War Theatre Simulator

Generated by Doxygen 1.9.4

1 War Theatre Simulator 1

2 Namespace Index 3

2.1 Namespace List . . . 3

3 Hierarchical Index 5

3.1 Class Hierarchy . . . 5

4 Class Index 7

4.1 Class List . . . 7

5 File Index 9

5.1 File List . . . 9

6 Namespace Documentation 11

6.1 baseAttack Namespace Reference . . . 11

6.1.1 Detailed Description . . . 11

6.1.2 Variable Documentation . . . 11

6.1.2.1 ARTILLERY_TO_ARTILLERY . . . 11

6.1.2.2 ARTILLERY_TO_INFANTRY . . . 12

6.1.2.3 ERROR_PAIR . . . 12

6.1.2.4 INFANTRY_TO_ARTILLERY . . . 12

6.1.2.5 INFRANTRY_TO_INFANTRY . . . 12

6.2 Terrain Namespace Reference . . . 12

6.2.1 Detailed Description . . . 12

6.2.2 Variable Documentation . . . 13

6.2.2.1 FOREST . . . 13

6.2.2.2 NONE . . . 13

6.2.2.3 PLAIN . . . 13

6.2.2.4 URBAN . . . 13

6.3 UnitCategory Namespace Reference . . . 13

6.3.1 Detailed Description . . . 14

6.3.2 Variable Documentation . . . 14

6.3.2.1 ARTILLERY . . . 14

6.3.2.2 INFANTRY . . . 14

6.3.2.3 NONE . . . 14

6.3.2.4 TANK . . . 14

7 Class Documentation 15

7.1 Area Class Reference . . . 15

7.1.1 Detailed Description . . . 16

7.1.2 Member Function Documentation . . . 16

7.1.2.1 getUnitOnPosition() . . . 16

7.1.2.2 getUnitsOfFaction() . . . 17

7.1.2.3 isRealPosition() . . . 17
7.1.3 Member Data Documentation . . . 17
7.1.3.1 fields . . . 18
7.1.3.2 units . . . 18
7.2 AreaManager Class Reference . . . 18
7.2.1 Detailed Description . . . 19
7.3 Artillery Class Reference . . . 20
7.3.1 Detailed Description . . . 21
7.3.2 Member Function Documentation . . . 21
7.3.2.1 getBaseAttack() . . . 22
7.3.2.2 getType() . . . 22
7.4 Field Class Reference . . . 23
7.4.1 Detailed Description . . . 23
7.4.2 Member Function Documentation . . . 23
7.4.2.1 getPosition() . . . 24
7.4.2.2 getTerrainType() . . . 24
7.4.3 Member Data Documentation . . . 24
7.4.3.1 fieldType . . . 24
7.5 FieldType Class Reference . . . 25
7.5.1 Detailed Description . . . 25
7.5.2 Member Function Documentation . . . 26
7.5.2.1 getTerrainType() . . . 26
7.6 ForestTerrain Class Reference . . . 26
7.6.1 Detailed Description . . . 27
7.7 Infantry Class Reference . . . 28
7.7.1 Detailed Description . . . 29
7.7.2 Member Function Documentation . . . 29
7.7.2.1 getBaseAttack() . . . 30
7.7.2.2 getType() . . . 30
7.8 PlainTerrain Class Reference . . . 30
7.8.1 Detailed Description . . . 31
7.9 Position Class Reference . . . 31
7.9.1 Detailed Description . . . 32
7.10 Tank Class Reference . . . 32
7.10.1 Detailed Description . . . 33
7.10.2 Member Function Documentation . . . 33
7.10.2.1 getBaseAttack() . . . 34
7.10.2.2 getType() . . . 34
7.11 Unit Class Reference . . . 34
7.11.1 Detailed Description . . . 37
7.11.2 Member Function Documentation . . . 37
7.11.2.1 getBaseAttack() . . . 37

7.11.2.2 getPosition() . . . 37
7.11.2.3 getType() . . . 38
7.11.2.4 getUnitFactionID() . . . 38
7.11.3 Member Data Documentation . . . 38
7.11.3.1 baseStrength . . . 38
7.11.3.2 ID . . . 38
7.11.3.3 organization . . . 39
7.11.3.4 position . . . 39
7.11.3.5 supplyLevel . . . 39
7.11.3.6 unitFactionId . . . 39
7.11.3.7 unitRange . . . 40
7.12 UrbanTerrain Class Reference . . . 40
7.12.1 Detailed Description . . . 41

8 File Documentation **43**

8.1 include/area.h File Reference . . . 43
8.1.1 Detailed Description . . . 44
8.2 area.h . . . 44
8.3 include/areamanager.h File Reference . . . 45
8.3.1 Detailed Description . . . 45
8.4 areamanager.h . . . 46
8.5 include/field.h File Reference . . . 46
8.5.1 Detailed Description . . . 48
8.5.2 Typedef Documentation . . . 48
8.5.2.1 FieldID . . . 48
8.5.2.2 TerrainType . . . 48
8.6 field.h . . . 49
8.7 include/units.h File Reference . . . 49
8.7.1 Detailed Description . . . 51
8.7.2 Typedef Documentation . . . 51
8.7.2.1 UnitFactionID . . . 51
8.7.2.2 UnitID . . . 52
8.7.2.3 UnitType . . . 52
8.8 units.h . . . 52

