TankGame

Generated by Doxygen 1.8.13

Contents

Chapter 1

Namespace Documentation

1.1 DefaultServerParams Namespace Reference

Variables

- const int port = 23856

1.1.1 Variable Documentation

1.1.1.1 port

```
const int DefaultServerParams::port = 23856
```

Definition at line 7 of file coreconst.h.

1.2 limits Namespace Reference

Variables

- const uint64_t maxCountOfItems = 18446744073709551615
- const uint64_t defaultBasisEnergy = 1000
- const uint64_t maxRadiusVisionOfBasis = 100

1.2.1 Variable Documentation

1.2.1.1 defaultBasisEnergy

```
const uint64_t limits::defaultBasisEnergy = 1000
```

Definition at line 12 of file coreconst.h.

1.2.1.2 maxCountOfItems

```
const uint64_t limits::maxCountOfItems = 18446744073709551615
```

Definition at line 11 of file coreconst.h.

1.2.1.3 maxRadiusVisionOfBasis

```
const uint64_t limits::maxRadiusVisionOfBasis = 100
```

Definition at line 13 of file coreconst.h.

1.3 staticBockTypes Namespace Reference

Variables

- int grass = 1

1.3.1 Variable Documentation

1.3.1.1 grass

```
int staticBockTypes::grass = 1
```

Definition at line 7 of file const.h.

1.4 Ui Namespace Reference

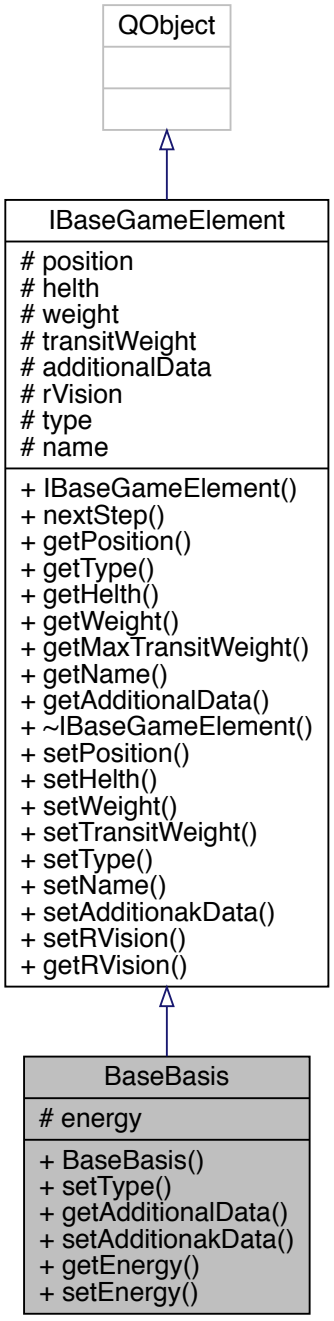
Chapter 2

Class Documentation

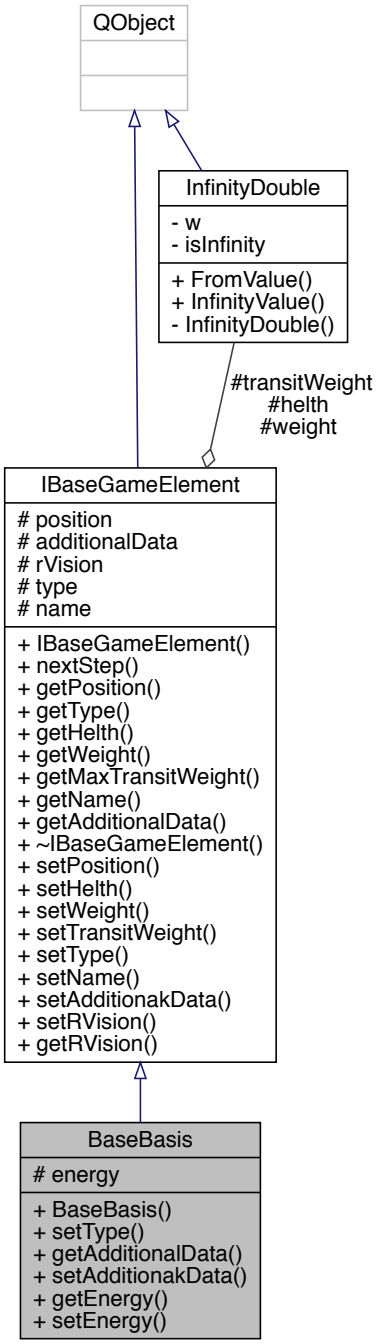
2.1 BaseBasis Class Reference

```
#include "basebasis.h"
```

Inheritance diagram for BaseBasis:



Collaboration diagram for BaseBasis:



Public Member Functions

- BaseBasis ()
- virtual void setType (int value) override
- virtual QByteArray * getAdditionalData ()
- virtual void setAdditionakData (QByteArray *data)
- virtual int getEnergy ()

- virtual void setEnergy (int _enery)
- virtual void nextStep ()
- virtual QVector3D * getPosition ()
- virtual int getType ()
- virtual InfinityDouble * getHelth ()
- virtual InfinityDouble * getWeight ()
- virtual InfinityDouble * getMaxTransitWeight ()
- virtual QString * getName ()
- virtual void setPosition (QVector3D *value)
- virtual void setHelth (InfinityDouble *value)
- virtual void setWeight (InfinityDouble *value)
- virtual void setTransitWeight (InfinityDouble *value)
- virtual void setName (QString name)
- virtual void setRVision (int _rVison)
- virtual int getRVision ()

Protected Attributes

- int energy = 100
- QVector3D * position = nullptr
- InfinityDouble * helth = nullptr
- InfinityDouble * weight = nullptr
- InfinityDouble * transitWeight = nullptr
- QByteArray * additionalData = nullptr
- int rVision = 1
- int type = -1
- QString name

2.1.1 Detailed Description

Definition at line 6 of file basebasis.h.

2.1.2 Constructor & Destructor Documentation

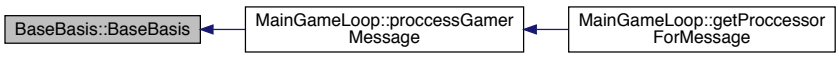
2.1.2.1 BaseBasis()

```
BaseBasis::BaseBasis ( )
```

Definition at line 4 of file basebasis.cpp.

Here is the call graph for this function:

Here is the caller graph for this function:



2.1.3 Member Function Documentation

2.1.3.1 getAdditionalData()

```
QByteArray * BaseBasis::getAdditionalData ( )  [virtual]
```

Reimplemented from IBaseGameElement.

Definition at line 17 of file basebasis.cpp.

2.1.3.2 getEnergy()

```
virtual int BaseBasis::getEnergy ( )  [inline], [virtual]
```

Definition at line 13 of file basebasis.h.

2.1.3.3 getHelth()

```
virtual InfinityDouble* IBaseGameElement::getHelth ( )  [inline], [virtual], [inherited]
```

Definition at line 33 of file ibasegameelement.h.

2.1.3.4 getMaxTransitWeight()

```
virtual InfinityDouble* IBaseGameElement::getMaxTransitWeight ( )  [inline], [virtual], [inherited]
```

Definition at line 35 of file ibasegameelement.h.

2.1.3.5 getName()

```
virtual QString* IBaseGameElement::getName ( )  [inline], [virtual], [inherited]
```

Definition at line 36 of file ibasegameelement.h.

2.1.3.6 getPosition()

```
virtual QVector3D* IBaseGameElement::getPosition ( )  [inline], [virtual], [inherited]
```

Definition at line 31 of file ibasegameelement.h.

2.1.3.7 getRVision()

```
virtual int IBaseGameElement::getRVision ( )  [inline], [virtual], [inherited]
```

Definition at line 81 of file ibasegameelement.h.

2.1.3.8 getType()

```
virtual int IBaseGameElement::getType ( )  [inline], [virtual], [inherited]
```

Definition at line 32 of file ibasegameelement.h.

2.1.3.9 getWeight()

```
virtual InfinityDouble* IBaseGameElement::getWeight ( )  [inline], [virtual], [inherited]
```

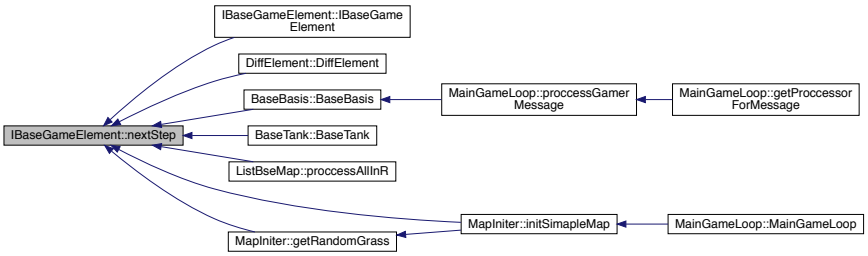
Definition at line 34 of file ibasegameelement.h.

2.1.3.10 nextStep()

```
virtual void IBaseGameElement::nextStep ( )  [inline], [virtual], [inherited]
```

Definition at line 29 of file ibasegameelement.h.

Here is the caller graph for this function:



2.1.3.11 setAdditionakData()

```
void BaseBasis::setAdditionakData (
            QByteArray * data )  [virtual]
```

Reimplemented from IBaseGameElement.

Definition at line 23 of file basebasis.cpp.

2.1.3.12 setEnergy()

```
virtual void BaseBasis::setEnergy (
            int _enery )  [inline], [virtual]
```

Definition at line 14 of file basebasis.h.

2.1.3.13 setHelth()

```
virtual void IBaseGameElement::setHelth (
            InfinityDouble * value )  [inline], [virtual], [inherited]
```

Definition at line 63 of file ibasegameelement.h.

Here is the caller graph for this function:

2.1.3.14 setName()

```
virtual void IBaseGameElement::setName (
            QString name )  [inline], [virtual], [inherited]
```

Definition at line 73 of file ibasegameelement.h.

2.1.3.15 setPosition()

```
virtual void IBaseGameElement::setPosition (
            QVector3D * value )  [inline], [virtual], [inherited]
```

Definition at line 61 of file ibasegameelement.h.

2.1.3.16 setRVision()

```
virtual void IBaseGameElement::setRVision (
            int _rVison )  [inline], [virtual], [inherited]
```

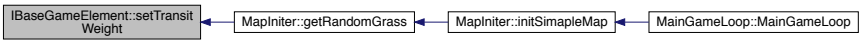
Definition at line 79 of file ibasegameelement.h.

2.1.3.17 setTransitWeight()

```
virtual void IBaseGameElement::setTransitWeight (
            InfinityDouble * value )  [inline], [virtual], [inherited]
```

Definition at line 67 of file ibasegameelement.h.

Here is the caller graph for this function:



2.1.3.18 setType()

```
void BaseBasis::setType (
            int value )  [override], [virtual]
```

Reimplemented from IBaseGameElement.

Definition at line 10 of file basebasis.cpp.

2.1.3.19 setWeight()

```
virtual void IBaseGameElement::setWeight (
            InfinityDouble * value )  [inline], [virtual], [inherited]
```

Definition at line 65 of file ibasegameelement.h.

2.1.4 Member Data Documentation

2.1.4.1 additionalData

```
QByteArray* IBaseGameElement::additionalData = nullptr  [protected], [inherited]
```

Definition at line 92 of file ibasegameelement.h.

2.1.4.2 energy

```
int BaseBasis::energy = 100  [protected]
```

Definition at line 17 of file basebasis.h.

2.1.4.3 helth

```
InfinityDouble* IBaseGameElement::helth = nullptr  [protected], [inherited]
```

Definition at line 89 of file ibasegameelement.h.

2.1.4.4 name

```
QString IBaseGameElement::name  [protected], [inherited]
```

Definition at line 96 of file ibasegameelement.h.

2.1.4.5 position

```
QVector3D* IBaseGameElement::position = nullptr  [protected], [inherited]
```

Definition at line 88 of file ibasegameelement.h.

2.1.4.6 rVision

```
int IBaseGameElement::rVision = 1  [protected], [inherited]
```

Definition at line 93 of file ibasegameelement.h.

2.1.4.7 transitWeight

```
InfinityDouble* IBaseGameElement::transitWeight = nullptr  [protected], [inherited]
```

Definition at line 91 of file ibasegameelement.h.

2.1.4.8 type

```
int IBaseGameElement::type = -1  [protected], [inherited]
```

Definition at line 95 of file ibasegameelement.h.

2.1.4.9 weight

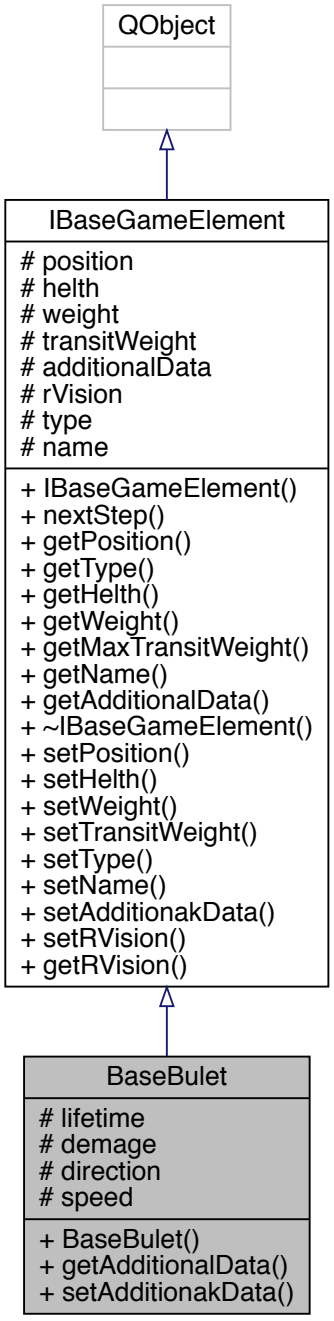
```
InfinityDouble* IBaseGameElement::weight = nullptr  [protected], [inherited]
```

Definition at line 90 of file ibasegameelement.h.

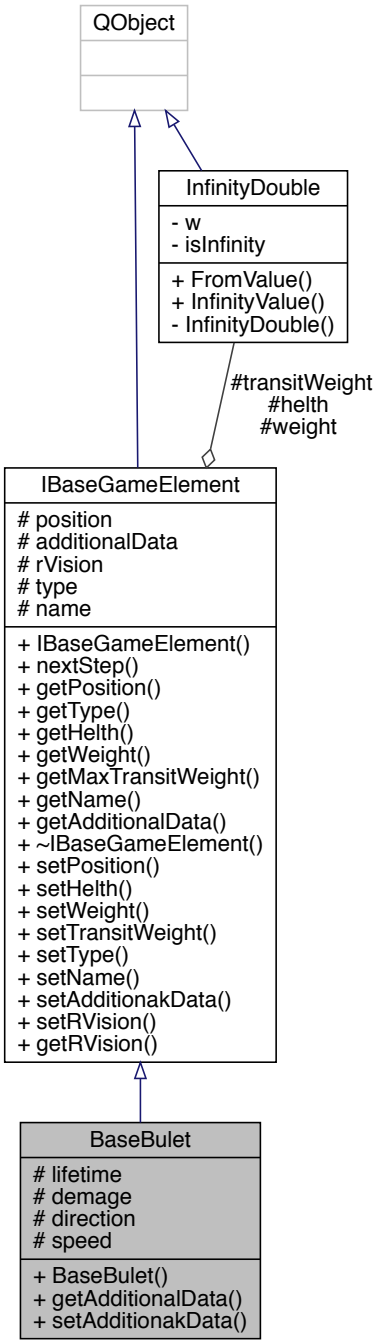
2.2 BaseBulet Class Reference

```
#include "basebulet.h"
```

Inheritance diagram for BaseBulet:



Collaboration diagram for BaseBulet:



Public Member Functions

- BaseBulet ()
- QByteArray * getAdditionalData ()
- void setAdditionakData (QByteArray *data)
- virtual void nextStep ()
- virtual QVector3D * getPosition ()

- virtual int getType ()
- virtual InfinityDouble * getHelth ()
- virtual InfinityDouble * getWeight ()
- virtual InfinityDouble * getMaxTransitWeight ()
- virtual QString * getName ()
- virtual void setPosition (QVector3D *value)
- virtual void setHelth (InfinityDouble *value)
- virtual void setWeight (InfinityDouble *value)
- virtual void setTransitWeight (InfinityDouble *value)
- virtual void setType (int value)
- virtual void setName (QString name)
- virtual void setRVision (int _rVison)
- virtual int getRVision ()

Protected Attributes

- int lifetime = 0
- double demage = 0
- double direction
- double speed
- QVector3D * position = nullptr
- InfinityDouble * helth = nullptr
- InfinityDouble * weight = nullptr
- InfinityDouble * transitWeight = nullptr
- QByteArray * additionalData = nullptr
- int rVision = 1
- int type = -1
- QString name

2.2.1 Detailed Description

Definition at line 6 of file basebulet.h.

2.2.2 Constructor & Destructor Documentation

2.2.2.1 BaseBulet()

```
BaseBulet::BaseBulet ( )
```

Definition at line 3 of file basebulet.cpp.

2.2.3 Member Function Documentation

2.2.3.1 getAdditionalData()

```
QByteArray * BaseBulet::getAdditionalData ( )  [virtual]
```

Reimplemented from IBaseGameElement.

Definition at line 5 of file basebulet.cpp.

2.2.3.2 getHelth()

```
virtual InfinityDouble* IBaseGameElement::getHelth ( )  [inline], [virtual], [inherited]
```

Definition at line 33 of file ibasegameelement.h.

2.2.3.3 getMaxTransitWeight()

```
virtual InfinityDouble* IBaseGameElement::getMaxTransitWeight ( )  [inline], [virtual], [inherited]
```

Definition at line 35 of file ibasegameelement.h.

2.2.3.4 getName()

```
virtual QString* IBaseGameElement::getName ( )  [inline], [virtual], [inherited]
```

Definition at line 36 of file ibasegameelement.h.

2.2.3.5 getPosition()

```
virtual QVector3D* IBaseGameElement::getPosition ( )  [inline], [virtual], [inherited]
```

Definition at line 31 of file ibasegameelement.h.

2.2.3.6 getRVision()

```
virtual int IBaseGameElement::getRVision ( )  [inline], [virtual], [inherited]
```

Definition at line 81 of file ibasegameelement.h.

2.2.3.7 getType()

```
virtual int IBaseGameElement::getType ( )  [inline], [virtual], [inherited]
```

Definition at line 32 of file ibasegameelement.h.

2.2.3.8 getWeight()

```
virtual InfinityDouble* IBaseGameElement::getWeight ( )  [inline], [virtual], [inherited]
```

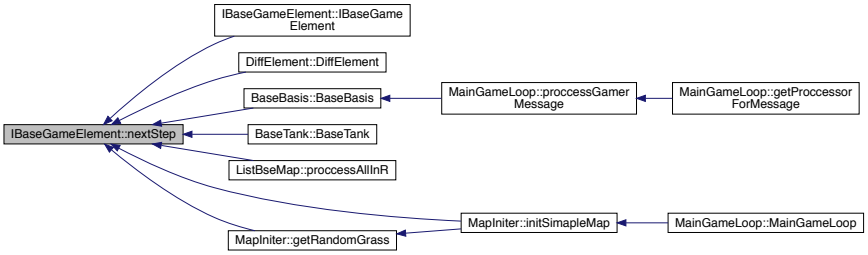
Definition at line 34 of file ibasegameelement.h.

2.2.3.9 nextStep()

```
virtual void IBaseGameElement::nextStep ( )  [inline], [virtual], [inherited]
```

Definition at line 29 of file ibasegameelement.h.

Here is the caller graph for this function:



2.2.3.10 setAdditionakData()

```
void BaseBulet::setAdditionakData (
            QByteArray * data )  [virtual]
```

Reimplemented from IBaseGameElement.

Definition at line 7 of file basebulet.cpp.

2.2.3.11 setHelth()

```
virtual void IBaseGameElement::setHelth (
            InfinityDouble * value )  [inline], [virtual], [inherited]
```

Definition at line 63 of file ibasegameelement.h.

Here is the caller graph for this function:

2.2.3.12 setName()

```
virtual void IBaseGameElement::setName (
            QString name )  [inline], [virtual], [inherited]
```

Definition at line 73 of file ibasegameelement.h.

2.2.3.13 setPosition()

```
virtual void IBaseGameElement::setPosition (
            QVector3D * value )  [inline], [virtual], [inherited]
```

Definition at line 61 of file ibasegameelement.h.

2.2.3.14 setRVision()

```
virtual void IBaseGameElement::setRVision (
            int _rVison )  [inline], [virtual], [inherited]
```

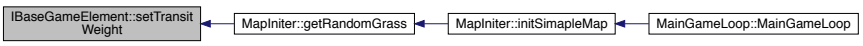
Definition at line 79 of file ibasegameelement.h.

2.2.3.15 setTransitWeight()

```
virtual void IBaseGameElement::setTransitWeight (
            InfinityDouble * value )  [inline], [virtual], [inherited]
```

Definition at line 67 of file ibasegameelement.h.

Here is the caller graph for this function:



2.2.3.16 setType()

```
virtual void IBaseGameElement::setType (
            int value )  [inline], [virtual], [inherited]
```

Reimplemented in BaseTank, and BaseBasis.

Definition at line 71 of file ibasegameelement.h.

Here is the caller graph for this function:

2.2.3.17 setWeight()

```
virtual void IBaseGameElement::setWeight (
            InfinityDouble * value )  [inline], [virtual], [inherited]
```

Definition at line 65 of file ibasegameelement.h.

2.2.4 Member Data Documentation

2.2.4.1 additionalData

```
QByteArray* IBaseGameElement::additionalData = nullptr  [protected], [inherited]
```

Definition at line 92 of file ibasegameelement.h.

2.2.4.2 demage

```
double BaseBulet::demage = 0  [protected]
```

Definition at line 12 of file basebulet.h.

2.2.4.3 direction

```
double BaseBulet::direction  [protected]
```

Definition at line 13 of file basebulet.h.

2.2.4.4 helth

```
InfinityDouble* IBaseGameElement::helth = nullptr  [protected], [inherited]
```

Definition at line 89 of file ibasegameelement.h.

2.2.4.5 lifetime

```
int BaseBulet::lifetime = 0  [protected]
```

Definition at line 11 of file basebulet.h.

2.2.4.6 name

```
QString IBaseGameElement::name  [protected], [inherited]
```

Definition at line 96 of file ibasegameelement.h.

2.2.4.7 position

```
QVector3D* IBaseGameElement::position = nullptr  [protected], [inherited]
```

Definition at line 88 of file ibasegameelement.h.

2.2.4.8 rVision

```
int IBaseGameElement::rVision = 1  [protected], [inherited]
```

Definition at line 93 of file ibasegameelement.h.

2.2.4.9 speed

```
double BaseBulet::speed  [protected]
```

Definition at line 14 of file basebulet.h.

2.2.4.10 transitWeight

```
InfinityDouble* IBaseGameElement::transitWeight = nullptr  [protected], [inherited]
```

Definition at line 91 of file ibasegameelement.h.

2.2.4.11 type

```
int IBaseGameElement::type = -1  [protected], [inherited]
```

Definition at line 95 of file ibasegameelement.h.

2.2.4.12 weight

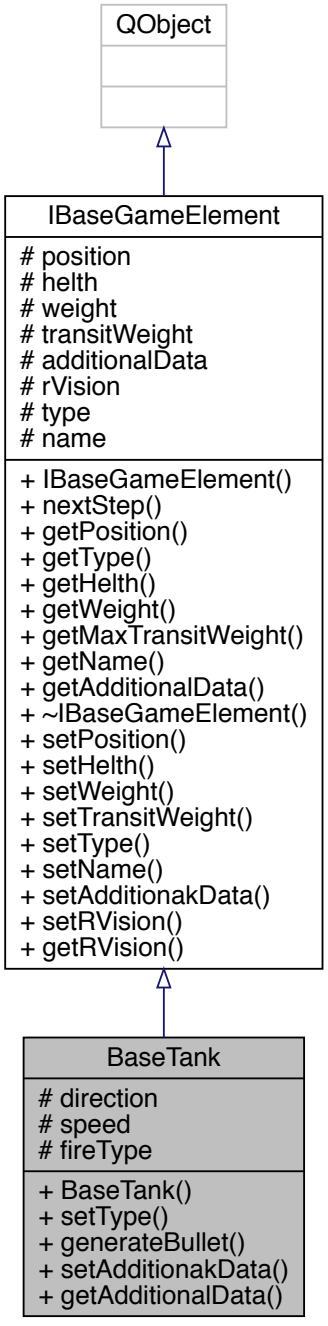
```
InfinityDouble* IBaseGameElement::weight = nullptr  [protected], [inherited]
```

Definition at line 90 of file ibasegameelement.h.

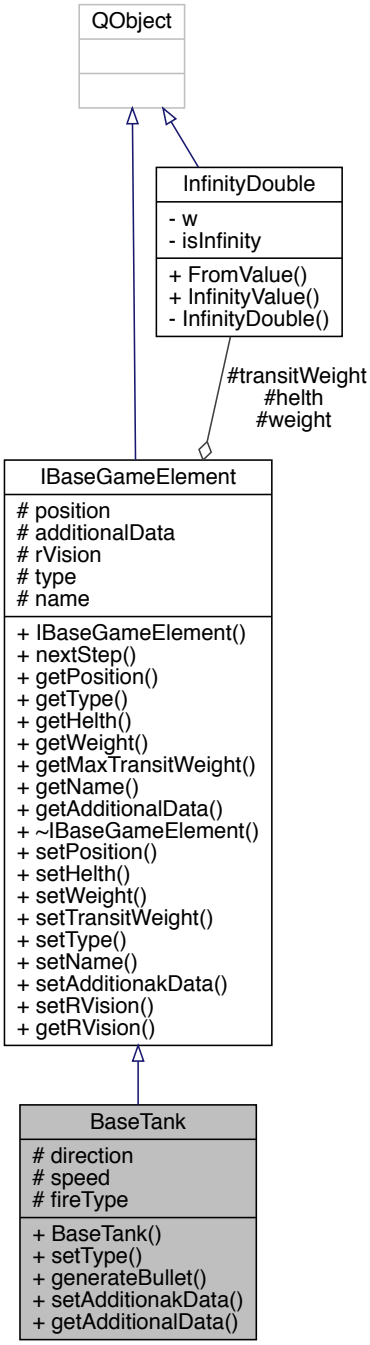
2.3 BaseTank Class Reference

```
#include "basetank.h"
```

Inheritance diagram for BaseTank:



Collaboration diagram for BaseTank:



Public Member Functions

- BaseTank ()
- virtual void setType (int value) override
- virtual BaseBulet generateBullet ()
- virtual void setAdditionakData (QByteArray ∗data)
- virtual QByteArray ∗ getAdditionalData ()

- virtual void nextStep ()
- virtual QVector3D ∗ getPosition ()
- virtual int getType ()
- virtual InfinityDouble ∗ getHelth ()
- virtual InfinityDouble ∗ getWeight ()
- virtual InfinityDouble ∗ getMaxTransitWeight ()
- virtual QString ∗ getName ()
- virtual void setPosition (QVector3D ∗value)
- virtual void setHelth (InfinityDouble ∗value)
- virtual void setWeight (InfinityDouble ∗value)
- virtual void setTransitWeight (InfinityDouble ∗value)
- virtual void setName (QString name)
- virtual void setRVision (int _rVison)
- virtual int getRVision ()

Protected Attributes

- double direction = 0
- double speed = 0
- int fireType = 0

  fireType 0 for non fire -1 for single fire if fireType > 0 then fire will be call evry fireType tik of game
- QVector3D ∗ position = nullptr
- InfinityDouble ∗ helth = nullptr
- InfinityDouble ∗ weight = nullptr
- InfinityDouble ∗ transitWeight = nullptr
- QByteArray ∗ additionalData = nullptr
- int rVision = 1
- int type = -1
- QString name

2.3.1 Detailed Description

Definition at line 6 of file basetank.h.

