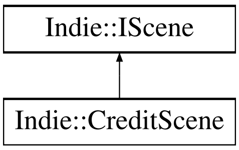
Gets the Scene Manager.

Returns

irr::scene::ISceneManager∗ The scene manager

4.13 Indie::CreditScene Class Reference

Inheritance diagram for Indie::CreditScene:



Public Member Functions

- **CreditScene** (ContextManager &context)
- void init () final

  Inits the scene.
- void reset () final

  Resets the scene.
- void update (irr::f32 deltaTime) final

  Updates the scene.
- void renderPre3D () final

  Renders before 3D rendering.
- void renderPost3D () final

  Renders after 3D rendering.

4.13.1 Member Function Documentation

4.13.1.1 renderPost3D()

```
void Indie::CreditScene::renderPost3D ( )  [final], [virtual]
```

Renders after 3D rendering.

Implements Indie::IScene.

4.13.1.2 update()

```
void Indie::CreditScene::update (
             irr::f32 deltaTime )  [final], [virtual]
```

Updates the scene.

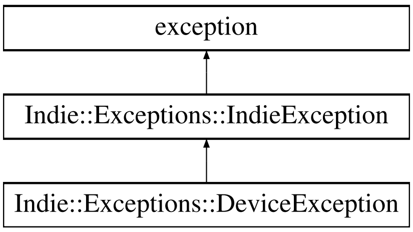
Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.

Implements Indie::IScene.

4.14 Indie::Exceptions::DeviceException Class Reference

Inheritance diagram for Indie::Exceptions::DeviceException:


exception
Indie::Exceptions::IndieException
Indie::Exceptions::DeviceException

Public Member Functions

- **DeviceException** (const std::string &component, const std::string &message)

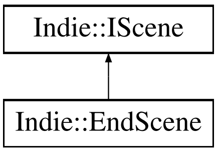
4.15 Indie::EndGame Struct Reference

Public Attributes

- std::vector< std::pair< std::string, int > > **scores**
- unsigned int **xp**
- MATCH_PLAY **matchPlay**

4.16 Indie::EndScene Class Reference

Inheritance diagram for Indie::EndScene:



Public Member Functions

- **EndScene** (ContextManager &context)
- void init () final

  Inits the scene.
- void reset () final

  Resets the scene.
- void update (irr::f32 deltaTime) final

  Updates the scene.
- void renderPre3D () final

  Renders before 3D rendering.
- void renderPost3D () final

  Renders after 3D rendering.
- const Indie::EndGame & **getEndGame** (void) const
- void **setEndGame** (const Indie::EndGame &endGame)
- void **setPlayerNames** (const std::vector< std::string > &names)

4.16.1 Member Function Documentation

4.16.1.1 renderPost3D()

```
void Indie::EndScene::renderPost3D ( )  [final], [virtual]
```

Renders after 3D rendering.

Implements Indie::IScene.

4.16.1.2 update()

```
void Indie::EndScene::update (
            irr::f32 deltaTime )  [final], [virtual]
```

Updates the scene.

Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.

Implements Indie::IScene.

4.17 Indie::Entity Class Reference

The Entity class.

```
#include <Entity.hpp>
```

Public Member Functions

- **Entity** (int id)
- template<class T >
  bool has () const

  Checks if the entity has the T component.
- template<typename T , typename TNext , typename... TList>
  bool has () const

  Checks if the entity has all given components.
- template<typename T >
  T ∗ getComponent () const

  Gets wanted component.

- template<typename T , typename... Args>
  void addComponent (Args &&... args)

  Adds a component to the entity.
- template<typename T >
  void removeComponent ()

  Removes given component.
- void removeAllComponents ()

  Deletes all components of the entity.
- int getId () const

  Gets the entity's id.
- bool isPendingDestroy () const

  Checks if the entity must be destroyed.
- void needDestroy ()

  By calling this, the entity is set as it will be destroyed soon.

4.17.1 Detailed Description

The Entity class.

4.17.2 Member Function Documentation

4.17.2.1 addComponent()

```
template<typename T , typename... Args>
void Indie::Entity::addComponent (
             Args &&... args ) [inline]
```

Adds a component to the entity.

Template Parameters

T	the component to add
Args	Component's ctor args

Parameters

args	Component's ctor args

4.17.2.2 getComponent()

```
template<typename T >
T* Indie::Entity::getComponent ( ) const [inline]
```

Gets wanted component.

Template Parameters

T	The component to get

Returns

T∗ The component found

4.17.2.3 getId()

```
int Indie::Entity::getId ( ) const  [inline]
```

Gets the entity's id.

Returns

int

4.17.2.4 has() [1/2]

```
template<class T >
bool Indie::Entity::has ( ) const  [inline]
```

Checks if the entity has the T component.

Template Parameters

T	The component

Returns

true The entity has the given component
false The entity hasn't the given component

4.17.2.5 has() [2/2]

```
template<typename T , typename TNext , typename...  TList>
bool Indie::Entity::has ( ) const  [inline]
```

Checks if the entity has all given components.

Template Parameters

T	The component to check
TNext	The next component to check
TList	The rest of the list

Returns

> true The component has all given components
> false The component hasn't all given components

4.17.2.6 isPendingDestroy()

```
bool Indie::Entity::isPendingDestroy ( ) const  [inline]
```

Checks if the entity must be destroyed.

Returns

> true The entity must be destroyed
> false The entity shouldn't be destroyed

4.17.2.7 removeComponent()

```
template<typename T >
void Indie::Entity::removeComponent ( )  [inline]
```

Removes given component.

Template Parameters

T	The component to remove

4.18 Indie::EntityBuilder Class Reference

The EntityBuilder class.

```
#include <EntityBuilder.hpp>
```

Public Member Functions

- Entity ∗ createGround (const irr::core::vector3df &position, const std::string &modelPath, const std::string &texturePath)

Creates a ground entity.

- Entity ∗ createPlayer (const irr::core::vector3df &position, const std::string &modelPath, const std::string &texturePath, std::unordered_map< irr::EKEY_CODE, Components::KEY_TYPE > keys, const std↩ ::string &playerName, Components::PlayerComponent::PLAYER_COLOR playerColor, Components::↩ PlayerComponent::PLAYER_START_POSITION startPosition)

  Creates a player entity.
- Entity ∗ createAi (const irr::core::vector3df &position, const std::string &modelPath, const std::string &texturePath, const std::string &playerName, Components::PlayerComponent::PLAYER_COLOR player↩ Color, Components::PlayerComponent::PLAYER_START_POSITION startPosition)

  Creates a AI entity.
- Entity ∗ createWall (const irr::core::vector3df &position, const std::string &modelPath, const std::string &texturePath, bool canBeDestroyed)

  Creates a Wall entity.
- Entity ∗ createBomb (const irr::core::vector3df &position, const std::string &modelPath, const std::string &texturePath, int idOwner, unsigned int range)

  Creates a Bomb entity.
- Entity ∗ createMap (const irr::core::vector2di &dimension, const Indie::Components::MAP_TYPE &type, const Indie::Components::THEME &theme, const std::string &mapPath="")

  Creates a new Map entity.
- Entity ∗ createLava (const irr::core::vector3df &position, const std::string &modelPath, const std::string &texturePath, float angle, int ownerId)

  Creates a new Lava entity.
- Entity ∗ createPowerUp (irr::core::vector3df position, const std::string &modelPath, const std::string &texturePath, Components::POWERUP_TYPE type)

  Creates a new power up entity.
- Entity ∗ createPowerDown (irr::core::vector3df position, const std::string &modelPath, const std::string &texturePath, Components::POWERDOWN_TYPE type)

  Creates a new power down entity.
- Entity ∗ createShake (void)

  Creates a Shake entity.

4.18.1 Detailed Description

The EntityBuilder class.

4.18.2 Member Function Documentation

4.18.2.1 createAi()

```
Entity* Indie::EntityBuilder::createAi (
            const irr::core::vector3df & position,
            const std::string & modelPath,
            const std::string & texturePath,
            const std::string & playerName,
            Components::PlayerComponent::PLAYER_COLOR playerColor,
            Components::PlayerComponent::PLAYER_START_POSITION startPosition )
```

Creates a AI entity.

Parameters

position	The position of the AI
modelPath	The model of the AI
texturePath	The texture of the AI
playerNb	The name of the AI
playerColor	The color of the AI
startPosition	Where the AI starts in the map

Returns

Entity∗ The new AI entity

4.18.2.2 createBomb()

```
Entity* Indie::EntityBuilder::createBomb (
            const irr::core::vector3df & position,
            const std::string & modelPath,
            const std::string & texturePath,
            int idOwner,
            unsigned int range )
```

Creates a Bomb entity.

Parameters

position	The position of the bomb
modelPath	The model of the bomb
texturePath	The texture of the bomb
idOwner	The owner id
range	The range of the bomb

Returns

Entity∗ The new bomb entity

4.18.2.3 createGround()

```
Entity* Indie::EntityBuilder::createGround (
            const irr::core::vector3df & position,
            const std::string & modelPath,
            const std::string & texturePath )
```

Creates a ground entity.

Parameters

position	The position of the ground
modelPath	The model of the ground
texturePath	The texture of the ground

Returns

Entity∗ The new ground entity

4.18.2.4 createLava()

```
Entity* Indie::EntityBuilder::createLava (
            const irr::core::vector3df & position,
            const std::string & modelPath,
            const std::string & texturePath,
            float angle,
            int ownerId )
```

Creates a new Lava entity.

Parameters

position	The position of the lava
modelPath	The model of the lava
texturePath	The texture of the lava
angle	The angle of the lava
ownerId	The owner id

Returns

Entity∗ The new lava entity

4.18.2.5 createMap()

```
Entity* Indie::EntityBuilder::createMap (
            const irr::core::vector2di & dimension,
            const Indie::Components::MAP_TYPE & type,
            const Indie::Components::THEME & theme,
            const std::string & mapPath = "" )
```

Creates a new Map entity.

Parameters

dimension	Dimensions of the map
type	Type of the map
theme	Theme of the map
mapPath	The path of the map (not mandatory)

Returns

Entity∗ The new map entity

4.18.2.6 createPlayer()

```
Entity* Indie::EntityBuilder::createPlayer (
            const irr::core::vector3df & position,
            const std::string & modelPath,
            const std::string & texturePath,
            std::unordered_map< irr::EKEY_CODE, Components::KEY_TYPE > keys,
            const std::string & playerName,
            Components::PlayerComponent::PLAYER_COLOR playerColor,
            Components::PlayerComponent::PLAYER_START_POSITION startPosition )
```

Creates a player entity.

Parameters

position	The position of the player
modelPath	The model of the player
texturePath	The texture of the player
keys	The keys of the player
playerName	The name of the player
playerColor	The color of the player
startPosition	Where the player starts in the map

Returns

Entity∗ The new player entity

4.18.2.7 createPowerDown()

```
Entity* Indie::EntityBuilder::createPowerDown (
            irr::core::vector3df position,
            const std::string & modelPath,
            const std::string & texturePath,
            Components::POWERDOWN_TYPE type )
```

Creates a new power down entity.

Parameters

position	The position of the power down
modelPath	The model of the power down
texturePath	The texture of the power down
type	The type of the power down

Returns

Entity∗ The new power pown

4.18.2.8 createPowerUp()

```
Entity* Indie::EntityBuilder::createPowerUp (
            irr::core::vector3df position,
            const std::string & modelPath,
            const std::string & texturePath,
            Components::POWERUP_TYPE type )
```

Creates a new power up entity.

Parameters

position	The position of the power up
modelPath	The model of the power up
texturePath	The texture of the power up
type	The type of the power up

Returns

Entity∗ The new power up entity

4.18.2.9 createShake()

```
Entity* Indie::EntityBuilder::createShake (
            void )
```

Creates a Shake entity.

Returns

Entity∗ The new shake entity

4.18.2.10 createWall()

```
Entity* Indie::EntityBuilder::createWall (
            const irr::core::vector3df & position,
            const std::string & modelPath,
            const std::string & texturePath,
            bool canBeDestroyed )
```

Creates a Wall entity.