Index **55**

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:

baseAttack
Namespace with constants of X_Y attack. X_Y means that X attacks Y with this basic damage 11
Terrain
Namespace consisting of terrain types IDs 12
UnitCategory
Namespace with unit types category IDs 13

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

- Area . . . 15
  - AreaManager . . . 18
- Field . . . 23
- FieldType . . . 25
  - ForestTerrain . . . 26
  - PlainTerrain . . . 30
  - UrbanTerrain . . . 40
- Position . . . 31
- Unit . . . 34
  - Artillery . . . 20
  - Infantry . . . 28
  - Tank . . . 32

Chapter 4

Class Index

4.1 Class List

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

Area 15
AreaManager 18
Artillery 20
Field 23
FieldType
    All fields will give certain punishments for movement and attack thus need to be distinguished . 25
ForestTerrain 26
Infantry 28
PlainTerrain 30
Position
    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) 31
Tank 32
Unit
    Basic class for all unit types 34
UrbanTerrain 40

Chapter 5

File Index

5.1 File List

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

include/area.h
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 43
include/areamanager.h
This file manages the game area and its metadadata It has no safeguards, so it should not be used in top-level UI. This class won't calculate any results of any events 45
include/field.h
File with field classes. Field is characterized with position (q,r, -q-r) and terrain type (forest, urban etc.) 46
include/units.h
This file consists of classes of Units, i.e. Infantry, Tank, Artillery 49

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

6.1.2 Variable Documentation

6.1.2.1 ARTILLERY_TO_ARTILLERY

```
constexpr int baseAttack::ARTILLERY_TO_ARTILLERY = 100  [constexpr]
```

Definition at line 29 of file units.h.

6.1.2.2 ARTILLERY_TO_INFANTRY

```
constexpr int baseAttack::ARTILLERY_TO_INFANTRY = 100  [constexpr]
```

Definition at line 30 of file units.h.

6.1.2.3 ERROR_PAIR

```
constexpr int baseAttack::ERROR_PAIR = -1  [constexpr]
```

base attack when type of units are wrong

Definition at line 24 of file units.h.

6.1.2.4 INFANTRY_TO_ARTILLERY

```
constexpr int baseAttack::INFANTRY_TO_ARTILLERY = 100  [constexpr]
```

Definition at line 27 of file units.h.

6.1.2.5 INFRANTRY_TO_INFANTRY

```
constexpr int baseAttack::INFRANTRY_TO_INFANTRY = 60  [constexpr]
```

Definition at line 26 of file units.h.

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

6.2.2 Variable Documentation

6.2.2.1 FOREST

```
const TerrainType Terrain::FOREST = 1
```

Definition at line 20 of file field.h.

6.2.2.2 NONE

```
const TerrainType Terrain::NONE = 0
```

Definition at line 19 of file field.h.

6.2.2.3 PLAIN

```
const TerrainType Terrain::PLAIN = 3
```

Definition at line 22 of file field.h.

6.2.2.4 URBAN

```
const TerrainType Terrain::URBAN = 2
```

Definition at line 21 of file field.h.

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

6.3.2 Variable Documentation

6.3.2.1 ARTILLERY

```
constexpr UnitType UnitCategory::ARTILLERY = 3  [constexpr]
```

Definition at line 17 of file units.h.

6.3.2.2 INFANTRY

```
constexpr UnitType UnitCategory::INFANTRY = 1  [constexpr]
```

Definition at line 15 of file units.h.

6.3.2.3 NONE

```
constexpr UnitType UnitCategory::NONE = 0  [constexpr]
```

Definition at line 14 of file units.h.

6.3.2.4 TANK

```
constexpr UnitType UnitCategory::TANK = 2  [constexpr]
```

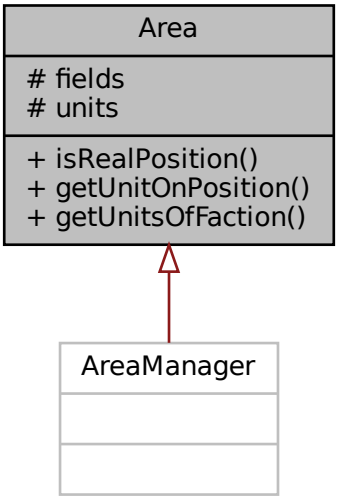
Definition at line 16 of file units.h.

Chapter 7

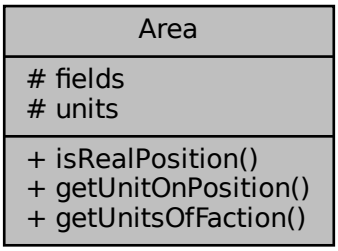
Class Documentation

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

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 Detailed Description

Definition at line 14 of file area.h.

7.1.2 Member Function Documentation

7.1.2.1 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

7.1.2.2 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

7.1.2.3 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

7.1.3 Member Data Documentation

7.1.3.1 fields

```
std::map<FieldID, Field> Area::fields  [protected]
```

Array with fields of the area.

Definition at line 18 of file area.h.

7.1.3.2 units

```
std::map<UnitID, Unit> Area::units  [protected]
```

Units present in arrea.

