War Theatre Simulator

Chapter 1

War Theatre Simulator

This application allows to simulate operational level of military combats on a hexagonal grid.

Chapter 2

Namespace Index

2.1 Namespace List

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

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

Chapter 4

Class Index

4.1 Class List

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

Chapter 5

File Index

5.1 File List

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

Chapter 6

Namespace Documentation

6.1 baseAttack Namespace Reference

namespace with constants of X_Y attack. X_Y means that X attacks Y with this basic damage

Variables

- constexpr int **ERROR_PAIR** = -1

  base attack when type of units are wrong
- constexpr int **INFRANTRY_TO_INFANTRY** = 60
- constexpr int **INFANTRY_TO_ARTILLERY** = 100
- constexpr int **ARTILLERY_TO_ARTILLERY** = 100
- constexpr int **ARTILLERY_TO_INFANTRY** = 100

6.1.1 Detailed Description

namespace with constants of X_Y attack. X_Y means that X attacks Y with this basic damage

6.2 Terrain Namespace Reference

namespace consisting of terrain types IDs

Variables

- const TerrainType **NONE** = 0
- const TerrainType **FOREST** = 1
- const TerrainType **URBAN** = 2
- const TerrainType **PLAIN** = 3

6.2.1 Detailed Description

namespace consisting of terrain types IDs

6.3 UnitCategory Namespace Reference

namespace with unit types category IDs

Variables

- constexpr UnitType **NONE** = 0
- constexpr UnitType **INFANTRY** = 1
- constexpr UnitType **TANK** = 2
- constexpr UnitType **ARTILLERY** = 3

6.3.1 Detailed Description

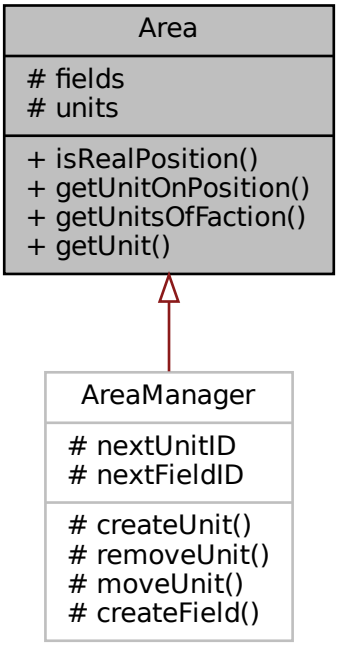
namespace with unit types category IDs

Chapter 7

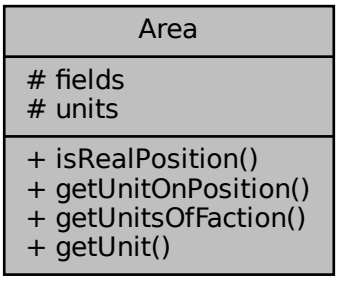
Class Documentation

7.1 Area Class Reference

Inheritance diagram for Area:



Collaboration diagram for Area:



Public Member Functions

- bool isRealPosition (const Position &position)

  Check if given position exists in the area.
- Unit & getUnitOnPosition (const Position &position)

  get unit on position
- std::vector< UnitID > getUnitsOfFaction (const UnitFactionID &unitFactionID) const

  get units of faction
- const Unit & getUnit (const UnitID &unitID)

  get unit of certain ID

Protected Attributes

- std::map< FieldID, Field > **fields**

  Array with fields of the area.
- std::map< UnitID, Unit > **units**

  Units present in arrea.

7.1.1 Member Function Documentation

7.1.1.1 getUnit()

```
const Unit & Area::getUnit (
            const UnitID & unitID )
```

get unit of certain ID

Parameters

unitID	ID of wanted unit

Returns

constant pointer to unit or NULL if not found

7.1.1.2 getUnitOnPosition()

```
Unit & Area::getUnitOnPosition (
            const Position & position )
```

get unit on position

Parameters

position	position to search unit on

Returns

pointer to unit on a given position. units.end() is returned if none is available

7.1.1.3 getUnitsOfFaction()

```
std::vector< UnitID > Area::getUnitsOfFaction (
            const UnitFactionID & unitFactionID ) const
```

get units of faction

Parameters

unitFactionID	faction of unit

Returns

vector of units that belong to faction

7.1.1.4 isRealPosition()

```
bool Area::isRealPosition (
            const Position & position )
```

Check if given position exists in the area.