2.3.2 Constructor & Destructor Documentation

2.3.2.1 BaseTank()

```
BaseTank::BaseTank ( )
```

Definition at line 4 of file basetank.cpp.

Here is the call graph for this function:

2.3.3 Member Function Documentation

2.3.3.1 generateBullet()

```
BaseBulet BaseTank::generateBullet ( )  [virtual]
```

Definition at line 13 of file basetank.cpp.

2.3.3.2 getAdditionalData()

```
QByteArray * BaseTank::getAdditionalData ( )  [virtual]
```

Reimplemented from IBaseGameElement.

Definition at line 21 of file basetank.cpp.

2.3.3.3 getHelth()

```
virtual InfinityDouble* IBaseGameElement::getHelth ( )  [inline], [virtual], [inherited]
```

Definition at line 33 of file ibasegameelement.h.

2.3.3.4 getMaxTransitWeight()

```
virtual InfinityDouble* IBaseGameElement::getMaxTransitWeight ( )  [inline], [virtual], [inherited]
```

Definition at line 35 of file ibasegameelement.h.

2.3.3.5 getName()

```
virtual QString* IBaseGameElement::getName ( )  [inline], [virtual], [inherited]
```

Definition at line 36 of file ibasegameelement.h.

2.3.3.6 getPosition()

```
virtual QVector3D* IBaseGameElement::getPosition ( )  [inline], [virtual], [inherited]
```

Definition at line 31 of file ibasegameelement.h.

2.3.3.7 getRVision()

```
virtual int IBaseGameElement::getRVision ( )  [inline], [virtual], [inherited]
```

Definition at line 81 of file ibasegameelement.h.

2.3.3.8 getType()

```
virtual int IBaseGameElement::getType ( )  [inline], [virtual], [inherited]
```

Definition at line 32 of file ibasegameelement.h.

2.3.3.9 getWeight()

```
virtual InfinityDouble* IBaseGameElement::getWeight ( )  [inline], [virtual], [inherited]
```

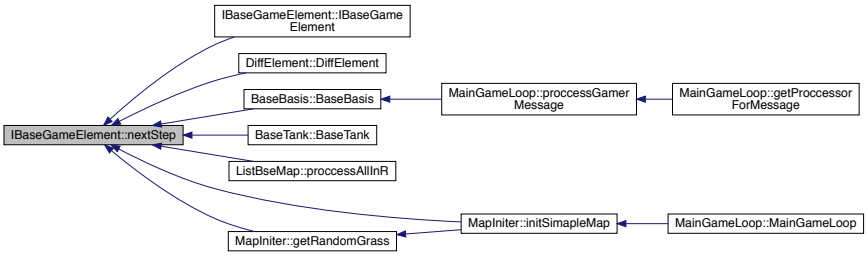
Definition at line 34 of file ibasegameelement.h.

2.3.3.10 nextStep()

```
virtual void IBaseGameElement::nextStep ( )  [inline], [virtual], [inherited]
```

Definition at line 29 of file ibasegameelement.h.

Here is the caller graph for this function:



2.3.3.11 setAdditionakData()

```
void BaseTank::setAdditionakData (
            QByteArray * data )  [virtual]
```

Reimplemented from IBaseGameElement.

Definition at line 15 of file basetank.cpp.

2.3.3.12 setHelth()

```
virtual void IBaseGameElement::setHelth (
            InfinityDouble * value )  [inline], [virtual], [inherited]
```

Definition at line 63 of file ibasegameelement.h.

Here is the caller graph for this function:

2.3.3.13 setName()

```
virtual void IBaseGameElement::setName (
            QString name )  [inline], [virtual], [inherited]
```

Definition at line 73 of file ibasegameelement.h.

2.3.3.14 setPosition()

```
virtual void IBaseGameElement::setPosition (
            QVector3D * value )  [inline], [virtual], [inherited]
```

Definition at line 61 of file ibasegameelement.h.

2.3.3.15 setRVision()

```
virtual void IBaseGameElement::setRVision (
            int _rVison )  [inline], [virtual], [inherited]
```

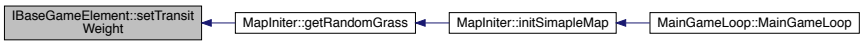
Definition at line 79 of file ibasegameelement.h.

2.3.3.16 setTransitWeight()

```
virtual void IBaseGameElement::setTransitWeight (
            InfinityDouble * value )  [inline], [virtual], [inherited]
```

Definition at line 67 of file ibasegameelement.h.

Here is the caller graph for this function:



2.3.3.17 setType()

```
void BaseTank::setType (
            int value )  [override], [virtual]
```

Reimplemented from IBaseGameElement.

Definition at line 6 of file basetank.cpp.

2.3.3.18 setWeight()

```
virtual void IBaseGameElement::setWeight (
            InfinityDouble * value )  [inline], [virtual], [inherited]
```

Definition at line 65 of file ibasegameelement.h.

2.3.4 Member Data Documentation

2.3.4.1 additionalData

```
QByteArray* IBaseGameElement::additionalData = nullptr  [protected], [inherited]
```

Definition at line 92 of file ibasegameelement.h.

2.3.4.2 direction

```
double BaseTank::direction = 0  [protected]
```

Definition at line 17 of file basetank.h.

2.3.4.3 fireType

```
int BaseTank::fireType = 0  [protected]
```

fireType 0 for non fire -1 for single fire if fireType $>$ 0 then fire will be call evry fireType tik of game

Definition at line 25 of file basetank.h.

2.3.4.4 helth

```
InfinityDouble* IBaseGameElement::helth = nullptr  [protected], [inherited]
```

Definition at line 89 of file ibasegameelement.h.

2.3.4.5 name

```
QString IBaseGameElement::name  [protected], [inherited]
```

Definition at line 96 of file ibasegameelement.h.

2.3.4.6 position

```
QVector3D* IBaseGameElement::position = nullptr  [protected], [inherited]
```

Definition at line 88 of file ibasegameelement.h.

2.3.4.7 rVision

```
int IBaseGameElement::rVision = 1  [protected], [inherited]
```

Definition at line 93 of file ibasegameelement.h.

2.3.4.8 speed

```
double BaseTank::speed = 0  [protected]
```

Definition at line 18 of file basetank.h.

2.3.4.9 transitWeight

```
InfinityDouble* IBaseGameElement::transitWeight = nullptr  [protected], [inherited]
```

Definition at line 91 of file ibasegameelement.h.

2.3.4.10 type

```
int IBaseGameElement::type = -1  [protected], [inherited]
```

Definition at line 95 of file ibasegameelement.h.

2.3.4.11 weight

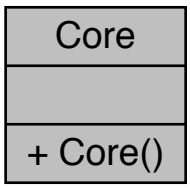
```
InfinityDouble* IBaseGameElement::weight = nullptr  [protected], [inherited]
```

Definition at line 90 of file ibasegameelement.h.

2.4 Core Class Reference

```
#include "core.h"
```

Collaboration diagram for Core:



Public Member Functions

- Core ()

2.4.1 Detailed Description

Definition at line 6 of file core.h.

2.4.2 Constructor & Destructor Documentation

2.4.2.1 Core()

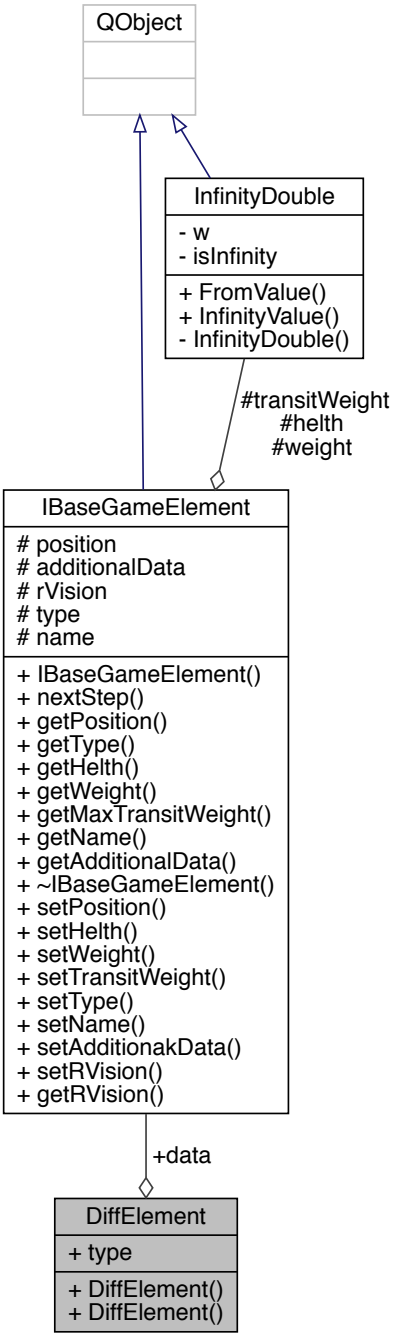
```
Core::Core ( )
```

Definition at line 4 of file core.cpp.

2.5 DiffElement Class Reference

```
#include "ibasegameelement.h"
```

Collaboration diagram for DiffElement:



Public Member Functions

- DiffElement ()
- DiffElement (eDiffType type, IBaseGameElement *data)

Public Attributes

- eDiffType type

- IBaseGameElement * data

Friends

- QDataStream & operator<< (QDataStream &stream, const DiffElement &myclass)
- QDataStream & operator>> (QDataStream &stream, DiffElement &myclass)

2.5.1 Detailed Description

Definition at line 99 of file ibasegameelement.h.

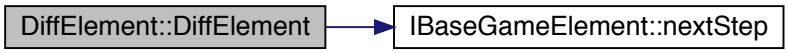
2.5.2 Constructor & Destructor Documentation

2.5.2.1 DiffElement() [1/2]

```
DiffElement::DiffElement ( )  [inline]
```

Definition at line 101 of file ibasegameelement.h.

Here is the call graph for this function:



2.5.2.2 DiffElement() [2/2]

```
DiffElement::DiffElement (
            eDiffType type,
            IBaseGameElement * data )  [inline]
```

Definition at line 102 of file ibasegameelement.h.

2.5.3 Friends And Related Function Documentation

2.5.3.1 operator<<

```
QDataStream& operator<< (
            QDataStream & stream,
            const DiffElement & myclass )  [friend]
```

Definition at line 107 of file ibasegameelement.h.

2.5.3.2 operator>>

```
QDataStream& operator>> (
            QDataStream & stream,
            DiffElement & myclass )  [friend]
```

Definition at line 113 of file ibasegameelement.h.

2.5.4 Member Data Documentation

2.5.4.1 data

```
IBaseGameElement* DiffElement::data
```

Definition at line 121 of file ibasegameelement.h.

2.5.4.2 type

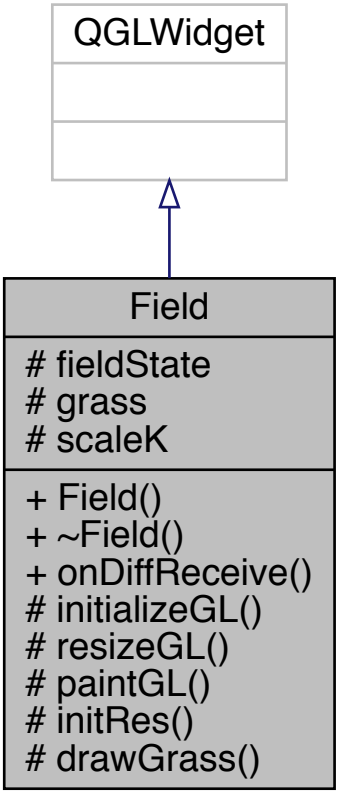
```
eDiffType DiffElement::type
```

Definition at line 120 of file ibasegameelement.h.

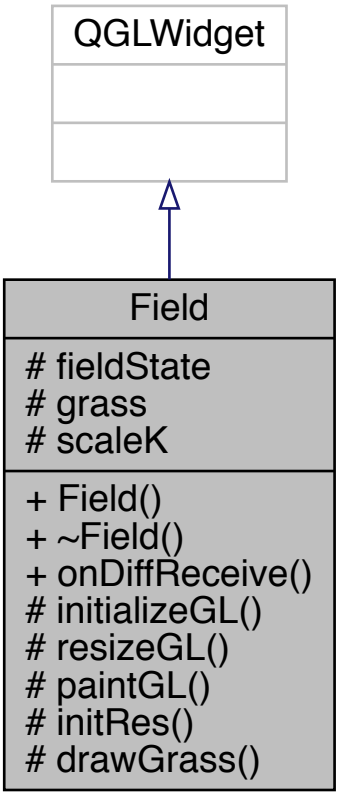
2.6 Field Class Reference

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

Inheritance diagram for Field:



Collaboration diagram for Field:



Public Slots

- void onDiffReceive (QList< DiffElement *> *diff)

Public Member Functions

- Field (QWidget *parent=0)
- ~Field ()

Protected Member Functions

- void initializeGL () Q_DECL_OVERRIDE
- void resizeGL (int w, int h) Q_DECL_OVERRIDE
- void paintGL () Q_DECL_OVERRIDE
- void initRes ()
- void drawGrass (float x, float y)

Protected Attributes

- QList< IBaseGameElement * > * fieldState
- GLuint grass
- float scaleK = 0.005

2.6.1 Detailed Description

Definition at line 6 of file field.h.

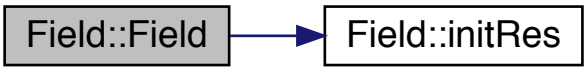
2.6.2 Constructor & Destructor Documentation

2.6.2.1 Field()

```
Field::Field (
            QWidget * parent = 0 )
```

Definition at line 4 of file field.cpp.

Here is the call graph for this function:



2.6.2.2 ~Field()

```
Field::~Field ( )
```

Definition at line 9 of file field.cpp.

2.6.3 Member Function Documentation

2.6.3.1 drawGrass()

```
void Field::drawGrass (
            float x,
            float y )  [protected]
```

Definition at line 69 of file field.cpp.

2.6.3.2 initializeGL()

```
void Field::initializeGL ( )  [protected]
```

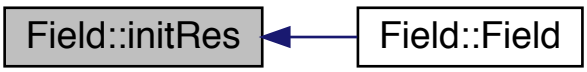
Definition at line 21 of file field.cpp.

2.6.3.3 initRes()

```
void Field::initRes ( )  [protected]
```

Definition at line 65 of file field.cpp.

Here is the caller graph for this function:



2.6.3.4 onDiffReceive

```
void Field::onDiffReceive (
            QList< DiffElement *> * diff )  [slot]
```

Definition at line 13 of file field.cpp.

2.6.3.5 paintGL()

```
void Field::paintGL ( )  [protected]
```

Definition at line 25 of file field.cpp.

2.6.3.6 resizeGL()

```
void Field::resizeGL (
            int w,
            int h )  [protected]
```

Definition at line 23 of file field.cpp.

2.6.4 Member Data Documentation

2.6.4.1 fieldState

```
QList<IBaseGameElement*>* Field::fieldState  [protected]
```

Definition at line 19 of file field.h.

2.6.4.2 grass

```
GLuint Field::grass  [protected]
```

Definition at line 24 of file field.h.

2.6.4.3 scaleK

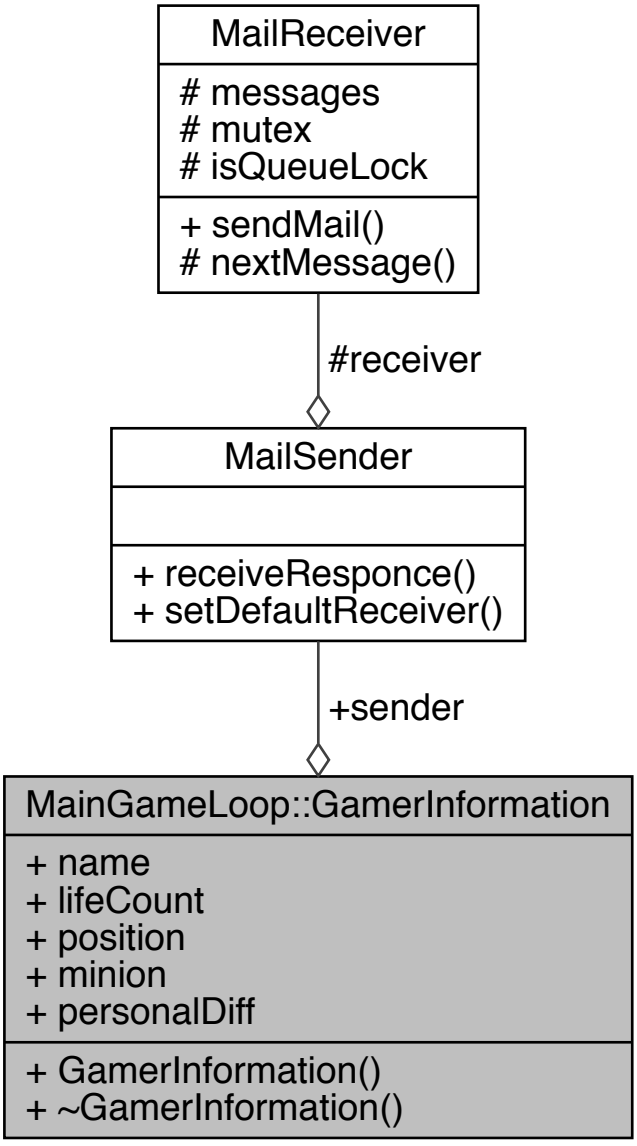
```
float Field::scaleK = 0.005  [protected]
```

Definition at line 26 of file field.h.

2.7 MainGameLoop::GamerInformation Class Reference

```
#include "maingameloop.h"
```

Collaboration diagram for MainGameLoop::GamerInformation:



Public Member Functions

- GamerInformation (IMap *map)
- ~GamerInformation ()

Public Attributes

- QString name
- uint64_t lifeCount
- QVector3D position
- MailSender * sender
- QList< IBaseGameElement * > * minion
- QList< DiffElement * > * personalDiff

2.7.1 Detailed Description

Definition at line 36 of file maingameloop.h.

2.7.2 Constructor & Destructor Documentation

2.7.2.1 GamerInformation()

```
MainGameLoop::GamerInformation::GamerInformation (
            IMap * map )
```

Definition at line 104 of file maingameloop.cpp.

2.7.2.2 ~GamerInformation()

```
MainGameLoop::GamerInformation::~GamerInformation ( )
```

Definition at line 115 of file maingameloop.cpp.

2.7.3 Member Data Documentation

2.7.3.1 lifeCount

```
uint64_t MainGameLoop::GamerInformation::lifeCount
```

Definition at line 39 of file maingameloop.h.

2.7.3.2 minion

```
QList<IBaseGameElement*>* MainGameLoop::GamerInformation::minion
```

Definition at line 42 of file maingameloop.h.

2.7.3.3 name

```
QString MainGameLoop::GamerInformation::name
```

Definition at line 38 of file maingameloop.h.

2.7.3.4 personalDiff

```
QList<DiffElement*>* MainGameLoop::GamerInformation::personalDiff
```

Definition at line 43 of file maingameloop.h.

2.7.3.5 position

```
QVector3D MainGameLoop::GamerInformation::position
```

Definition at line 40 of file maingameloop.h.

2.7.3.6 sender

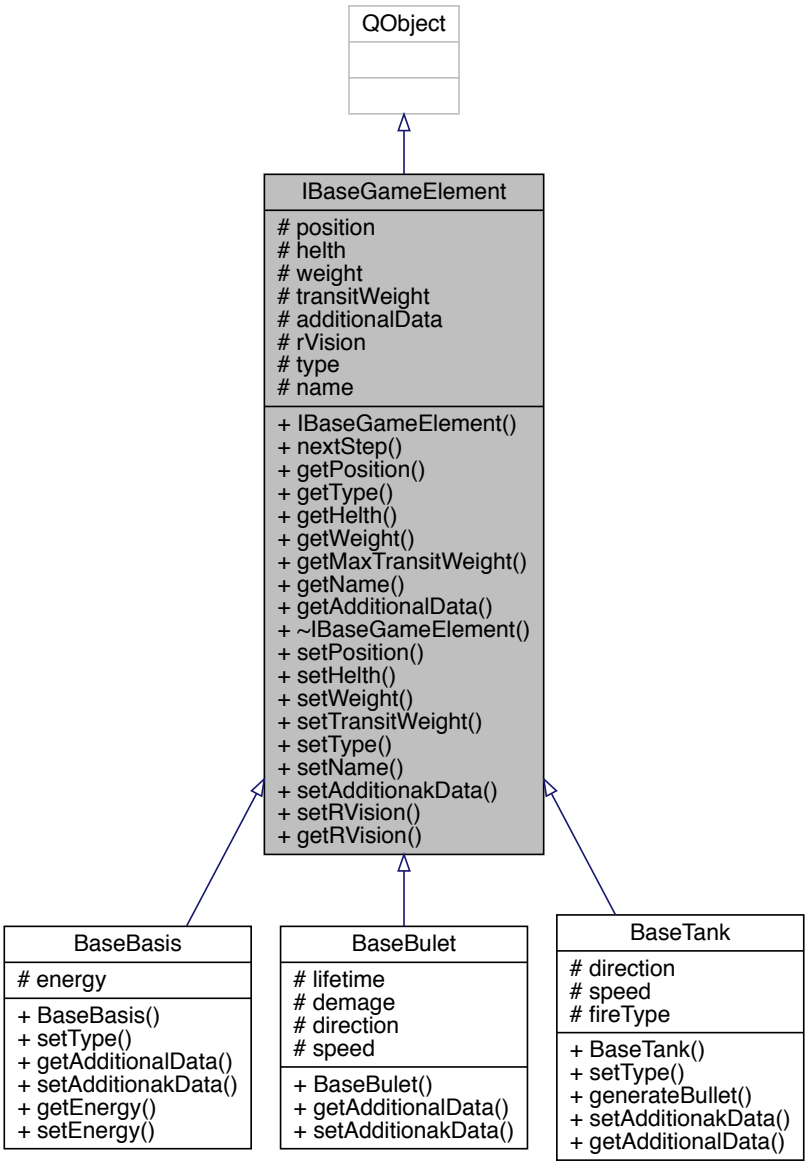
```
MailSender* MainGameLoop::GamerInformation::sender
```

Definition at line 41 of file maingameloop.h.

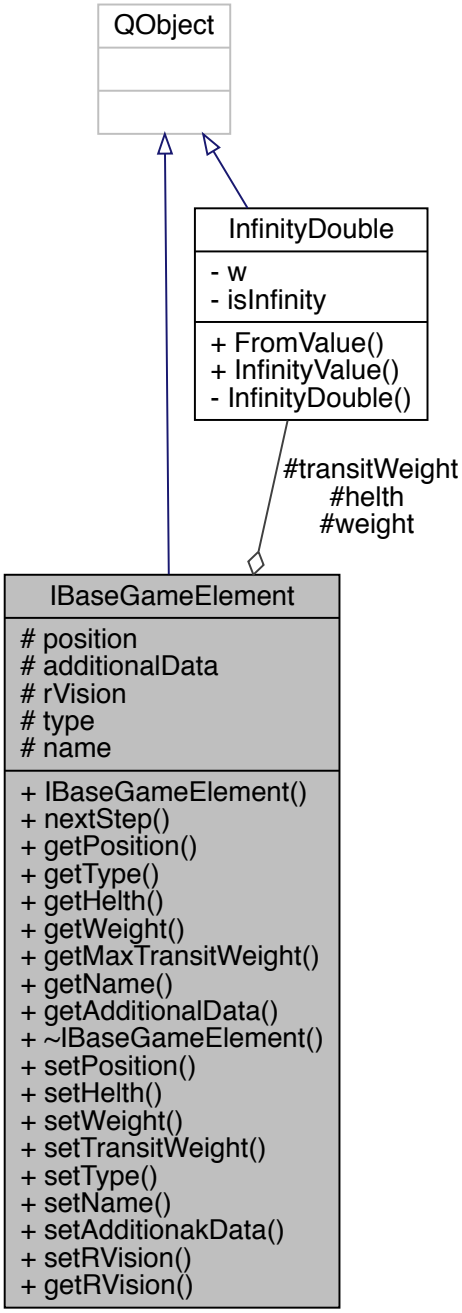
2.8 IBaseGameElement Class Reference

```
#include "ibasegameelement.h"
```

Inheritance diagram for IBaseGameElement:



Collaboration diagram for IBaseGameElement:



Public Member Functions

- IBaseGameElement ()
- virtual void nextStep ()
- virtual QVector3D * getPosition ()
- virtual int getType ()
- virtual InfinityDouble * getHelth ()

- virtual InfinityDouble * getWeight ()
- virtual InfinityDouble * getMaxTransitWeight ()
- virtual QString * getName ()
- virtual QByteArray * getAdditionalData ()
- virtual ~IBaseGameElement ()
- virtual void setPosition (QVector3D *value)
- virtual void setHelth (InfinityDouble *value)
- virtual void setWeight (InfinityDouble *value)
- virtual void setTransitWeight (InfinityDouble *value)
- virtual void setType (int value)
- virtual void setName (QString name)
- virtual void setAdditionakData (QByteArray *data)
- virtual void setRVision (int _rVison)
- virtual int getRVision ()

Protected Attributes

- QVector3D * position = nullptr
- InfinityDouble * helth = nullptr
- InfinityDouble * weight = nullptr
- InfinityDouble * transitWeight = nullptr
- QByteArray * additionalData = nullptr
- int rVision = 1
- int type = -1
- QString name

Friends

- QDataStream & operator<< (QDataStream &stream, const IBaseGameElement &myclass)
- QDataStream & operator>> (QDataStream &stream, IBaseGameElement &myclass)

2.8.1 Detailed Description

Definition at line 17 of file ibasegameelement.h.

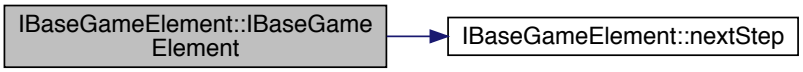
2.8.2 Constructor & Destructor Documentation

2.8.2.1 IBaseGameElement()

```
IBaseGameElement::IBaseGameElement ( )  [inline]
```

Definition at line 20 of file ibasegameelement.h.

Here is the call graph for this function:



2.8.2.2 ~IBaseGameElement()

```
virtual IBaseGameElement::~IBaseGameElement ( )  [inline], [virtual]
```

Definition at line 54 of file ibasegameelement.h.

2.8.3 Member Function Documentation

2.8.3.1 getAdditionalData()

```
virtual QByteArray* IBaseGameElement::getAdditionalData ( )  [inline], [virtual]
```

Reimplemented in BaseBulet, BaseTank, and BaseBasis.

Definition at line 37 of file ibasegameelement.h.

2.8.3.2 getHelth()

```
virtual InfinityDouble* IBaseGameElement::getHelth ( )  [inline], [virtual]
```

Definition at line 33 of file ibasegameelement.h.

2.8.3.3 getMaxTransitWeight()

```
virtual InfinityDouble* IBaseGameElement::getMaxTransitWeight ( )  [inline], [virtual]
```

Definition at line 35 of file ibasegameelement.h.

2.8.3.4 getName()

```
virtual QString* IBaseGameElement::getName ( )  [inline], [virtual]
```

Definition at line 36 of file ibasegameelement.h.

2.8.3.5 getPosition()

```
virtual QVector3D* IBaseGameElement::getPosition ( )  [inline], [virtual]
```

Definition at line 31 of file ibasegameelement.h.

2.8.3.6 getRVision()

```
virtual int IBaseGameElement::getRVision ( )  [inline], [virtual]
```

Definition at line 81 of file ibasegameelement.h.