Parameters

position	The position of the wall
modelPath	The model of the wall
texturePath	The texture of the wall
canBeDestroyed	The wall is a solid of destructible one

Returns

Entity∗ The new wall entity

4.19 Indie::EntityIterator< Types > Class Template Reference

Public Member Functions

- **EntityIterator** (const EntityManager ∗entityManager, size_t index, bool isEnd)
- bool **isEnd** () const
- Entity ∗ **get** () const
- Entity ∗ **operator∗** () const
- bool **operator==** (const EntityIterator< Types... > &rhs) const
- bool **operator!=** (const EntityIterator< Types... > &rhs) const
- EntityIterator< Types... > & **operator++** ()

4.20 Indie::EntityManager Class Reference

Public Member Functions

- Entity ∗ **createEntity** ()
- template<typename T , typename... Args>
  Entity ∗ **createUniqueEntity** (Args &&... args)
- template<typename T >
  Entity ∗ **getUniqueEntity** ()
- template<typename... Types>
  EntityView< Types... > **each** () const
- Entity ∗ **getByIndex** (size_t index) const
- Entity ∗ **getById** (int id) const
- size_t **getCount** () const
- void **cleanup** ()
- void **reset** ()

4.21 Indie::EntityView< Types > Class Template Reference

Public Member Functions

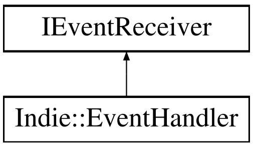
- **EntityView** (EntityIterator< Types... > begin, EntityIterator< Types... > end)
- const EntityIterator< Types... > & **begin** () const
- const EntityIterator< Types... > & **end** () const

4.22 Indie::EventHandler Class Reference

EventHandler singleton class.

```
#include <EventHandler.hpp>
```

Inheritance diagram for Indie::EventHandler:



Public Member Functions

- bool OnEvent (const irr::SEvent &event) final

  Called automatically by irrlicht when an event happens. It stores key informations.
- bool isKeyPressed (irr::EKEY_CODE key) const

  Checks if given key is pressed or not.
- bool isKeyPressedAtOnce (irr::EKEY_CODE key) const

  Checks if the given key is pressed at once.
- bool isKeyReleased (irr::EKEY_CODE key) const

  Checks if given key has been released.
- bool isAnyKeyPressed () const

  Checks if any key is pressed.
- bool isAnyKeyPressedAtOnce () const

  Checks if any key is pressed at once.
- void resetKeys (void)

  Resets all the key sets.
- void resetKeysStatus (void)

  Resets the keys status.
- void resetKeysStatusOnce (void)

  Resets the keys status pressed once.
- void resetKeysReleased (void)

  Resets the keys status released.
- void **operator=** (EventHandler const &)=delete
- **EventHandler** (EventHandler const &)=delete

Static Public Member Functions

- static EventHandler & getInstance ()

  Gets the instance of the class.

4.22.1 Detailed Description

EventHandler singleton class.

4.22.2 Member Function Documentation

4.22.2.1 getInstance()

```
static EventHandler& Indie::EventHandler::getInstance ( )  [static]
```

Gets the instance of the class.

Returns

EventHandler& The instance

4.22.2.2 isAnyKeyPressed()

```
bool Indie::EventHandler::isAnyKeyPressed ( ) const
```

Checks if any key is pressed.

Returns

true A key has been pressed
false No key has been pressed

4.22.2.3 isAnyKeyPressedAtOnce()

```
bool Indie::EventHandler::isAnyKeyPressedAtOnce ( ) const
```

Checks if any key is pressed at once.

Returns

true A key has been pressed
false No key has been pressed

4.22.2.4 isKeyPressed()

```
bool Indie::EventHandler::isKeyPressed (
            irr::EKEY_CODE key ) const
```

Checks if given key is pressed or not.

Parameters

key	The key to check

Returns

true The key is pressed
false The key is not pressed

4.22.2.5 isKeyPressedAtOnce()

```
bool Indie::EventHandler::isKeyPressedAtOnce (
            irr::EKEY_CODE key ) const
```

Checks if the given key is pressed at once.

Parameters

key	The key to check

Returns

true The key is pressed
false The key is not pressed

4.22.2.6 isKeyReleased()

```
bool Indie::EventHandler::isKeyReleased (
            irr::EKEY_CODE key ) const
```

Checks if given key has been released.

Parameters

key	The key to check

Returns

true The key has been released
false The key hasn't been released

4.22.2.7 OnEvent()

```
bool Indie::EventHandler::OnEvent (
            const irr::SEvent & event )  [final]
```

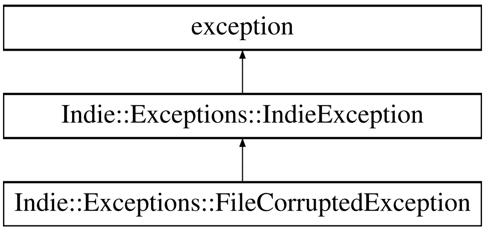
Called automatically by irrlicht when an event happens. It stores key informations.

Parameters

event	The generated irrlicht event

4.23 Indie::Exceptions::FileCorruptedException Class Reference

Inheritance diagram for Indie::Exceptions::FileCorruptedException:

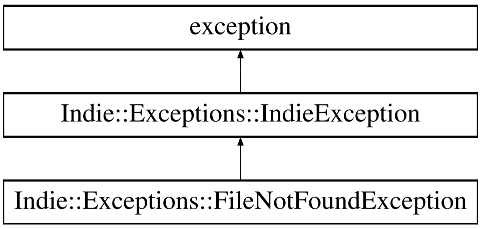

exception
Indie::Exceptions::IndieException
Indie::Exceptions::FileCorruptedException

Public Member Functions

- **FileCorruptedException** (const std::string &component, const std::string &message)

4.24 Indie::Exceptions::FileNotFoundException Class Reference

Inheritance diagram for Indie::Exceptions::FileNotFoundException:


exception
Indie::Exceptions::IndieException
Indie::Exceptions::FileNotFoundException

Public Member Functions

- **FileNotFoundException** (const std::string &component, const std::string &message)

4.25 Indie::FileParser Class Reference

FileParser class.

```
#include <FileParser.hpp>
```

Public Member Functions

- std::unordered_map< std::string, std::string > parse (const std::string &filepath, size_t nbValues=2) const

  Reads values from given file which must be in CSV format and return the data parsed.
- void writeToFile (const std::string &filepath, const std::unordered_map< std::string, std::string > &data) const

  Writes given data in a file located at given filepath.

4.25.1 Detailed Description

FileParser class.

4.25.2 Member Function Documentation

4.25.2.1 parse()

```
std::unordered_map<std::string, std::string> Indie::FileParser::parse (
            const std::string & filepath,
            size_t nbValues = 2 ) const
```

Reads values from given file which must be in CSV format and return the data parsed.

Parameters

filepath	The filepath of the file to parse

Returns

std::vector<std::vector<std::string>> The parsed data

4.25.2.2 writeToFile()

```
void Indie::FileParser::writeToFile (
            const std::string & filepath,
            const std::unordered_map< std::string, std::string > & data ) const
```

Writes given data in a file located at given filepath.

Parameters

filepath	File location
data	Data to write

4.26 Indie::Components::GameComponent Class Reference

Public Member Functions

- **GameComponent** (irr::f32 timeToEnd)
- MATCH_PLAY **getGameStatus** () const
- void **setGameStatus** (MATCH_PLAY status)
- irr::f32 **getTimeToEnd** (void) const
- void **setTimeToEnd** (irr::f32 value)

4.27 Indie::GameEngine Class Reference

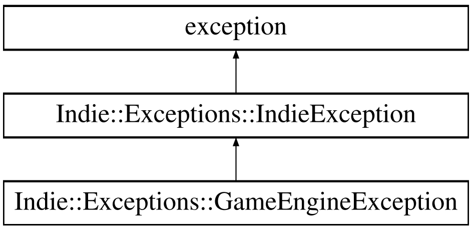
Public Member Functions

- void startGame (void)

  Starts the game.

4.28 Indie::Exceptions::GameEngineException Class Reference

Inheritance diagram for Indie::Exceptions::GameEngineException:


exception
Indie::Exceptions::IndieException
Indie::Exceptions::GameEngineException

Public Member Functions

- **GameEngineException** (const std::string &component, const std::string &message)

4.29 Indie::GameInfos Struct Reference

Public Attributes

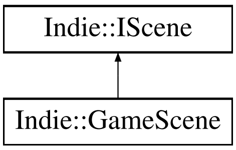
- unsigned int **xp**
- unsigned int **lvl**
- std::unordered_map< std::string, int > **scores_map**

Static Public Attributes

- static const std::vector< unsigned int > **xp_level**

4.30 Indie::GameScene Class Reference

Inheritance diagram for Indie::GameScene:



Public Types

- enum **MODE** { **SOLO**, **MULTI** }

Public Member Functions

- **GameScene** (ContextManager &context)
- void init () final

  Inits the scene.
- void reset () final

  Resets the scene.
- void update (irr::f32 deltaTime) final

  Updates the scene.
- void renderPre3D () final

  Renders before 3D rendering.
- void renderPost3D () final

  Renders after 3D rendering.
- InitGame ∗ **getInitGame** (void) const
- void **setInitGame** (const InitGame &initGame)

4.30.1 Member Function Documentation

4.30.1.1 renderPost3D()

```
void Indie::GameScene::renderPost3D ( )  [final], [virtual]
```

Renders after 3D rendering.

Implements Indie::IScene.

4.30.1.2 update()

```
void Indie::GameScene::update (
             irr::f32 deltaTime )  [final], [virtual]
```

Updates the scene.

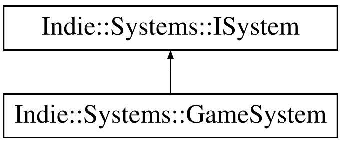
Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.

Implements Indie::IScene.

4.31 Indie::Systems::GameSystem Class Reference

Inheritance diagram for Indie::Systems::GameSystem:



Public Member Functions

- void onUpdate (irr::f32 deltaTime, EntityManager &entityManager) const final

  Updates things according to system type.

4.31.1 Member Function Documentation

4.31.1.1 onUpdate()

```
void Indie::Systems::GameSystem::onUpdate (
            irr::f32 deltaTime,
            EntityManager & entityManager ) const  [final], [virtual]
```

Updates things according to system type.

Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.
entityManager	The entity manager

Implements Indie::Systems::ISystem.

4.32 Indie::Components::HitboxComponent Class Reference

Hitbox component.

```
#include <HitboxComponent.hpp>
```

Public Member Functions

- **HitboxComponent** (irr::core::vector3df, const ContextManager &)
- irr::scene::IAnimatedMeshSceneNode ∗ getMesh (void) const

  Gets the stored Mesh.

4.32.1 Detailed Description

Hitbox component.

4.32.2 Member Function Documentation

4.32.2.1 getMesh()

```
irr::scene::IAnimatedMeshSceneNode* Indie::Components::HitboxComponent::getMesh (
            void ) const
```

Gets the stored Mesh.