Parameters

position	checked position

Returns

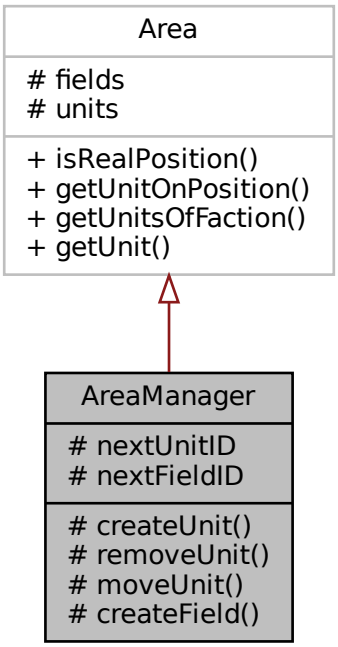
true if field of given position exists

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

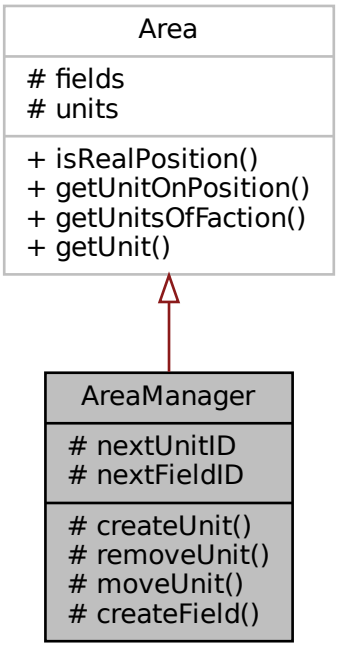
- include/area.h

7.2 AreaManager Class Reference

Inheritance diagram for AreaManager:



Collaboration diagram for AreaManager:



Protected Member Functions

- void createUnit (const Position &position, const Unit &unit)

  Spawn unit on a given position.
- void removeUnit (const UnitID &unitID)

  Remove unit from area.
- void moveUnit (const UnitID &UnitID, const Position &position)

  Move unit to position.
- void createField (const FieldType &fieldType, const Position &position)

  Create new field of given type and position.

Protected Attributes

- UnitID **nextUnitID** = 1

  next unique identifier of a unit. Should increase when unit is created. Should not ever be decreased
- FieldID **nextFieldID** = 1

  next unique identifier of a field. Should increase for each new field. Should not ever be decreased

7.2.1 Member Function Documentation

7.2.1.1 createField()

```
void AreaManager::createField (
            const FieldType & fieldType,
            const Position & position )  [protected]
```

Create new field of given type and position.

Parameters

fieldType	type of terrain of the field
position	position of the new field

7.2.1.2 createUnit()

```
void AreaManager::createUnit (
            const Position & position,
            const Unit & unit )  [protected]
```

Spawn unit on a given position.

Parameters

position	Position of a unit
unit	- unit to be spawned

7.2.1.3 moveUnit()

```
void AreaManager::moveUnit (
            const UnitID & UnitID,
            const Position & position )  [protected]
```

Move unit to position.

Parameters

UnitID	ID of unit to be moved
position	position to be reached by unit

7.2.1.4 removeUnit()

```
void AreaManager::removeUnit (
            const UnitID & unitID )  [protected]
```

Remove unit from area.

Parameters

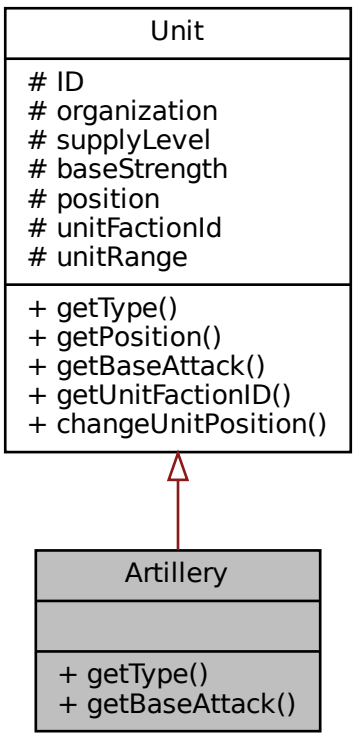
unitID	ID of removed unit

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

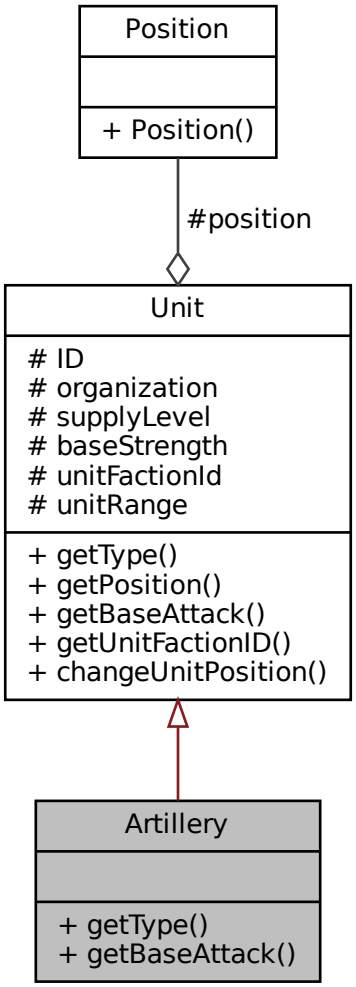
- include/areamanager.h

7.3 Artillery Class Reference

Inheritance diagram for Artillery:



Collaboration diagram for Artillery:



Public Member Functions

- const UnitType getType ()
- const int getBaseAttack (const UnitID &unitType)

  get Base attack of artillery

7.3.1 Member Function Documentation

7.3.1.1 getBaseAttack()

```
const int Artillery::getBaseAttack (
            const UnitID & unitType )  [virtual]
```

get Base attack of artillery

Parameters

unitType	type of enemy unit

Returns

base attack for artillery when dealing with certain enemy

Reimplemented from Unit.

7.3.1.2 getType()

```
const UnitType Artillery::getType ( )  [inline], [virtual]
```

Returns

UnitCategory::ARTILLERY

Reimplemented from Unit.

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

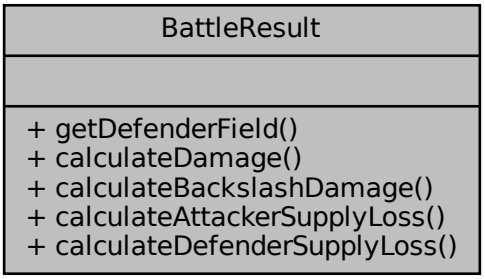
- include/units.h

7.4 BattleResult Class Reference

Class containing information regarding battle results.

```
#include <battle.h>
```

Collaboration diagram for BattleResult:



Public Member Functions

- const FieldID getDefenderField ()

  get field of defender
- int calculateDamage ()

  Calculate damage dealt to defender. Damage is equivalent to organization loss of defender.
- int calculateBackslashDamage ()

  Calculate damage dealt to attacker.
- int calculateAttackerSupplyLoss ()

  Calculate attacker loss in equipment.
- int calculateDefenderSupplyLoss ()

  Calculate defender loss in equipment.

7.4.1 Detailed Description

Class containing information regarding battle results.

7.4.2 Member Function Documentation

7.4.2.1 calculateAttackerSupplyLoss()

```
int BattleResult::calculateAttackerSupplyLoss ( )
```

Calculate attacker loss in equipment.

Returns

loss of supply level of attacker

7.4.2.2 calculateBackslashDamage()

```
int BattleResult::calculateBackslashDamage ( )
```

Calculate damage dealt to attacker.

Returns

organization loss of attacker

7.4.2.3 calculateDamage()

```
int BattleResult::calculateDamage ( )
```

Calculate damage dealt to defender. Damage is equivalent to organization loss of defender.

Returns

organization loss of defender

7.4.2.4 calculateDefenderSupplyLoss()

```
int BattleResult::calculateDefenderSupplyLoss ( )
```

Calculate defender loss in equipment.

Returns

loss of supply level of defender

7.4.2.5 getDefenderField()

```
const FieldID BattleResult::getDefenderField ( )
```

get field of defender

Returns

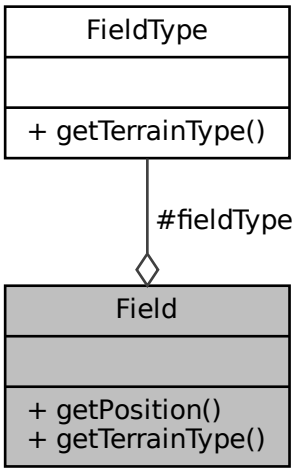
id of defended field

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

- include/battle.h

7.5 Field Class Reference

Collaboration diagram for Field:



Public Member Functions

- const Position getPosition ()

  get position of a field
- virtual const FieldType getTerrainType ()

Protected Attributes

- FieldType **fieldType**

7.5.1 Member Function Documentation

7.5.1.1 getPosition()

```
const Position Field::getPosition ( )
```

get position of a field

Returns

position as (q,r,s)

7.5.1.2 getTerrainType()

```
virtual const FieldType Field::getTerrainType ( )  [virtual]
```