2.8.3.7 getType()

```
virtual int IBaseGameElement::getType ( )  [inline], [virtual]
```

Definition at line 32 of file ibasegameelement.h.

2.8.3.8 getWeight()

```
virtual InfinityDouble* IBaseGameElement::getWeight ( )  [inline], [virtual]
```

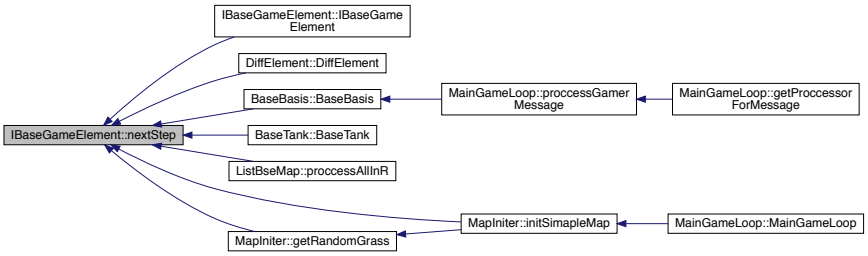
Definition at line 34 of file ibasegameelement.h.

2.8.3.9 nextStep()

```
virtual void IBaseGameElement::nextStep ( )  [inline], [virtual]
```

Definition at line 29 of file ibasegameelement.h.

Here is the caller graph for this function:



2.8.3.10 setAdditionakData()

```
virtual void IBaseGameElement::setAdditionakData (
            QByteArray * data )  [inline], [virtual]
```

Reimplemented in BaseBulet, BaseTank, and BaseBasis.

Definition at line 75 of file ibasegameelement.h.

2.8.3.11 setHelth()

```
virtual void IBaseGameElement::setHelth (
            InfinityDouble * value )  [inline], [virtual]
```

Definition at line 63 of file ibasegameelement.h.

Here is the caller graph for this function:

2.8.3.12 setName()

```
virtual void IBaseGameElement::setName (
            QString name )  [inline], [virtual]
```

Definition at line 73 of file ibasegameelement.h.

2.8.3.13 setPosition()

```
virtual void IBaseGameElement::setPosition (
            QVector3D * value )  [inline], [virtual]
```

Definition at line 61 of file ibasegameelement.h.

2.8.3.14 setRVision()

```
virtual void IBaseGameElement::setRVision (
            int _rVison )  [inline], [virtual]
```

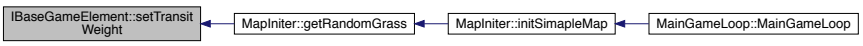
Definition at line 79 of file ibasegameelement.h.

2.8.3.15 setTransitWeight()

```
virtual void IBaseGameElement::setTransitWeight (
            InfinityDouble * value )  [inline], [virtual]
```

Definition at line 67 of file ibasegameelement.h.

Here is the caller graph for this function:



2.8.3.16 setType()

```
virtual void IBaseGameElement::setType (
            int value )  [inline], [virtual]
```

Reimplemented in BaseTank, and BaseBasis.

Definition at line 71 of file ibasegameelement.h.

Here is the caller graph for this function:

2.8.3.17 setWeight()

```
virtual void IBaseGameElement::setWeight (
            InfinityDouble * value )  [inline], [virtual]
```

Definition at line 65 of file ibasegameelement.h.

2.8.4 Friends And Related Function Documentation

2.8.4.1 operator<<

```
QDataStream& operator<< (
            QDataStream & stream,
            const IBaseGameElement & myclass )  [friend]
```

Definition at line 39 of file ibasegameelement.h.

2.8.4.2 operator>>

```
QDataStream& operator>> (
            QDataStream & stream,
            IBaseGameElement & myclass )  [friend]
```

Definition at line 46 of file ibasegameelement.h.

2.8.5 Member Data Documentation

2.8.5.1 additionalData

```
QByteArray* IBaseGameElement::additionalData = nullptr  [protected]
```

Definition at line 92 of file ibasegameelement.h.

2.8.5.2 helth

```
InfinityDouble* IBaseGameElement::helth = nullptr  [protected]
```

Definition at line 89 of file ibasegameelement.h.

2.8.5.3 name

```
QString IBaseGameElement::name  [protected]
```

Definition at line 96 of file ibasegameelement.h.

2.8.5.4 position

```
QVector3D* IBaseGameElement::position = nullptr  [protected]
```

Definition at line 88 of file ibasegameelement.h.

2.8.5.5 rVision

```
int IBaseGameElement::rVision = 1  [protected]
```

Definition at line 93 of file ibasegameelement.h.

2.8.5.6 transitWeight

```
InfinityDouble* IBaseGameElement::transitWeight = nullptr  [protected]
```

Definition at line 91 of file ibasegameelement.h.

2.8.5.7 type

```
int IBaseGameElement::type = -1  [protected]
```

Definition at line 95 of file ibasegameelement.h.

2.8.5.8 weight

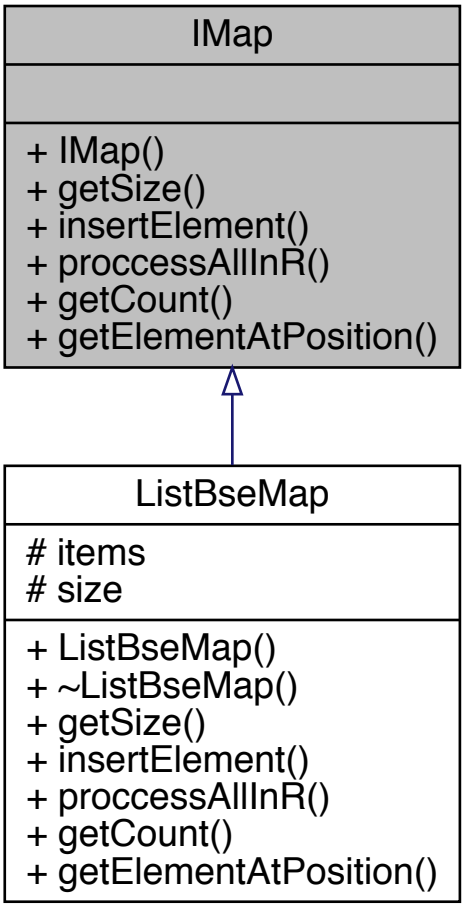
```
InfinityDouble* IBaseGameElement::weight = nullptr  [protected]
```

Definition at line 90 of file ibasegameelement.h.

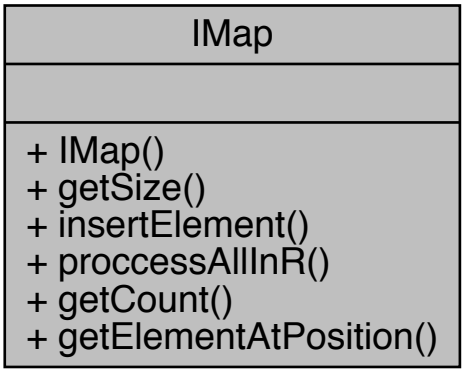
2.9 IMap Class Reference

```
#include "imap.h"
```

Inheritance diagram for IMap:



Collaboration diagram for IMap:



Public Member Functions

- IMap ()
- virtual QSizeF * getSize ()=0
- virtual void insertElement (IBaseGameElement *element, QVector3D point)=0
- virtual void proccessAllInR (IBaseGameElement *element, double r, bool(&mapOperator)(IBaseGame↩ Element *element))=0
- virtual int getCount ()=0
- virtual IBaseGameElement * getElementAtPosition (int pos)=0

2.9.1 Detailed Description

Definition at line 8 of file imap.h.

2.9.2 Constructor & Destructor Documentation

2.9.2.1 IMap()

```
IMap::IMap ( )
```

Definition at line 4 of file imap.cpp.

2.9.3 Member Function Documentation

2.9.3.1 getCount()

```
virtual int IMap::getCount ( )  [pure virtual]
```

Implemented in ListBseMap.

2.9.3.2 getElementAtPosition()

```
virtual IBaseGameElement* IMap::getElementAtPosition (
            int pos )  [pure virtual]
```

Implemented in ListBseMap.

2.9.3.3 getSize()

```
virtual QSizeF* IMap::getSize ( )  [pure virtual]
```

Implemented in ListBseMap.

2.9.3.4 insertElement()

```
virtual void IMap::insertElement (
            IBaseGameElement * element,
            QVector3D point )  [pure virtual]
```

Implemented in ListBseMap.

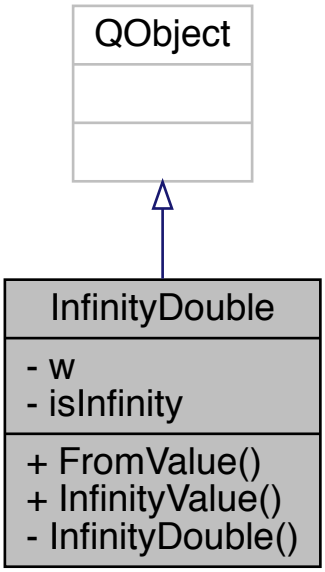
2.9.3.5 proccessAllInR()

```
virtual void IMap::proccessAllInR (
            IBaseGameElement * element,
            double r,
            bool(&)(IBaseGameElement *element) mapOperator )  [pure virtual]
```

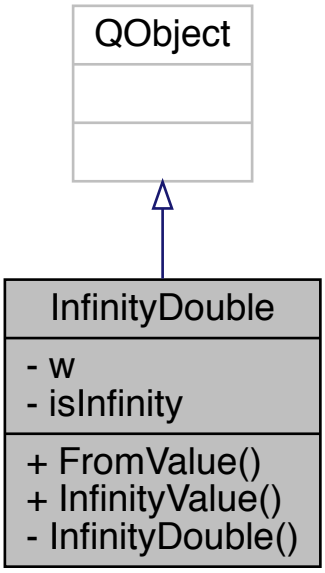
2.10 InfinityDouble Class Reference

```
#include "infinitydouble.h"
```

Inheritance diagram for InfinityDouble:



Collaboration diagram for InfinityDouble:



Static Public Member Functions

- static InfinityDouble * FromValue (double w)
- static InfinityDouble * InfinityValue ()

Private Member Functions

- InfinityDouble ()

Private Attributes

- double w
- bool isInfinity = false

Friends

- QDataStream & operator<< (QDataStream &stream, const InfinityDouble &myclass)
- QDataStream & operator>> (QDataStream &stream, InfinityDouble &myclass)

2.10.1 Detailed Description

Definition at line 8 of file infinitydouble.h.

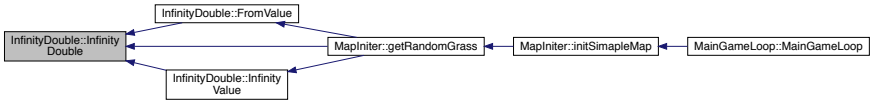
2.10.2 Constructor & Destructor Documentation

2.10.2.1 InfinityDouble()

```
InfinityDouble::InfinityDouble ( )  [private]
```

Definition at line 3 of file infinitydouble.cpp.

Here is the caller graph for this function:



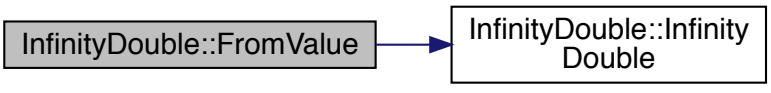
2.10.3 Member Function Documentation

2.10.3.1 FromValue()

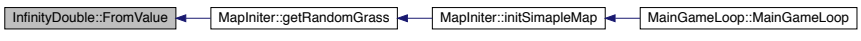
```
InfinityDouble * InfinityDouble::FromValue (
            double w )  [static]
```

Definition at line 8 of file infinitydouble.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

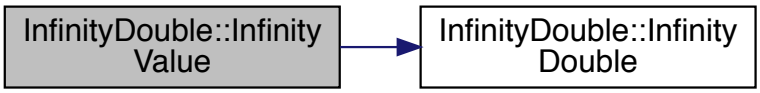


2.10.3.2 InfinityValue()

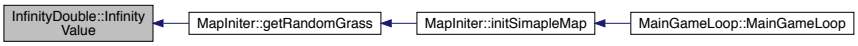
```
InfinityDouble * InfinityDouble::InfinityValue ( )  [static]
```

Definition at line 16 of file infinitydouble.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:



2.10.4 Friends And Related Function Documentation

2.10.4.1 operator<<

```
QDataStream& operator<< (
            QDataStream & stream,
            const InfinityDouble & myclass )  [friend]
```

Definition at line 12 of file infinitydouble.h.

2.10.4.2 operator>>

```
QDataStream& operator>> (
            QDataStream & stream,
            InfinityDouble & myclass )  [friend]
```

Definition at line 17 of file infinitydouble.h.

2.10.5 Member Data Documentation

2.10.5.1 isInfinity

```
bool InfinityDouble::isInfinity = false  [private]
```

Definition at line 25 of file infinitydouble.h.

2.10.5.2 w

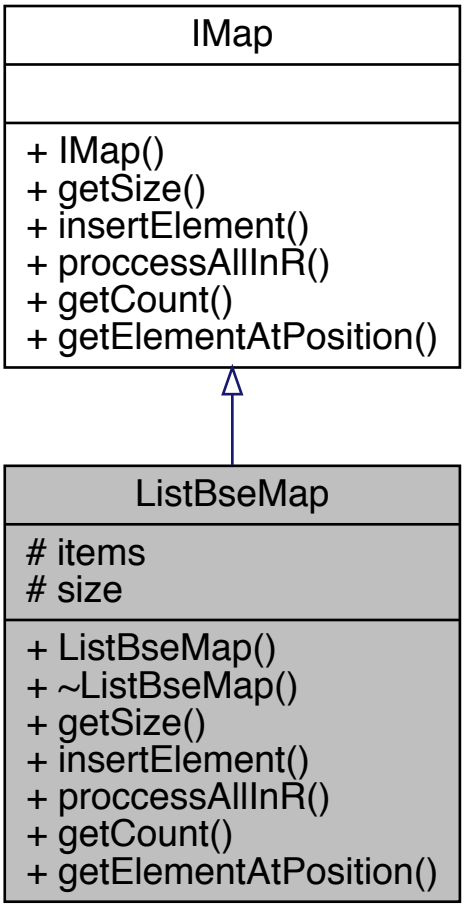
```
double InfinityDouble::w  [private]
```

Definition at line 24 of file infinitydouble.h.

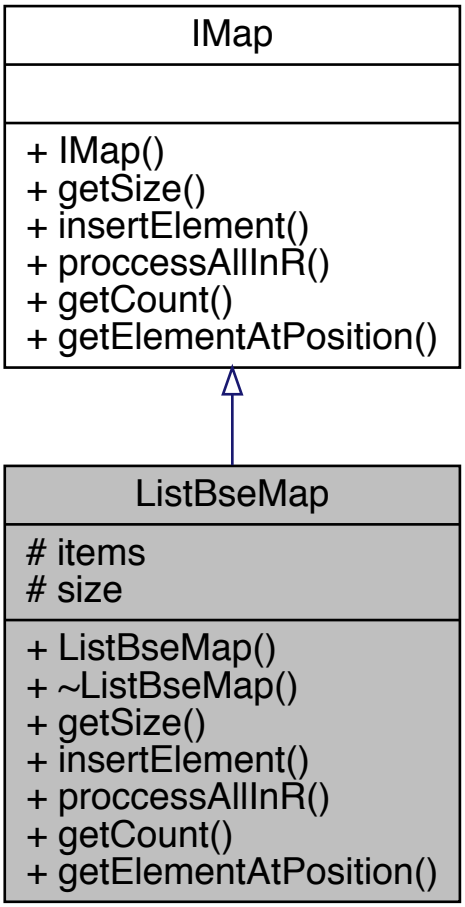
2.11 ListBseMap Class Reference

```
#include "listbsemap.h"
```

Inheritance diagram for ListBseMap:



Collaboration diagram for ListBseMap:



Public Member Functions

- ListBseMap (double width, double heigth)
- ~ListBseMap ()
- QSizeF * getSize ()
- void insertElement (IBaseGameElement *element, QVector3D point)
- void proccessAllInR (IBaseGameElement *element, double r, bool(&mapOperator)(IBaseGameElement *))
- int getCount ()
- IBaseGameElement * getElementAtPosition (int pos)
- virtual void proccessAllInR (IBaseGameElement *element, double r, bool(&mapOperator)(IBaseGameElement *element))=0

Protected Attributes

- QList< IBaseGameElement * > * items
- QSizeF * size

2.11.1 Detailed Description

Definition at line 7 of file listbsemap.h.

2.11.2 Constructor & Destructor Documentation

2.11.2.1 ListBseMap()

```
ListBseMap::ListBseMap (
            double width,
            double heigth )
```

Definition at line 4 of file listbsemap.cpp.

2.11.2.2 ~ListBseMap()

```
ListBseMap::~ListBseMap ( )
```

Definition at line 9 of file listbsemap.cpp.

2.11.3 Member Function Documentation

2.11.3.1 getCount()

```
int ListBseMap::getCount ( )  [virtual]
```

Implements IMap.

Definition at line 32 of file listbsemap.cpp.

2.11.3.2 getElementAtPosition()

```
IBaseGameElement * ListBseMap::getElementAtPosition (
            int pos )  [virtual]
```

Implements IMap.

Definition at line 36 of file listbsemap.cpp.

2.11.3.3 getSize()

```
QSizeF * ListBseMap::getSize ( )  [virtual]
```

Implements IMap.

Definition at line 14 of file listbsemap.cpp.

2.11.3.4 insertElement()

```
void ListBseMap::insertElement (
            IBaseGameElement * element,
            QVector3D point )  [virtual]
```

Implements IMap.

Definition at line 18 of file listbsemap.cpp.

2.11.3.5 proccessAllInR() [1/2]

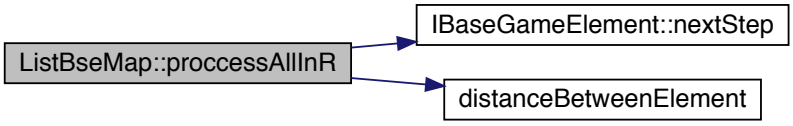
```
virtual void IMap::proccessAllInR (
            IBaseGameElement * element,
            double r,
            bool(&)(IBaseGameElement *element) mapOperator )  [pure virtual], [inherited]
```

2.11.3.6 proccessAllInR() [2/2]

```
void ListBseMap::proccessAllInR (
            IBaseGameElement * element,
            double r,
            bool(&)(IBaseGameElement *) mapOperator )
```

Definition at line 22 of file listbsemap.cpp.

Here is the call graph for this function:



2.11.4 Member Data Documentation

2.11.4.1 items

```
QList<IBaseGameElement*>* ListBseMap::items  [protected]
```

Definition at line 22 of file listbsemap.h.

2.11.4.2 size

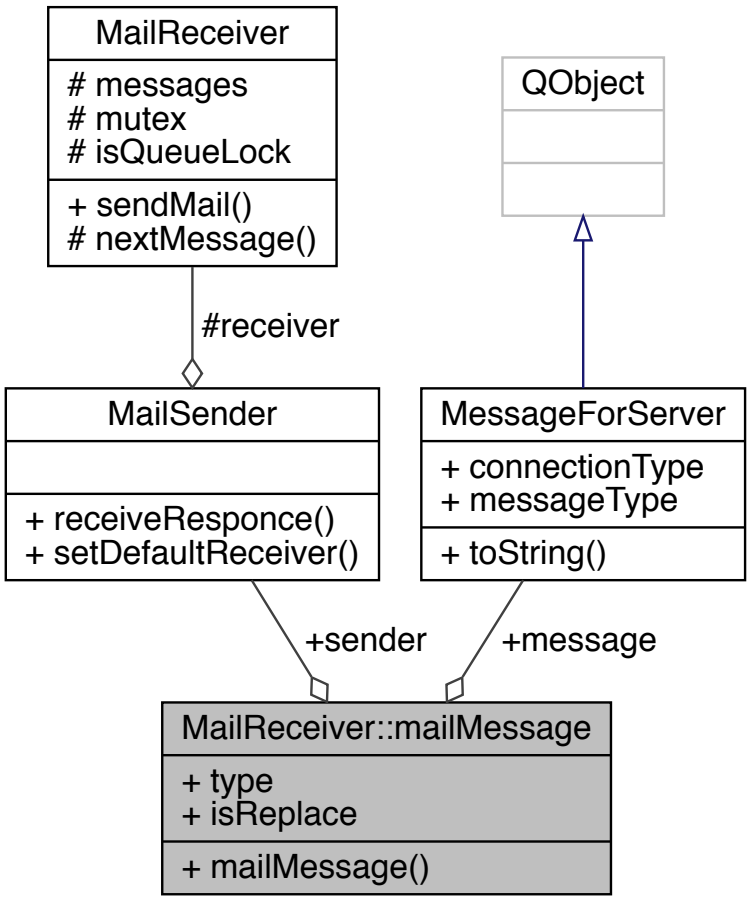
```
QSizeF* ListBseMap::size  [protected]
```

Definition at line 23 of file listbsemap.h.

2.12 MailReceiver::mailMessage Class Reference

```
#include "mailboxelement.h"
```

Collaboration diagram for MailReceiver::mailMessage:



Public Member Functions

- mailMessage (MessageForServer *message, MailSender *sender, int type, bool isReplace=true)

Public Attributes

- MessageForServer * message
- MailSender * sender
- int type
- bool isReplace

Friends

- QDebug operator<< (QDebug debug, const mailMessage &c)

2.12.1 Detailed Description

Definition at line 51 of file mailboxelement.h.

2.12.2 Constructor & Destructor Documentation

2.12.2.1 mailMessage()

```
MailReceiver::mailMessage::mailMessage (
            MessageForServer * message,
            MailSender * sender,
            int type,
            bool isReplace = true )  [inline]
```

Definition at line 57 of file mailboxelement.h.

2.12.3 Friends And Related Function Documentation

2.12.3.1 operator<<

```
QDebug operator<< (
            QDebug debug,
            const mailMessage & c )  [friend]
```

Definition at line 66 of file mailboxelement.h.

2.12.4 Member Data Documentation

2.12.4.1 isReplace

```
bool MailReceiver::mailMessage::isReplace
```

Definition at line 56 of file mailboxelement.h.

2.12.4.2 message

```
MessageForServer* MailReceiver::mailMessage::message
```

Definition at line 53 of file mailboxelement.h.

2.12.4.3 sender

```
MailSender* MailReceiver::mailMessage::sender
```

Definition at line 54 of file mailboxelement.h.

2.12.4.4 type

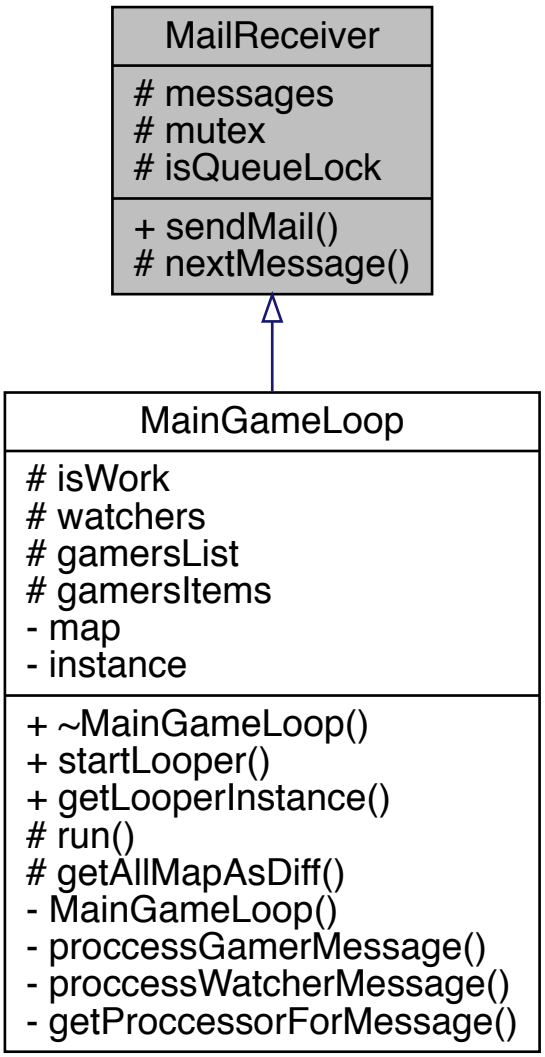
```
int MailReceiver::mailMessage::type
```

Definition at line 55 of file mailboxelement.h.

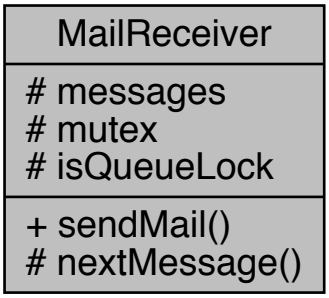
2.13 MailReceiver Class Reference

```
#include "mailboxelement.h"
```

Inheritance diagram for MailReceiver:



Collaboration diagram for MailReceiver:



Classes

- class mailMessage

Public Member Functions

- virtual void sendMail (MessageForServer *message, MailSender *sender, int type, bool isReplace=true)

Protected Member Functions

- virtual mailMessage * nextMessage ()

Protected Attributes

- QQueue< mailMessage * > messages
- QMutex mutex
- volatile bool isQueueLock = false

2.13.1 Detailed Description

Definition at line 16 of file mailboxelement.h.

2.13.2 Member Function Documentation

2.13.2.1 nextMessage()

```
virtual mailMessage* MailReceiver::nextMessage ( )  [inline], [protected], [virtual]
```

Definition at line 82 of file mailboxelement.h.

2.13.2.2 sendMail()

```
virtual void MailReceiver::sendMail (
            MessageForServer * message,
            MailSender * sender,
            int type,
            bool isReplace = true )  [inline], [virtual]
```

Definition at line 18 of file mailboxelement.h.

2.13.3 Member Data Documentation

2.13.3.1 isQueueLock

```
volatile bool MailReceiver::isQueueLock = false  [protected]
```

Definition at line 80 of file mailboxelement.h.

2.13.3.2 messages

```
QQueue<mailMessage*> MailReceiver::messages  [protected]
```

Definition at line 78 of file mailboxelement.h.

2.13.3.3 mutex

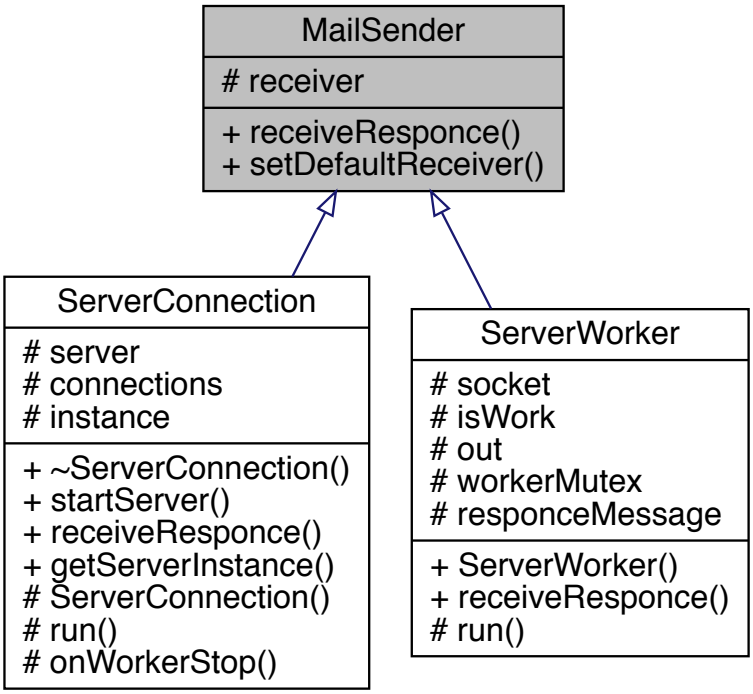
```
QMutex MailReceiver::mutex  [protected]
```

Definition at line 79 of file mailboxelement.h.