Returns

irr::scene::IAnimatedMeshSceneNode∗ The mesh

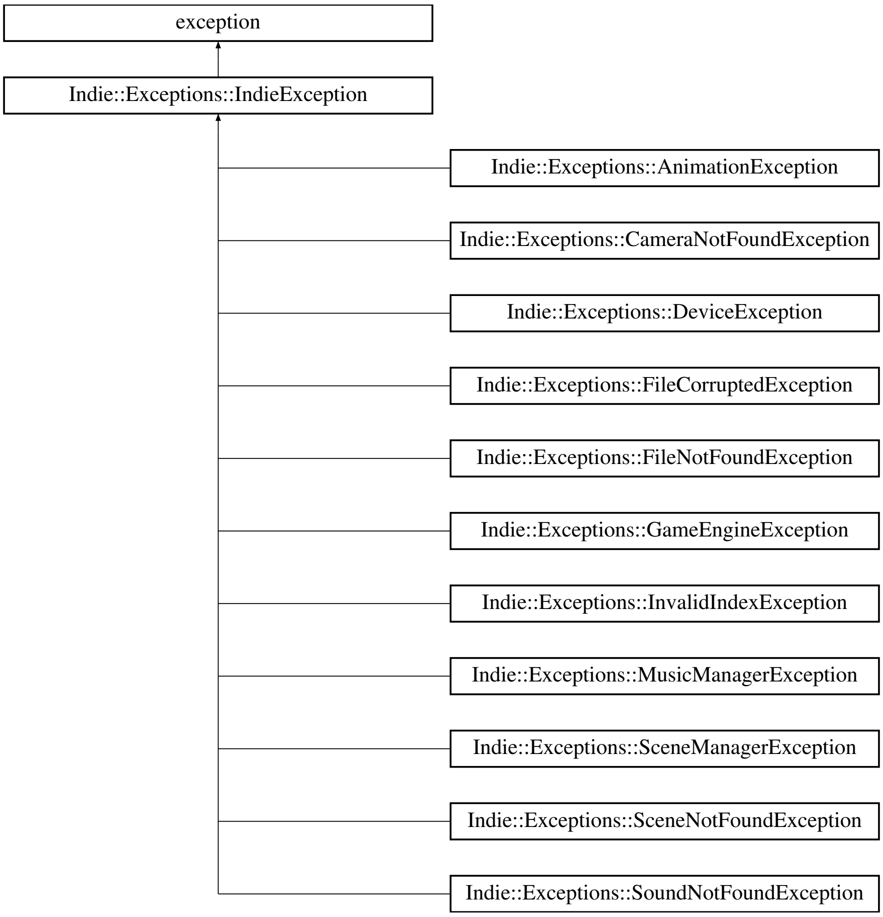
4.33 Indie::ImageLoader Class Reference

Public Member Functions

- Image ∗ **getImage** (std::string const &filepath)

4.34 Indie::Exceptions::IndieException Class Reference

Inheritance diagram for Indie::Exceptions::IndieException:


exception
Indie::Exceptions::IndieException
Indie::Exceptions::AnimationException
Indie::Exceptions::CameraNotFoundException
Indie::Exceptions::DeviceException
Indie::Exceptions::FileCorruptedException
Indie::Exceptions::FileNotFoundException
Indie::Exceptions::GameEngineException
Indie::Exceptions::InvalidIndexException
Indie::Exceptions::MusicManagerException
Indie::Exceptions::SceneManagerException
Indie::Exceptions::SceneNotFoundException
Indie::Exceptions::SoundNotFoundException

Public Member Functions

- **IndieException** (const std::string &component, const std::string &message)
- const char ∗ what () const noexcept final

  Returns a C-style character string describing the general cause of the current error.
- const std::string & getComponent (void) const noexcept

  Gets the component where the exception has been thrown.

4.34.1 Member Function Documentation

4.34.1.1 getComponent()

```
const std::string& Indie::Exceptions::IndieException::getComponent (
            void  ) const  [noexcept]
```

Gets the component where the exception has been thrown.

Returns

const std::string& The component's name

4.34.1.2 what()

```
const char* Indie::Exceptions::IndieException::what ( ) const  [final], [noexcept]
```

Returns a C-style character string describing the general cause of the current error.

Returns

const char∗ The string

4.35 Indie::InitGame Struct Reference

Public Attributes

- GameScene::MODE **mode** = GameScene::MODE::SOLO
- int **nbAi** = 0
- Components::THEME **mapTheme** = Components::THEME::DIRT
- Components::MAP_TYPE **mapType** = Components::MAP_TYPE::DEFAULT
- bool **powerUp** = false
- std::string **mapPath** = ""
- time_t **timeLimit** = 0
- std::vector< PlayerParams > **playersParams** = {}

4.36 Indie::Components::InputComponent Class Reference

Hitbox component.

```
#include <InputComponent.hpp>
```

Public Member Functions

- **InputComponent** (std::unordered_map< irr::EKEY_CODE, KEY_TYPE > keys)
- bool isKeyPressed (KEY_TYPE key) const

  Checks if given key is pressed or not.
- const std::unordered_map< irr::EKEY_CODE, bool > & getKeysState (void) const

  Gets watched keys state.
- void setKeys (std::unordered_map< irr::EKEY_CODE, KEY_TYPE > newKeys)

  Sets keys to watch.
- void setKeysState (std::unordered_map< irr::EKEY_CODE, bool > newKeysState)

  Sets watched keys state.
- KEY_TYPE getKeyType (irr::EKEY_CODE key) const

  Gets KEY_TYPE which corresponds to its irr::EKEY_CODE.

4.36.1 Detailed Description

Hitbox component.

4.36.2 Member Function Documentation

4.36.2.1 getKeysState()

```
const std::unordered_map<irr::EKEY_CODE, bool>& Indie::Components::InputComponent::getKeys↩
State (
            void ) const
```

Gets watched keys state.

Returns

const std::unordered_map<irr::EKEY_CODE, bool>& List with the state of each watched key

4.36.2.2 getKeyType()

```
KEY_TYPE Indie::Components::InputComponent::getKeyType (
            irr::EKEY_CODE key ) const
```

Gets KEY_TYPE which corresponds to its irr::EKEY_CODE.

Parameters

key	The key code

Returns

KEY_TYPE The key type

4.36.2.3 isKeyPressed()

```
bool Indie::Components::InputComponent::isKeyPressed (
            KEY_TYPE key ) const
```

Checks if given key is pressed or not.

Parameters

key	The KEY_TYPE (see enum)

Returns

true The key is pressed
false The key is not pressed

4.36.2.4 setKeys()

```
void Indie::Components::InputComponent::setKeys (
             std::unordered_map< irr::EKEY_CODE, KEY_TYPE > newKeys )
```

Sets keys to watch.

Parameters

newKeys	Keys to watch

4.36.2.5 setKeysState()

```
void Indie::Components::InputComponent::setKeysState (
             std::unordered_map< irr::EKEY_CODE, bool > newKeysState )
```

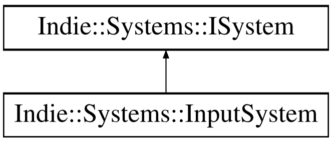
Sets watched keys state.

Parameters

newKeysState	Watched keys state

4.37 Indie::Systems::InputSystem Class Reference

Inheritance diagram for Indie::Systems::InputSystem:



Public Member Functions

- void onUpdate (irr::f32 deltaTime, EntityManager &entityManager) const final

  Updates things according to system type.

4.37.1 Member Function Documentation

4.37.1.1 onUpdate()

```
void Indie::Systems::InputSystem::onUpdate (
            irr::f32 deltaTime,
            EntityManager & entityManager ) const  [final], [virtual]
```

Updates things according to system type.

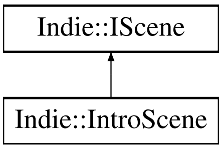
Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.
entityManager	The entity manager

Implements Indie::Systems::ISystem.

4.38 Indie::IntroScene Class Reference

Inheritance diagram for Indie::IntroScene:



Public Member Functions

- **IntroScene** (ContextManager &context)
- void init () final

  Inits the scene.
- void reset () final

  Resets the scene.
- void update (irr::f32 deltaTime) final

  Updates the scene.
- void renderPre3D () final

  Renders before 3D rendering.
- void renderPost3D () final

  Renders after 3D rendering.
- void **skipScene** (void)

4.38.1 Member Function Documentation

4.38.1.1 renderPost3D()

```
void Indie::IntroScene::renderPost3D ( )  [final], [virtual]
```

Renders after 3D rendering.

Implements Indie::IScene.

4.38.1.2 update()

```
void Indie::IntroScene::update (
            irr::f32 deltaTime )  [final], [virtual]
```

Updates the scene.

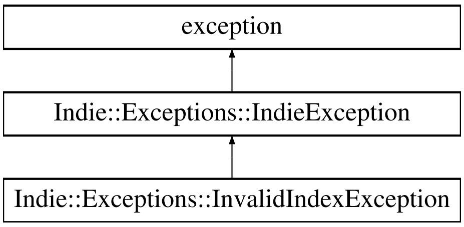
Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.

Implements Indie::IScene.

4.39 Indie::Exceptions::InvalidIndexException Class Reference

Inheritance diagram for Indie::Exceptions::InvalidIndexException:

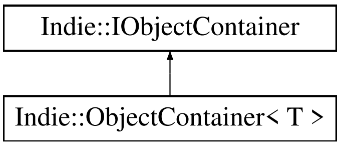

exception
Indie::Exceptions::IndieException
Indie::Exceptions::InvalidIndexException

Public Member Functions

- **InvalidIndexException** (const std::string &component, const std::string &message)

4.40 Indie::IObjectContainer Class Reference

Inheritance diagram for Indie::IObjectContainer:

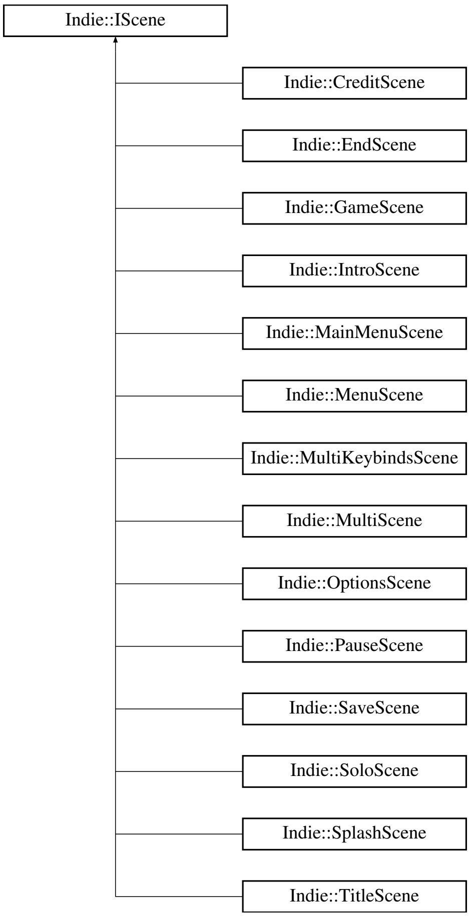


4.41 Indie::IScene Class Reference

IScene interface. All scenes must inherit this interface.

```
#include <IScene.hpp>
```

Inheritance diagram for Indie::IScene:



Public Member Functions

- virtual void init ()=0

  Inits the scene.
- virtual void reset ()=0

  Resets the scene.
- virtual void update (irr::f32 deltaTime)=0

  Updates the scene.
- virtual void renderPre3D ()=0

  Renders before 3D rendering.
- virtual void renderPost3D ()=0

  Renders after 3D rendering.

4.41.1 Detailed Description

IScene interface. All scenes must inherit this interface.

4.41.2 Member Function Documentation

4.41.2.1 renderPost3D()

```
virtual void Indie::IScene::renderPost3D ( )  [pure virtual]
```

Renders after 3D rendering.

Implemented in Indie::PauseScene, Indie::SoloScene, Indie::SplashScene, Indie::GameScene, Indie::MultiKeybindsScene, Indie::EndScene, Indie::MultiScene, Indie::CreditScene, Indie::MainMenuScene, Indie::MenuScene, Indie::SaveScene, Indie::TitleScene, Indie::OptionsScene, and Indie::IntroScene.

4.41.2.2 update()

```
virtual void Indie::IScene::update (
            irr::f32 deltaTime )  [pure virtual]
```

Updates the scene.

Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.