Returns

type of terrain of the field

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

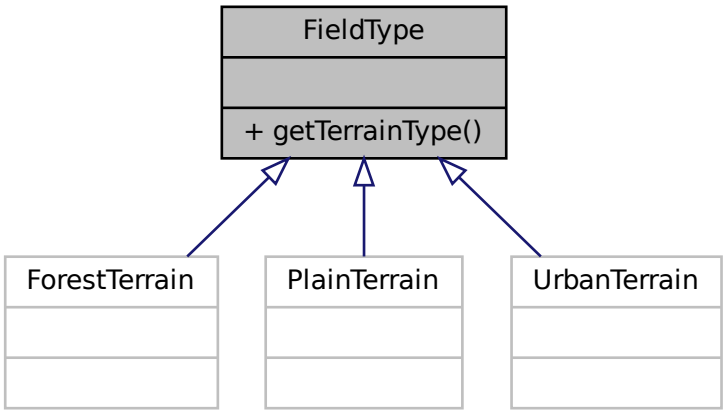
- include/field.h

7.6 FieldType Class Reference

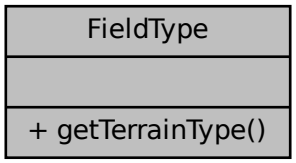
All fields will give certain punishments for movement and attack thus need to be distinguished.

`#include <field.h>`

Inheritance diagram for FieldType:



Collaboration diagram for FieldType:



Public Member Functions

- virtual const TerrainType getTerrainType ()

  get ID of terrain type of this field

7.6.1 Detailed Description

All fields will give certain punishments for movement and attack thus need to be distinguished.

7.6.2 Member Function Documentation

7.6.2.1 getTerrainType()

```
virtual const TerrainType FieldType::getTerrainType ( )  [virtual]
```

get ID of terrain type of this field

Returns

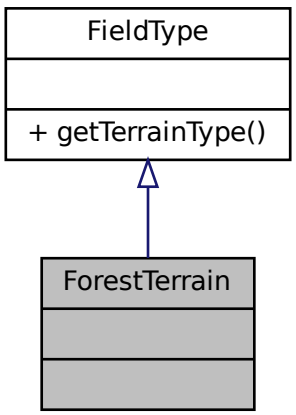
ID of terrain type for this field

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

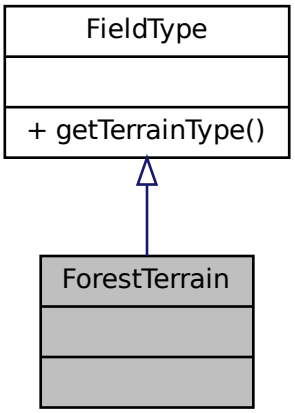
- include/field.h

7.7 ForestTerrain Class Reference

Inheritance diagram for ForestTerrain:



Collaboration diagram for ForestTerrain:



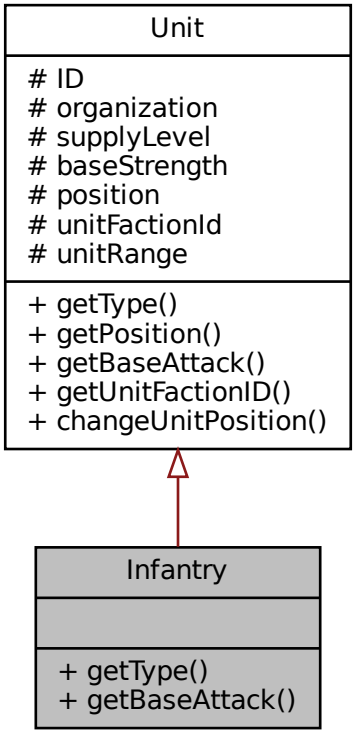
Additional Inherited Members

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

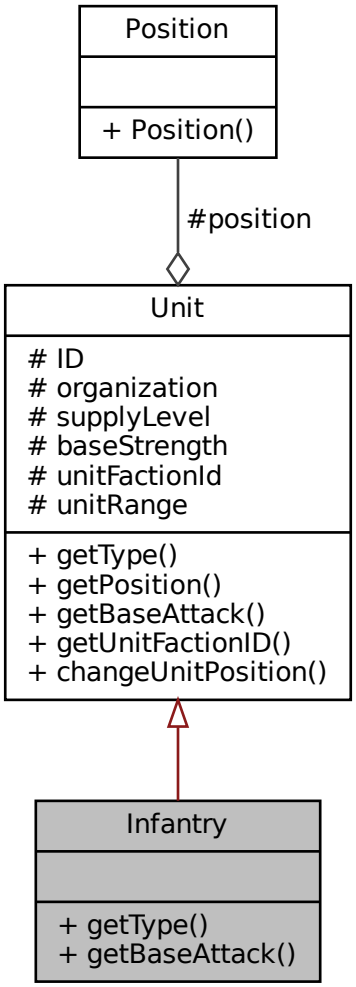
- include/field.h

7.8 Infantry Class Reference

Inheritance diagram for Infantry:



Collaboration diagram for Infantry:



Public Member Functions

- const UnitType getType ()
- const int getBaseAttack (const UnitID &unitType)

  get Base attack of infantry

7.8.1 Member Function Documentation

7.8.1.1 getBaseAttack()

```
const int Infantry::getBaseAttack (
            const UnitID & unitType )  [virtual]
```

get Base attack of infantry

Parameters

unitType	type of enemy unit

Returns

base attack for infantry when dealing with certain enemy

Reimplemented from Unit.