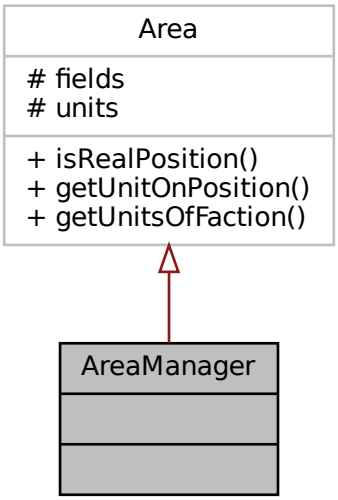
Definition at line 21 of file area.h.

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

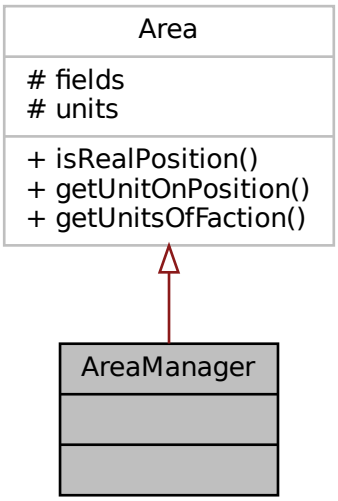
- include/area.h

7.2 AreaManager Class Reference

Inheritance diagram for AreaManager:



Collaboration diagram for AreaManager:



7.2.1 Detailed Description

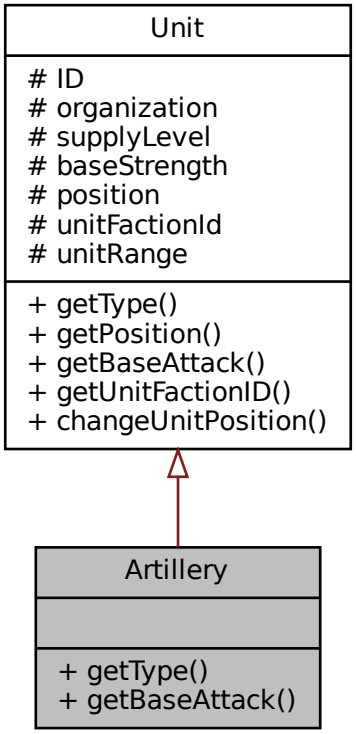
Definition at line 13 of file areamanager.h.

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

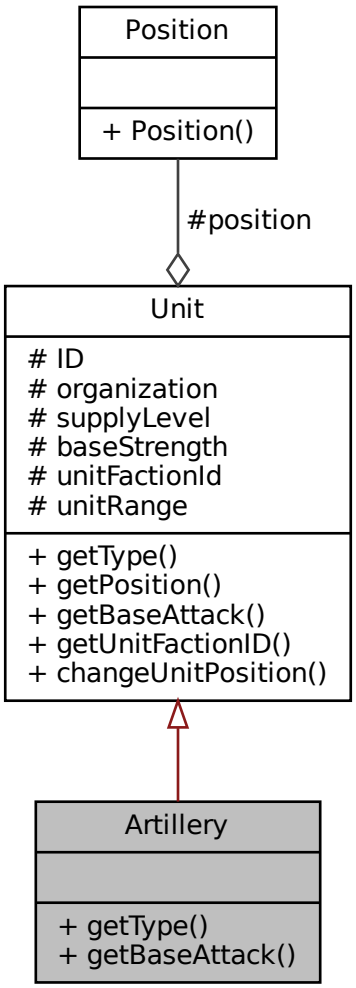
- include/areamanager.h

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 Detailed Description

Definition at line 115 of file units.h.

7.3.2 Member Function Documentation

7.3.2.1 getBaseAttack()

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

get Base attack of artillery

Parameters

unitType	type of enemy uniut

Returns

base attack for artillery when dealing with certain enemy

Reimplemented from Unit.

7.3.2.2 getType()

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

Returns

UnitCategory::ARTILLERY

Reimplemented from Unit.

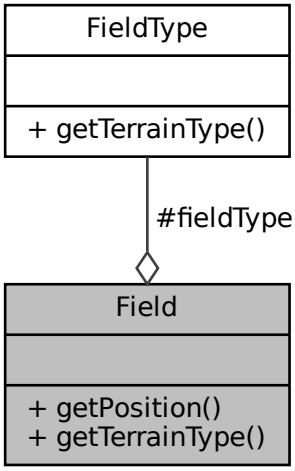
Definition at line 118 of file units.h.

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

- include/units.h

7.4 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

7.4.1 Detailed Description

Definition at line 67 of file field.h.

7.4.2 Member Function Documentation

7.4.2.1 getPosition()

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

get position of a field

Returns

position as (q,r,s)

7.4.2.2 getTerrainType()

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

Returns

type of terrain of the field

7.4.3 Member Data Documentation

7.4.3.1 fieldType

```
FieldType Field::fieldType  [protected]
```

Definition at line 73 of file field.h.