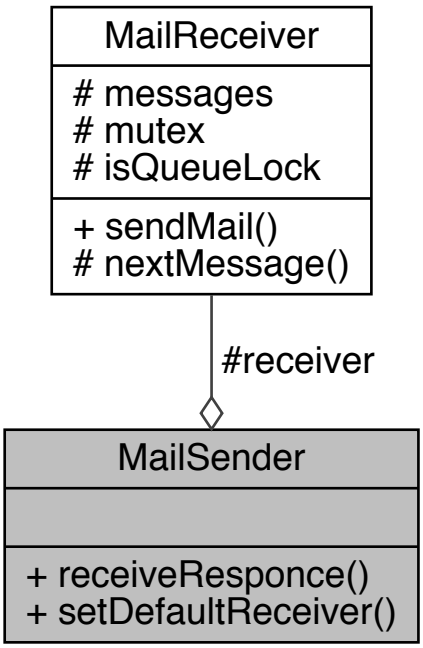
2.14 MailSender Class Reference

```
#include "mailboxelement.h"
```

Inheritance diagram for MailSender:



Collaboration diagram for MailSender:



Public Member Functions

- virtual void receiveResponce (QList< DiffElement *> *diff, MessageForServer *message)=0
- virtual void setDefaultReceiver (MailReceiver *receiver)

Protected Attributes

- MailReceiver * receiver

2.14.1 Detailed Description

Definition at line 96 of file mailboxelement.h.

2.14.2 Member Function Documentation

2.14.2.1 receiveResponce()

```
virtual void MailSender::receiveResponce (
            QList< DiffElement *> * diff,
            MessageForServer * message )  [pure virtual]
```

Implemented in ServerWorker, and ServerConnection.

2.14.2.2 setDefaultReceiver()

```
virtual void MailSender::setDefaultReceiver (
            MailReceiver * receiver )  [inline], [virtual]
```

Definition at line 100 of file mailboxelement.h.

2.14.3 Member Data Documentation

2.14.3.1 receiver

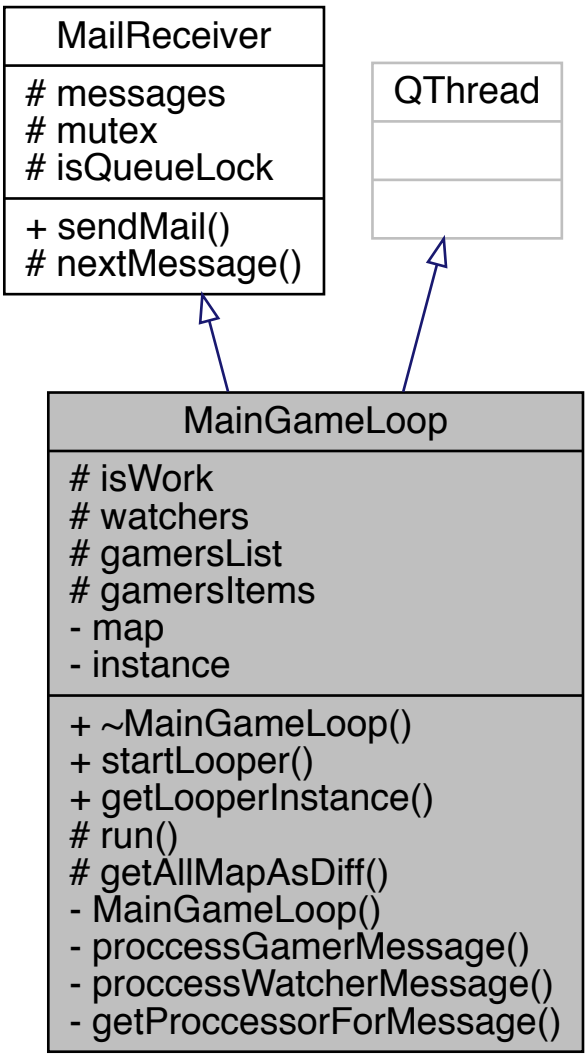
```
MailReceiver* MailSender::receiver  [protected]
```

Definition at line 105 of file mailboxelement.h.

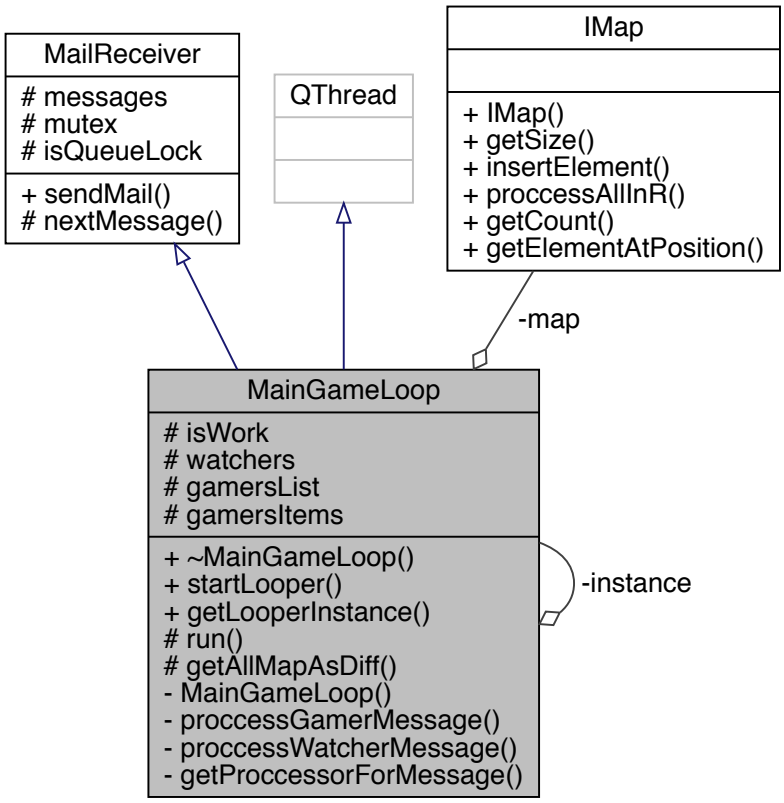
2.15 MainGameLoop Class Reference

```
#include "maingameloop.h"
```

Inheritance diagram for MainGameLoop:



Collaboration diagram for MainGameLoop:



Classes

- class GamerInformation

Public Member Functions

- ~MainGameLoop ()
- void startLooper ()
- virtual void sendMail (MessageForServer *message, MailSender *sender, int type, bool isReplace=true)

Static Public Member Functions

- static MainGameLoop * getLooperInstance ()

Protected Member Functions

- void run ()
- QList< DiffElement * > * getAllMapAsDiff ()
- virtual mailMessage * nextMessage ()

Protected Attributes

- bool isWork = true
- QList< MailSender * > watchers
- QList< GamerInformation * > gamersList

  gamersList
- QList< QList< IBaseGameElement * > * > gamersItems

  gamersItems
- QQueue< mailMessage * > messages
- QMutex mutex
- volatile bool isQueueLock = false

Private Types

- typedef void(MainGameLoop::* messageProccessor) (MailReceiver::mailMessage *, MailSender *)

Private Member Functions

- MainGameLoop ()
- void proccessGamerMessage (mailMessage *msg, MailSender *receiver)
- void proccessWatcherMessage (mailMessage *msg, MailSender *receiver)
- messageProccessor getProccessorForMessage (eConnectionType type)

Private Attributes

- IMap * map

Static Private Attributes

- static MainGameLoop * instance = nullptr

2.15.1 Detailed Description

Definition at line 12 of file maingameloop.h.

2.15.2 Member Typedef Documentation

2.15.2.1 messageProccessor

```
typedef void(MainGameLoop::* MainGameLoop::messageProccessor) (MailReceiver::mailMessage *,
MailSender *)  [private]
```

Definition at line 25 of file maingameloop.h.

2.15.3 Constructor & Destructor Documentation

2.15.3.1 ~MainGameLoop()

```
MainGameLoop::~MainGameLoop ( )
```

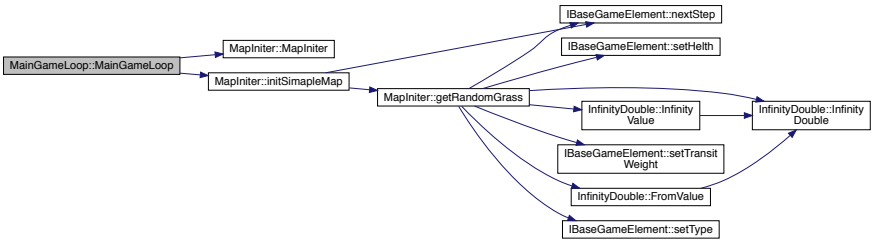
Definition at line 91 of file maingameloop.cpp.

2.15.3.2 MainGameLoop()

```
MainGameLoop::MainGameLoop ( )  [private]
```

Definition at line 8 of file maingameloop.cpp.

Here is the call graph for this function:



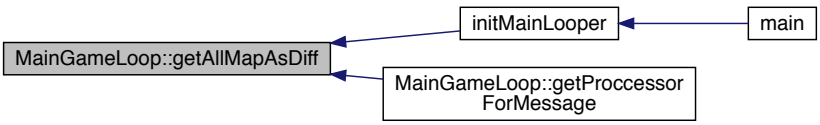
2.15.4 Member Function Documentation

2.15.4.1 getAllMapAsDiff()

```
QList< DiffElement * > * MainGameLoop::getAllMapAsDiff ( )  [protected]
```

Definition at line 82 of file maingameloop.cpp.

Here is the caller graph for this function:

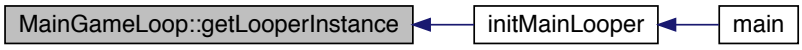


2.15.4.2 getLooperInstance()

```
static MainGameLoop* MainGameLoop::getLooperInstance ( )  [inline], [static]
```

Definition at line 15 of file maingameloop.h.

Here is the caller graph for this function:

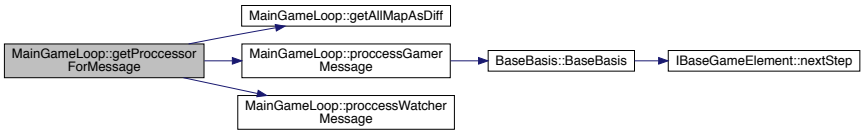


2.15.4.3 getProccessorForMessage()

```
MainGameLoop::messageProccessor MainGameLoop::getProccessorForMessage (
            eConnectionType type )  [private]
```

Definition at line 57 of file maingameloop.cpp.

Here is the call graph for this function:



2.15.4.4 nextMessage()

```
virtual mailMessage* MailReceiver::nextMessage ( )  [inline], [protected], [virtual], [inherited]
```

Definition at line 82 of file mailboxelement.h.

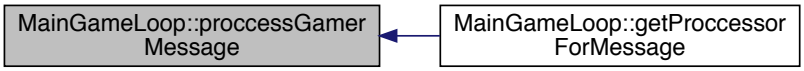
2.15.4.5 proccessGamerMessage()

```
void MainGameLoop::proccessGamerMessage (
            MailReceiver::mailMessage * msg,
            MailSender * receiver )  [private]
```

Definition at line 11 of file maingameloop.cpp.

Here is the call graph for this function:

Here is the caller graph for this function:

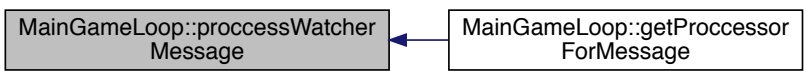


2.15.4.6 processWatcherMessage()

```
void MainGameLoop::proccessWatcherMessage (
            MailReceiver::mailMessage * msg,
            MailSender * receiver )  [private]
```

Definition at line 39 of file maingameloop.cpp.

Here is the caller graph for this function:



2.15.4.7 run()

```
void MainGameLoop::run ( )  [protected]
```

Definition at line 69 of file maingameloop.cpp.

2.15.4.8 sendMail()

```
virtual void MailReceiver::sendMail (
            MessageForServer * message,
            MailSender * sender,
            int type,
            bool isReplace = true )  [inline], [virtual], [inherited]
```

Definition at line 18 of file mailboxelement.h.

2.15.4.9 startLooper()

```
void MainGameLoop::startLooper ( )
```

Definition at line 98 of file maingameloop.cpp.

Here is the caller graph for this function:

2.15.5 Member Data Documentation

2.15.5.1 gamersItems

```
QList<QList<IBaseGameElement*>*> MainGameLoop::gamersItems  [protected]
```

gamersItems

list of gamer object; Each list represent game element of each gamer; firs element of each gamer object is basis

Definition at line 62 of file maingameloop.h.

2.15.5.2 gamersList

```
QList<GamerInformation*> MainGameLoop::gamersList  [protected]
```

gamersList

list of client as gamers

Definition at line 56 of file maingameloop.h.

2.15.5.3 instance

```
MainGameLoop * MainGameLoop::instance = nullptr  [static], [private]
```

Definition at line 27 of file maingameloop.h.

2.15.5.4 isQueueLock

```
volatile bool MailReceiver::isQueueLock = false  [protected], [inherited]
```

Definition at line 80 of file mailboxelement.h.

2.15.5.5 isWork

```
bool MainGameLoop::isWork = true  [protected]
```

Definition at line 49 of file maingameloop.h.

2.15.5.6 map

```
IMap* MainGameLoop::map  [private]
```

Definition at line 28 of file maingameloop.h.

2.15.5.7 messages

```
QQueue<mailMessage*> MailReceiver::messages  [protected], [inherited]
```

Definition at line 78 of file mailboxelement.h.

2.15.5.8 mutex

```
QMutex MailReceiver::mutex  [protected], [inherited]
```

Definition at line 79 of file mailboxelement.h.

2.15.5.9 watchers

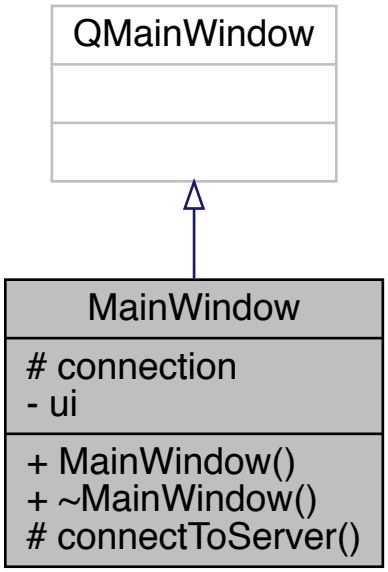
```
QList<MailSender*> MainGameLoop::watchers  [protected]
```

Definition at line 51 of file maingameloop.h.

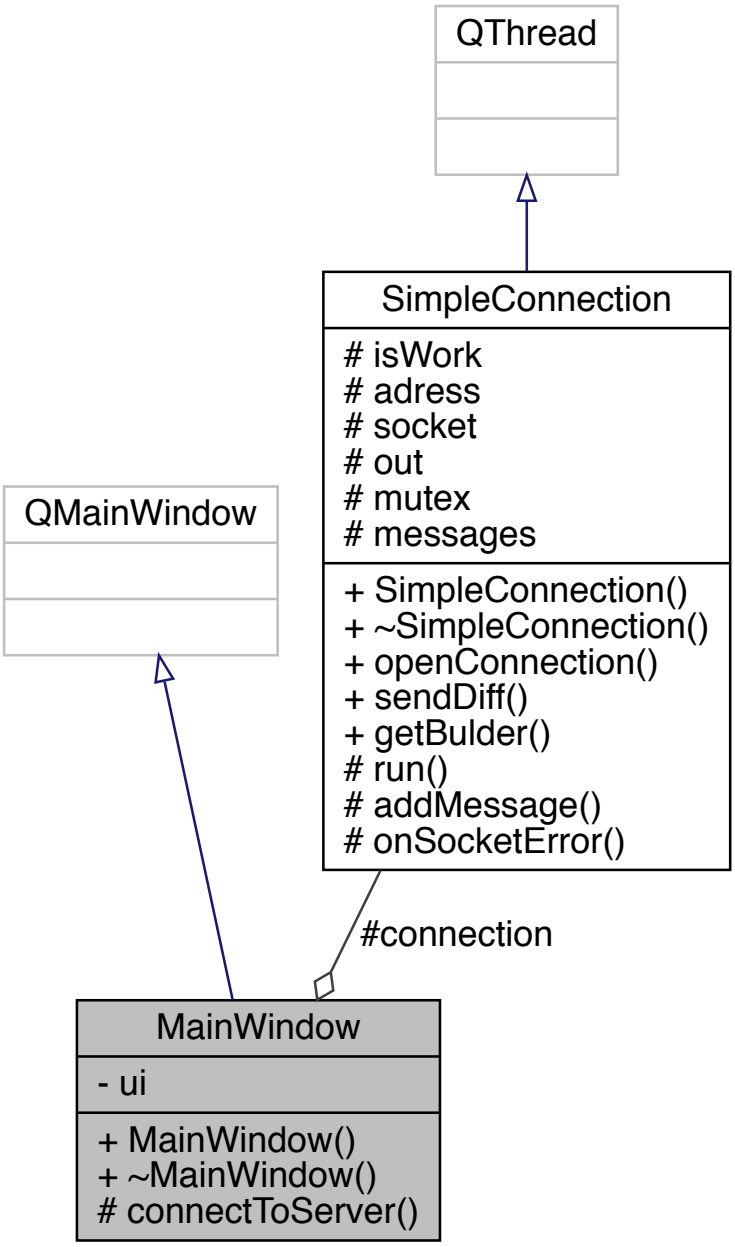
2.16 MainWindow Class Reference

```
#include "mainwindow.h"
```

Inheritance diagram for MainWindow:



Collaboration diagram for MainWindow:



Public Member Functions

- MainWindow (QWidget *parent=0)
- ~MainWindow ()

Protected Member Functions

- void connectToServer ()

Protected Attributes

- SimpleConnection connection

Private Attributes

- Ui::MainWindow * ui

2.16.1 Detailed Description

Definition at line 11 of file mainwindow.h.

2.16.2 Constructor & Destructor Documentation

2.16.2.1 MainWindow()

```
MainWindow::MainWindow (
            QWidget * parent = 0 )  [explicit]
```

Definition at line 4 of file mainwindow.cpp.

Here is the call graph for this function:

Here is the caller graph for this function:

2.16.2.2 ~MainWindow()

```
MainWindow::~MainWindow ( )
```

Definition at line 12 of file mainwindow.cpp.

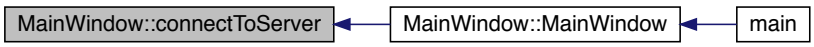
2.16.3 Member Function Documentation

2.16.3.1 connectToServer()

```
void MainWindow::connectToServer ( )  [protected]
```

Definition at line 16 of file mainwindow.cpp.

Here is the caller graph for this function:



2.16.4 Member Data Documentation

2.16.4.1 connection

```
SimpleConnection MainWindow::connection  [protected]
```

Definition at line 20 of file mainwindow.h.

2.16.4.2 ui

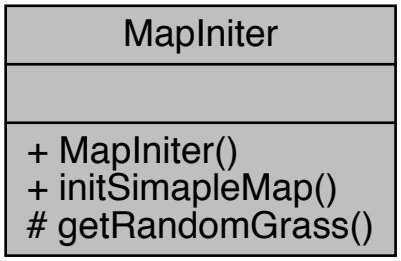
```
Ui::MainWindow* MainWindow::ui  [private]
```

Definition at line 23 of file mainwindow.h.

2.17 MapIniter Class Reference

`#include "mapiniter.h"`

Collaboration diagram for MapIniter:



Public Member Functions

- MapIniter ()
- IMap ∗ initSimapleMap ()

Protected Member Functions

- IBaseGameElement ∗ getRandomGrass (double maxWidth, double maxHeigth)

2.17.1 Detailed Description

Definition at line 8 of file mapiniter.h.

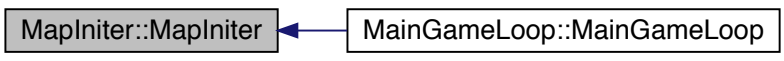
2.17.2 Constructor & Destructor Documentation

2.17.2.1 MapIniter()

`MapIniter::MapIniter ( )`

Definition at line 3 of file mapiniter.cpp.

Here is the caller graph for this function:



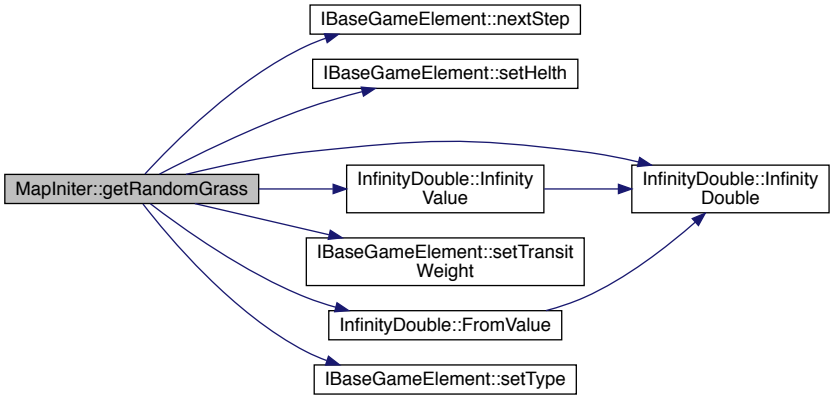
2.17.3 Member Function Documentation

2.17.3.1 getRandomGrass()

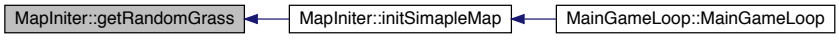
```
IBaseGameElement * MapIniter::getRandomGrass (
            double maxWidth,
            double maxHeigth )  [protected]
```

Definition at line 17 of file mapiniter.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

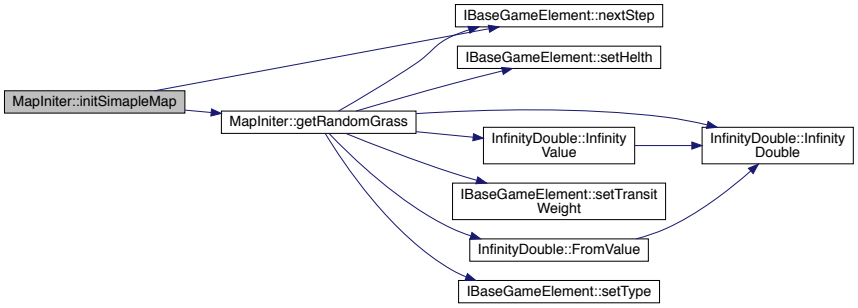


2.17.3.2 initSimapleMap()

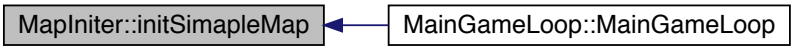
```
IMap * MapIniter::initSimapleMap ( )
```

Definition at line 5 of file mapiniter.cpp.

Here is the call graph for this function:



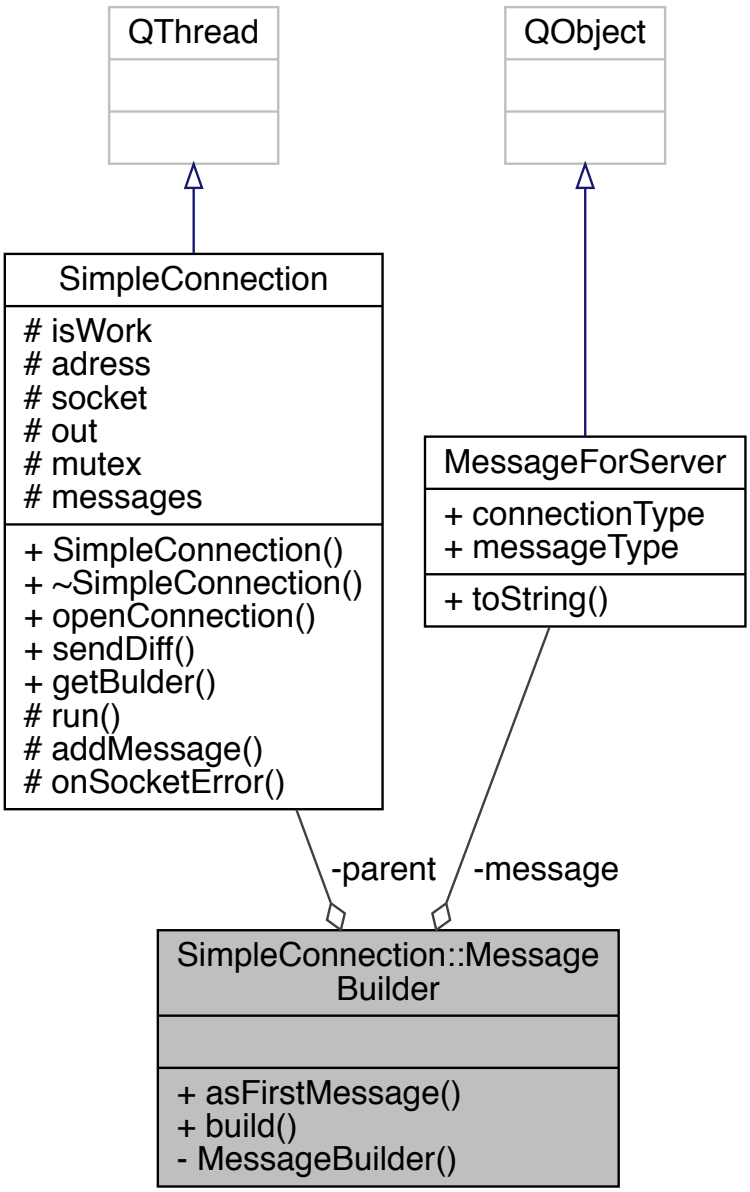
Here is the caller graph for this function:



2.18 SimpleConnection::MessageBuilder Class Reference

```
#include "simpleconnection.h"
```

Collaboration diagram for SimpleConnection::MessageBuilder:



Public Member Functions

- MessageBuilder * asFirstMessage (eConnectionType type)
- void build ()

Private Member Functions

- MessageBuilder (SimpleConnection *sender)

Private Attributes

- MessageForServer * message
- SimpleConnection * parent

Friends

- class SimpleConnection

2.18.1 Detailed Description

Definition at line 58 of file simpleconnection.h.

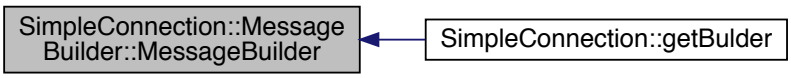
2.18.2 Constructor & Destructor Documentation

2.18.2.1 MessageBuilder()

```
SimpleConnection::MessageBuilder::MessageBuilder (
            SimpleConnection * sender )  [private]
```

Definition at line 71 of file simpleconnection.cpp.

Here is the caller graph for this function:



2.18.3 Member Function Documentation

2.18.3.1 asFirstMessage()

```
SimpleConnection::MessageBuilder * SimpleConnection::MessageBuilder::asFirstMessage (
            eConnectionType type )
```

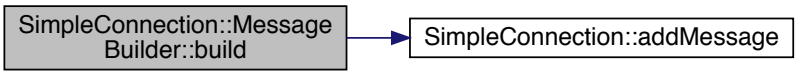
Definition at line 77 of file simpleconnection.cpp.

2.18.3.2 build()

```
void SimpleConnection::MessageBuilder::build ( )
```

Definition at line 83 of file simpleconnection.cpp.

Here is the call graph for this function:



2.18.4 Friends And Related Function Documentation

2.18.4.1 SimpleConnection

```
friend class SimpleConnection  [friend]
```

Definition at line 59 of file simpleconnection.h.