7.8.1.2 getType()

```
const UnitType Infantry::getType ( )  [virtual]
```

Returns

UnitCategory::INFANTRY

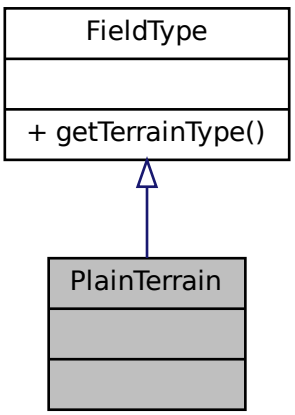
Reimplemented from Unit.

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

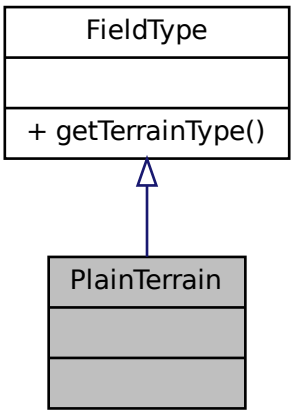
- include/units.h

7.9 PlainTerrain Class Reference

Inheritance diagram for PlainTerrain:



Collaboration diagram for PlainTerrain:



Additional Inherited Members

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

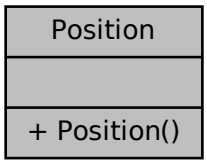
- include/field.h

7.10 Position Class Reference

Class to store position of a field.

`#include <field.h>`

Collaboration diagram for Position:



Public Member Functions

- **Position** (const int &_q, const int &_r)

7.10.1 Detailed Description

Class to store position of a field.

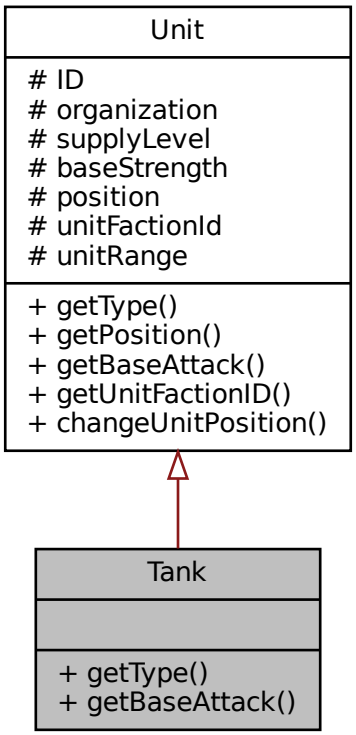
Coordinate systems which is going to be used for this hexagonal simulation is q, r for two axes and s for the third (however, s is "artificial" since s = -q - r)

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

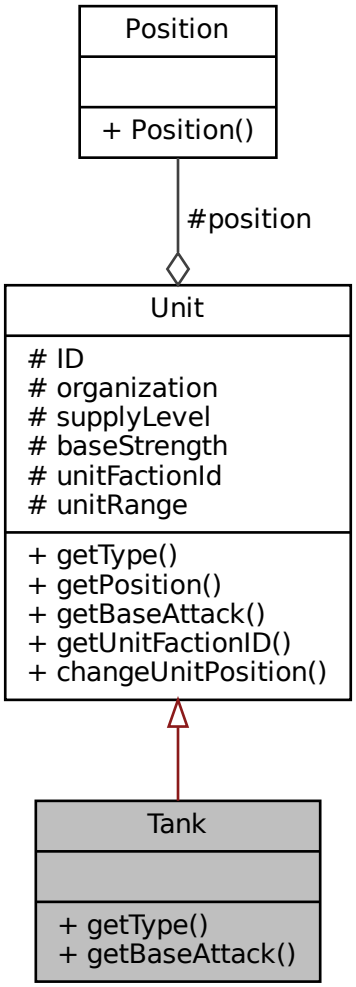
- include/field.h

7.11 Tank Class Reference

Inheritance diagram for Tank:



Collaboration diagram for Tank:



Public Member Functions

- const UnitType getType ()
- const int getBaseAttack (const UnitID &unitType)

  get Base attack of tank

7.11.1 Member Function Documentation

7.11.1.1 getBaseAttack()

```
const int Tank::getBaseAttack (
            const UnitID & unitType )  [virtual]
```

get Base attack of tank

Parameters

unitType	type of enemy unit

Returns

base attack for tank when dealing with certain enemy

Reimplemented from Unit.

7.11.1.2 getType()

```
const UnitType Tank::getType ( )  [inline], [virtual]
```

Returns

UnitCategory::TANK

Reimplemented from Unit.