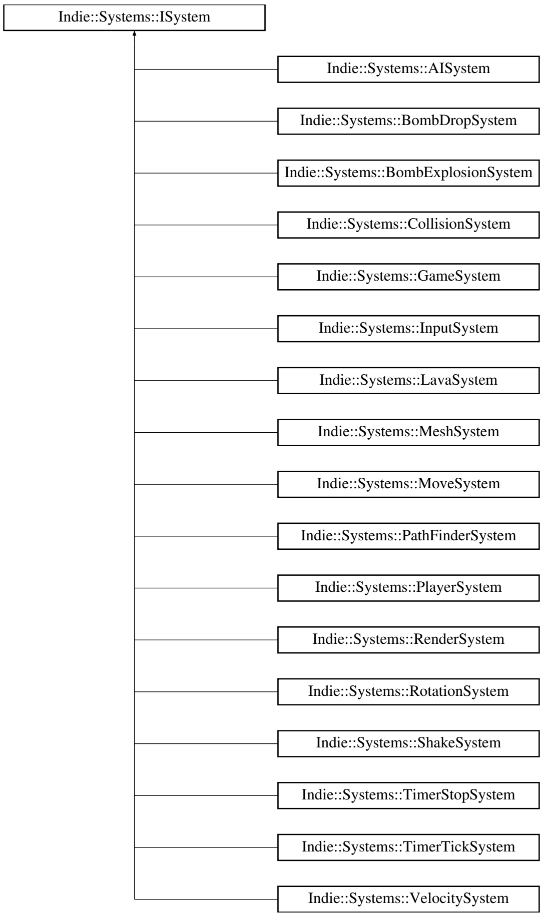
Implemented in Indie::PauseScene, Indie::SoloScene, Indie::SplashScene, Indie::GameScene, Indie::MultiKeybindsScene, Indie::EndScene, Indie::MultiScene, Indie::CreditScene, Indie::MainMenuScene, Indie::MenuScene, Indie::SaveScene, Indie::TitleScene, Indie::OptionsScene, and Indie::IntroScene.

4.42 Indie::Systems::ISystem Class Reference

ISystem interface. All systems must inherit this interface.

```
#include <ISystem.hpp>
```

Inheritance diagram for Indie::Systems::ISystem:


Indie::Systems::ISystem
Indie::Systems::AISystem
Indie::Systems::BombDropSystem
Indie::Systems::BombExplosionSystem
Indie::Systems::CollisionSystem
Indie::Systems::GameSystem
Indie::Systems::InputSystem
Indie::Systems::LavaSystem
Indie::Systems::MeshSystem
Indie::Systems::MoveSystem
Indie::Systems::PathFinderSystem
Indie::Systems::PlayerSystem
Indie::Systems::RenderSystem
Indie::Systems::RotationSystem
Indie::Systems::ShakeSystem
Indie::Systems::TimerStopSystem
Indie::Systems::TimerTickSystem
Indie::Systems::VelocitySystem

Public Member Functions

- virtual void onUpdate (irr::f32 deltaTime, EntityManager &entityManager) const =0

  Updates things according to system type.

4.42.1 Detailed Description

ISystem interface. All systems must inherit this interface.

4.42.2 Member Function Documentation

4.42.2.1 onUpdate()

```
virtual void Indie::Systems::ISystem::onUpdate (
            irr::f32 deltaTime,
            EntityManager & entityManager ) const  [pure virtual]
```

Updates things according to system type.

Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.
entityManager	The entity manager

Implemented in Indie::Systems::AISystem, Indie::Systems::GameSystem, Indie::Systems::PathFinderSystem, Indie::Systems::BombExplosionSystem, Indie::Systems::InputSystem, Indie::Systems::BombDropSystem, Indie::Systems::CollisionSystem, Indie::Systems::LavaSystem, Indie::Systems::MeshSystem, Indie::Systems::MoveSystem, Indie::Systems::RenderSystem, Indie::Systems::PlayerSystem, Indie::Systems::RotationSystem, Indie::Systems::ShakeSystem, Indie::Systems::VelocitySystem, Indie::Systems::TimerStopSystem, and Indie::Systems::TimerTickSystem.

4.43 Indie::Keybind Class Reference

Public Member Functions

- **Keybind** (ContextManager &context, irr::EKEY_CODE key)
- void **init** (std::string const &filepath, int posX, int posY, irr::core::position2d< irr::s32 > pos)
- void **update** (std::pair< int, int >)
- void **draw** ()
- bool **getStatus** () const
- irr::EKEY_CODE **getKey** () const
- void **setStatus** (bool status)
- void **setUsedKeys** (std::vector< std::pair< Indie::Components::KEY_TYPE, std::unique_ptr< Keybind >>> &list)
- void **setUsedKeys** (std::vector< std::pair< Indie::Components::KEY_TYPE, std::shared_ptr< Keybind >>> &list)

Public Attributes

- Image ∗ **tick**

Static Public Attributes

- static const std::vector< std::pair< irr::EKEY_CODE, irr::core::stringw > > **keyCodes**

4.44 Indie::Components::KillComponent Class Reference

Kill component.

```
#include <KillComponent.hpp>
```

Public Member Functions

- **KillComponent** (int ownerId=-1)
- int getOwnerId () const

  Gets the Id of the player who droped the bomb.

4.44.1 Detailed Description

Kill component.

4.44.2 Member Function Documentation

4.44.2.1 getOwnerId()

```
int Indie::Components::KillComponent::getOwnerId ( ) const
```

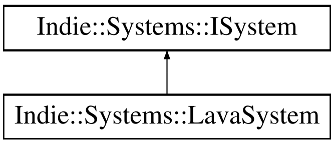
Gets the Id of the player who droped the bomb.

Returns

int The player's Id

4.45 Indie::Systems::LavaSystem Class Reference

Inheritance diagram for Indie::Systems::LavaSystem:



Public Member Functions

- void onUpdate (irr::f32 deltaTime, EntityManager &entityManager) const final

  Updates things according to system type.

4.45.1 Member Function Documentation

4.45.1.1 onUpdate()

```
void Indie::Systems::LavaSystem::onUpdate (
            irr::f32 deltaTime,
            EntityManager & entityManager ) const  [final], [virtual]
```

Updates things according to system type.

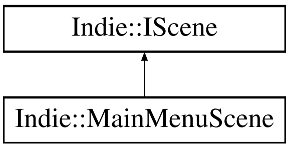
Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.
entityManager	The entity manager

Implements Indie::Systems::ISystem.

4.46 Indie::MainMenuScene Class Reference

Inheritance diagram for Indie::MainMenuScene:



Public Member Functions

- **MainMenuScene** (ContextManager &context)
- void init () final

  Inits the scene.
- void reset () final

  Resets the scene.
- void update (irr::f32 deltaTime) final

  Updates the scene.
- void renderPre3D () final

  Renders before 3D rendering.
- void renderPost3D () final

  Renders after 3D rendering.

4.46.1 Member Function Documentation

4.46.1.1 renderPost3D()

```
void Indie::MainMenuScene::renderPost3D ( )  [final], [virtual]
```

Renders after 3D rendering.

Implements Indie::IScene.

4.46.1.2 update()

```
void Indie::MainMenuScene::update (
            irr::f32 deltaTime )  [final], [virtual]
```

Updates the scene.

Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.

Implements Indie::IScene.

4.47 Indie::Components::MapComponent Class Reference

Map component.

```
#include <MapComponent.hpp>
```

Public Member Functions

- **MapComponent** (const irr::core::vector2di &dimension=irr::core::vector2di(15, 15), MAP_TYPE type=MAP_TYPE::DEFAULT, THEME theme=THEME::STONE, const std::string &mapPath="")
- const MAP_TYPE & getType () const

  Gets the map type.
- const THEME & getTheme () const

  Gets the map theme.
- const irr::core::vector2di & getDimension () const

  Gets the map's dimensions.
- const std::vector< std::vector< OBJECT > > & getMap () const

  Gets the map.
- const MAP_STATE & getMapState () const

  Gets the map state.
- const std::string & getMapPath (void) const

  Gets the map's path.
- void setMap (const std::vector< std::vector< OBJECT >> &map)

  Sets the map.
- void setMapState (MAP_STATE newState)

  Sets the map's state.

4.47.1 Detailed Description

Map component.

4.47.2 Member Function Documentation

4.47.2.1 getDimension()

```
const irr::core::vector2di& Indie::Components::MapComponent::getDimension ( ) const
```

Gets the map's dimensions.

Returns

const irr::core::vector2di& The map's dimensions

4.47.2.2 getMap()

```
const std::vector<std::vector<OBJECT> >& Indie::Components::MapComponent::getMap ( ) const
```

Gets the map.

Returns

const std::vector<std::vector<OBJECT>>& The map

4.47.2.3 getMapPath()

```
const std::string& Indie::Components::MapComponent::getMapPath (
             void  ) const
```

Gets the map's path.

Returns

const std::string& The map's path

4.47.2.4 getMapState()

```
const MAP_STATE& Indie::Components::MapComponent::getMapState ( ) const
```

Gets the map state.

Returns

const MAP_STATE& The map state

4.47.2.5 getTheme()

```
const THEME& Indie::Components::MapComponent::getTheme ( ) const
```

Gets the map theme.

Returns

const THEME& The map them

4.47.2.6 getType()

```
const MAP_TYPE& Indie::Components::MapComponent::getType ( ) const
```

Gets the map type.

Returns

const MAP_TYPE& The map type

4.47.2.7 setMap()

```
void Indie::Components::MapComponent::setMap (
            const std::vector< std::vector< OBJECT >> & map )
```

Sets the map.

Parameters

map	The map

4.47.2.8 setMapState()

```
void Indie::Components::MapComponent::setMapState (
            MAP_STATE newState )
```

Sets the map's state.

Parameters

newState	The state

4.48 Indie::MapGenerator Class Reference

Public Member Functions

- **MapGenerator** (EntityBuilder &entityBuilder, irr::core::vector2di vector, Components::MAP_TYPE type, Components::THEME theme, const std::string &mapPath)
- void generate (EntityManager &entityManager, EntityBuilder &entityBuilder)

  Generates a map which take in account parameters given in ctor.

4.48.1 Member Function Documentation

4.48.1.1 generate()

```
void Indie::MapGenerator::generate (
            EntityManager & entityManager,
            EntityBuilder & entityBuilder )
```

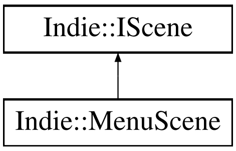
Generates a map which take in account parameters given in ctor.

Parameters

entityManager	The entity manager
entityBuilder	The entity builder

4.49 Indie::MenuScene Class Reference

Inheritance diagram for Indie::MenuScene:



Public Member Functions

- **MenuScene** (ContextManager &context)
- void init () final

  Inits the scene.
- void reset () final

  Resets the scene.
- void update (irr::f32 deltaTime) final

  Updates the scene.

- void renderPre3D () final

  Renders before 3D rendering.
- void renderPost3D () final

  Renders after 3D rendering.
- void **setColor** (irr::video::SColor color)
- void **saveHighScoreMap** (std::string mapPath, int score)
- const GameInfos & **getGameInfos** (void) const
- void **setGameInfos** (const GameInfos &gameInfos)

4.49.1 Member Function Documentation

4.49.1.1 renderPost3D()

```
void Indie::MenuScene::renderPost3D ( )  [final], [virtual]
```

Renders after 3D rendering.

Implements Indie::IScene.

4.49.1.2 update()

```
void Indie::MenuScene::update (
             irr::f32 deltaTime )  [final], [virtual]
```

Updates the scene.

Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.

Implements Indie::IScene.

4.50 Indie::Components::MeshComponent Class Reference

Mesh component.

```
#include <MeshComponent.hpp>
```

Public Types

- enum **MESH_STATE** { **STAND**, **RUN**, **DIE** }

Public Member Functions

- **MeshComponent** (const ContextManager &contextManager, const std::string &texturePath)
- irr::video::ITexture ∗ getTexture (void) const

  Gets the mesh's texture.
- const MESH_STATE & getCurrentMeshState (void) const

  Gets the mesh's state.
- void setCurrentMeshState (MESH_STATE newState)

  Sets the current mesh's state.
- irr::scene::IAnimatedMesh ∗ getMeshByState (MESH_STATE state) const

  Get a mesh according to specified state.

4.50.1 Detailed Description

Mesh component.