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

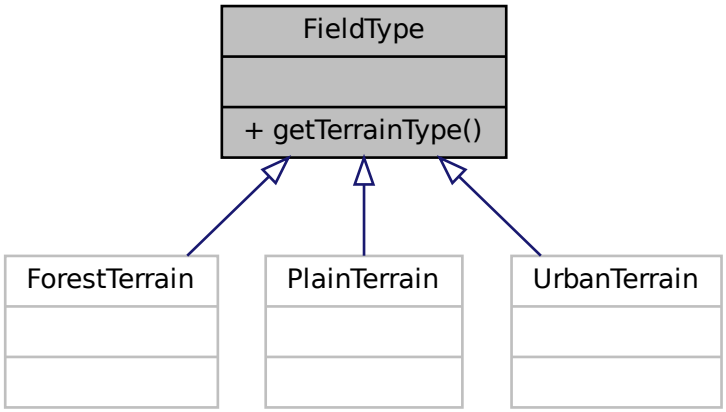
- include/field.h

7.5 FieldType Class Reference

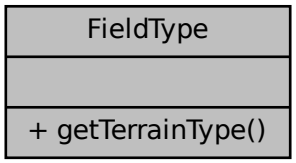
All fields will give certain punishments for movement and attack thus need to be distinguished.

`#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

7.5.1 Detailed Description

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

Definition at line 43 of file field.h.

7.5.2 Member Function Documentation

7.5.2.1 getTerrainType()

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

get ID of terrain type of this field

Returns

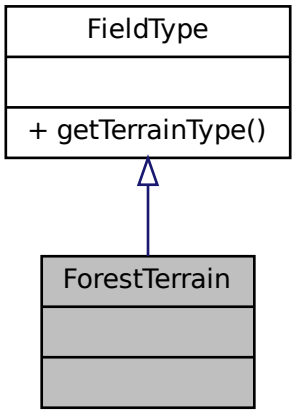
ID of terrain type for this field

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

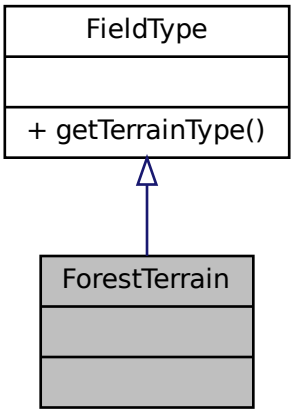
- include/field.h

7.6 ForestTerrain Class Reference

Inheritance diagram for ForestTerrain:



Collaboration diagram for ForestTerrain:



Additional Inherited Members

7.6.1 Detailed Description

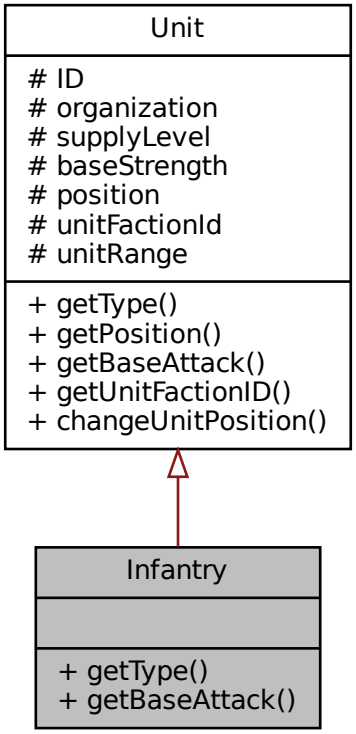
Definition at line 50 of file field.h.

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

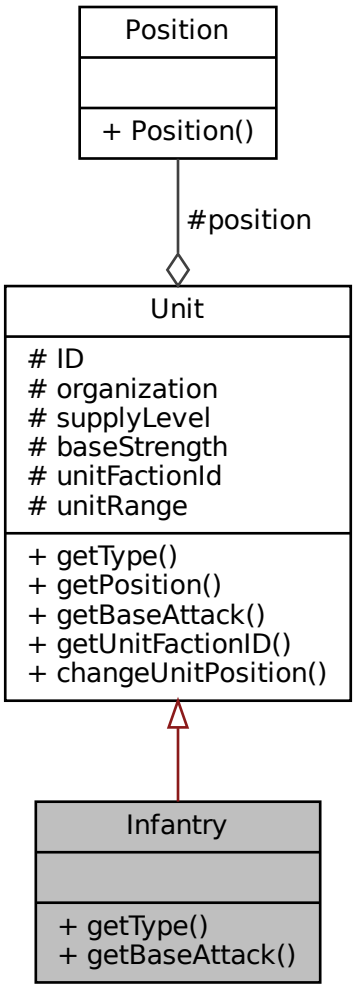
- include/field.h

7.7 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

7.7.1 Detailed Description

Definition at line 90 of file units.h.

7.7.2 Member Function Documentation

7.7.2.1 getBaseAttack()

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

get Base attack of infantry

Parameters

unitType	type of enemy uniut

Returns

base attack for infantry when dealing with certain enemy

Reimplemented from Unit.

7.7.2.2 getType()

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

Returns

UnitCategory::INFANTRY

Reimplemented from Unit.

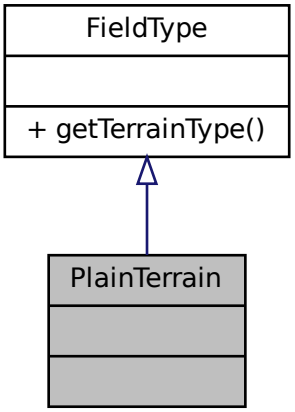
Definition at line 93 of file units.h.

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

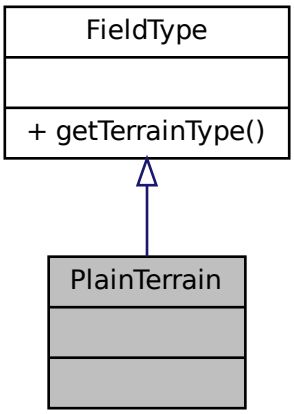
- include/units.h

7.8 PlainTerrain Class Reference

Inheritance diagram for PlainTerrain:



Collaboration diagram for PlainTerrain:



Additional Inherited Members

7.8.1 Detailed Description

Definition at line 60 of file field.h.