2.18.5 Member Data Documentation

2.18.5.1 message

```
MessageForServer* SimpleConnection::MessageBuilder::message  [private]
```

Definition at line 68 of file simpleconnection.h.

2.18.5.2 parent

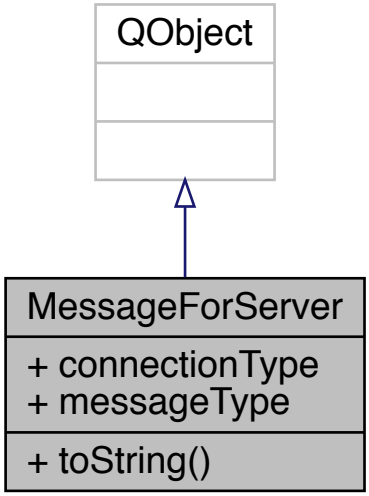
```
SimpleConnection* SimpleConnection::MessageBuilder::parent  [private]
```

Definition at line 69 of file simpleconnection.h.

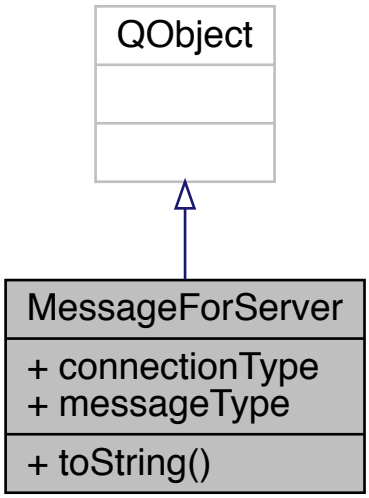
2.19 MessageForServer Class Reference

```
#include "simpleconnection.h"
```

Inheritance diagram for MessageForServer:



Collaboration diagram for MessageForServer:



Public Member Functions

- QString toString ()

Public Attributes

- eConnectionType connectionType
- eMessageType messageType

Friends

- QDataStream & operator<< (QDataStream &stream, const MessageForServer &myclass)
- QDataStream & operator>> (QDataStream &stream, MessageForServer &myclass)
- QDebug operator<< (QDebug debug, MessageForServer &c)

2.19.1 Detailed Description

Definition at line 18 of file simpleconnection.h.

2.19.2 Member Function Documentation

2.19.2.1 toString()

```
QString MessageForServer::toString ( )
```

Definition at line 91 of file simpleconnection.cpp.

2.19.3 Friends And Related Function Documentation

2.19.3.1 operator<< [1/2]

```
QDataStream& operator<< (
            QDataStream & stream,
            const MessageForServer & myclass )  [friend]
```

Definition at line 25 of file simpleconnection.h.

2.19.3.2 operator<< [2/2]

```
QDebug operator<< (
            QDebug debug,
            MessageForServer & c )  [friend]
```

Definition at line 40 of file simpleconnection.h.

2.19.3.3 operator>>

```
QDataStream& operator>> (
            QDataStream & stream,
            MessageForServer & myclass )  [friend]
```

Definition at line 30 of file simpleconnection.h.

2.19.4 Member Data Documentation

2.19.4.1 connectionType

```
eConnectionType MessageForServer::connectionType
```

Definition at line 22 of file simpleconnection.h.

2.19.4.2 messageType

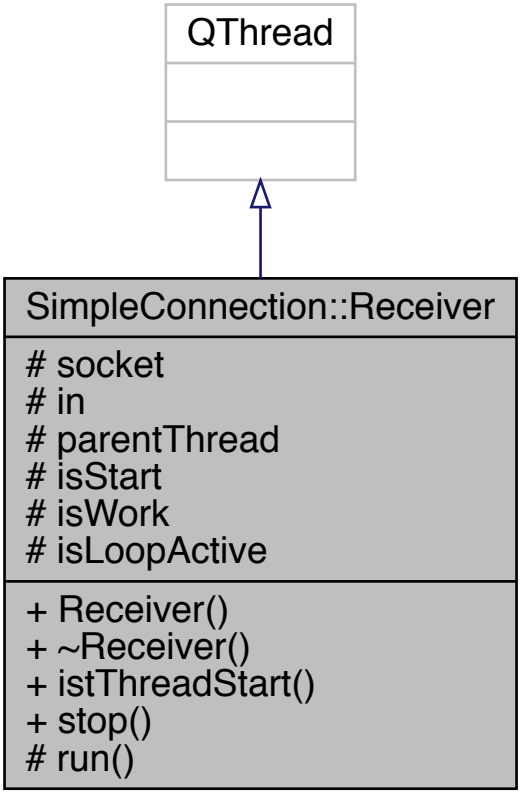
```
eMessageType MessageForServer::messageType
```

Definition at line 23 of file simpleconnection.h.

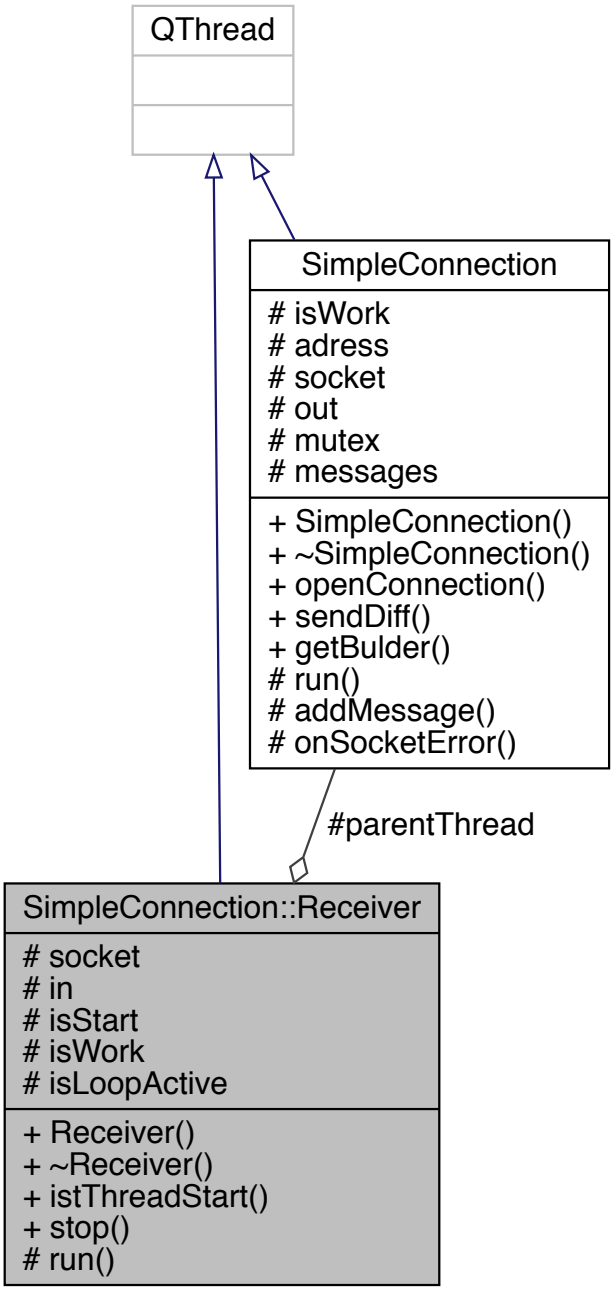
2.20 SimpleConnection::Receiver Class Reference

```
#include "simpleconnection.h"
```

Inheritance diagram for SimpleConnection::Receiver:



Collaboration diagram for SimpleConnection::Receiver:



Public Member Functions

- Receiver (QTcpSocket *socket, SimpleConnection *parentThread, QObject *parent=0)
- virtual ~Receiver ()
- bool istThreadStart ()
- void stop ()

Protected Member Functions

- void run ()

Protected Attributes

- QTcpSocket * socket
- QDataStream * in
- SimpleConnection * parentThread
- volatile bool isStart = false
- volatile bool isWork = true
- volatile bool isLoopActive = false

2.20.1 Detailed Description

Definition at line 96 of file simpleconnection.h.

2.20.2 Constructor & Destructor Documentation

2.20.2.1 Receiver()

```
SimpleConnection::Receiver::Receiver (
            QTcpSocket * socket,
            SimpleConnection * parentThread,
            QObject * parent = 0 )
```

Definition at line 116 of file simpleconnection.cpp.

2.20.2.2 ~Receiver()

```
SimpleConnection::Receiver::~Receiver ( )  [virtual]
```

Definition at line 125 of file simpleconnection.cpp.

2.20.3 Member Function Documentation

2.20.3.1 istThreadStart()

```
bool SimpleConnection::Receiver::istThreadStart ( )
```

Definition at line 129 of file simpleconnection.cpp.

2.20.3.2 run()

```
void SimpleConnection::Receiver::run ( )  [protected]
```

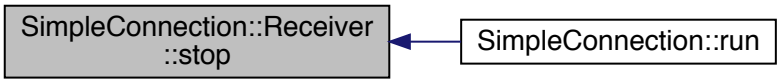
Definition at line 141 of file simpleconnection.cpp.

2.20.3.3 stop()

```
void SimpleConnection::Receiver::stop ( )
```

Definition at line 134 of file simpleconnection.cpp.

Here is the caller graph for this function:



2.20.4 Member Data Documentation

2.20.4.1 in

```
QDataStream* SimpleConnection::Receiver::in  [protected]
```

Definition at line 107 of file simpleconnection.h.

2.20.4.2 isLoopActive

```
volatile bool SimpleConnection::Receiver::isLoopActive = false  [protected]
```

Definition at line 111 of file simpleconnection.h.

2.20.4.3 isStart

```
volatile bool SimpleConnection::Receiver::isStart = false  [protected]
```

Definition at line 109 of file simpleconnection.h.

2.20.4.4 isWork

```
volatile bool SimpleConnection::Receiver::isWork = true  [protected]
```

Definition at line 110 of file simpleconnection.h.

2.20.4.5 parentThread

```
SimpleConnection* SimpleConnection::Receiver::parentThread  [protected]
```

Definition at line 108 of file simpleconnection.h.

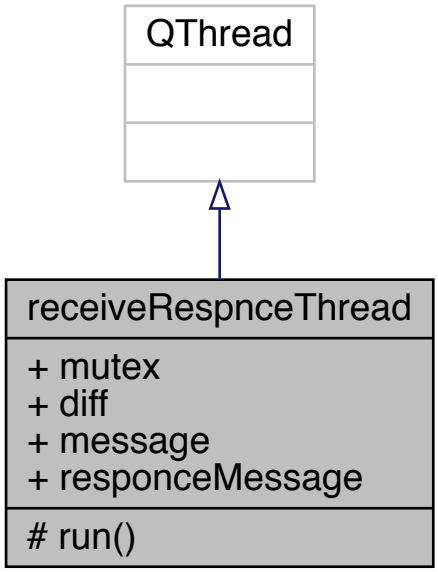
2.20.4.6 socket

```
QTcpSocket* SimpleConnection::Receiver::socket  [protected]
```

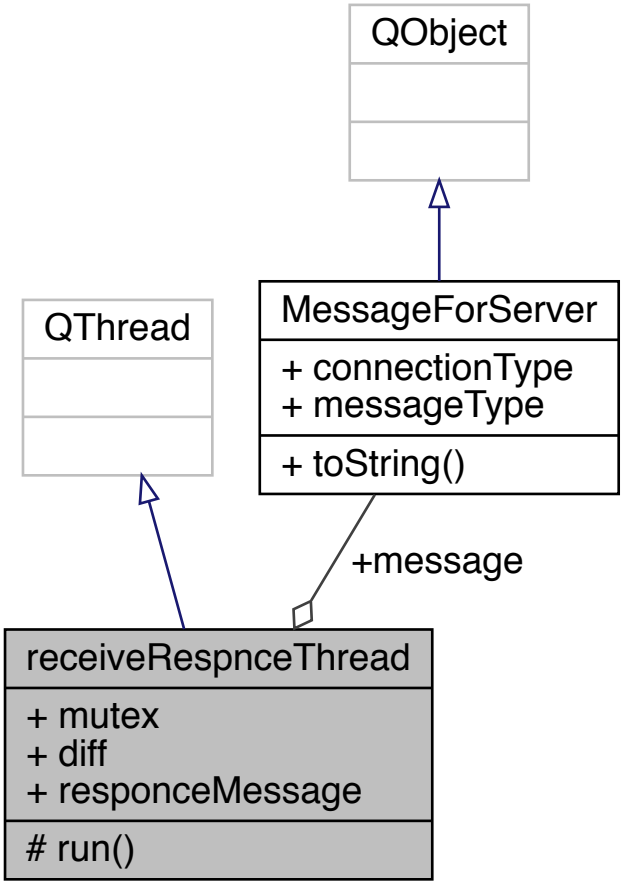
Definition at line 106 of file simpleconnection.h.

2.21 receiveRespnceThread Class Reference

Inheritance diagram for receiveRespnceThread:



Collaboration diagram for receiveRespnceThread:



Public Attributes

- QMutex * mutex
- QList< DiffElement * > * diff
- MessageForServer * message
- QQueue< ServerWorker::responceData > * responceMessage

Protected Member Functions

- void run ()

2.21.1 Detailed Description

Definition at line 45 of file serverworker.cpp.

2.21.2 Member Function Documentation

2.21.2.1 run()

```
void receiveRespnceThread::run ( )  [inline], [protected]
```

Definition at line 54 of file serverworker.cpp.

2.21.3 Member Data Documentation

2.21.3.1 diff

```
QList<DiffElement*>* receiveRespnceThread::diff
```

Definition at line 48 of file serverworker.cpp.

2.21.3.2 message

```
MessageForServer* receiveRespnceThread::message
```

Definition at line 49 of file serverworker.cpp.

2.21.3.3 mutex

```
QMutex* receiveRespnceThread::mutex
```

Definition at line 47 of file serverworker.cpp.

2.21.3.4 responceMessage

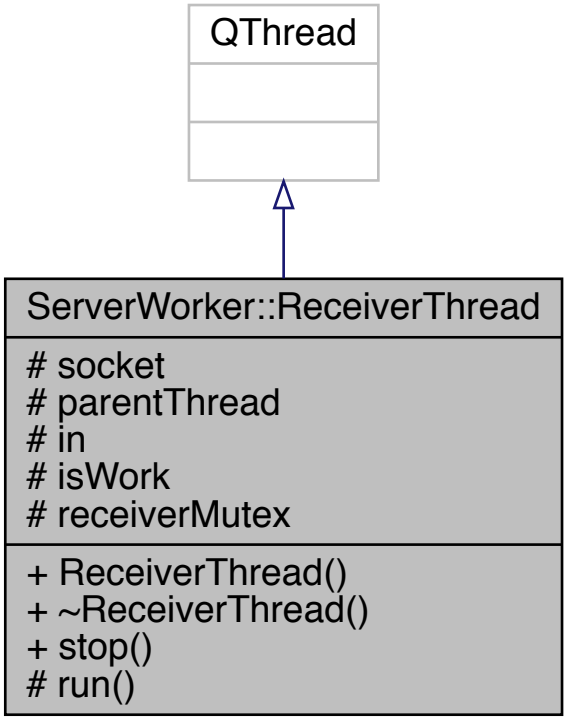
```
QQueue<ServerWorker::responceData>* receiveRespnceThread::responceMessage
```

Definition at line 50 of file serverworker.cpp.

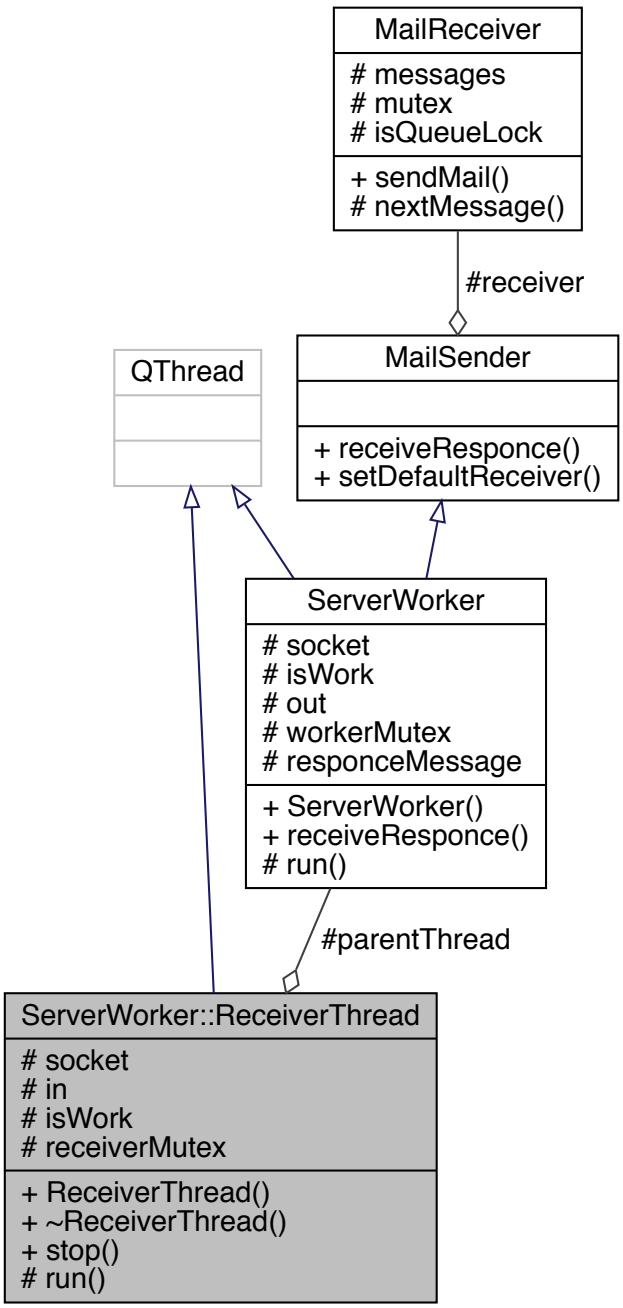
2.22 ServerWorker::ReceiverThread Class Reference

```
#include "serverworker.h"
```

Inheritance diagram for ServerWorker::ReceiverThread:



Collaboration diagram for ServerWorker::ReceiverThread:



Public Member Functions

- ReceiverThread (ServerWorker *parent, QTcpSocket *socket)
- virtual ~ReceiverThread ()
- void stop ()

Protected Member Functions

- void run ()

Protected Attributes

- QTcpSocket * socket
- ServerWorker * parentThread
- QDataStream * in
- bool isWork = true
- QMutex receiverMutex

2.22.1 Detailed Description

Definition at line 41 of file serverworker.h.

2.22.2 Constructor & Destructor Documentation

2.22.2.1 ReceiverThread()

```
ServerWorker::ReceiverThread::ReceiverThread (
            ServerWorker * parent,
            QTcpSocket * socket )
```

Definition at line 78 of file serverworker.cpp.

2.22.2.2 ~ReceiverThread()

```
ServerWorker::ReceiverThread::~ReceiverThread ( )  [virtual]
```

Definition at line 85 of file serverworker.cpp.

2.22.3 Member Function Documentation

2.22.3.1 run()

```
void ServerWorker::ReceiverThread::run ( )  [protected]
```

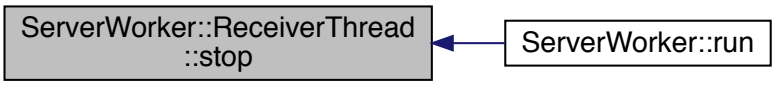
Definition at line 95 of file serverworker.cpp.

2.22.3.2 stop()

```
void ServerWorker::ReceiverThread::stop ( )
```

Definition at line 89 of file serverworker.cpp.

Here is the caller graph for this function:



2.22.4 Member Data Documentation

2.22.4.1 in

```
QDataStream* ServerWorker::ReceiverThread::in  [protected]
```

Definition at line 52 of file serverworker.h.

2.22.4.2 isWork

```
bool ServerWorker::ReceiverThread::isWork = true  [protected]
```

Definition at line 53 of file serverworker.h.

2.22.4.3 parentThread

```
ServerWorker* ServerWorker::ReceiverThread::parentThread  [protected]
```

Definition at line 51 of file serverworker.h.

2.22.4.4 receiverMutex

```
QMutex ServerWorker::ReceiverThread::receiverMutex  [protected]
```

Definition at line 54 of file serverworker.h.

2.22.4.5 socket

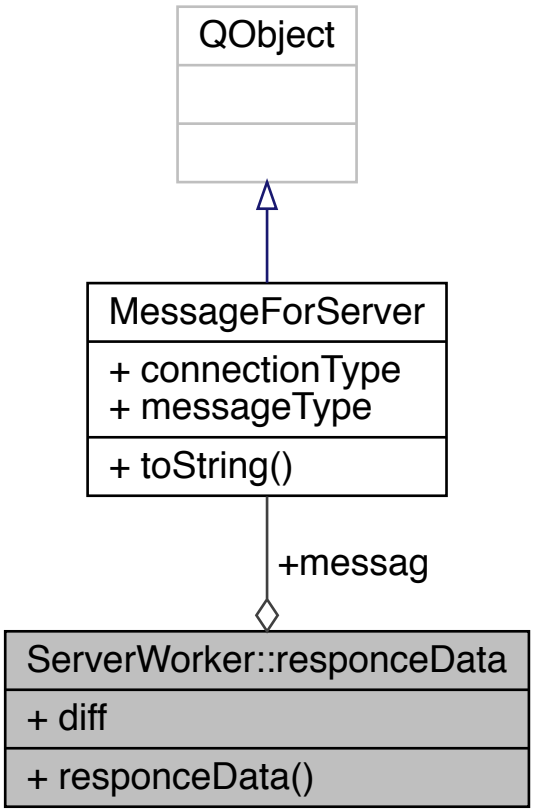
```
QTcpSocket* ServerWorker::ReceiverThread::socket  [protected]
```

Definition at line 50 of file serverworker.h.

2.23 ServerWorker::responceData Struct Reference

```
#include "serverworker.h"
```

Collaboration diagram for ServerWorker::responceData:



Public Member Functions

- responceData (QList< DiffElement *> *diff, MessageForServer *messag)

Public Attributes

- QList< DiffElement * > * diff
- MessageForServer * messag

2.23.1 Detailed Description

Definition at line 24 of file serverworker.h.

2.23.2 Constructor & Destructor Documentation

2.23.2.1 responceData()

```
ServerWorker::responceData::responceData (
            QList< DiffElement *> * diff,
            MessageForServer * messag )  [inline]
```

Definition at line 26 of file serverworker.h.

2.23.3 Member Data Documentation

2.23.3.1 diff

```
QList<DiffElement*>* ServerWorker::responceData::diff
```

Definition at line 31 of file serverworker.h.

2.23.3.2 messag

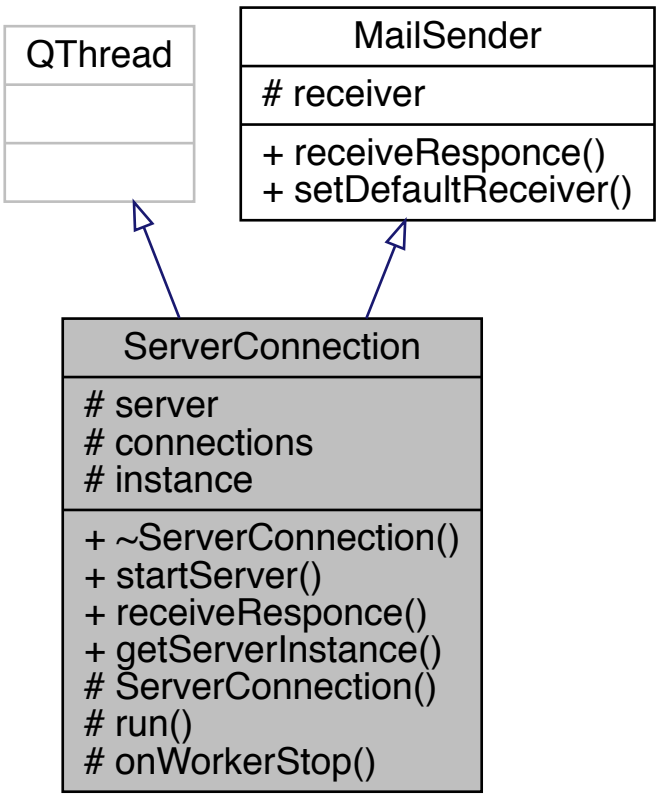
```
MessageForServer* ServerWorker::responceData::messag
```

Definition at line 32 of file serverworker.h.

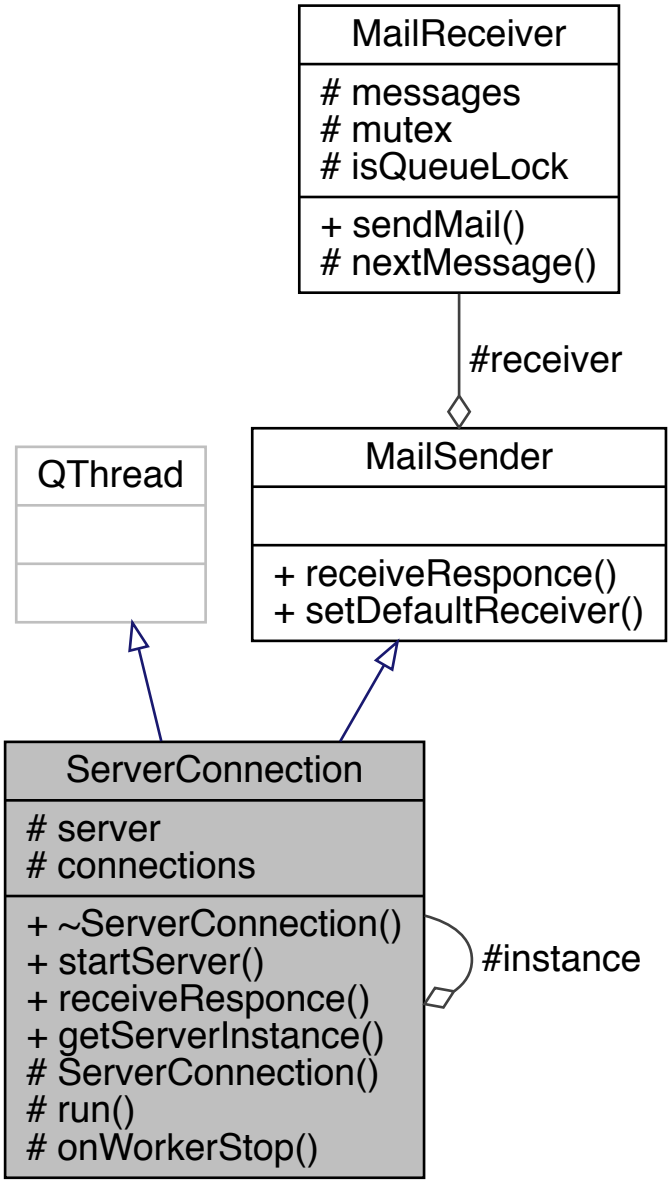
2.24 ServerConnection Class Reference

```
#include "serverconnection.h"
```

Inheritance diagram for ServerConnection:



Collaboration diagram for ServerConnection:



Signals

- void onServerError (serverError error)

Public Member Functions

- ~ServerConnection ()
- void startServer ()
- void receiveResponce (QList< DiffElement *> *diff, MessageForServer *message)
- virtual void setDefaultReceiver (MailReceiver *receiver)

Static Public Member Functions

- static ServerConnection * getServerInstance ()

Protected Slots

- void onWorkerStop (ServerWorker *worker)

Protected Member Functions

- ServerConnection ()
- void run ()

Protected Attributes

- QTcpServer * server
- QList< ServerWorker * > * connections
- MailReceiver * receiver

Static Protected Attributes

- static ServerConnection * instance = nullptr

2.24.1 Detailed Description

Definition at line 12 of file serverconnection.h.