4.50.2 Member Function Documentation

4.50.2.1 getCurrentMeshState()

```
const MESH_STATE& Indie::Components::MeshComponent::getCurrentMeshState (
            void  ) const
```

Gets the mesh's state.

Returns

const MESH_STATE& The mesh's state

4.50.2.2 getMeshByState()

```
irr::scene::IAnimatedMesh* Indie::Components::MeshComponent::getMeshByState (
            MESH_STATE state ) const
```

Get a mesh according to specified state.

Parameters

state	The wanted state

Returns

irr::scene::IAnimatedMesh∗ The mesh

4.50.2.3 getTexture()

```
irr::video::ITexture* Indie::Components::MeshComponent::getTexture (
            void ) const
```

Gets the mesh's texture.

Returns

irr::video::ITexture∗ The texture

4.50.2.4 setCurrentMeshState()

```
void Indie::Components::MeshComponent::setCurrentMeshState (
            MESH_STATE newState )
```

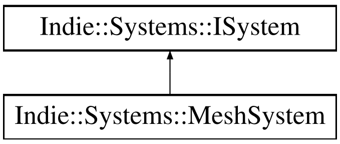
Sets the current mesh's state.

Parameters

newState	The new state

4.51 Indie::Systems::MeshSystem Class Reference

Inheritance diagram for Indie::Systems::MeshSystem:



Public Member Functions

- void onUpdate (irr::f32 deltaTime, EntityManager &entityManager) const final

  Updates things according to system type.

4.51.1 Member Function Documentation

4.51.1.1 onUpdate()

```
void Indie::Systems::MeshSystem::onUpdate (
            irr::f32 deltaTime,
            EntityManager & entityManager ) const  [final], [virtual]
```

Updates things according to system type.

Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.
entityManager	The entity manager

Implements Indie::Systems::ISystem.

4.52 Indie::Components::MoveComponent Class Reference

Move component.

`#include <MoveComponent.hpp>`

Public Member Functions

- void setUp (bool)

  Sets up state.
- void setDown (bool)

  Sets down state.
- void setLeft (bool)

  Sets left state.
- void setRight (bool)

  Sets right state.
- void setDrop (bool)

  Sets drip state.
- bool getUp (void) const

  Gets up state.
- bool getDown (void) const

  Get down state.
- bool getLeft (void) const

  Gets left state.
- bool getRight (void) const

  Gets right state.
- bool getDrop (void) const

  Gets drop state.

4.52.1 Detailed Description

Move component.

4.52.2 Member Function Documentation

4.52.2.1 getDown()

```
bool Indie::Components::MoveComponent::getDown (
            void  ) const
```

Get down state.

Returns

true Should move down
false Shouldn't move down

4.52.2.2 getDrop()

```
bool Indie::Components::MoveComponent::getDrop (
            void  ) const
```

Gets drop state.

Returns

true Should drop bomb
false Shouldn't drop bomb

4.52.2.3 getLeft()

```
bool Indie::Components::MoveComponent::getLeft (
            void  ) const
```

Gets left state.

Returns

true Should move left
false Shouldn't move left

4.52.2.4 getRight()

```
bool Indie::Components::MoveComponent::getRight (
            void  ) const
```

Gets right state.

Returns

true Should move right
false Shouldn't move right

4.52.2.5 getUp()

```
bool Indie::Components::MoveComponent::getUp (
            void  ) const
```

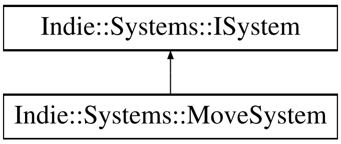
Gets up state.

Returns

true Should move up
false Shouldn't move up

4.53 Indie::Systems::MoveSystem Class Reference

Inheritance diagram for Indie::Systems::MoveSystem:



Public Member Functions

- int **getCenter** (int value) const
- void onUpdate (irr::f32 deltaTime, EntityManager &entityManager) const final

  Updates things according to system type.

4.53.1 Member Function Documentation

4.53.1.1 onUpdate()

```
void Indie::Systems::MoveSystem::onUpdate (
            irr::f32 deltaTime,
            EntityManager & entityManager ) const  [final], [virtual]
```

Updates things according to system type.

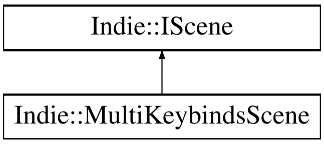
Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.
entityManager	The entity manager

Implements Indie::Systems::ISystem.

4.54 Indie::MultiKeybindsScene Class Reference

Inheritance diagram for Indie::MultiKeybindsScene:



Public Member Functions

- **MultiKeybindsScene** (ContextManager &context)
- void init () final

  Inits the scene.
- void reset () final

  Resets the scene.
- void update (irr::f32 deltaTime) final

  Updates the scene.
- void renderPre3D () final

  Renders before 3D rendering.
- void renderPost3D () final

  Renders after 3D rendering.
- void **setData** (InitGame ∗initGame)

4.54.1 Member Function Documentation

4.54.1.1 renderPost3D()

```
void Indie::MultiKeybindsScene::renderPost3D ( )  [final], [virtual]
```

Renders after 3D rendering.

Implements Indie::IScene.

4.54.1.2 update()

```
void Indie::MultiKeybindsScene::update (
             irr::f32 deltaTime )  [final], [virtual]
```

Updates the scene.

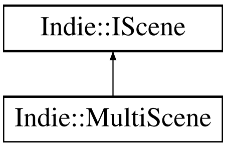
Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.

Implements Indie::IScene.

4.55 Indie::MultiScene Class Reference

Inheritance diagram for Indie::MultiScene:



Public Member Functions

- **MultiScene** (ContextManager &context)
- void init () final

  Inits the scene.
- void reset () final

  Resets the scene.
- void update (irr::f32 deltaTime) final

  Updates the scene.
- void renderPre3D () final

  Renders before 3D rendering.
- void renderPost3D () final

  Renders after 3D rendering.

4.55.1 Member Function Documentation

4.55.1.1 renderPost3D()

```
void Indie::MultiScene::renderPost3D ( )  [final], [virtual]
```

Renders after 3D rendering.

Implements Indie::IScene.

4.55.1.2 update()

```
void Indie::MultiScene::update (
            irr::f32 deltaTime )  [final], [virtual]
```

Updates the scene.

Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.

Implements Indie::IScene.

4.56 Indie::Music Class Reference

Public Types

- enum **Status** : int { **Intro**, **Loop**, **Outro**, **NotPlaying** }

Public Member Functions

- **Music** (std::string filepath)
- float **getVolume** () const
- void **setVolume** (float volume)
- void **loop** ()
- void **unLoop** ()
- void **mute** ()
- void **unMute** ()
- void **playMusic** ()
- void **pauseMusic** ()
- void **stopMusic** ()
- void **restartMusic** ()
- void **update** ()
- void **setStatus** (Status)
- Status **getStatus** ()

4.57 Indie::MusicManager Class Reference

MusicManager class.

```
#include <MusicManager.hpp>
```

Public Member Functions

- void addMusic (const std::string filepath)

  Adds a music.
- void setMusic (size_t id)

  Sets the current music.
- void setVolume (int vol)

  Sets the current volume.
- int getMusicVolume (void) const

  Gets the current volume.
- void mute ()

  Mutes the current music.
- void unMute ()

  Unmutes the current music.
- void playMusic ()

  Plays the current music.
- void pauseMusic ()

  Pauses the current music.
- void stopMusic ()

  Stops the current music.
- void restartMusic ()

  Restarts the current music.
- void update ()

  Updates the musics.
- Music::Status getStatus ()

  Gets the status of the current music.
- void setStatus (Music::Status)

  Sets the status of the current music.
- bool isMusicMuted () const

  Checks if the current music is muted or not.

4.57.1 Detailed Description

MusicManager class.

4.57.2 Member Function Documentation

4.57.2.1 addMusic()

```
void Indie::MusicManager::addMusic (
            const std::string filepath )
```

Adds a music.

Parameters

filepath	Path to the music

4.57.2.2 getMusicVolume()

```
int Indie::MusicManager::getMusicVolume (
            void  ) const
```

Gets the current volume.

Returns

float The current volume

4.57.2.3 getStatus()

```
Music::Status Indie::MusicManager::getStatus ( )
```

Gets the status of the current music.

Returns

Music::Status The status of the music

4.57.2.4 isMusicMuted()

```
bool Indie::MusicManager::isMusicMuted ( ) const
```

Checks if the current music is muted or not.

Returns

true The music is muted
false The music is not muted

4.57.2.5 pauseMusic()

```
void Indie::MusicManager::pauseMusic ( )
```

Pauses the current music.

4.57.2.6 restartMusic()

```
void Indie::MusicManager::restartMusic ( )
```

Restarts the current music.

4.57.2.7 setMusic()

```
void Indie::MusicManager::setMusic (
             size_t id )
```

Sets the current music.

Parameters

id	ID of the music to set

4.57.2.8 setVolume()

```
void Indie::MusicManager::setVolume (
             int vol )
```

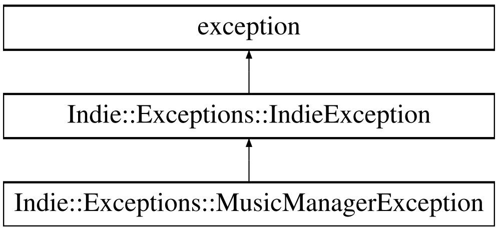
Sets the current volume.

Parameters

vol	The new volume

4.58 Indie::Exceptions::MusicManagerException Class Reference

Inheritance diagram for Indie::Exceptions::MusicManagerException:

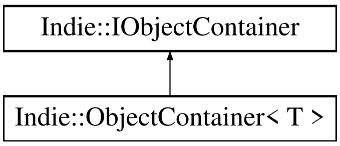

exception
Indie::Exceptions::IndieException
Indie::Exceptions::MusicManagerException

Public Member Functions

- **MusicManagerException** (const std::string &component, const std::string &message)

4.59 Indie::ObjectContainer< T > Class Template Reference

Inheritance diagram for Indie::ObjectContainer< T >:



Public Member Functions

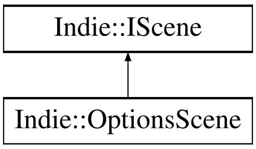
- template<typename... Args>
  ObjectContainer (Args &&... args)

Public Attributes

- T **data**

4.60 Indie::OptionsScene Class Reference

Inheritance diagram for Indie::OptionsScene:



Public Member Functions

- **OptionsScene** (ContextManager &context)
- void init () final

  Inits the scene.
- void reset () final

  Resets the scene.
- void update (irr::f32 deltaTime) final

  Updates the scene.
- void renderPre3D () final

  Renders before 3D rendering.
- void renderPost3D () final

  Renders after 3D rendering.

4.60.1 Member Function Documentation

4.60.1.1 renderPost3D()

```
void Indie::OptionsScene::renderPost3D ( )  [final], [virtual]
```

Renders after 3D rendering.

Implements Indie::IScene.

4.60.1.2 update()

```
void Indie::OptionsScene::update (
            irr::f32 deltaTime )  [final], [virtual]
```

Updates the scene.

Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.

Implements Indie::IScene.

4.61 Indie::Parallax Class Reference

Public Member Functions

- **Parallax** (ContextManager &context)
- void **init** (ContextManager &context, const std::string &filepath, irr::core::position2d< irr::s32 > size, irr::core::position2d< float > vel)
- void **update** (irr::f32 ticks)
- void **draw** ()
- void **setColor** (irr::video::SColor color=irr::video::SColor(255, 255, 255, 255))

4.62 Indie::Components::PathFinderComponent Class Reference

Path finder component.

```
#include <PathFinderComponent.hpp>
```

Public Member Functions

- void setMap (std::vector< std::vector< OBJECT >> map)

  Sets the path map of AI.
- void setMapBomb (std::vector< std::vector< OBJECT >> map)

  Sets the map with current explosions and future explosions with the range for each bomb.
- void setEndMapPos (irr::core::vector2di endPos)

  Sets the final position for the movement of AI.
- std::vector< std::vector< OBJECT > > getMap () const

  Gets the path map.
- std::vector< std::vector< OBJECT > > getMapBomb () const

  Gets the bombed map.
- irr::core::vector2di getEndMapPos () const

  Gets the final position of AI.