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

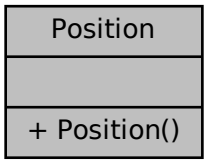
- include/field.h

7.9 Position Class Reference

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)

`#include <field.h>`

Collaboration diagram for Position:



Public Member Functions

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

7.9.1 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)

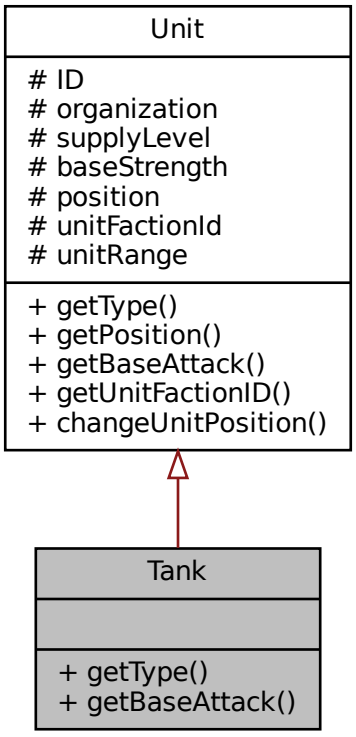
Definition at line 30 of file field.h.

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

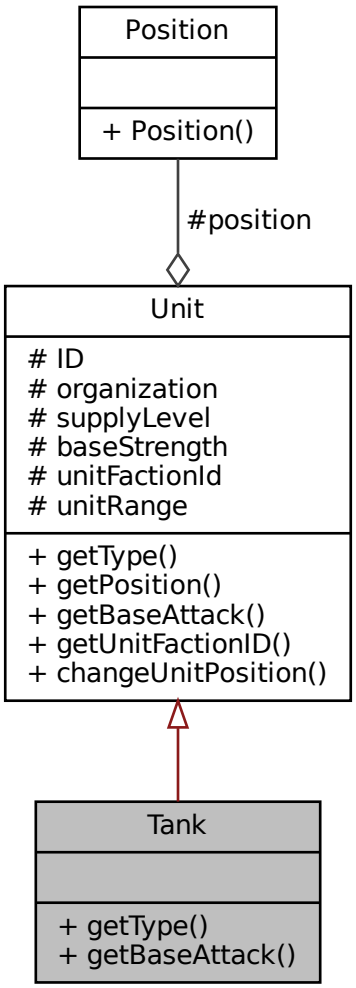
- include/field.h

7.10 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

7.10.1 Detailed Description

Definition at line 102 of file units.h.

7.10.2 Member Function Documentation

7.10.2.1 getBaseAttack()

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

get Base attack of tank

Parameters

unitType	type of enemy uniut

Returns

base attack for tank when dealing with certain enemy

Reimplemented from Unit.

7.10.2.2 getType()

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

Returns

UnitCategory::TANK

Reimplemented from Unit.

Definition at line 105 of file units.h.