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

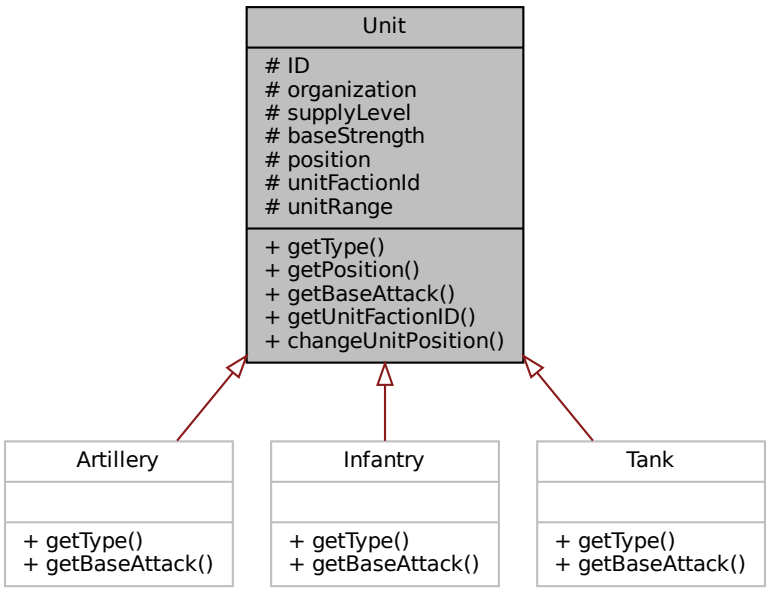
- include/units.h

7.12 Unit Class Reference

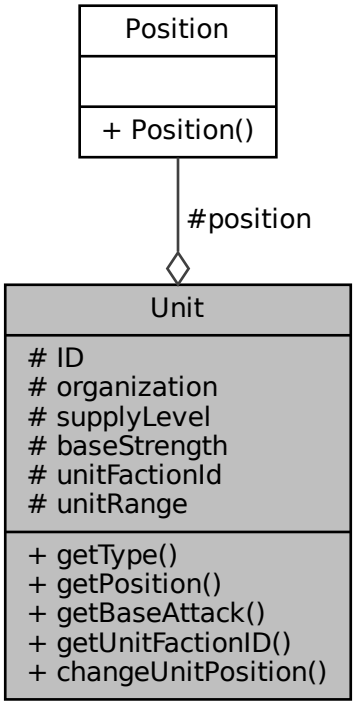
Basic class for all unit types.

`#include <units.h>`

Inheritance diagram for Unit:



Collaboration diagram for Unit:



Public Member Functions

- virtual const UnitType getType ()

  get type ID of a unit
- const Position getPosition () const
- virtual const int getBaseAttack (const UnitType &unitType)

  Get base attack, a value that might be unique for every pair (UNIT_TYPE, DEFENDER_UNIT_TYPE)
- virtual const UnitFactionID getUnitFactionID () const

  get ID of unit faction
- void **changeUnitPosition** (const Position &position)

  change position of a unit

Protected Attributes

- UnitID **ID**

  individual identifier for a unit
- int organization

  organization stat of a unit.
- int supplyLevel

  Level of logistic supply of a unit.
- int **baseStrength**

basic strength of a unit (it is determined only by type of unit)

- Position **position**

  position of a unit

- UnitFactionID **unitFactionId**

  ID of faction that owns the unit`.

- int **unitRange**

  range of an attack of a unit unitRange = X means that every enemy unit in (manhattan distance) X of our unit can be attacked

7.12.1 Detailed Description

Basic class for all unit types.

7.12.2 Member Function Documentation

7.12.2.1 getBaseAttack()

```
virtual const int Unit::getBaseAttack (
            const UnitType & unitType )  [virtual]
```

Get base attack, a value that might be unique for every pair (UNIT_TYPE, DEFENDER_UNIT_TYPE)

Parameters

unitType	type of enemy unit that we deal with

Returns

base attack of this unit type

Reimplemented in Infantry, Tank, and Artillery.

7.12.2.2 getPosition()

```
const Position Unit::getPosition ( ) const
```

Returns

Position of unit

7.12.2.3 getType()

```
virtual const UnitType Unit::getType ( )  [virtual]
```

get type ID of a unit

Returns

ID of unit type

Reimplemented in Infantry, Tank, and Artillery.

7.12.2.4 getUnitFactionID()

```
virtual const UnitFactionID Unit::getUnitFactionID ( ) const  [virtual]
```

get ID of unit faction

Returns

ID of a faction of unit

7.12.3 Member Data Documentation

7.12.3.1 organization

```
int Unit::organization  [protected]
```

organization stat of a unit.