2.24.2 Constructor & Destructor Documentation

2.24.2.1 ~ServerConnection()

```
ServerConnection::~ServerConnection ( )
```

Definition at line 6 of file serverconnection.cpp.

2.24.2.2 ServerConnection()

```
ServerConnection::ServerConnection ( )  [protected]
```

Definition at line 12 of file serverconnection.cpp.

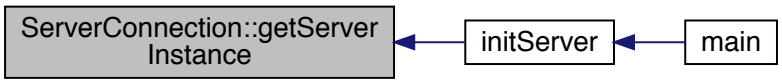
2.24.3 Member Function Documentation

2.24.3.1 getServerInstance()

```
static ServerConnection* ServerConnection::getServerInstance ( )  [inline], [static]
```

Definition at line 18 of file serverconnection.h.

Here is the caller graph for this function:



2.24.3.2 onServerError

```
void ServerConnection::onServerError (
            serverError error )  [signal]
```

2.24.3.3 onWorkerStop

```
void ServerConnection::onWorkerStop (
            ServerWorker * worker )  [protected], [slot]
```

Definition at line 16 of file serverconnection.cpp.

Here is the caller graph for this function:

2.24.3.4 receiveResponce()

```
void ServerConnection::receiveResponce (
            QList< DiffElement *> * diff,
            MessageForServer * message )  [virtual]
```

Implements MailSender.

Definition at line 51 of file serverconnection.cpp.

2.24.3.5 run()

```
void ServerConnection::run ( )  [protected]
```

Definition at line 27 of file serverconnection.cpp.

2.24.3.6 setDefaultReceiver()

```
virtual void MailSender::setDefaultReceiver (
            MailReceiver * receiver )  [inline], [virtual], [inherited]
```

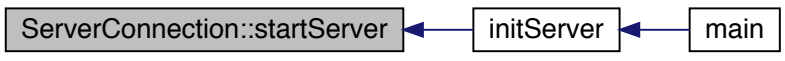
Definition at line 100 of file mailboxelement.h.

2.24.3.7 startServer()

```
void ServerConnection::startServer ( )
```

Definition at line 21 of file serverconnection.cpp.

Here is the caller graph for this function:



2.24.4 Member Data Documentation

2.24.4.1 connections

```
QList<ServerWorker*>* ServerConnection::connections  [protected]
```

Definition at line 32 of file serverconnection.h.

2.24.4.2 instance

```
ServerConnection * ServerConnection::instance = nullptr  [static], [protected]
```

Definition at line 29 of file serverconnection.h.

2.24.4.3 receiver

```
MailReceiver* MailSender::receiver  [protected], [inherited]
```

Definition at line 105 of file mailboxelement.h.

2.24.4.4 server

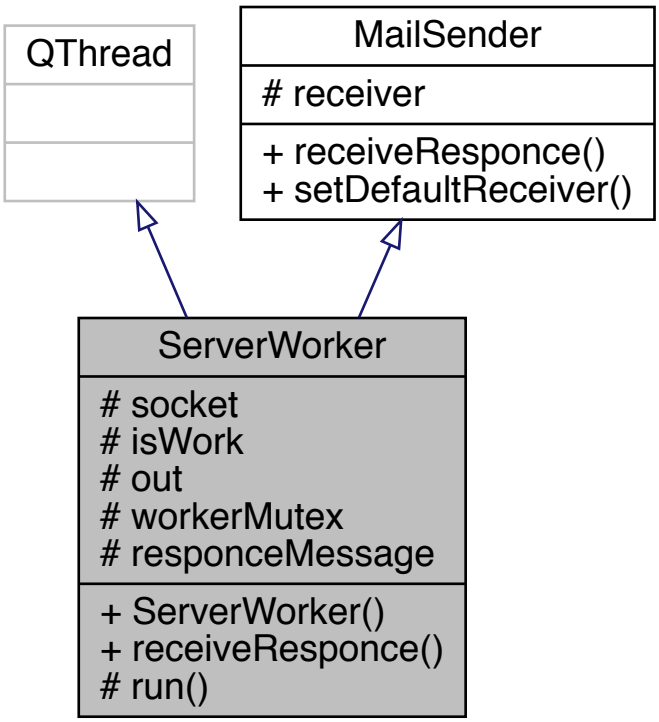
```
QTcpServer* ServerConnection::server  [protected]
```

Definition at line 31 of file serverconnection.h.

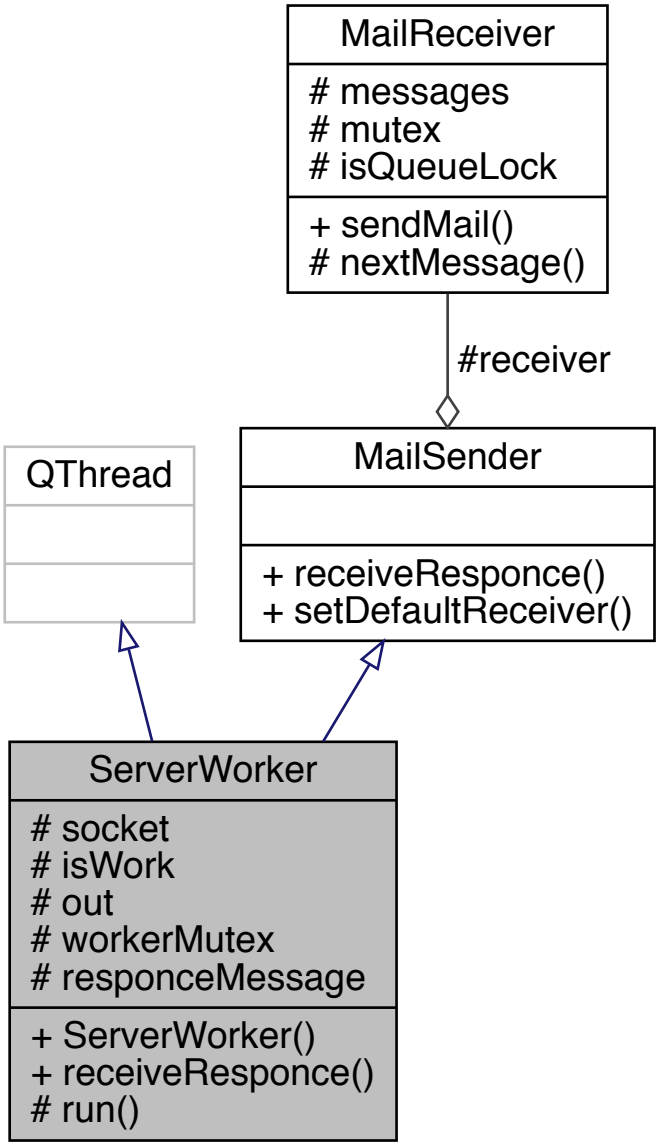
2.25 ServerWorker Class Reference

```
#include "serverworker.h"
```

Inheritance diagram for ServerWorker:



Collaboration diagram for ServerWorker:



Classes

- class ReceiverThread
- struct responceData

Signals

- void onStop (ServerWorker *worker)

Public Member Functions

- ServerWorker (QTcpSocket *socket)
- void receiveResponce (QList< DiffElement *> *diff, MessageForServer *message)
- virtual void setDefaultReceiver (MailReceiver *receiver)

Protected Member Functions

- void run ()

Protected Attributes

- QTcpSocket * socket
- volatile bool isWork = true
- QDataStream * out
- QMutex workerMutex
- QQueue< responceData > responceMessage
- MailReceiver * receiver

Friends

- class receiveRespnceThread

2.25.1 Detailed Description

Definition at line 10 of file serverworker.h.

2.25.2 Constructor & Destructor Documentation

2.25.2.1 ServerWorker()

```
ServerWorker::ServerWorker (
            QTcpSocket * socket )
```

Definition at line 6 of file serverworker.cpp.

2.25.3 Member Function Documentation

2.25.3.1 onStop

```
void ServerWorker::onStop (
            ServerWorker * worker )  [signal]
```

Here is the caller graph for this function:

2.25.3.2 receiveResponce()

```
void ServerWorker::receiveResponce (
            QList< DiffElement *> * diff,
            MessageForServer * message )  [virtual]
```

Implements MailSender.

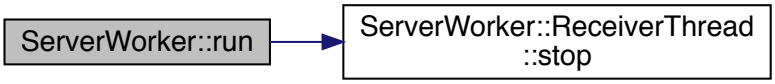
Definition at line 61 of file serverworker.cpp.

2.25.3.3 run()

```
void ServerWorker::run ( )  [protected]
```

Definition at line 10 of file serverworker.cpp.

Here is the call graph for this function:



2.25.3.4 setDefaultReceiver()

```
virtual void MailSender::setDefaultReceiver (
            MailReceiver * receiver )  [inline], [virtual], [inherited]
```

Definition at line 100 of file mailboxelement.h.

2.25.4 Friends And Related Function Documentation

2.25.4.1 receiveRespnceThread

```
friend class receiveRespnceThread  [friend]
```

Definition at line 12 of file serverworker.h.

2.25.5 Member Data Documentation

2.25.5.1 isWork

```
volatile bool ServerWorker::isWork = true  [protected]
```

Definition at line 36 of file serverworker.h.

2.25.5.2 out

```
QDataStream* ServerWorker::out  [protected]
```

Definition at line 37 of file serverworker.h.

2.25.5.3 receiver

```
MailReceiver* MailSender::receiver  [protected], [inherited]
```

Definition at line 105 of file mailboxelement.h.

2.25.5.4 responceMessage

```
QQueue<responceData> ServerWorker::responceMessage  [protected]
```

Definition at line 39 of file serverworker.h.

2.25.5.5 socket

```
QTcpSocket* ServerWorker::socket  [protected]
```

Definition at line 35 of file serverworker.h.

2.25.5.6 workerMutex

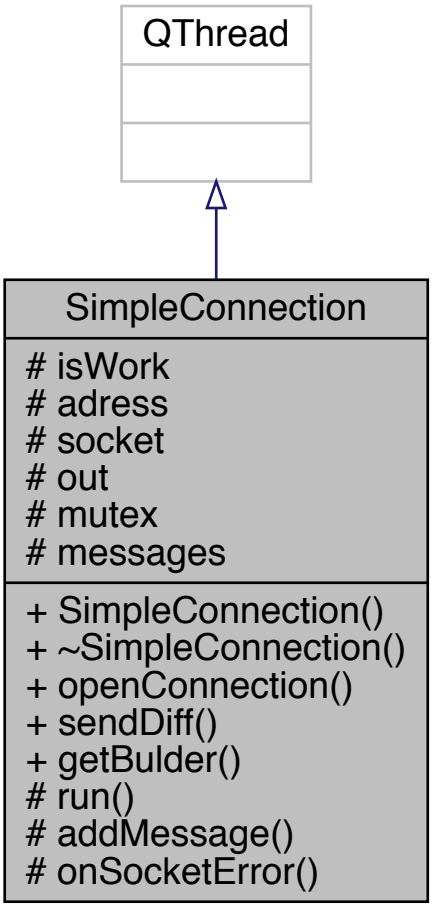
```
QMutex ServerWorker::workerMutex  [protected]
```

Definition at line 38 of file serverworker.h.

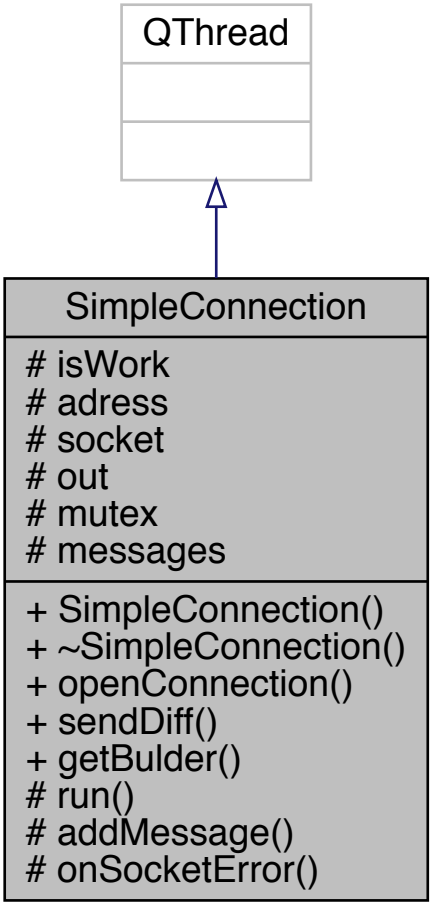
2.26 SimpleConnection Class Reference

```
#include "simpleconnection.h"
```

Inheritance diagram for SimpleConnection:



Collaboration diagram for SimpleConnection:



Classes

- class MessageBuilder
- class Receiver

Signals

- void onDiffReceive (QList< DiffElement *> *diffs)

Public Member Functions

- SimpleConnection (QHostAddress adress, QObject *parent=0)
- virtual ~SimpleConnection ()
- void openConnection ()
- void sendDiff (QList< DiffElement *> *diffs)
- MessageBuilder * getBulder ()

Protected Slots

- void onSocketError (QAbstractSocket::SocketError error)

Protected Member Functions

- void run ()
- void addMessage (MessageBuilder *messages)

Protected Attributes

- volatile bool isWork = true
- QHostAddress adress
- QTcpSocket * socket
- QDataStream * out
- QMutex mutex
- QQueue< MessageForServer * > messages

2.26.1 Detailed Description

Definition at line 50 of file simpleconnection.h.

2.26.2 Constructor & Destructor Documentation

2.26.2.1 SimpleConnection()

```
SimpleConnection::SimpleConnection (
            QHostAddress adress,
            QObject * parent = 0 )  [explicit]
```

Definition at line 5 of file simpleconnection.cpp.

2.26.2.2 ~SimpleConnection()

```
SimpleConnection::~SimpleConnection ( )  [virtual]
```

Definition at line 10 of file simpleconnection.cpp.

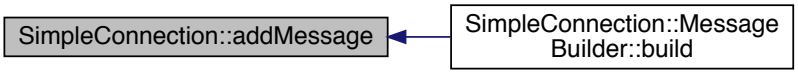
2.26.3 Member Function Documentation

2.26.3.1 addMessage()

```
void SimpleConnection::addMessage (
            MessageBuilder * messages )  [protected]
```

Definition at line 64 of file simpleconnection.cpp.

Here is the caller graph for this function:

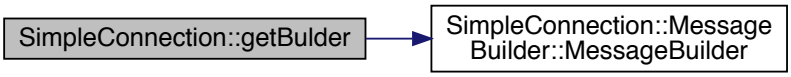


2.26.3.2 getBulder()

```
SimpleConnection::MessageBuilder * SimpleConnection::getBulder ( )
```

Definition at line 19 of file simpleconnection.cpp.

Here is the call graph for this function:



2.26.3.3 onDiffReceive

```
void SimpleConnection::onDiffReceive (
            QList< DiffElement *> * diffs )  [signal]
```

2.26.3.4 onSocketError

```
void SimpleConnection::onSocketError (
            QAbstractSocket::SocketError error )  [protected], [slot]
```

Definition at line 23 of file simpleconnection.cpp.

2.26.3.5 openConnection()

```
void SimpleConnection::openConnection ( )
```

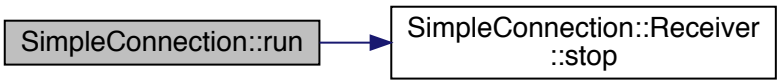
Definition at line 15 of file simpleconnection.cpp.

2.26.3.6 run()

```
void SimpleConnection::run ( )  [protected]
```

Definition at line 29 of file simpleconnection.cpp.

Here is the call graph for this function:



2.26.3.7 sendDiff()

```
void SimpleConnection::sendDiff (
            QList< DiffElement *> * diffs )
```

Definition at line 87 of file simpleconnection.cpp.

2.26.4 Member Data Documentation

2.26.4.1 adress

```
QHostAddress SimpleConnection::adress  [protected]
```

Definition at line 86 of file simpleconnection.h.

2.26.4.2 isWork

```
volatile bool SimpleConnection::isWork = true  [protected]
```

Definition at line 84 of file simpleconnection.h.

2.26.4.3 messages

```
QQueue<MessageForServer*> SimpleConnection::messages  [protected]
```

Definition at line 92 of file simpleconnection.h.

2.26.4.4 mutex

```
QMutex SimpleConnection::mutex  [protected]
```

Definition at line 91 of file simpleconnection.h.

2.26.4.5 out

```
QDataStream* SimpleConnection::out  [protected]
```

Definition at line 89 of file simpleconnection.h.

2.26.4.6 socket

```
QTcpSocket* SimpleConnection::socket  [protected]
```

Definition at line 87 of file simpleconnection.h.

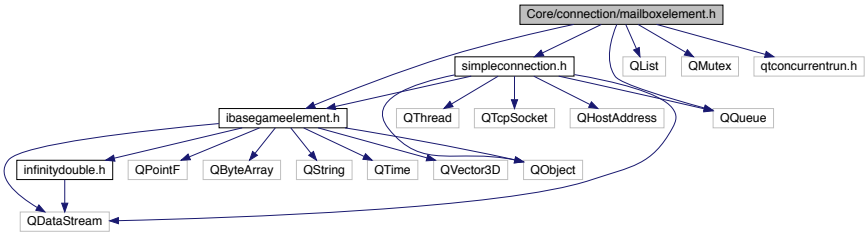
Chapter 3

File Documentation

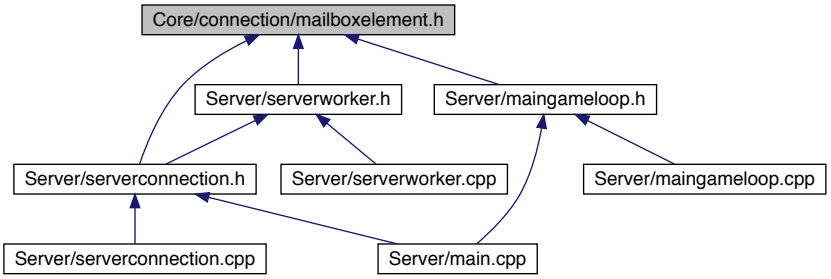
3.1 Core/connection/mailboxelement.h File Reference

```
#include "simpleconnection.h"
#include "../ibasegameelement.h"
#include <QList>
#include <QMutex>
#include <QQueue>
#include <qtconcurrentrun.h>
```

Include dependency graph for mailboxelement.h:



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



Classes

- class MailReceiver
- class MailReceiver::mailMessage
- class MailSender

3.2 mailboxelement.h

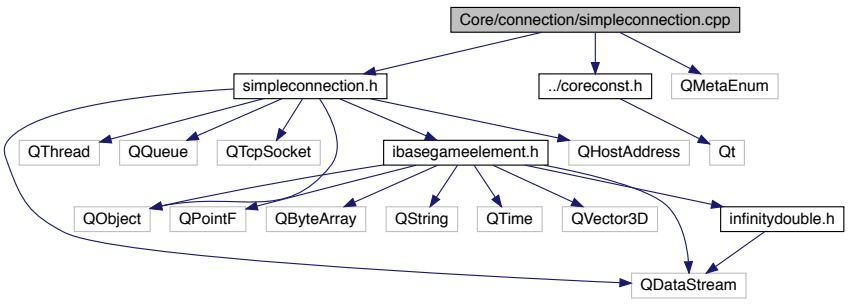
```
00001 #ifndef MAILBOXELEMENT_H
00002 #define MAILBOXELEMENT_H
00003
00004 #include "simpleconnection.h"
00005 #include "../ibasegameelement.h"
00006 #include <QList>
00007 #include <QMutex>
00008 #include <QQueue>
00009 #include <qtconcurrentrun.h>
00010
00011 using namespace QtConcurrent;
00012
00013 class MailReceiver;
00014 class MailSender;
00015
00016 class MailReceiver {
00017  public:
00018   virtual void sendMail(MessageForServer* message,
00019                         MailSender* sender,
00020                         int type,
00021                         bool isReplace = true) {
00022     run(QThreadPool::globalInstance(), [=] {
00023       qInfo() << "MailReceiver::sendMail - befor mutex";
00024       isQueueLock = true;
00025       mutex.lock();
00026       qInfo() << "MailReceiver::sendMail - in mutex";
00027       qInfo() << "MailReceiver :: adding new message item in to queue";
00028       mailMessage* msg;
00029       if (isReplace && messages.length() != 0) {
00030         for (int i = 0; i < messages.length(); i++) {
00031           msg = messages.at(i);
00032           if ((msg->sender == sender && msg->type == type) ||
00033               i == messages.length() - 1) {
00034             messages.removeAt(i);
00035             messages.insert(i,
00036                             new mailMessage(message, sender, type, isReplace));
00037             break;
00038           }
00039         }
00040       } else
00041         messages.push_back(new mailMessage(message, sender, type, isReplace));
00042       mailMessage* test = messages.last();
00043       qInfo() << "getting new message in queue :: " << (*messages.last());
00044       mutex.unlock();
00045       isQueueLock = false;
00046       qInfo() << "MailReceiver::sendMail - after mutex";
00047     });
00048   }
00049
00050  protected:
00051   class mailMessage {
00052    public:
00053     MessageForServer* message;
00054     MailSender* sender;
00055     int type;
00056     bool isReplace;
00057     mailMessage(MessageForServer* message,
00058                 MailSender* sender,
00059                 int type,
00060                 bool isReplace = true) {
00061       this->message = message;
00062       this->sender = sender;
00063       this->type = type;
00064       this->isReplace = isReplace;
00065     }
00066     friend QDebug operator<<(QDebug debug, const
      mailMessage& c) {
00067       QDebugStateSaver saver(debug);
00068
00069       if (c.message != nullptr) {
00070         debug.nospace() << "mailMessage {message:" << (*c.
```

```
      message) << "}";
00071       } else {
00072         debug.nospace() << "mailMessage {message:empty}";
00073       }
00074
00075       return debug;
00076     }
00077   };
00078   QQueue<mailMessage*> messages;
00079   QMutex mutex;
00080   volatile bool isQueueLock = false;
00081
00082   virtual mailMessage* nextMessage() {
00083     if (isQueueLock)
00084       return nullptr;
00085     mutex.lock();
00086     mailMessage* result = nullptr;
00087     if (messages.length() > 0) {
00088       qInfo() << "enqueue message";
00089       result = messages.takeFirst();
00090     }
00091     mutex.unlock();
00092     return result;
00093   }
00094 };
00095
00096 class MailSender {
00097  public:
00098   virtual void receiveResponce(QList<DiffElement*>*
      diff,
00099                                MessageForServer* message) = 0;
00100   virtual void setDefaultReceiver(MailReceiver* receiver) {
00101     this->receiver = receiver;
00102   }
00103
00104  protected:
00105   MailReceiver* receiver;
00106 };
00107
00108 #endif  // MAILBOXELEMENT_H
```

3.3 Core/connection/simpleconnection.cpp File Reference

```
#include "simpleconnection.h"
#include "../coreconst.h"
#include <QMetaEnum>
```

Include dependency graph for simpleconnection.cpp:



Functions

- QString stringify (eConnectionType e)
- QString stringify (eMessageType e)

3.3.1 Function Documentation

3.3.1.1 stringify() [1/2]

```
QString stringify (
            eConnectionType e )
```

Definition at line 96 of file simpleconnection.cpp.

3.3.1.2 stringify() [2/2]

```
QString stringify (
            eMessageType e )
```

Definition at line 106 of file simpleconnection.cpp.

3.4 simpleconnection.cpp

```
00001 #include "simpleconnection.h"
00002 #include "../coreconst.h"
00003 #include <QMetaEnum>
00004
00005 SimpleConnection::SimpleConnection(QHostAddress adress, QObject* parent)
00006     : QThread(parent) {
00007   this->adress = adress;
00008 }
00009
00010 SimpleConnection::~SimpleConnection() {
00011   delete socket;
00012   delete out;
00013 }
00014
00015 void SimpleConnection::openConnection() {
00016   this->start();
00017 }
00018
00019 SimpleConnection::MessageBuilder*
      SimpleConnection::getBulder() {
00020   return new MessageBuilder(this);
00021 }
00022
00023 void SimpleConnection::onSocketError(QAbstractSocket::SocketError err) {
00024   qCritical() << err;
00025   isWork = false;
00026   // TODO add something
00027 }
00028
00029 void SimpleConnection::run() {
00030   socket = new QTcpSocket();
00031   connect(socket, SIGNAL(error(QAbstractSocket::SocketError)), this,
00032           SLOT(onSocketError(QAbstractSocket::SocketError)),
00033           Qt::DirectConnection);
00034   socket->connectToHost(adress, DefaultServerParams::port);
00035   isWork = socket->waitForConnected();
00036   qDebug() << "Connect to server on " << socket->peerAddress().toString() << ":"
00037            << socket->peerPort();
00038   out = new QDataStream();
00039   out->setDevice(socket);
00040   out->setVersion(QDataStream::Qt_5_7);
00041
00042   Receiver* receiver = new Receiver(socket, this);
00043   while (!receiver->istThreadStart())
00044     qInfo() << "whaiting for receiver loop start........";
```

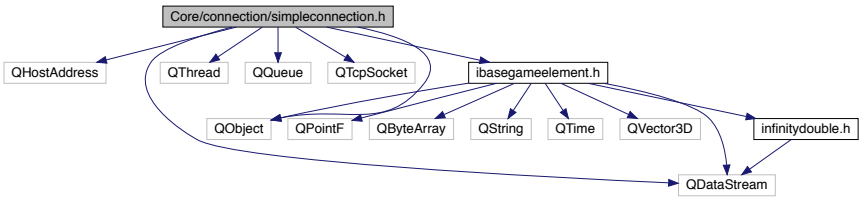
```
00045
00046   qInfo() << "starting sending to server message loop";
00047   while (isWork) {
00048     if (messages.length() > 0) {
00049       MessageForServer* newMessage = messages.takeFirst();
00050       out->startTransaction();
00051       (*out) << (*newMessage);
00052       out->commitTransaction();
00053       socket->flush();
00054       qInfo() << "sending some message for server";
00055       continue;
00056     } else {
00057       msleep(100);
00058     }
00059   }
00060   receiver->stop();
00061   delete receiver;
00062 }
00063
00064 void SimpleConnection::addMessage(
      MessageBuilder* message) {
00065   mutex.lock();
00066   this->messages.push_back(message->message);
00067   delete message;
00068   mutex.unlock();
00069 }
00070
00071 SimpleConnection::MessageBuilder::
      MessageBuilder(SimpleConnection* sender) {
00072   this->parent = sender;
00073   this->message = new MessageForServer();
00074 }
00075
00076 SimpleConnection::MessageBuilder*
00077 SimpleConnection::MessageBuilder::
      asFirstMessage(eConnectionType type) {
00078   this->message->connectionType = type;
00079   this->message->messageType = eFirstMessae;
00080   return this;
00081 }
00082
00083 void SimpleConnection::MessageBuilder::
      build() {
00084   parent->addMessage(this);
00085 }
00086
00087 void SimpleConnection::sendDiff(QList<DiffElement*>* diffs) {
00088   emit onDiffReceive(diffs);
00089 }
00090
00091 QString MessageForServer::toString() {
00092   return "MessageForServer :: connectionType = " + stringify(connectionType) +
00093          "; messageType = " + stringify(messageType);
00094 }
00095
00096 QString stringify(eConnectionType e) {
00097   switch (e) {
00098     case eGamer:
00099       return "eGamer";
00100     case eWatcher:
00101       return "eWatcher";
00102   }
00103   return "";
00104 }
00105
00106 QString stringify(eMessageType e) {
00107   switch (e) {
00108     case eFirstMessae:
00109       return "eFirstMessae";
00110     case eGetUpdateMessage:
00111       return "eGetUpdateMessage";
00112   }
00113   return "";
00114 }
00115
00116 SimpleConnection::Receiver::Receiver(QTcpSocket* socket,
00117                                      SimpleConnection* parentThread,
00118                                      QObject* parent)
00119     : QThread(parent) {
00120   this->socket = socket;
00121   this->parentThread = parentThread;
00122   this->start();
00123 }
00124
00125 SimpleConnection::Receiver::~Receiver() {
00126   delete in;
00127 }
```

```
00128
00129 bool SimpleConnection::Receiver::istThreadStart() {
00130   return isStart;
00131   msleep(100);
00132 }
00133
00134 void SimpleConnection::Receiver::stop() {
00135   isWork = false;
00136   while (isLoopActive) {
00137     msleep(100);
00138   }
00139 }
00140
00141 void SimpleConnection::Receiver::run() {
00142   in = new QDataStream();
00143   in->setDevice(socket);
00144   in->setVersion(QDataStream::Qt_5_7);
00145
00146   qInfo() << "starting message receiver loop";
00147   while (true) {
00148     isLoopActive = true;
00149     if (!isWork)
00150       break;
00151     isStart = true;
00152     socket->waitForReadyRead(-1);
00153     qInfo() << "starting reading some response";
00154     MessageForServer sendedMessage;
00155     (*in) >> sendedMessage;
00156     int diffsLenth;
00157     (*in) >> diffsLenth;
00158     QList<DiffElement*>* result = new QList<DiffElement*>();
00159     for (int i = 0; i < diffsLenth; i++) {
00160       DiffElement* newItem = new DiffElement();
00161       (*in) >> (*newItem);
00162       result->append(newItem);
00163     }
00164     parentThread->sendDiff(result);
00165     qInfo() << "something read from server";
00166   }
00167   isLoopActive = false;
00168 }
```

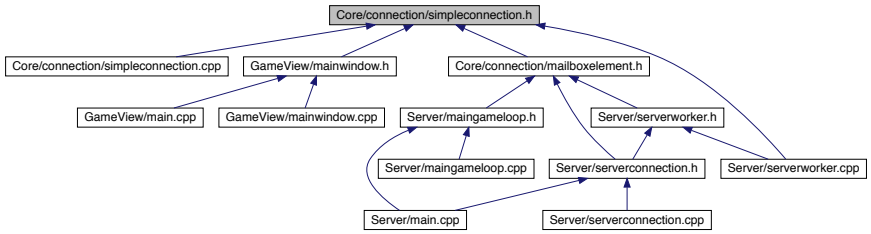
3.5 Core/connection/simpleconnection.h File Reference

```
#include <QHostAddress>
#include <QObject>
#include <QThread>
#include <QQueue>
#include <QTcpSocket>
#include <QDataStream>
#include <ibasegameelement.h>
```

Include dependency graph for simpleconnection.h:



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



Classes

- class MessageForServer
- class SimpleConnection
- class SimpleConnection::MessageBuilder
- class SimpleConnection::Receiver

Enumerations

- enum eConnectionType { eGamer, eWatcher }
- enum eMessageType { eFirstMessae, eGetUpdateMessage }

Functions

- QString stringify (eConnectionType e)
- QString stringify (eMessageType e)

3.5.1 Enumeration Type Documentation

3.5.1.1 eConnectionType

```
enum eConnectionType
```

Enumerator

eGamer	
eWatcher	

Definition at line 12 of file simpleconnection.h.