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

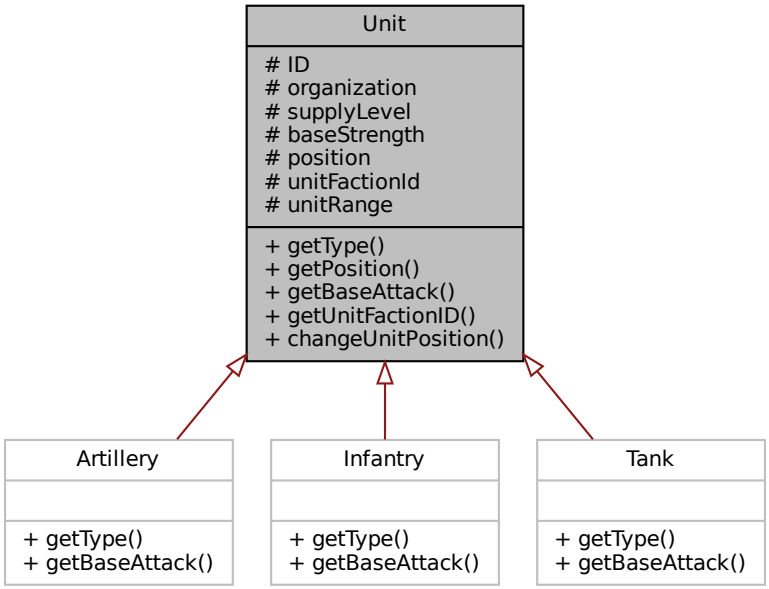
- include/units.h

7.11 Unit Class Reference

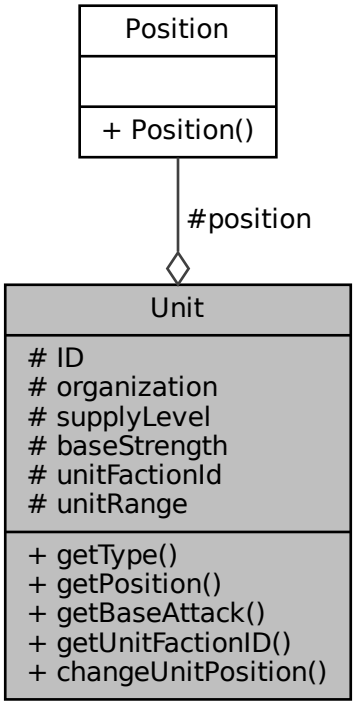
Basic class for all unit types.

```
#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

7.11.1 Detailed Description

Basic class for all unit types.

Definition at line 37 of file units.h.

7.11.2 Member Function Documentation

7.11.2.1 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.

7.11.2.2 getPosition()

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

Returns

Position of unit

7.11.2.3 getType()

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

get type ID of a unit

Returns

ID of unit type

Reimplemented in Infantry, Tank, and Artillery.

7.11.2.4 getUnitFactionID()

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

get ID of unit faction

Returns

ID of a faction of unit

7.11.3 Member Data Documentation

7.11.3.1 baseStrength

```
int Unit::baseStrength  [protected]
```

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

Definition at line 56 of file units.h.

7.11.3.2 ID

```
UnitID Unit::ID  [protected]
```

individual identifier for a unit

Definition at line 40 of file units.h.

7.11.3.3 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

Definition at line 46 of file units.h.

7.11.3.4 position

```
Position Unit::position  [protected]
```

position of a unit

Definition at line 59 of file units.h.

7.11.3.5 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)

Definition at line 53 of file units.h.

7.11.3.6 unitFactionId

```
UnitFactionID Unit::unitFactionId  [protected]
```

ID of faction that owns the unit`.

Definition at line 62 of file units.h.

7.11.3.7 unitRange

```
int Unit::unitRange  [protected]
```

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

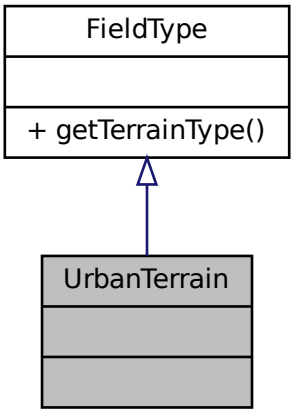
Definition at line 66 of file units.h.

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

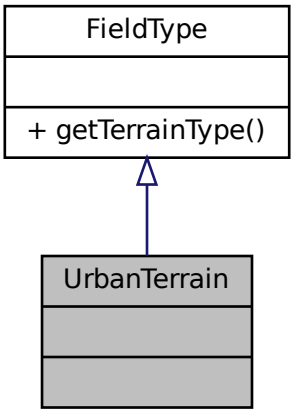
- include/units.h

7.12 UrbanTerrain Class Reference

Inheritance diagram for UrbanTerrain:



Collaboration diagram for UrbanTerrain:



Additional Inherited Members

7.12.1 Detailed Description

Definition at line 55 of file field.h.

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

- include/field.h

Chapter 8

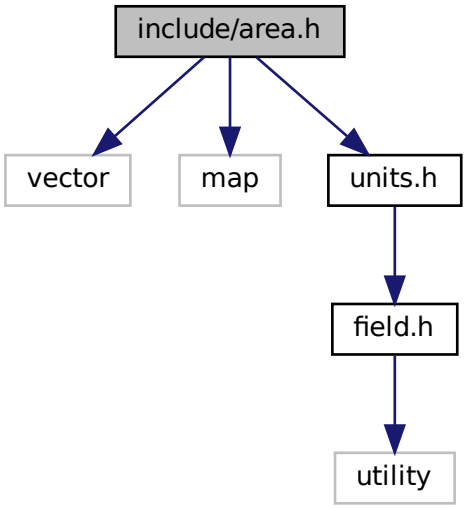
File Documentation

8.1 include/area.h File Reference

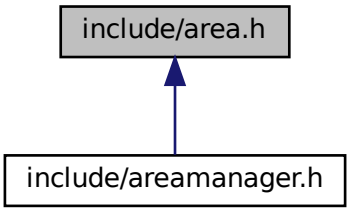
This class manages whole war area This manages states of a world, including units stats, fields, their terrain types etc. It DOES NOT handle changes of such world.

```
#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.

Author

Maciej Sikorski

Version

0.1

Date

20.05.2023

Definition in file area.h.

8.2 area.h

Go to the documentation of this file.

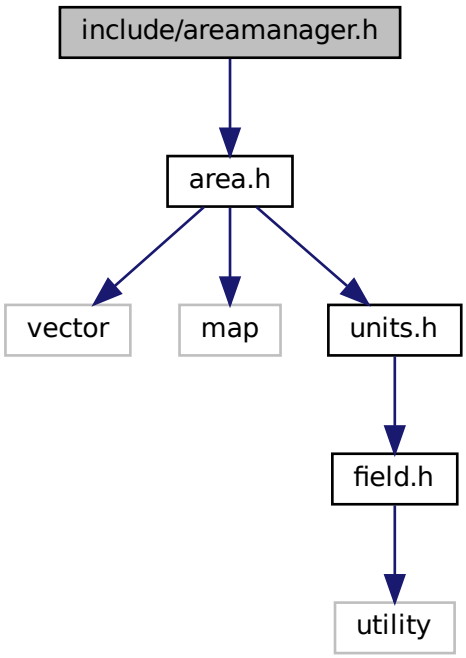
```
00001
00011 #include <vector>
00012 #include <map>
00013 #include "units.h"
00014 class Area{
00015
00016     protected:
00018         std::map<FieldID, Field> fields;
00019
00021         std::map<UnitID, Unit> units;
00022     public:
00023
00027         bool isRealPosition(const Position &position);
00028
00032         Unit& getUnitOnPosition(const Position &position);
00033
00034
00038         std::vector<UnitID> getUnitsOfFaction(const UnitFactionID &unitFactionID) const;
00039
00040
00041
00042 };
```

8.3 include/areamanager.h File Reference

This file manages the game area and its metadadata It has no safeguards, so it should not be used in top-level UI. This class won't calculate any results of any events.