It is similar to HP in strategy games. When organization is below 0, a unit dissolves. Movement, attack and defence cost some degree of organization

7.12.3.2 supplyLevel

```
int Unit::supplyLevel  [protected]
```

Level of logistic supply of a unit.

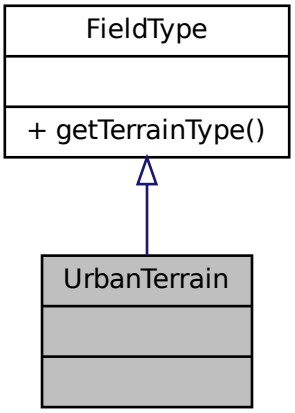
Level of supply is determined by fuel source nearby. It decreases during: defence, attack, movement. 100 supply level is considered enough (more gives no bonuses nor punishments)

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

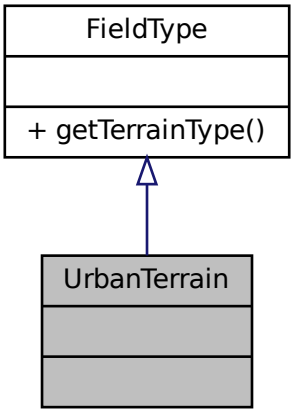
- include/units.h

7.13 UrbanTerrain Class Reference

Inheritance diagram for UrbanTerrain:



Collaboration diagram for UrbanTerrain:



Additional Inherited Members

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

- include/field.h

Chapter 8

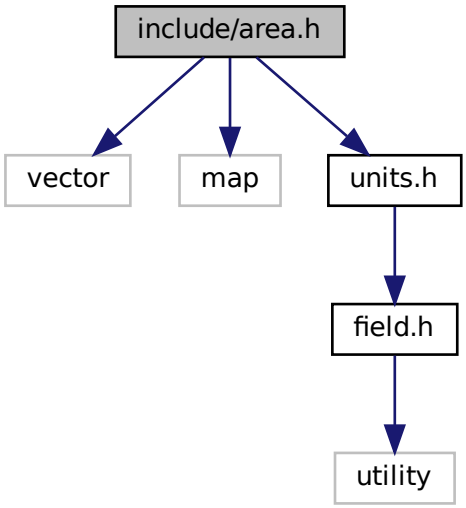
File Documentation

8.1 include/area.h File Reference

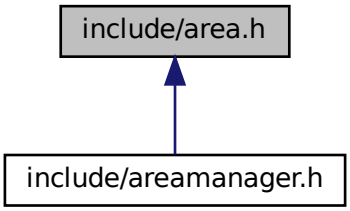
This class manages whole war area.

```
#include <vector>
#include <map>
#include "units.h"
```

Include dependency graph for area.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Area

8.1.1 Detailed Description

This class manages whole war area.

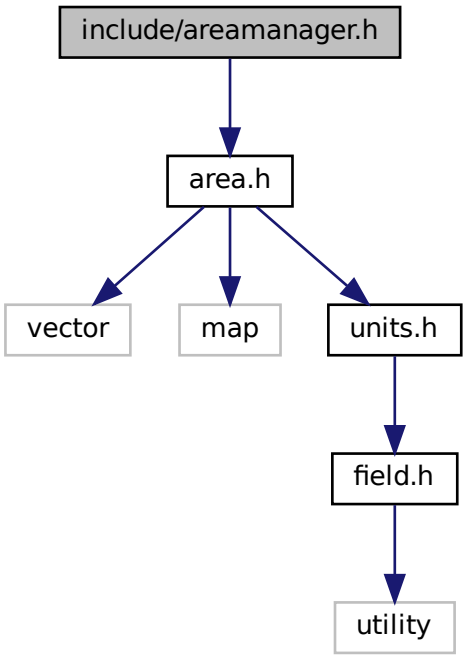
This manages states of a world, including units stats, fields, their terrain types etc. It DOES NOT handle changes of such world.

8.2 include/areamanager.h File Reference

This file manages the game area and its metadadata This class can be used to It has no safeguards, so it should not be used in top-level UI.

```
#include "area.h"
```

Include dependency graph for areamanager.h:



Classes

- class AreaManager

8.2.1 Detailed Description

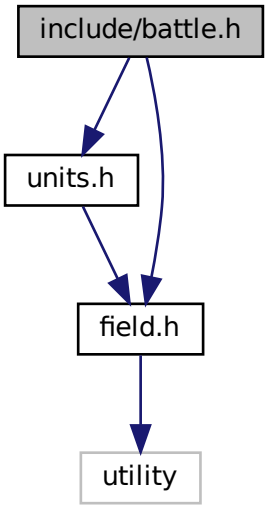
This file manages the game area and its metadadata This class can be used to It has no safeguards, so it should not be used in top-level UI.