3.5.1.2 eMessageType

```
enum eMessageType
```

Enumerator

eFirstMessae	
eGetUpdateMessage	

Definition at line 13 of file simpleconnection.h.

3.5.2 Function Documentation

3.5.2.1 stringify() [1/2]

```
QString stringify (
            eConnectionType e )
```

Definition at line 96 of file simpleconnection.cpp.

3.5.2.2 stringify() [2/2]

```
QString stringify (
            eMessageType e )
```

Definition at line 106 of file simpleconnection.cpp.

3.6 simpleconnection.h

```
00001 #ifndef SIMPLECONNECTION_H
00002 #define SIMPLECONNECTION_H
00003
00004 #include <QHostAddress>
00005 #include <QObject>
00006 #include <QThread>
00007 #include <QQueue>
00008 #include <QTcpSocket>
00009 #include <QDataStream>
00010 #include <ibasegameelement.h>
00011
00012 enum eConnectionType { eGamer, eWatcher };
00013 enum eMessageType { eFirstMessae,
      eGetUpdateMessage };
00014
00015 QString stringify(eConnectionType e);
00016 QString stringify(eMessageType e);
00017
00018 class MessageForServer : public QObject {
00019   Q_OBJECT
00020
00021  public:
00022   eConnectionType connectionType;
```

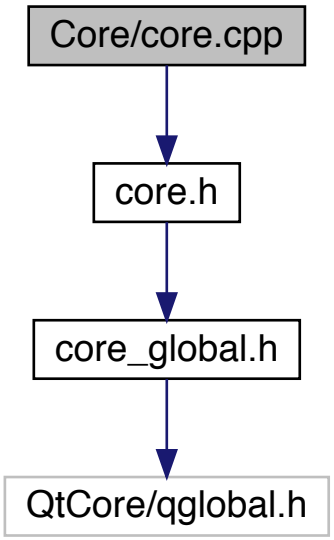
```
00023   eMessageType messageType;
00024
00025   friend QDataStream& operator<<(QDataStream&
      stream,
00026                                  const MessageForServer&
      myclass) {
00027     return stream << ((int)myclass.connectionType)
00028                   << ((int)myclass.messageType);
00029   }
00030   friend QDataStream& operator>>(QDataStream&
      stream,
00031                                  MessageForServer& myclass) {
00032     int con;
00033     int msg;
00034     QDataStream& result = (stream >> con >> msg);
00035     myclass.connectionType = (eConnectionType)
      con;
00036     myclass.messageType = (eMessageType)msg;
00037     return result;
00038   }
00039
00040   friend QDebug operator<<(QDebug debug,
      MessageForServer& c) {
00041     QDebugStateSaver saver(debug);
00042     debug.nospace() << c.toString();
00043
00044     return debug;
00045   }
00046
00047   QString toString();
00048 };
00049
00050 class SimpleConnection : public QThread {
00051   Q_OBJECT
00052  public:
00053   explicit SimpleConnection(QHostAddress adress,
      QObject* parent = 0);
00054   virtual ~SimpleConnection();
00055   void openConnection();
00056   void sendDiff(QList<DiffElement*>* diffs);
00057
00058   class MessageBuilder {
00059     friend class SimpleConnection;
00060
00061     MessageBuilder(SimpleConnection* sender);
00062
00063    public:
00064     MessageBuilder* asFirstMessage(
      eConnectionType type);
00065     void build();
00066
00067    private:
00068     MessageForServer* message;
00069     SimpleConnection* parent;
00070   };
00071
00072   MessageBuilder* getBulder();
00073
00074  signals:
00075   void onDiffReceive(QList<DiffElement*>*
      diffs);
00076
00077  public slots:
00078
00079  protected slots:
00080   void onSocketError(QAbstractSocket::
      SocketError error);
00081
00082   // QThread interface
00083  protected:
00084   volatile bool isWork = true;
00085   void run();
00086   QHostAddress adress;
00087   QTcpSocket* socket;
00088
00089   QDataStream* out;
00090
00091   QMutex mutex;
00092   QQueue<MessageForServer*> messages;
00093
00094   void addMessage(MessageBuilder* messages);
00095
00096   class Receiver : public QThread {
00097    public:
00098     Receiver(QTcpSocket* socket,
00099              SimpleConnection* parentThread,
00100              QObject* parent = 0);
```

```
00101     virtual ~Receiver();
00102     bool istThreadStart();
00103     void stop();
00104
00105    protected:
00106     QTcpSocket* socket;
00107     QDataStream* in;
00108     SimpleConnection* parentThread;
00109     volatile bool isStart = false;
00110     volatile bool isWork = true;
00111     volatile bool isLoopActive = false;
00112
00113     // QThread interface
00114    protected:
00115     void run();
00116   };
00117 };
00118
00119 #endif  // SIMPLECONNECTION_H
```

3.7 Core/core.cpp File Reference

```
#include "core.h"
```

Include dependency graph for core.cpp:

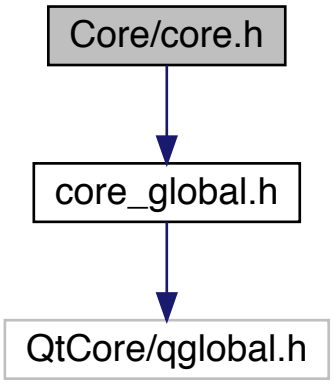


3.8 core.cpp

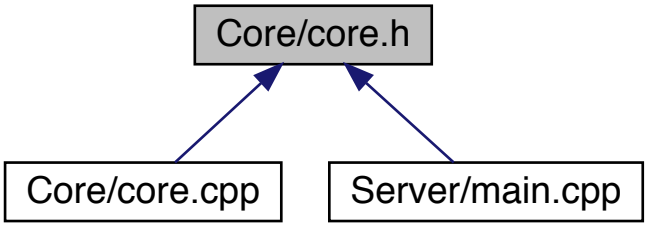
```
00001 #include "core.h"
00002
00003
00004 Core::Core()
00005 {
00006 }
```

3.9 Core/core.h File Reference

`#include "core_global.h"`
Include dependency graph for core.h:



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



Classes

- class Core

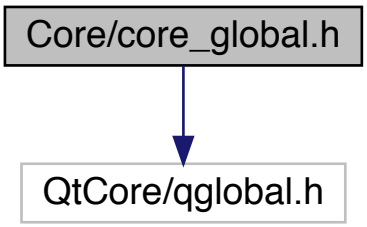
3.10 core.h

```
00001 #ifndef CORE_H
00002 #define CORE_H
00003
00004 #include "core_global.h"
00005
```

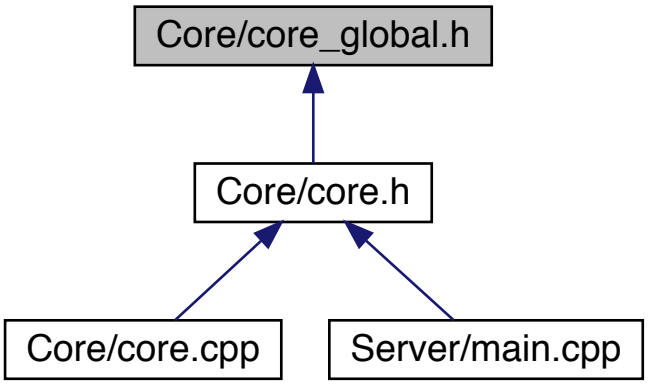
```
00006 class CORESHARED_EXPORT Core
00007 {
00008
00009 public:
00010     Core();
00011 };
00012
00013 #endif // CORE_H
```

3.11 Core/core_global.h File Reference

`#include <QtCore/qglobal.h>`
Include dependency graph for core_global.h:



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



Macros

- #define CORESHARED_EXPORT Q_DECL_IMPORT

3.11.1 Macro Definition Documentation

3.11.1.1 CORESHARED_EXPORT

```
#define CORESHARED_EXPORT Q_DECL_IMPORT
```

Definition at line 9 of file core_global.h.

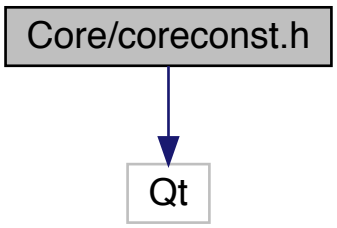
3.12 core_global.h

```
00001 #ifndef CORE_GLOBAL_H
00002 #define CORE_GLOBAL_H
00003
00004 #include <QtCore/qglobal.h>
00005
00006 #if defined(CORE_LIBRARY)
00007 #  define CORESHARED_EXPORT Q_DECL_EXPORT
00008 #else
00009 #  define CORESHARED_EXPORT Q_DECL_IMPORT
00010 #endif
00011
00012 #endif // CORE_GLOBAL_H
```

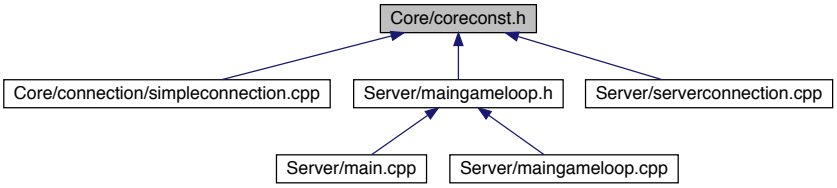
3.13 Core/coreconst.h File Reference

```
#include <Qt>
```

Include dependency graph for coreconst.h:



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



Namespaces

- DefaultServerParams
- limits

Variables

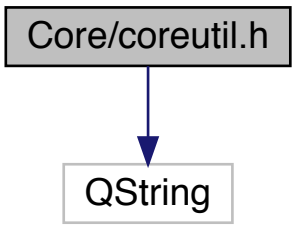
- const int DefaultServerParams::port = 23856
- const uint64_t limits::maxCountOfItems = 18446744073709551615
- const uint64_t limits::defaultBasisEnergy = 1000
- const uint64_t limits::maxRadiusVisionOfBasis = 100

3.14 coreconst.h

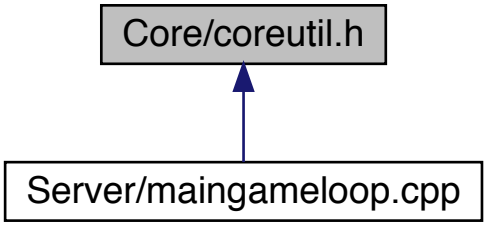
```
00001 #ifndef CONST_H
00002 #define CONST_H
00003
00004 #include <Qt>
00005
00006 namespace DefaultServerParams {
00007 const int port = 23856;
00008 }
00009
00010 namespace limits {
00011 const uint64_t maxCountOfItems = 18446744073709551615;
00012 const uint64_t defaultBasisEnergy = 1000;
00013 const uint64_t maxRadiusVisionOfBasis = 100;
00014 }
00015
00016 #endif  // CONST_H
```

3.15 Core/coreutil.h File Reference

`#include <QString>`
Include dependency graph for coreutil.h:



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



Functions

- QString codingNum (uint64_t num)

Variables

- const char alphabit [ ]
- const int alphabetSize = sizeof(alphabit) / sizeof(char)

3.15.1 Function Documentation

3.15.1.1 codingNum()

```
QString codingNum (
            uint64_t num )  [inline]
```

Definition at line 16 of file coreutil.h.

3.15.2 Variable Documentation

3.15.2.1 alphabetSize

```
const int alphabetSize = sizeof(alphabit) / sizeof(char)
```

Definition at line 14 of file coreutil.h.

3.15.2.2 alphabit

```
const char alphabit[]
```

Initial value:

```
= {
    '0', '1', '2', '3', '4', '5', '6', '7',  '8', '9', ':', ';', '<', '=', '>',
    '?', '@', 'A', 'B', 'C', 'D', 'E', 'F',  'G', 'H', 'I', 'J', 'K', 'L', 'M',
    'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U',  'V', 'W', 'X', 'Y', 'Z', '[', ']',
    '^', '_', '`', 'a', 'b', 'c', 'd', 'e',  'f', 'g', 'h', 'i', 'j', 'k', 'l',
    'm', 'n', 'o', 'p', 'q', 'r', 's', 't',  'u', 'v', 'w', 'x', 'y', 'z', '{',
    '|', '}', '~', '#', '$', '%', '&', '\'', '(', ')', '*', '+', ',', '-', '.'}
```

Definition at line 6 of file coreutil.h.

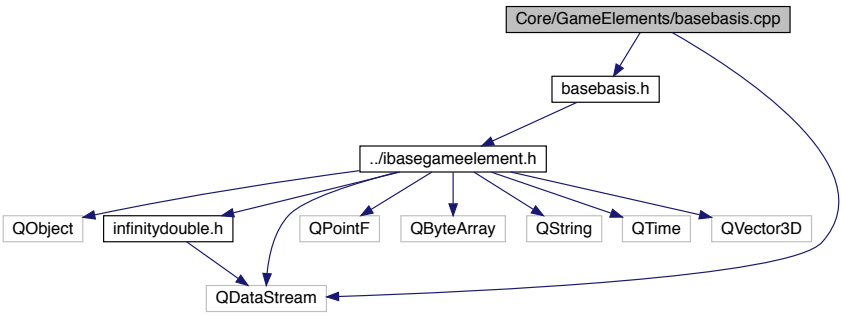
3.16 coreutil.h

```
00001 #ifndef UTIL_H
00002 #define UTIL_H
00003
00004 #include <QString>
00005
00006 const char alphabit[] = {
00007     '0', '1', '2', '3', '4', '5', '6', '7',  '8', '9', ':', ';', '<', '=', '>',
00008     '?', '@', 'A', 'B', 'C', 'D', 'E', 'F',  'G', 'H', 'I', 'J', 'K', 'L', 'M',
00009     'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U',  'V', 'W', 'X', 'Y', 'Z', '[', ']',
00010     '^', '_', '`', 'a', 'b', 'c', 'd', 'e',  'f', 'g', 'h', 'i', 'j', 'k', 'l',
00011     'm', 'n', 'o', 'p', 'q', 'r', 's', 't',  'u', 'v', 'w', 'x', 'y', 'z', '{',
00012     '|', '}', '~', '#', '$', '%', '&', '\'', '(', ')', '*', '+', ',', '-', '.'};
00013
00014 const int alphabetSize = sizeof(alphabit) / sizeof(char);
00015
00016 inline QString codingNum(uint64_t num) {
00017   QString result = "";
00018   if (num == 0)
00019     result = "0";
00020   while (num > 0) {
00021     result += alphabit[num % alphabetSize];
00022     num = num / alphabetSize;
00023   }
00024   return result;
00025 }
00026
00027 #endif  // UTIL_H
```

3.17 Core/GameElements/basebasis.cpp File Reference

```
#include "basebasis.h"
#include <QDataStream>
```

Include dependency graph for basebasis.cpp:

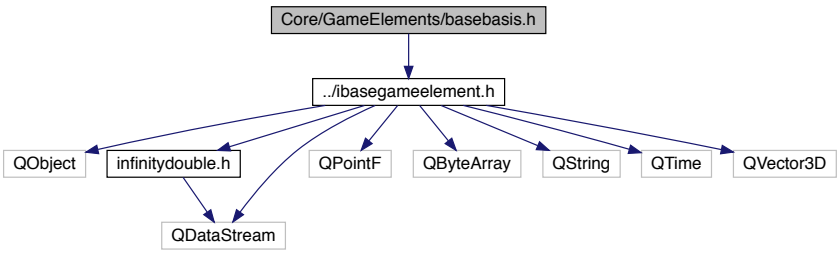


3.18 basebasis.cpp

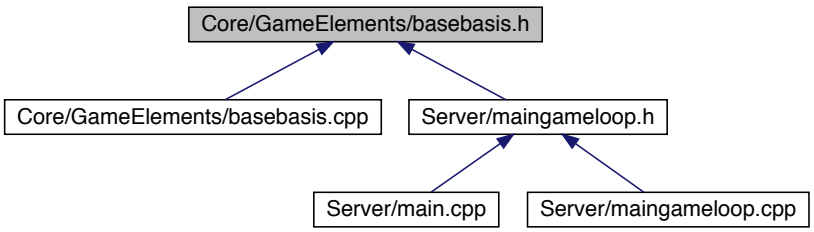
```
00001 #include "basebasis.h"
00002 #include <QDataStream>
00003
00004 BaseBasis::BaseBasis() : IBaseGameElement() {
00005   helth = InfinityDouble::InfinityValue();
00006   weight = InfinityDouble::InfinityValue();
00007   transitWeight = InfinityDouble::InfinityValue();
00008 }
00009
00010 void BaseBasis::setType(int value) {
00011   if (value != ((int)eBasis)) {
00012     Q_ASSERT_X(false, "game logic", "Wrong type for basis");
00013   }
00014   IBaseGameElement::setType(value);
00015 }
00016
00017 QByteArray* BaseBasis::getAdditionalData() {
00018   additionalData->clear();
00019   QDataStream stream(additionalData, QIODevice::WriteOnly);
00020   stream << energy;
00021 }
00022
00023 void BaseBasis::setAdditionakData(QByteArray* data) {
00024   QDataStream stream(*data);
00025   stream >> energy;
00026   IBaseGameElement::setAdditionakData(data);
00027 }
```

3.19 Core/GameElements/basebasis.h File Reference

`#include "../ibasegameelement.h"`
Include dependency graph for basebasis.h:



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



Classes

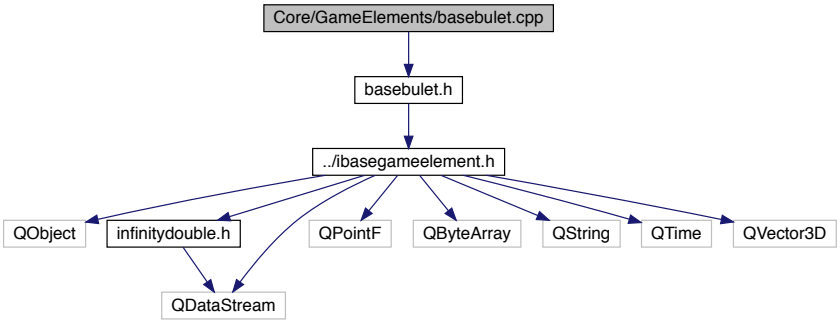
- class BaseBasis

3.20 basebasis.h

```
00001 #ifndef BASEBASIS_H
00002 #define BASEBASIS_H
00003
00004 #include "../ibasegameelement.h"
00005
00006 class BaseBasis : public IBaseGameElement {
00007  public:
00008   BaseBasis();
00009   virtual void setType(int value) override;
00010   virtual QByteArray* getAdditionalData();
00011   virtual void setAdditionakData(QByteArray* data);
00012
00013   virtual int getEnergy() { return energy; }
00014   virtual void setEnergy(int _enery) { this->energy = _enery; }
00015
00016  protected:
00017   int energy = 100;
00018 };
00019
00020 #endif  // BASEBASIS_H
```

3.21 Core/GameElements/basebulet.cpp File Reference

`#include "basebulet.h"`
Include dependency graph for basebulet.cpp:

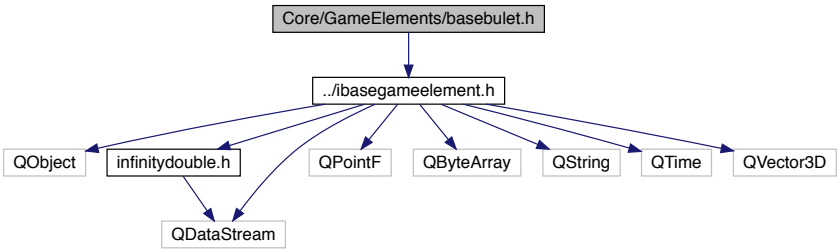


3.22 basebulet.cpp

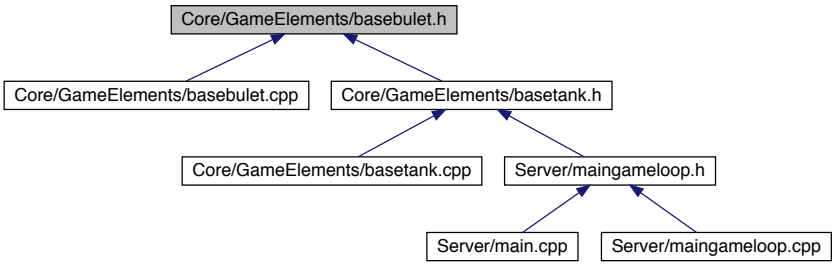
```
00001 #include "basebulet.h"
00002
00003 BaseBulet::BaseBulet() {}
00004
00005 QByteArray* BaseBulet::getAdditionalData() {}
00006
00007 void BaseBulet::setAdditionakData(QByteArray* data) {}
```

3.23 Core/GameElements/basebulet.h File Reference

`#include "../ibasegameelement.h"`
Include dependency graph for basebulet.h:



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



Classes

- class BaseBulet

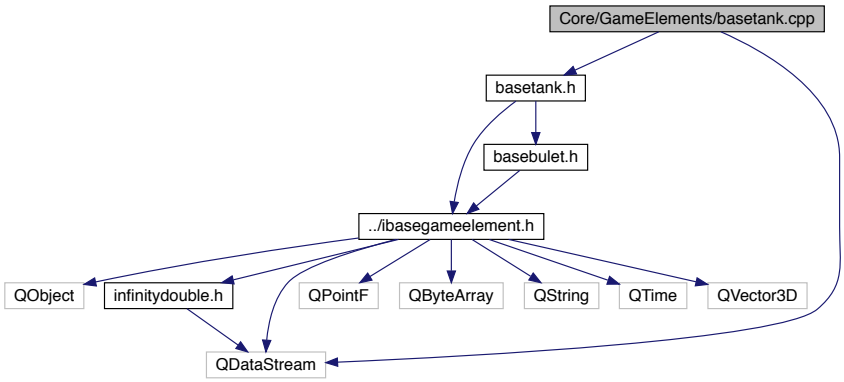
3.24 basebulet.h

```
00001 #ifndef BASEBULET_H
00002 #define BASEBULET_H
00003
00004 #include "../ibasegameelement.h"
00005
00006 class BaseBulet : public IBaseGameElement {
00007  public:
00008   BaseBulet();
00009
00010  protected:
00011   int lifetime = 0;
00012   double demage = 0;
00013   double direction;
00014   double speed;
00015
00016   // IBaseGameElement interface
00017  public:
00018   QByteArray* getAdditionalData();
00019   void setAdditionakData(QByteArray* data);
00020 };
00021
00022 #endif  // BASEBULET_H
```

3.25 Core/GameElements/basetank.cpp File Reference

```
#include "basetank.h"
#include <QDataStream>
```

Include dependency graph for basetank.cpp:



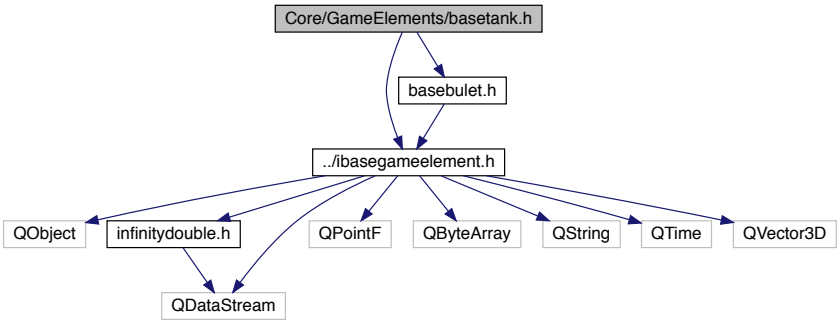
3.26 basetank.cpp

```
00001 #include "basetank.h"
00002 #include <QDataStream>
00003
00004 BaseTank::BaseTank() : IBaseGameElement() {}
00005
00006 void BaseTank::setType(int value) {
00007   if (value != ((int)eSimpleTank)) {
00008     Q_ASSERT_X(false, "game logic", "Wrong type for tank");
00009   }
00010   IBaseGameElement::setType(value);
00011 }
00012
00013 BaseBulet BaseTank::generateBullet() {}
00014
00015 void BaseTank::setAdditionakData(QByteArray* data) {
00016   IBaseGameElement::setAdditionakData(data);
00017   QDataStream stream(*data);
00018   stream >> direction >> speed >> fireType;
00019 }
00020
00021 QByteArray* BaseTank::getAdditionalData() {
00022   this->additionalData->clear();
00023   QDataStream stream(this->additionalData, QIODevice::WriteOnly);
00024   stream << direction << speed << fireType;
00025   return additionalData;
00026 }
```