4.62.1 Detailed Description

Path finder component.

4.62.2 Member Function Documentation

4.62.2.1 getEndMapPos()

```
irr::core::vector2di Indie::Components::PathFinderComponent::getEndMapPos ( ) const
```

Gets the final position of AI.

Returns

The final position vector

4.62.2.2 getMap()

```
std::vector<std::vector<OBJECT> > Indie::Components::PathFinderComponent::getMap ( ) const
```

Gets the path map.

Returns

The path map

4.62.2.3 getMapBomb()

```
std::vector<std::vector<OBJECT> > Indie::Components::PathFinderComponent::getMapBomb ( )
const
```

Gets the bombed map.

Returns

The bombed map

4.62.2.4 setEndMapPos()

```
void Indie::Components::PathFinderComponent::setEndMapPos (
            irr::core::vector2di endPos )
```

Sets the final position for the movement of AI.

Parameters

endPos	The final position vector

4.62.2.5 setMap()

```
void Indie::Components::PathFinderComponent::setMap (
            std::vector< std::vector< OBJECT >> map )
```

Sets the path map of AI.

Parameters

map	The path map

4.62.2.6 setMapBomb()

```
void Indie::Components::PathFinderComponent::setMapBomb (
            std::vector< std::vector< OBJECT >> map )
```

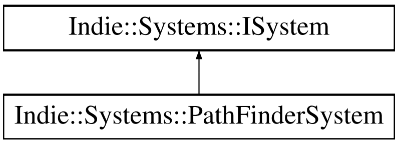
Sets the map with current explosions and future explosions with the range for each bomb.

Parameters

map	The bombed map

4.63 Indie::Systems::PathFinderSystem Class Reference

Inheritance diagram for Indie::Systems::PathFinderSystem:



Public Member Functions

- void onUpdate (irr::f32 deltaTime, EntityManager &entityManager) const final

  Updates things according to system type.

4.63.1 Member Function Documentation

4.63.1.1 onUpdate()

```
void Indie::Systems::PathFinderSystem::onUpdate (
            irr::f32 deltaTime,
            EntityManager & entityManager ) const  [final], [virtual]
```

Updates things according to system type.

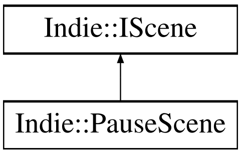
Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.
entityManager	The entity manager

Implements Indie::Systems::ISystem.

4.64 Indie::PauseScene Class Reference

Inheritance diagram for Indie::PauseScene:



Public Types

- enum **PAUSE_ASSETS** {
  CONTINUE, **MENU**, **QUIT**, **RESTART**,
  BOMBER, **TITLE**, **BG** }

Public Member Functions

- **PauseScene** (ContextManager &context)
- void init () final

  Inits the scene.
- void reset () final

  Resets the scene.
- void update (irr::f32 deltaTime) final

  Updates the scene.
- void renderPre3D () final

  Renders before 3D rendering.
- void renderPost3D () final

  Renders after 3D rendering.

4.64.1 Member Function Documentation

4.64.1.1 renderPost3D()

```
void Indie::PauseScene::renderPost3D ( )  [final], [virtual]
```

Renders after 3D rendering.

Implements Indie::IScene.

4.64.1.2 update()

```
void Indie::PauseScene::update (
            irr::f32 deltaTime )  [final], [virtual]
```

Updates the scene.

Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.

Implements Indie::IScene.

4.65 Indie::Components::PlayerComponent Class Reference

Player component.

```
#include <PlayerComponent.hpp>
```

Public Types

- enum **PLAYER_COLOR** {
  RED = 0, **GREEN** = 1, **BLUE** = 2, **YELLOW** = 3,
  PURPLE = 4, **GREY** = 5 }
- enum **PLAYER_START_POSITION** { **TOP_LEFT** = 0, **TOP_RIGHT** = 1, **DOWN_LEFT** = 2, **DOWN_RIGHT** = 3 }

Public Member Functions

- **PlayerComponent** (const std::string &name, PLAYER_COLOR playerColor, PLAYER_START_POSITION startPosition, bool isHuman)
- bool isHuman (void) const

  Used to check if entity is linked to a human player.
- unsigned int getBombsRange (void) const

  Gets bombs range.
- unsigned int getMaxBombNb (void) const

  Gets the maximum number of bombs that can be dropped simultaneously.
- unsigned int getCurrentBombNb (void) const

  Gets the current number of bombs that can be dropped.
- unsigned int getVelocityLevel (void) const

  Gets the velocity level of the player.
- unsigned int getXp (void) const

  Gets the XP count.
- int getScore (void) const

  Gets the score value.
- bool getWallPass (void) const

  Gets wall pass boolean value in order to check if the player took this power up.
- bool isDead (void) const

  Gets the player's state : dead or alive.
- const std::string & getName (void) const

  Gets the player's name.
- PLAYER_COLOR getPlayerColor (void) const

  Gets the player's color.
- PLAYER_START_POSITION getStartPosition (void) const

Gets the player's start position.

- void setBombsRange (unsigned int value)

  Sets bombs range.
- void setMaxBombNb (unsigned int value)

  Sets the maximum number of bombs that can be dropped simultaneously.
- void setCurrentBombNb (unsigned int value)

  Sets the current number of bombs that can be dropped.
- void setVelocityLevel (unsigned int value)

  Sets the velocity level.
- void setXpCount (unsigned int value)

  Sets the XP count.
- void setScore (int score)

  Sets the score value.
- void setWallPass (bool value)

  Sets wall pass boolean value.
- void setIsDead (bool value)

  Sets the player's state.

4.65.1 Detailed Description

Player component.

4.65.2 Member Function Documentation

4.65.2.1 getBombsRange()

```
unsigned int Indie::Components::PlayerComponent::getBombsRange (
            void  ) const
```

Gets bombs range.

Returns

unsigned int Bombs range

4.65.2.2 getCurrentBombNb()

```
unsigned int Indie::Components::PlayerComponent::getCurrentBombNb (
            void  ) const
```

Gets the current number of bombs that can be dropped.

Returns

unsigned int The number

4.65.2.3 getMaxBombNb()

```
unsigned int Indie::Components::PlayerComponent::getMaxBombNb (
            void ) const
```

Gets the maximum number of bombs that can be dropped simultaneously.

Returns

unsigned int The number

4.65.2.4 getName()

```
const std::string& Indie::Components::PlayerComponent::getName (
            void ) const
```

Gets the player's name.

Returns

const std::string& The name

4.65.2.5 getPlayerColor()

```
PLAYER_COLOR Indie::Components::PlayerComponent::getPlayerColor (
            void ) const
```

Gets the player's color.

Returns

PLAYER_COLOR The color

4.65.2.6 getScore()

```
int Indie::Components::PlayerComponent::getScore (
            void ) const
```

Gets the score value.

Returns

int The score value

4.65.2.7 getStartPosition()

```
PLAYER_START_POSITION Indie::Components::PlayerComponent::getStartPosition (
            void ) const
```

Gets the player's start position.

Returns

PLAYER_START_POSITION The start position

4.65.2.8 getVelocityLevel()

```
unsigned int Indie::Components::PlayerComponent::getVelocityLevel (
            void ) const
```

Gets the velocity level of the player.

Returns

unsigned int The level

4.65.2.9 getWallPass()

```
bool Indie::Components::PlayerComponent::getWallPass (
            void ) const
```

Gets wall pass boolean value in order to check if the player took this power up.

Returns

true The player took the power up
false The player didn't took the power up

4.65.2.10 getXp()

```
unsigned int Indie::Components::PlayerComponent::getXp (
            void ) const
```

Gets the XP count.

Returns

unsigned int The XP count

4.65.2.11 isDead()

```
bool Indie::Components::PlayerComponent::isDead (
            void ) const
```

Gets the player's state : dead or alive.

Returns

true The player is dead
false The player is alive

4.65.2.12 isHuman()

```
bool Indie::Components::PlayerComponent::isHuman (
            void ) const
```

Used to check if entity is linked to a human player.

Returns

true Player is a human player
false Player is not a human player

4.65.2.13 setBombsRange()

```
void Indie::Components::PlayerComponent::setBombsRange (
            unsigned int value )
```

Sets bombs range.

Parameters

value	The new range

4.65.2.14 setCurrentBombNb()

```
void Indie::Components::PlayerComponent::setCurrentBombNb (
            unsigned int value )
```

Sets the current number of bombs that can be dropped.

Parameters

value	The new number

4.65.2.15 setIsDead()

```
void Indie::Components::PlayerComponent::setIsDead (
            bool value )
```

Sets the player's state.

Parameters

value	The new state

4.65.2.16 setMaxBombNb()

```
void Indie::Components::PlayerComponent::setMaxBombNb (
            unsigned int value )
```

Sets the maximum number of bombs that can be dropped simultaneously.

Parameters

value	The new the maximum number of bombs that can be dropped simultaneously

4.65.2.17 setScore()

```
void Indie::Components::PlayerComponent::setScore (
            int score )
```

Sets the score value.

Parameters

score	The score value

4.65.2.18 setVelocityLevel()

```
void Indie::Components::PlayerComponent::setVelocityLevel (
            unsigned int value )
```

Sets the velocity level.

Parameters

value	The new leven

4.65.2.19 setWallPass()

```
void Indie::Components::PlayerComponent::setWallPass (
            bool value )
```

Sets wall pass boolean value.

Parameters

value	The new value

4.65.2.20 setXpCount()

```
void Indie::Components::PlayerComponent::setXpCount (
            unsigned int value )
```

Sets the XP count.

Parameters

value	The new XP count

4.66 Indie::PlayerMaps::PlayerMap Struct Reference

Public Attributes

- std::string **path**
- unsigned int **reqLvl**

4.67 Indie::PlayerMaps Class Reference

Classes

- struct PlayerMap

Static Public Attributes

- static const std::vector< PlayerMap > **mapPaths**

4.68 Indie::PlayerParams Struct Reference

Public Attributes

- std::string **playerTexture**
- std::unordered_map< irr::EKEY_CODE, Components::KEY_TYPE > **playerKeys**
- Components::PlayerComponent::PLAYER_COLOR **playerColor**

4.69 Indie::PlayerSkins::PlayerSkin Struct Reference

Public Attributes

- std::string **path**
- Components::PlayerComponent::PLAYER_COLOR **color**
- unsigned int **reqLvl**

4.70 Indie::PlayerSkins Class Reference

Classes

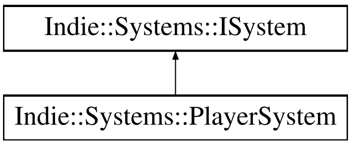
- struct PlayerSkin

Static Public Attributes

- static const std::vector< PlayerSkin > **skinPaths**

4.71 Indie::Systems::PlayerSystem Class Reference

Inheritance diagram for Indie::Systems::PlayerSystem:



Public Member Functions

- void onUpdate (irr::f32 deltaTime, EntityManager &entityManager) const final

  Updates things according to system type.

4.71.1 Member Function Documentation

4.71.1.1 onUpdate()

```
void Indie::Systems::PlayerSystem::onUpdate (
            irr::f32 deltaTime,
            EntityManager & entityManager ) const  [final], [virtual]
```

Updates things according to system type.

Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.
entityManager	The entity manager

Implements Indie::Systems::ISystem.

4.72 Indie::Components::PositionComponent Class Reference

Position component.

```
#include <PositionComponent.hpp>
```

Public Member Functions

- **PositionComponent** (float posX, float posY, float posZ)
- const irr::core::vector3df & getPosition (void) const

  Gets the stored position.
- void setPosition (irr::core::vector3df position)

  Sets the stored position.

4.72.1 Detailed Description

Position component.