8.3 include/battle.h File Reference

File with classes to resolve battles between two units.

```
#include "units.h"
#include "field.h"
```

Include dependency graph for battle.h:



Classes

- class BattleResult

  Class containing information regarding battle results.

8.3.1 Detailed Description

File with classes to resolve battles between two units.

Author

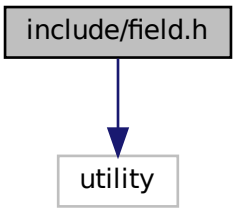
Maciej Sikorski

8.4 include/field.h File Reference

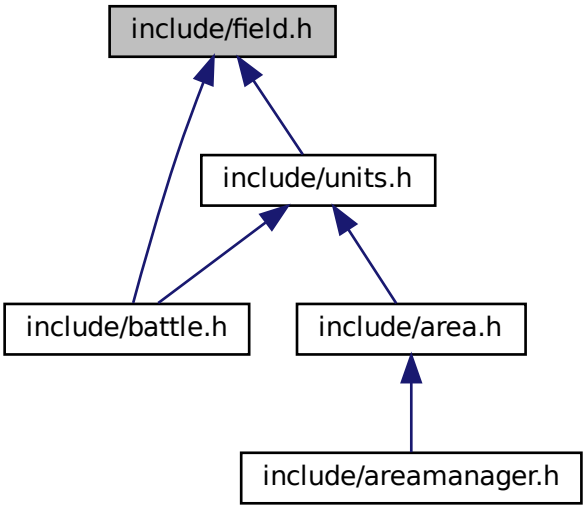
File with field classes.

```
#include <utility>
```

Include dependency graph for field.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Position

  Class to store position of a field.
- class FieldType

  All fields will give certain punishments for movement and attack thus need to be distinguished.
- class ForestTerrain
- class UrbanTerrain
- class PlainTerrain
- class Field

Namespaces

- namespace Terrain

  namespace consisting of terrain types IDs

Typedefs

- typedef int **TerrainType**

  type for identifiers of terrain
- typedef int **FieldID**

  unique identifier of a field

Variables

- const TerrainType **Terrain::NONE** = 0
- const TerrainType **Terrain::FOREST** = 1
- const TerrainType **Terrain::URBAN** = 2
- const TerrainType **Terrain::PLAIN** = 3

8.4.1 Detailed Description

File with field classes.

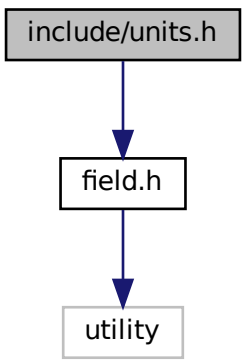
Field is characterized with position (q,r, -q-r) and terrain type (forest, urban etc.)

8.5 include/units.h File Reference

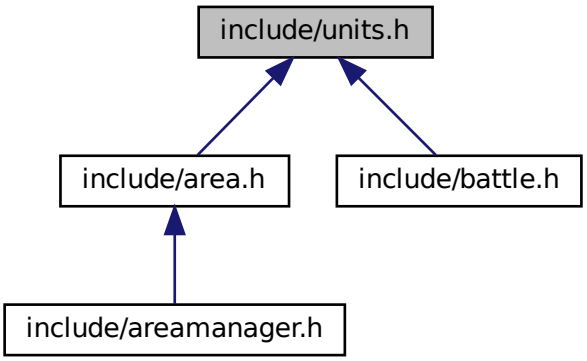
This file consists of classes of Units, i.e. Infantry, Tank, Artillery.

```
#include "field.h"
```

Include dependency graph for units.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Unit

  Basic class for all unit types.
- class Infantry
- class Tank
- class Artillery

Namespaces

- namespace UnitCategory

  namespace with unit types category IDs
- namespace baseAttack

  namespace with constants of X_Y attack. X_Y means that X attacks Y with this basic damage

Typedefs

- typedef int **UnitType**
- typedef int **UnitID**
- typedef int **UnitFactionID**

Variables

- constexpr UnitType **UnitCategory::NONE** = 0
- constexpr UnitType **UnitCategory::INFANTRY** = 1
- constexpr UnitType **UnitCategory::TANK** = 2
- constexpr UnitType **UnitCategory::ARTILLERY** = 3
- constexpr int **baseAttack::ERROR_PAIR** = -1

  base attack when type of units are wrong
- constexpr int **baseAttack::INFRANTRY_TO_INFANTRY** = 60
- constexpr int **baseAttack::INFANTRY_TO_ARTILLERY** = 100
- constexpr int **baseAttack::ARTILLERY_TO_ARTILLERY** = 100
- constexpr int **baseAttack::ARTILLERY_TO_INFANTRY** = 100

8.5.1 Detailed Description

This file consists of classes of Units, i.e. Infantry, Tank, Artillery.

Index