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

Include dependency graph for areamanager.h:



Classes

- class AreaManager

8.3.1 Detailed Description

This file manages the game area and its metadadata It has no safeguards, so it should not be used in top-level UI. This class won't calculate any results of any events.

Author

Maciej Sikorski

Version

0.1

Date

22.05.2023

Definition in file areamanager.h.

8.4 areamanager.h

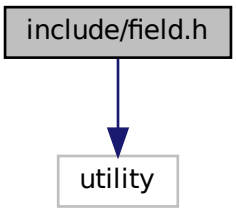
Go to the documentation of this file.

```
00001
00011 #include "area.h"
00012
00013 class AreaManager :  Area{
00014
00016     UnitID nextUnitID = 1;
00017
00019     FieldID nextFieldID = 1;
00020
00024     void createUnit(const Position &position, const Unit &unit);
00025
00028     void removeUnit(const UnitID &unitID);
00029
00033     void moveUnit(const UnitID &UnitID, const Position &position);
00034 };
```

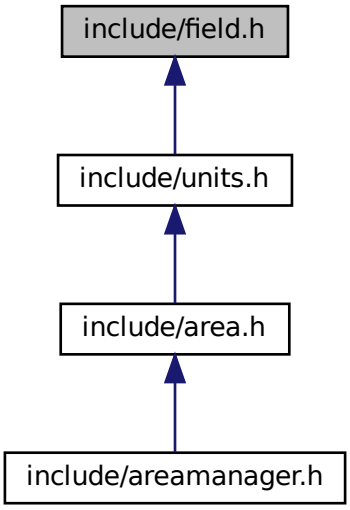
8.5 include/field.h File Reference

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

`#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. 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)
- 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

8.5.1 Detailed Description

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

Author

Maciej Sikorski

Version

0.1

Date

2023-05-19

Definition in file field.h.

8.5.2 Typedef Documentation

8.5.2.1 FieldID

```
typedef int FieldID
```

unique identifier of a field

Definition at line 15 of file field.h.

8.5.2.2 TerrainType

```
typedef int TerrainType
```

type for identifiers of terrain

Definition at line 12 of file field.h.

8.6 field.h

Go to the documentation of this file.

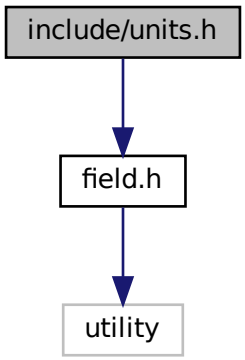
```
00001
00009 #include<utility>
00010
00012 typedef int TerrainType;
00013
00015 typedef int FieldID;
00016
00018 namespace Terrain {
00019     const TerrainType NONE = 0;
00020     const TerrainType FOREST = 1;
00021     const TerrainType URBAN = 2;
00022     const TerrainType PLAIN = 3;
00023 }
00024
00030 class Position{
00031     private:
00032         int q,r;
00033
00036         const int s();
00037     public:
00038         Position (const int &_q, const int &_r);
00039 };
00040
00041
00043 class FieldType{
00044     public:
00047         virtual const TerrainType getTerrainType();
00048 };
00049
00050 class ForestTerrain :  public FieldType{
00053     const TerrainType getTerrainType();
00054 };
00055 class UrbanTerrain :  public FieldType{
00058     const TerrainType getTerrainType();
00059 };
00060 class PlainTerrain :  public FieldType{
00063     const TerrainType getTerrainType();
00064 };
00065
00066
00067 class Field{
00068     private:
00069         Position position;
00070         FieldID fieldID;
00071
00072     protected:
00073         FieldType fieldType;
00074
00075     public:
00080         const Position getPosition();
00081
00085         virtual const FieldType getTerrainType();
00086 };
```

8.7 include/units.h File Reference

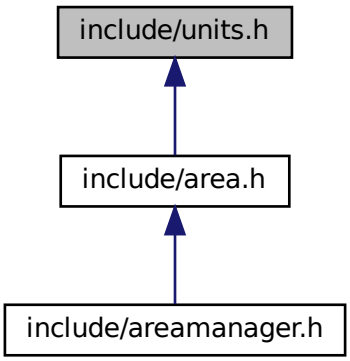
This file consists of classes of Units, i.e. Infantry, Tank, Artillery.

```
#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

8.7.1 Detailed Description

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

Definition in file units.h.

8.7.2 Typedef Documentation

8.7.2.1 UnitFactionID

```
typedef int UnitFactionID
```

Definition at line 10 of file units.h.

8.7.2.2 UnitID

```
typedef int UnitID
```

Definition at line 9 of file units.h.

8.7.2.3 UnitType

```
typedef int UnitType
```

Definition at line 8 of file units.h.

8.8 units.h

Go to the documentation of this file.