4.72.2 Member Function Documentation

4.72.2.1 getPosition()

```
const irr::core::vector3df& Indie::Components::PositionComponent::getPosition (
             void  ) const
```

Gets the stored position.

Returns

const irr::core::vector3df& The position

4.72.2.2 setPosition()

```
void Indie::Components::PositionComponent::setPosition (
             irr::core::vector3df position )
```

Sets the stored position.

Parameters

position	The new position

4.73 Indie::Components::PowerDownComponent Class Reference

Power down component.

```
#include <PowerDownComponent.hpp>
```

Public Member Functions

- **PowerDownComponent** (POWERDOWN_TYPE type)
- POWERDOWN_TYPE getType (void) const

  Gets the type of the power down.

4.73.1 Detailed Description

Power down component.

4.73.2 Member Function Documentation

4.73.2.1 getType()

```
POWERDOWN_TYPE Indie::Components::PowerDownComponent::getType (
            void ) const
```

Gets the type of the power down.

Returns

POWERDOWN_TYPE The type

4.74 Indie::Components::PowerUpComponent Class Reference

Power up component.

```
#include <PowerUpComponent.hpp>
```

Public Member Functions

- **PowerUpComponent** (POWERUP_TYPE type)
- POWERUP_TYPE getType (void) const

  Gets the type of the power up.

4.74.1 Detailed Description

Power up component.

4.74.2 Member Function Documentation

4.74.2.1 getType()

```
POWERUP_TYPE Indie::Components::PowerUpComponent::getType (
            void ) const
```

Gets the type of the power up.

Returns

POWERUP_TYPE The type

4.75 Indie::Prompt Class Reference

Public Member Functions

- **Prompt** (ContextManager &context)
- void **init** (std::string const &filepath, int posX, int posY, irr::core::position2d< irr::s32 > pos)
- void **update** (std::pair< int, int > pos)
- void **draw** ()
- std::string **getText** (void) const
- void **setText** (std::string newText)

Static Public Attributes

- static const std::vector< std::pair< irr::EKEY_CODE, std::string > > **keyCodes**
- static const std::vector< std::pair< irr::EKEY_CODE, std::string > > **keyCodesMaj**

4.76 Indie::Components::RenderComponent Class Reference

Render component.

```
#include <RenderComponent.hpp>
```

Public Member Functions

- **RenderComponent** (const std::string &meshPath, const std::string &texturePath, const ContextManager &contextManager, irr::core::vector3df position, bool lightSensible)
- irr::scene::IAnimatedMeshSceneNode ∗ getMesh (void) const

  Gets the current mesh.
- irr::video::ITexture ∗ getTexture (void) const

  Gets the current texture.
- const std::string & getMeshPath (void) const

  Gets the current mesh path.
- const std::string & getTexturePath (void) const

  Gets the current texture path.
- void setMesh (irr::scene::IAnimatedMeshSceneNode ∗newMesh)

  Sets the current mesh.

4.76.1 Detailed Description

Render component.

4.76.2 Member Function Documentation

4.76.2.1 getMesh()

```
irr::scene::IAnimatedMeshSceneNode* Indie::Components::RenderComponent::getMesh (
            void ) const
```

Gets the current mesh.

Returns

irr::scene::IAnimatedMeshSceneNode* The mesh

4.76.2.2 getMeshPath()

```
const std::string& Indie::Components::RenderComponent::getMeshPath (
            void ) const
```

Gets the current mesh path.

Returns

const std::string& The mesh path

4.76.2.3 getTexture()

```
irr::video::ITexture* Indie::Components::RenderComponent::getTexture (
            void ) const
```

Gets the current texture.

Returns

irr::video::ITexture* The texture

4.76.2.4 getTexturePath()

```
const std::string& Indie::Components::RenderComponent::getTexturePath (
            void ) const
```

Gets the current texture path.

Returns

const std::string& the texture path

4.76.2.5 setMesh()

```
void Indie::Components::RenderComponent::setMesh (
            irr::scene::IAnimatedMeshSceneNode * newMesh )
```

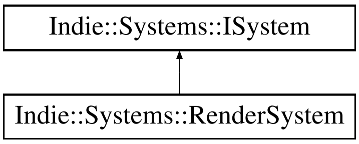
Sets the current mesh.

Parameters

newMesh	The new mesh

4.77 Indie::Systems::RenderSystem Class Reference

Inheritance diagram for Indie::Systems::RenderSystem:



Public Member Functions

- void onUpdate (irr::f32 deltaTime, EntityManager &entityManager) const final

  Updates things according to system type.

4.77.1 Member Function Documentation

4.77.1.1 onUpdate()

```
void Indie::Systems::RenderSystem::onUpdate (
            irr::f32 deltaTime,
            EntityManager & entityManager ) const  [final], [virtual]
```

Updates things according to system type.

Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.
entityManager	The entity manager

Implements Indie::Systems::ISystem.

4.78 Indie::Components::RotationComponent Class Reference

Rotation component.

```
#include <RotationComponent.hpp>
```

Public Member Functions

- **RotationComponent** (float angleY=180, float angleX=0)
- const irr::core::vector3df & getAngle (void) const

  Gets the current 3D rotation vector.
- void setYAngle (float newAngle)

  Sets the Y angle of the 3D rotation vector.

4.78.1 Detailed Description

Rotation component.

4.78.2 Member Function Documentation

4.78.2.1 getAngle()

```
const irr::core::vector3df& Indie::Components::RotationComponent::getAngle (
             void  ) const
```

Gets the current 3D rotation vector.

Returns

const irr::core::vector3df& The 3D rotation vector

4.78.2.2 setYAngle()

```
void Indie::Components::RotationComponent::setYAngle (
             float newAngle )
```

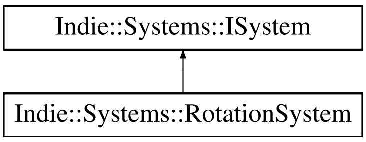
Sets the Y angle of the 3D rotation vector.

Parameters

newAngle	The new Y angle

4.79 Indie::Systems::RotationSystem Class Reference

Inheritance diagram for Indie::Systems::RotationSystem:



Public Member Functions

- void onUpdate (irr::f32 deltaTime, EntityManager &entityManager) const final

  Updates things according to system type.

4.79.1 Member Function Documentation

4.79.1.1 onUpdate()

```
void Indie::Systems::RotationSystem::onUpdate (
            irr::f32 deltaTime,
            EntityManager & entityManager ) const  [final], [virtual]
```

Updates things according to system type.

Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.
entityManager	The entity manager

Implements Indie::Systems::ISystem.

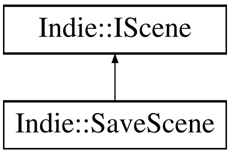
4.80 Indie::SaveManager Class Reference

Public Member Functions

- void **loadSave** (const std::string &filepath)
- void **saveCurrentSave** (void)
- void **loadMusicParams** (void)
- void **setCurrentSave** (const std::unordered_map< std::string, std::string > &newSave)
- void **saveValue** (const std::pair< std::string, std::string > &value)
- std::unordered_map< std::string, std::string > & **getCurrentSave** (void)
- std::vector< std::pair< std::string, time_t > > **getSavedGame** (void)
- void **resetCurentSave** ()

4.81 Indie::SaveScene Class Reference

Inheritance diagram for Indie::SaveScene:



Public Member Functions

- **SaveScene** (ContextManager &context)
- void init () final

  Inits the scene.
- void reset () final

  Resets the scene.
- void update (irr::f32 deltaTime) final

  Updates the scene.
- void renderPre3D () final

  Renders before 3D rendering.
- void renderPost3D () final

  Renders after 3D rendering.
- void **skipScene** (bool update, bool render, bool subUpdate, bool subRender)

4.81.1 Member Function Documentation

4.81.1.1 renderPost3D()

```
void Indie::SaveScene::renderPost3D ( )  [final], [virtual]
```

Renders after 3D rendering.

Implements Indie::IScene.

4.81.1.2 update()

```
void Indie::SaveScene::update (
            irr::f32 deltaTime )  [final], [virtual]
```

Updates the scene.

Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.

Implements Indie::IScene.

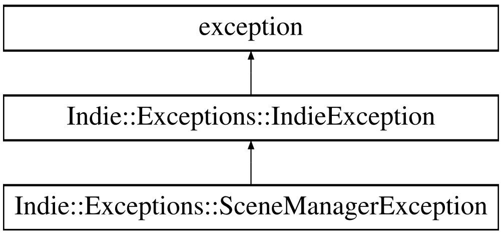
4.82 Indie::SceneManager Class Reference

Public Member Functions

- template<typename T >
  T ∗ **getScene** ()
- template<typename T >
  void **addScene** (ContextManager &context)
- template<typename T >
  void **restartScene** ()
- void **restartScenes** ()
- template<typename T >
  void **setScene** (ContextManager &context)
- template<typename T >
  void **setSubScene** ()
- void **setSceneUpdateActive** (bool status)
- void **setSubSceneUpdateActive** (bool status)
- void **setSceneRenderActive** (bool status)
- void **setSubSceneRenderActive** (bool status)
- void **update** (ContextManager &contextManager, irr::f32 deltaTime)
- void **initLoading** (ContextManager ∗context)

4.83 Indie::Exceptions::SceneManagerException Class Reference

Inheritance diagram for Indie::Exceptions::SceneManagerException:

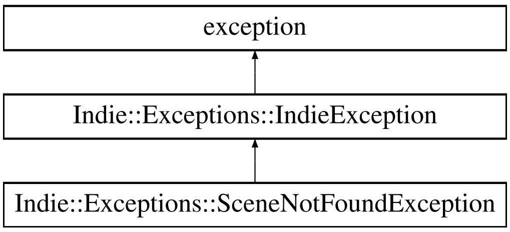

exception
Indie::Exceptions::IndieException
Indie::Exceptions::SceneManagerException

Public Member Functions

- **SceneManagerException** (const std::string &component, const std::string &message)

4.84 Indie::Exceptions::SceneNotFoundException Class Reference

Inheritance diagram for Indie::Exceptions::SceneNotFoundException:


exception
Indie::Exceptions::IndieException
Indie::Exceptions::SceneNotFoundException

Public Member Functions

- **SceneNotFoundException** (const std::string &component, const std::string &message)

4.85 Indie::ServiceLocator Class Reference

Public Member Functions

- **ServiceLocator** (const ServiceLocator &)=delete
- ServiceLocator & **operator=** (const ServiceLocator &)=delete
- template<typename T >

  T & **get** ()

Static Public Member Functions

- static ServiceLocator & **getInstance** ()

4.86 Indie::Components::ShakeComponent Class Reference

Shake component.

```
#include <ShakeComponent.hpp>
```

Public Member Functions

- bool getIsShaking (void) const

  Gets the isShaking boolean value.
- irr::f32 getDeltaTime (void) const

  Gets the delta time.
- const irr::core::vector3df & getInitialPosition (void) const

  Gets the initial camera's position.
- const irr::core::vector3df & getInitialTarget (void) const

  Gets the initial camera's target.
- void setIsShaking (bool value)

  Sets the isShaking boolean.
- void setDeltaTime (irr::f32 value)

  Sets the delta time.

4.86.1 Detailed Description

Shake component.

4.86.2 Member Function Documentation

4.86.2.1 getDeltaTime()

```
irr::f32 Indie::Components::ShakeComponent::getDeltaTime (
            void ) const
```

Gets the delta time.

Returns

irr::f32 The delta time

4.86.2.2 getInitialPosition()

```
const irr::core::vector3df& Indie::Components::ShakeComponent::getInitialPosition (
            void ) const
```

Gets the initial camera's position.

Returns

const irr::core::vector3df& The position

4.86.2.3 getInitialTarget()

```
const irr::core::vector3df& Indie::Components::ShakeComponent::getInitialTarget (
            void ) const
```

Gets the initial camera's target.

Returns

const irr::core::vector3df& The target

4.86.2.4 getIsShaking()

```
bool Indie::Components::ShakeComponent::getIsShaking (
            void  ) const
```

Gets the isShaking boolean value.

Returns

true The screen is shaking
false The screen is not shaking

4.86.2.5 setDeltaTime()

```
void Indie::Components::ShakeComponent::setDeltaTime (
            irr::f32 value )
```

Sets the delta time.

Parameters

value	The new delta time

4.86.2.6 setIsShaking()

```
void Indie::Components::ShakeComponent::setIsShaking (
            bool value )
```

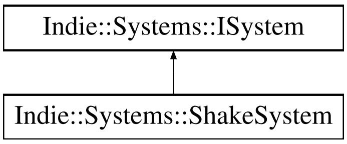
Sets the isShaking boolean.

Parameters

value	The new value

4.87 Indie::Systems::ShakeSystem Class Reference

Inheritance diagram for Indie::Systems::ShakeSystem:



Public Member Functions

- void onUpdate (irr::f32 deltaTime, EntityManager &entityManager) const final

  Updates things according to system type.

4.87.1 Member Function Documentation

4.87.1.1 onUpdate()

```
void Indie::Systems::ShakeSystem::onUpdate (
            irr::f32 deltaTime,
            EntityManager & entityManager ) const  [final], [virtual]
```

Updates things according to system type.

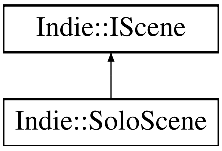
Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.
entityManager	The entity manager

Implements Indie::Systems::ISystem.

4.88 Indie::SoloScene Class Reference

Inheritance diagram for Indie::SoloScene:



Public Member Functions

- **SoloScene** (ContextManager &context)
- void init () final

  Inits the scene.
- void reset () final

  Resets the scene.
- void update (irr::f32 deltaTime) final

  Updates the scene.
- void renderPre3D () final

  Renders before 3D rendering.

- void renderPost3D () final

    Renders after 3D rendering.
- void **setKeybinds** (const std::vector< std::pair< Indie::Components::KEY_TYPE, std::unique_ptr< Keybind >>> &keybinds)
- const std::vector< std::pair< Indie::Components::KEY_TYPE, std::unique_ptr< Keybind > > > & **get↩ Keybinds** (void) const
- void **resetKeybinds** (void)
- std::string **getFileName** (std::string const &filepath)

4.88.1 Member Function Documentation

4.88.1.1 renderPost3D()

```
void Indie::SoloScene::renderPost3D ( )  [final], [virtual]
```

Renders after 3D rendering.

Implements Indie::IScene.

4.88.1.2 update()

```
void Indie::SoloScene::update (
            irr::f32 deltaTime )  [final], [virtual]
```

Updates the scene.

Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.

Implements Indie::IScene.

4.89 Indie::SoundManager Class Reference

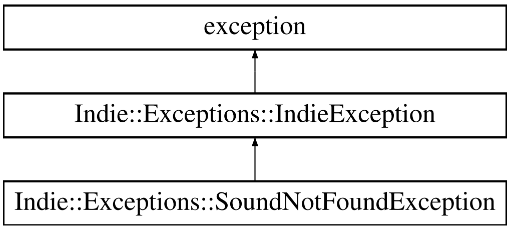
Public Member Functions

- void **update** ()
- void **setMute** (bool)
- void **setVolume** (int)
- bool **isMuted** () const
- int **getVolume** () const
- void **addSound** (const std::string &)

- void **playSound** (const std::string &)
- void **playSound** (const std::string &, float, float, float)
- void **playPitchedSound** (const std::string &)
- void **playPitchedSound** (const std::string &, float, float, float)

4.90 Indie::Exceptions::SoundNotFoundException Class Reference

Inheritance diagram for Indie::Exceptions::SoundNotFoundException:

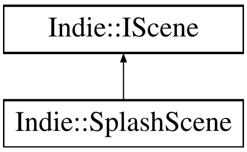

exception
Indie::Exceptions::IndieException
Indie::Exceptions::SoundNotFoundException

Public Member Functions

- **SoundNotFoundException** (const std::string &component, const std::string &message)

4.91 Indie::SplashScene Class Reference

Inheritance diagram for Indie::SplashScene:



Public Types

- enum **SPLASH_ASSETS** : int {
BG, **LOGO**, **PROJECT**, **BOMBER**,
END }

Public Member Functions

- **SplashScene** (ContextManager &context)
- void init () final

  Inits the scene.
- void reset () final

  Resets the scene.
- void update (irr::f32 deltaTime) final

  Updates the scene.
- void renderPre3D () final

  Renders before 3D rendering.
- void renderPost3D () final

  Renders after 3D rendering.

4.91.1 Member Function Documentation

4.91.1.1 renderPost3D()

```
void Indie::SplashScene::renderPost3D ( )  [final], [virtual]
```

Renders after 3D rendering.

Implements Indie::IScene.

4.91.1.2 update()

```
void Indie::SplashScene::update (
            irr::f32 deltaTime )  [final], [virtual]
```

Updates the scene.

Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.

Implements Indie::IScene.

4.92 Indie::SystemManager Class Reference

Public Member Functions

- template<typename T >
  T ∗ **getSystem** ()
- template<typename T , typename... Args>
  void **addSystem** (Args &&... args)
- SystemManager & **operator=** (const SystemManager &)=delete
- **SystemManager** (SystemManager const &)=delete

Static Public Member Functions

- static SystemManager & **getInstance** ()

4.93 Indie::Components::TimerComponent Class Reference

Timer component.

```
#include <TimerComponent.hpp>
```

Public Member Functions

- **TimerComponent** (irr::f32 timeToEnd)
- irr::f32 getTimePassed (void) const

  Gets the passed time.
- irr::f32 getTimeToEnd (void) const

  Gets the time when the timer should flag as being ended.
- void setTimePassed (irr::f32 timePassed)

  Sets the time passed.
- void setTimeToEnd (irr::f32 timeToEnd)

  Sets the time to end.

4.93.1 Detailed Description

Timer component.

4.93.2 Member Function Documentation

4.93.2.1 getTimePassed()

```
irr::f32 Indie::Components::TimerComponent::getTimePassed (
             void  ) const
```

Gets the passed time.

Returns

irr::f32 The passed time

4.93.2.2 getTimeToEnd()

```
irr::f32 Indie::Components::TimerComponent::getTimeToEnd (
             void  ) const
```

Gets the time when the timer should flag as being ended.

Returns

irr::f32

4.93.2.3 setTimePassed()

```
void Indie::Components::TimerComponent::setTimePassed (
             irr::f32 timePassed )
```

Sets the time passed.

Parameters

timePassed	The new value

4.93.2.4 setTimeToEnd()

```
void Indie::Components::TimerComponent::setTimeToEnd (
            irr::f32 timeToEnd )
```

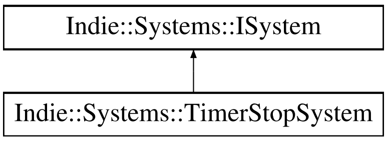
Sets the time to end.

Parameters

timeToEnd	The new value

4.94 Indie::Systems::TimerStopSystem Class Reference

Inheritance diagram for Indie::Systems::TimerStopSystem:



Public Member Functions

- void onUpdate (irr::f32 deltaTime, EntityManager &entityManager) const final

    Updates things according to system type.

4.94.1 Member Function Documentation

4.94.1.1 onUpdate()

```
void Indie::Systems::TimerStopSystem::onUpdate (
            irr::f32 deltaTime,
            EntityManager & entityManager ) const  [final], [virtual]
```

Updates things according to system type.

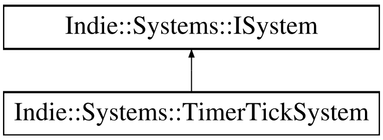
Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.
entityManager	The entity manager

Implements Indie::Systems::ISystem.

4.95 Indie::Systems::TimerTickSystem Class Reference

Inheritance diagram for Indie::Systems::TimerTickSystem:



Public Member Functions

- void onUpdate (irr::f32 deltaTime, EntityManager &entityManager) const final

    Updates things according to system type.

4.95.1 Member Function Documentation

4.95.1.1 onUpdate()

```
void Indie::Systems::TimerTickSystem::onUpdate (
            irr::f32 deltaTime,
            EntityManager & entityManager ) const  [final], [virtual]
```

Updates things according to system type.

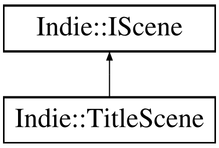
Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.
entityManager	The entity manager

Implements Indie::Systems::ISystem.

4.96 Indie::TitleScene Class Reference

Inheritance diagram for Indie::TitleScene:



Public Member Functions

- **TitleScene** (ContextManager &context)
- void init () final

  Inits the scene.
- void reset () final

  Resets the scene.
- void update (irr::f32 deltaTime) final

  Updates the scene.
- void renderPre3D () final

  Renders before 3D rendering.
- void renderPost3D () final

  Renders after 3D rendering.

4.96.1 Member Function Documentation

4.96.1.1 renderPost3D()

```
void Indie::TitleScene::renderPost3D ( )  [final], [virtual]
```

Renders after 3D rendering.

Implements Indie::IScene.

4.96.1.2 update()

```
void Indie::TitleScene::update (
            irr::f32 deltaTime )  [final], [virtual]
```

Updates the scene.

Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.

Implements Indie::IScene.

4.97 Indie::UiSelector Class Reference

Public Member Functions

- **UiSelector** (int sizeX, int sizeY, irr::EKEY_CODE up, irr::EKEY_CODE down, irr::EKEY_CODE left, irr::E↩ KEY_CODE right, bool x=true, bool y=true)
- void **setBLockSound** (bool, bool)
- std::pair< int, int > **getPos** () const
- void **setPos** (int posX, int posY)
- void **setSize** (int sizeX, int sizeY)
- void **update** ()

4.98 Indie::Components::VelocityComponent Class Reference

Velocity component.

```
#include <VelocityComponent.hpp>
```

Public Member Functions

- float getVelocity (void) const

  Gets the velocity.
- void setVelocity (float newVel)

  Sets the velocity.

4.98.1 Detailed Description

Velocity component.

4.98.2 Member Function Documentation

4.98.2.1 getVelocity()

```
float Indie::Components::VelocityComponent::getVelocity (
            void ) const
```

Gets the velocity.

Returns

float The velocity

4.98.2.2 setVelocity()

```
void Indie::Components::VelocityComponent::setVelocity (
            float newVel )
```

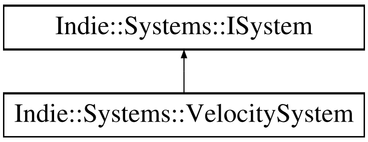
Sets the velocity.

Parameters

newVel	The new velocity

4.99 Indie::Systems::VelocitySystem Class Reference

Inheritance diagram for Indie::Systems::VelocitySystem:



Public Member Functions

- void onUpdate (irr::f32 deltaTime, EntityManager &entityManager) const final
  Updates things according to system type.

4.99.1 Member Function Documentation

4.99.1.1 onUpdate()

```
void Indie::Systems::VelocitySystem::onUpdate (
            irr::f32 deltaTime,
            EntityManager & entityManager ) const [final], [virtual]
```

Updates things according to system type.

Parameters

deltaTime	The time elapsed since the last time this method has been called. It's very useful for framerate independence.
entityManager	The entity manager

Implements Indie::Systems::ISystem.

4.100 Indie::Components::WallComponent Class Reference

Wall component.

```
#include <WallComponent.hpp>
```

Public Member Functions

- **WallComponent** (bool canBeDestroyed)
- bool getCanBeDestroyed () const

    Gets the canBeDestroyed value which is necessary for bomb explosion & wall pass power up.

4.100.1 Detailed Description

Wall component.

4.100.2 Member Function Documentation

4.100.2.1 getCanBeDestroyed()

```
bool Indie::Components::WallComponent::getCanBeDestroyed ( ) const
```

Gets the canBeDestroyed value which is necessary for bomb explosion & wall pass power up.

Returns

true The wall can be destroyed
false The wall is solid

Index