```
00001
00005 #include"field.h"
00006
00007
00008 typedef int UnitType;
00009 typedef int UnitID;
00010 typedef int UnitFactionID;
00011
00013 namespace UnitCategory{
00014     constexpr UnitType NONE = 0;
00015     constexpr UnitType INFANTRY = 1;
00016     constexpr UnitType TANK = 2;
00017     constexpr UnitType ARTILLERY = 3;
00018 };
00019
00022 namespace baseAttack{
00024     constexpr int ERROR_PAIR = -1;
00025
00026     constexpr int INFRANTRY_TO_INFANTRY = 60;
00027     constexpr int INFANTRY_TO_ARTILLERY = 100;
00028
00029     constexpr int ARTILLERY_TO_ARTILLERY = 100;
00030     constexpr int ARTILLERY_TO_INFANTRY = 100;
00031 }
00032
00037 class Unit{
00038     protected:
00040         UnitID ID;
00041
00043
00046         int organization;
00047
00049
00053         int supplyLevel;
00054
00056         int baseStrength;
00057
00059         Position position;
00060
00062         UnitFactionID unitFactionId;
00063
00066         int unitRange;
00067
00068     public:
00071         const virtual UnitType getType();
00072
00074         const Position getPosition() const;
00075
00079         virtual const int getBaseAttack(const UnitType &unitType);
00080
00081
00084         virtual const UnitFactionID getUnitFactionID() const;
00085
00087         void changeUnitPosition(const Position &position);
00088 };
```

```
00089
00090 class Infantry :  Unit {
00091     public:
00093         const UnitType getType(){
00094             return UnitCategory::INFANTRY;
00095         }
00099         const int getBaseAttack(const UnitID &unitType);
00100 };
00101
00102 class Tank :  Unit {
00103     public:
00105         const UnitType getType(){
00106             return UnitCategory::TANK;
00107         }
00111         const int getBaseAttack(const UnitID &unitType);
00112
00113 };
00114
00115 class Artillery :  Unit{
00116     public:
00118         const UnitType getType(){
00119             return UnitCategory::ARTILLERY;
00120         }
00121
00125         const int getBaseAttack(const UnitID &unitType);
00126 };
```

Index

Area, 15
    fields, 17
    getUnitOnPosition, 16
    getUnitsOfFaction, 17
    isRealPosition, 17
    units, 18
AreaManager, 18
ARTILLERY
    UnitCategory, 14
Artillery, 20
    getBaseAttack, 21
    getType, 22
ARTILLERY_TO_ARTILLERY
    baseAttack, 11
ARTILLERY_TO_INFANTRY
    baseAttack, 11

baseAttack, 11
    ARTILLERY_TO_ARTILLERY, 11
    ARTILLERY_TO_INFANTRY, 11
    ERROR_PAIR, 12
    INFANTRY_TO_ARTILLERY, 12
    INFRANTRY_TO_INFANTRY, 12
baseStrength
    Unit, 38

ERROR_PAIR
    baseAttack, 12

Field, 23
    fieldType, 24
    getPosition, 23
    getTerrainType, 24
field.h
    FieldID, 48
    TerrainType, 48
FieldID
    field.h, 48
fields
    Area, 17
FieldType, 25
    getTerrainType, 26
fieldType
    Field, 24
FOREST
    Terrain, 13
ForestTerrain, 26

getBaseAttack
    Artillery, 21
    Infantry, 29
    Tank, 33
    Unit, 37
getPosition
    Field, 23
    Unit, 37
getTerrainType
    Field, 24
    FieldType, 26
getType
    Artillery, 22
    Infantry, 30
    Tank, 34
    Unit, 37
getUnitFactionID
    Unit, 38
getUnitOnPosition
    Area, 16
getUnitsOfFaction
    Area, 17

ID
    Unit, 38
include/area.h, 43, 44
include/areamanager.h, 45, 46
include/field.h, 46, 49
include/units.h, 49, 52
INFANTRY
    UnitCategory, 14
Infantry, 28
    getBaseAttack, 29
    getType, 30
INFANTRY_TO_ARTILLERY
    baseAttack, 12
INFRANTRY_TO_INFANTRY
    baseAttack, 12
isRealPosition
    Area, 17

NONE
    Terrain, 13
    UnitCategory, 14

organization
    Unit, 38

PLAIN
    Terrain, 13
PlainTerrain, 30
Position, 31

position
Unit, 39

supplyLevel
Unit, 39

TANK
UnitCategory, 14
Tank, 32
getBaseAttack, 33
getType, 34
Terrain, 12
FOREST, 13
NONE, 13
PLAIN, 13
URBAN, 13
TerrainType
field.h, 48

Unit, 34
baseStrength, 38
getBaseAttack, 37
getPosition, 37
getType, 37
getUnitFactionID, 38
ID, 38
organization, 38
position, 39
supplyLevel, 39
unitFactionId, 39
unitRange, 39
UnitCategory, 13
ARTILLERY, 14
INFANTRY, 14
NONE, 14
TANK, 14
UnitFactionID
units.h, 51
unitFactionId
Unit, 39
UnitID
units.h, 51
unitRange
Unit, 39
units
Area, 18
units.h
UnitFactionID, 51
UnitID, 51
UnitType, 52
UnitType
units.h, 52
URBAN
Terrain, 13
UrbanTerrain, 40