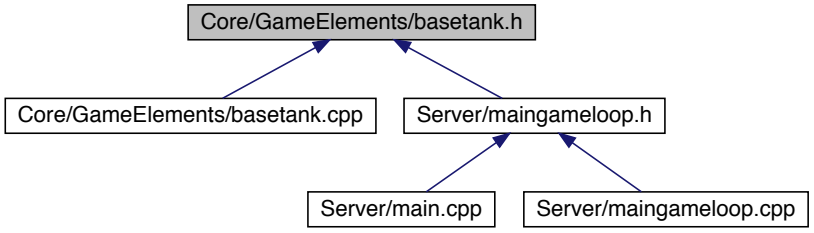
3.27 Core/GameElements/basetank.h File Reference

```
#include "../ibasegameelement.h"
#include "basebulet.h"
```

Include dependency graph for basetank.h:



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



Classes

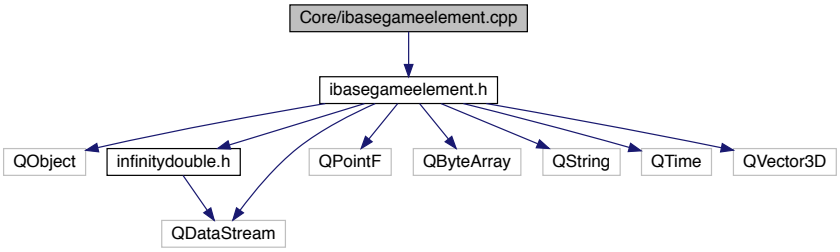
- class BaseTank

3.28 basetank.h

```
00001 #ifndef BASETANK_H
00002 #define BASETANK_H
00003 #include "../ibasegameelement.h"
00004 #include "basebulet.h"
00005
00006 class BaseTank : public IBaseGameElement {
00007  public:
00008   BaseTank();
00009
00010  public:
00011   virtual void setType(int value) override;
00012   virtual BaseBulet generateBullet();
00013   virtual void setAdditionakData(QByteArray* data);
00014   virtual QByteArray* getAdditionalData();
00015
00016  protected:
00017   double direction = 0;
00018   double speed = 0;
00019   /**
00020    * @brief fireType
00021    * 0  for non fire
00022    * -1 for single fire
00023    * if fireType > 0 then fire will be call evry fireType tik of game
00024    */
00025   int fireType = 0;
00026 };
00027
00028 #endif  // BASETANK_H
```

3.29 Core/ibasegameelement.cpp File Reference

```
#include "ibasegameelement.h"
```
Include dependency graph for ibasegameelement.cpp:

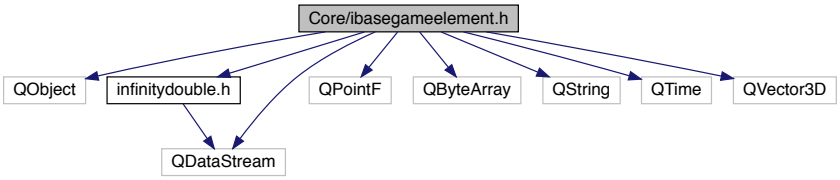


3.30 ibasegameelement.cpp

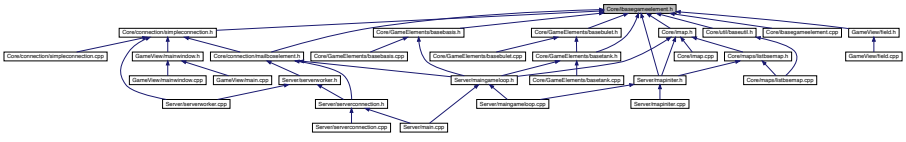
```
00001 #include "ibasegameelement.h"
```

3.31 Core/ibasegameelement.h File Reference

```
#include <QObject>
#include "infinitydouble.h"
#include <QPointF>
#include <QByteArray>
#include <QDataStream>
#include <QString>
#include <QTime>
#include <QVector3D>
```
Include dependency graph for ibasegameelement.h:



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



Classes

- class IBaseGameElement
- class DiffElement

Enumerations

- enum eDiffType { eNew, eChange, eDeleted }
- enum eBaseGameElementType { eGrass, eSimpleTank, eBasis }

3.31.1 Enumeration Type Documentation

3.31.1.1 eBaseGameElementType

```
enum eBaseGameElementType
```

Enumerator

eGrass	
eSimpleTank	
eBasis	

Definition at line 14 of file ibasegameelement.h.

3.31.1.2 eDiffType

```
enum eDiffType
```

Enumerator

eNew	
eChange	
eDeleted	

Definition at line 13 of file ibasegameelement.h.

3.32 ibasegameelement.h

```
00001 #ifndef IBASEGAMEELEMENT_H
00002 #define IBASEGAMEELEMENT_H
00003
00004 #include <QObject>
```

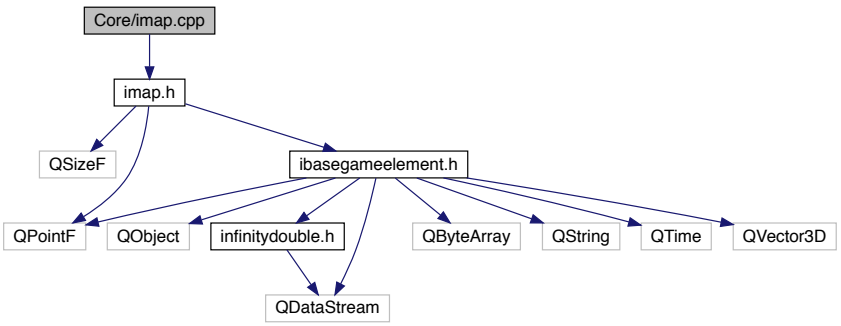
```
00005 #include "infinitydouble.h"
00006 #include <QPointF>
00007 #include <QByteArray>
00008 #include <QDataStream>
00009 #include <QString>
00010 #include <QTime>
00011 #include <QVector3D>
00012
00013 enum eDiffType { eNew, eChange, eDeleted };
00014 enum eBaseGameElementType { eGrass,
      eSimpleTank, eBasis };
00015 class IBaseGameElement;
00016
00017 class IBaseGameElement : public QObject {
00018   Q_OBJECT
00019  public:
00020   IBaseGameElement() {
00021     helth = InfinityDouble::FromValue(1);
00022     weight = InfinityDouble::FromValue(0);
00023     transitWeight = InfinityDouble::FromValue(0);
00024     position = new QVector3D(0, 0, 0);
00025     name = QString::number(QTime::currentTime().
      msec());
00026     additionalData = new QByteArray();
00027   }
00028
00029   virtual void nextStep(){};
00030
00031   virtual QVector3D* getPosition() { return position; }
00032   virtual int getType() { return type; }
00033   virtual InfinityDouble* getHelth() { return helth; }
00034   virtual InfinityDouble* getWeight() { return weight; }
00035   virtual InfinityDouble* getMaxTransitWeight() { return
      transitWeight; }
00036   virtual QString* getName() { return &name; }
00037   virtual QByteArray* getAdditionalData() { return additionalData; }
00038
00039   friend QDataStream& operator<<(QDataStream&
      stream,
00040                                  const IBaseGameElement&
      myclass) {
00041     return stream << (*myclass.position) << (*myclass.
      helth)
00042                   << (*myclass.weight) << (*myclass.
      transitWeight)
00043                   << (*myclass.additionalData) <<
      myclass.type << myclass.name
00044                   << rVision;
00045   }
00046   friend QDataStream& operator>>(QDataStream&
      stream,
00047                                  IBaseGameElement& myclass) {
00048     return stream >> (*myclass.position) >> (*myclass.
      helth) >>
00049            (*myclass.weight) >> (*myclass.transitWeight) >>
00050            (*myclass.additionalData) >> myclass.
      type >> myclass.name >> rVision;
00051     ;
00052   }
00053
00054   virtual ~IBaseGameElement() {
00055     delete helth;
00056     delete weight;
00057     delete transitWeight;
00058     delete position;
00059   }
00060
00061   virtual void setPosition(QVector3D* value) { this->position = value; }
00062
00063   virtual void setHelth(InfinityDouble* value) { this->helth = value; }
00064
00065   virtual void setWeight(InfinityDouble* value) { this->weight = value; }
00066
00067   virtual void setTransitWeight(InfinityDouble* value) {
00068     this->transitWeight = value;
00069   }
00070
00071   virtual void setType(int value) { this->type = value; }
00072
00073   virtual void setName(QString name) { this->name = name; }
00074
00075   virtual void setAdditionakData(QByteArray* data) {
00076     this->additionalData = data;
00077   }
00078
00079   virtual void setRVision(int _rVison) { rVision = _rVison; }
00080
```

```
00081   virtual int getRVision() { return rVision; }
00082
00083  signals:
00084
00085  public slots:
00086
00087  protected:
00088   QVector3D* position = nullptr;
00089   InfinityDouble* helth = nullptr;
00090   InfinityDouble* weight = nullptr;
00091   InfinityDouble* transitWeight = nullptr;
00092   QByteArray* additionalData = nullptr;
00093   int rVision = 1;
00094
00095   int type = -1;
00096   QString name;
00097 };
00098
00099 class DiffElement {
00100  public:
00101   DiffElement() { data = new IBaseGameElement(); }
00102   DiffElement(eDiffType type, IBaseGameElement* data) {
00103     this->type = type;
00104     this->data = data;
00105   }
00106
00107   friend QDataStream& operator<<(QDataStream&
      stream,
00108                                  const DiffElement& myclass) {
00109     stream << ((int)myclass.type);
00110     IBaseGameElement& gameElement = (*myclass.
      data);
00111     return stream << gameElement;
00112   }
00113   friend QDataStream& operator>>(QDataStream&
      stream, DiffElement& myclass) {
00114     int type;
00115     QDataStream& result = stream >> type >> (*
      myclass.data);
00116     myclass.type = (eDiffType)type;
00117     return result;
00118   }
00119
00120   eDiffType type;
00121   IBaseGameElement* data;
00122 };
00123
00124 #endif  // IBASEGAMEELEMENT_H
```

3.33 Core/imap.cpp File Reference

`#include "imap.h"`

Include dependency graph for imap.cpp:



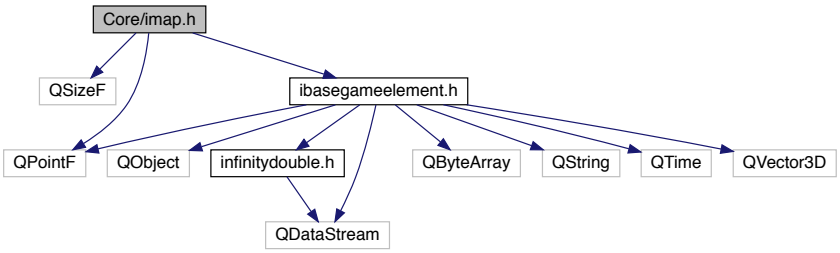
3.34 imap.cpp

```
00001
00002 #include "imap.h"
00003
00004 IMap::IMap()
00005 {
00006
00007 }
```

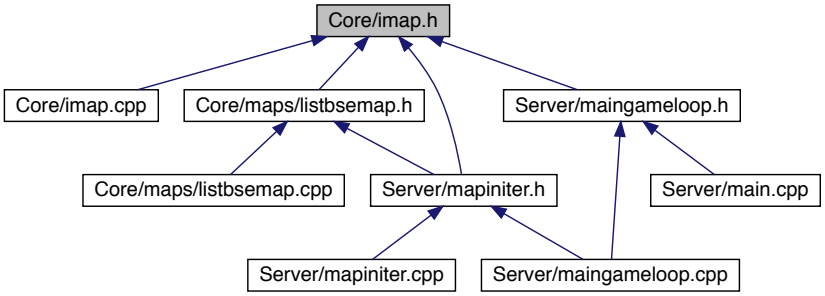
3.35 Core/imap.h File Reference

```
#include <QSizeF>
#include <QPointF>
#include "ibasegameelement.h"
```

Include dependency graph for imap.h:



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



Classes

- class IMap

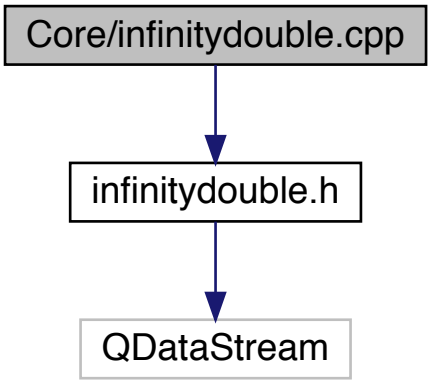
3.36 imap.h

```
00001 #ifndef IMAP_H
00002 #define IMAP_H
00003
00004 #include <QSizeF>
00005 #include <QPointF>
00006 #include "ibasegameelement.h"
00007
00008 class IMap {
00009  public:
00010   IMap();
00011   virtual QSizeF* getSize() = 0;
00012   virtual void insertElement(IBaseGameElement* element, QVector3D point)
      = 0;
00013   virtual void proccessAllInR(
00014       IBaseGameElement* element,
00015       double r,
00016       bool (&mapOperator)(IBaseGameElement* element)) = 0;
00017   virtual int getCount() = 0;
00018   virtual IBaseGameElement* getElementAtPosition(int pos) = 0;
00019 };
00020
00021 #endif  // IMAP_H
```

3.37 Core/infinitydouble.cpp File Reference

```
#include "infinitydouble.h"
```

Include dependency graph for infinitydouble.cpp:



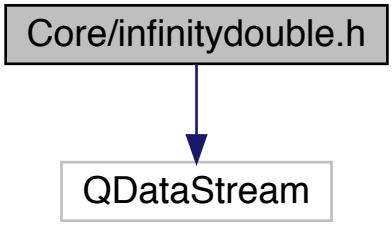
3.38 infinitydouble.cpp

```
00001 #include "infinitydouble.h"
00002
00003 InfinityDouble::InfinityDouble()
00004 {
00005
00006 }
00007
00008 InfinityDouble *InfinityDouble::FromValue(double w)
00009 {
00010     InfinityDouble *result = new InfinityDouble();
00011     result->w = w;
```

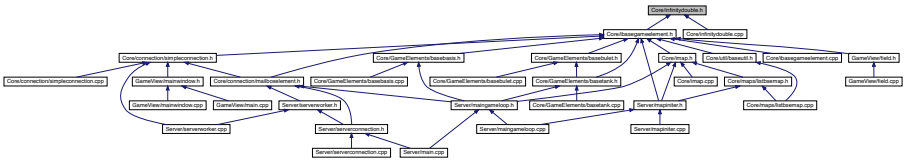
```
00012     result->isInfinity = false;
00013     return result;
00014 }
00015
00016 InfinityDouble *InfinityDouble::
      InfinityValue()
00017 {
00018     InfinityDouble *result=new InfinityDouble();
00019     result->isInfinity = true;
00020     return result;
00021 }
```

3.39 Core/infinitydouble.h File Reference

`#include <QDataStream>`
Include dependency graph for infinitydouble.h:



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



Classes

- class InfinityDouble

3.40 infinitydouble.h

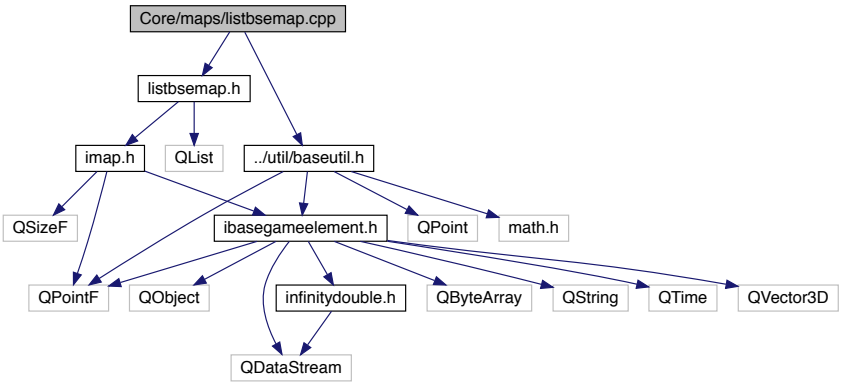
```
00001 #ifndef WEIGHT_H
00002 #define WEIGHT_H
00003
00004 #include <QDataStream>
00005
00006
00007
00008 class InfinityDouble:public QObject
00009 {
```

```
00010     Q_OBJECT
00011
00012     friend QDataStream& operator << (QDataStream&
      stream, const InfinityDouble& myclass)
00013     {
00014         return stream << myclass.isInfinity <<
      myclass.w;
00015     }
00016
00017     friend QDataStream& operator >> (QDataStream&
      stream, InfinityDouble& myclass)
00018     {
00019         return stream >> myclass.isInfinity >>
      myclass.w;
00020     }
00021
00022 private:
00023     InfinityDouble();
00024     double w;
00025     bool isInfinity = false;
00026
00027
00028 public:
00029     static InfinityDouble *FromValue(double w);
00030     static InfinityDouble *InfinityValue();
00031
00032
00033
00034
00035 };
00036
00037 #endif // WEIGHT_H
```

3.41 Core/maps/listbsemap.cpp File Reference

```
#include "listbsemap.h"
#include "../util/baseutil.h"
```

Include dependency graph for listbsemap.cpp:



3.42 listbsemap.cpp

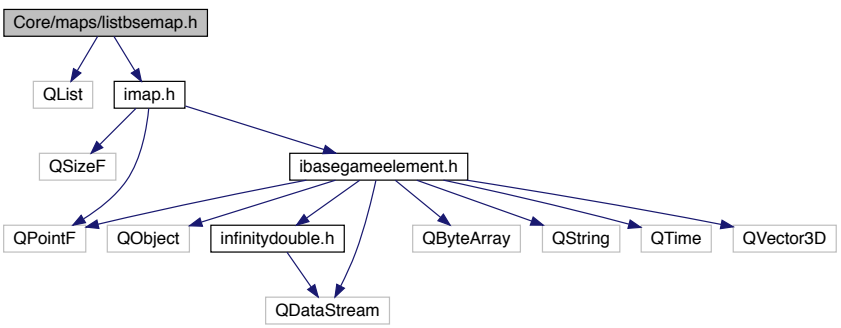
```
00001 #include "listbsemap.h"
00002 #include "../util/baseutil.h"
00003
00004 ListBseMap::ListBseMap(double width, double heigth) {
00005   size = new QSizeF(width, heigth);
00006   items = new QList<IBaseGameElement*>();
00007 }
00008
```

```
00009 ListBseMap::~ListBseMap() {
00010   delete items;
00011   delete size;
00012 }
00013
00014 QSizeF* ListBseMap::getSize() {
00015   delete size;
00016 }
00017
00018 void ListBseMap::insertElement(IBaseGameElement* element,
      QVector3D point) {
00019   items->append(element);
00020 }
00021
00022 void ListBseMap::proccessAllInR(
      IBaseGameElement* element,
00023                                 double r,
00024                                 bool (&mapOperator)(IBaseGameElement*)) {
00025   for (int i = 0; i < items->size(); i++) {
00026     IBaseGameElement* el = items->at(i);
00027     if (element != el && distanceBetweenElement(el, element
      ))
00028       mapOperator(el);
00029   }
00030 }
00031
00032 int ListBseMap::getCount() {
00033   return items->size();
00034 }
00035
00036 IBaseGameElement* ListBseMap::
      getElementAtPosition(int pos) {
00037   return items->at(pos);
00038 }
```

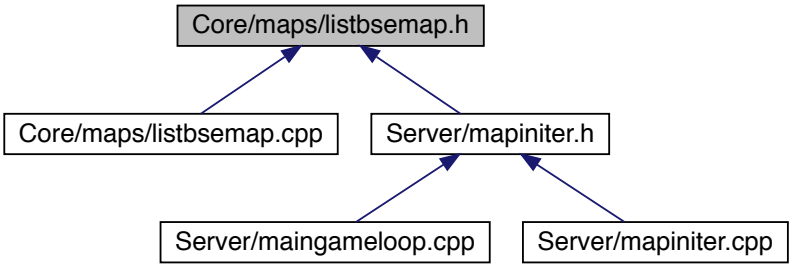
3.43 Core/maps/listbsemap.h File Reference

```
#include <QList>
#include "imap.h"
```

Include dependency graph for listbsemap.h:



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



Classes

- class ListBseMap

3.44 listbsemap.h

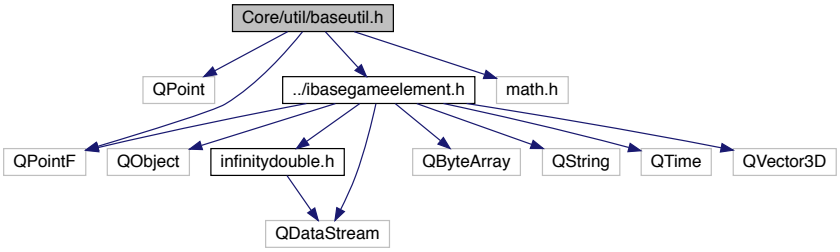
```
00001 #ifndef LISTBSEMAP_H
00002 #define LISTBSEMAP_H
00003
00004 #include <QList>
00005 #include "imap.h"
00006
00007 class ListBseMap : public IMap {
00008  public:
00009   ListBseMap(double width, double heigth);
00010   ~ListBseMap();
00011   // IMap interface
00012  public:
00013   QSizeF* getSize();
00014   void insertElement(IBaseGameElement* element, QVector3D point);
00015   void proccessAllInR(IBaseGameElement* element,
00016                       double r,
00017                       bool (&mapOperator)(IBaseGameElement*));
00018   int getCount();
00019   IBaseGameElement* getElementAtPosition(int pos);
00020
00021  protected:
00022   QList<IBaseGameElement*>* items;
00023   QSizeF* size;
00024 };
00025
00026 #endif  // LISTBSEMAP_H
```

3.45 Core/util/baseutil.h File Reference

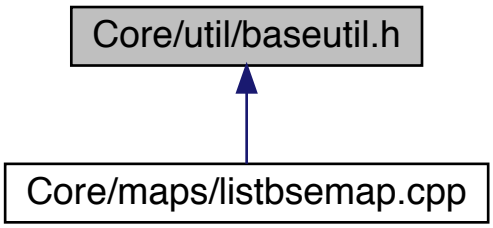
```
#include <QPoint>
#include <QPointF>
#include "../ibasegameelement.h"
```

```
#include "math.h"
```

Include dependency graph for baseutil.h:



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



Functions

- double distanceBetweenElement (IBaseGameElement *el1, IBaseGameElement *el2)

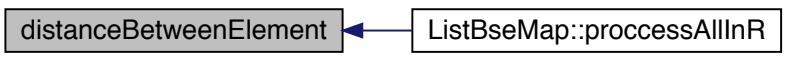
3.45.1 Function Documentation

3.45.1.1 distanceBetweenElement()

```
double distanceBetweenElement (
            IBaseGameElement * el1,
            IBaseGameElement * el2 )
```

Definition at line 6 of file baseutil.h.

Here is the caller graph for this function:



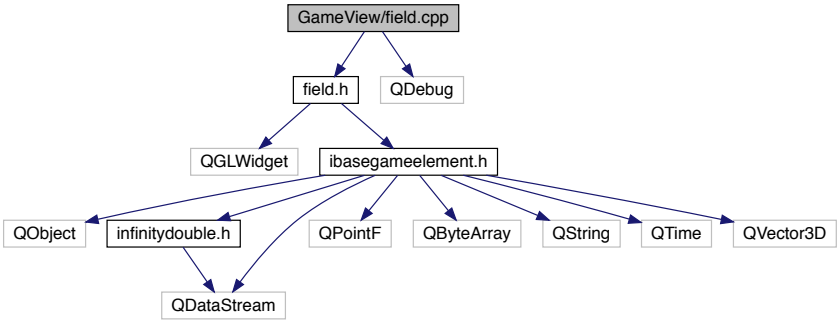
3.46 baseutil.h

```
00001 #include <QPoint>
00002 #include <QPointF>
00003 #include "../ibasegameelement.h"
00004 #include "math.h"
00005
00006 double distanceBetweenElement(IBaseGameElement* el1,
      IBaseGameElement* el2) {
00007   if (el1 == NULL || el2 == NULL)
00008     return 0;
00009   if (el1 == el2)
00010     return 0;
00011   QVector3D* p1 = el1->getPosition();
00012   QVector3D* p2 = el2->getPosition();
00013   double dx = p1->x() - p2->x();
00014   double dy = p1->y() - p2->x();
00015   double dz = p1->z() - p2->z();
00016   return sqrt(dx * dx + dy * dy + dz * dz);
00017 }
```

3.47 GameView/field.cpp File Reference

```
#include "field.h"
#include <QDebug>
```

Include dependency graph for field.cpp:



Functions

- QImage loadTexture2 (char *filename, GLuint &textureID)

3.47.1 Function Documentation

3.47.1.1 loadTexture2()

```
QImage loadTexture2 (
            char * filename,
            GLuint & textureID )
```

Definition at line 42 of file field.cpp.

3.48 field.cpp

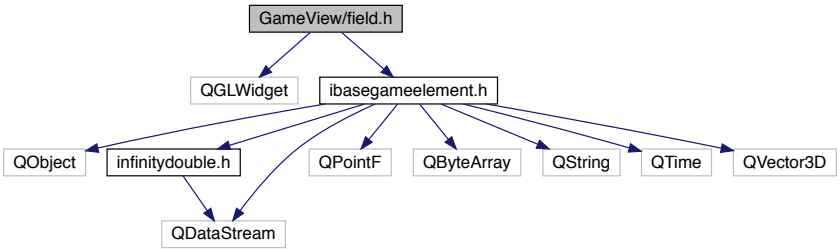
```
00001 #include "field.h"
00002 #include <QDebug>
00003
00004 Field::Field(QWidget* parent) : QGLWidget(parent) {
00005   fieldState = new QList<IBaseGameElement*>();
00006   initRes();
00007 }
00008
00009 Field::~Field() {
00010   delete fieldState;
00011 }
00012
00013 void Field::onDiffReceive(QList<DiffElement*>* diff) {
00014   for (int i = 0; i < diff->length(); i++) {
00015     if (diff->at(i)->type == eDiffType::eNew) {
00016       this->fieldState->append(diff->at(i)->data);
00017     }
00018   }
00019   update();
00020 }
00021 void Field::initializeGL() {}
00022
00023 void Field::resizeGL(int w, int h) {}
00024
00025 void Field::paintGL() {
00026   glClearColor(0.1f, 0.1f, 0.1f, 1.0f);
00027   glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
00028   glTranslatef(-1, -1, 0);
00029   for (int i = 0; i < fieldState->length(); i++) {
00030     auto currentElement = fieldState->at(i);
00031     if (currentElement->getType() == (int)eBaseGameElementType::eGrass) {
00032       drawGrass(currentElement->getPosition()->x(),
00033                 currentElement->getPosition()->y());
00034     }
00035   }
00036   glTranslatef(1, 1, 0);
00037   //  glEnd();
00038
00039   //  glDisable(GL_TEXTURE_2D);
00040 }
00041
00042 QImage loadTexture2(char* filename, GLuint& textureID) {
00043   glEnable(GL_TEXTURE_2D);  // Enable texturing
00044
00045   glGenTextures(1, &textureID);             // Obtain an id for the texture
00046   glBindTexture(GL_TEXTURE_2D, textureID);  // Set as the current texture
00047
00048   glPixelStorei(GL_UNPACK_ALIGNMENT, 1);
00049   glTexEnvf(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_DECAL);
00050
00051   QImage im(filename);
00052   QImage tex = QGLWidget::convertToGLFormat(im);
00053
00054   glTexImage2D(GL_TEXTURE_2D, 0, GL_RGBA, tex.width(), tex.height(), 0, GL_RGBA,
00055                GL_UNSIGNED_BYTE, tex.bits());
00056
00057   glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
00058   glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
00059
00060   glDisable(GL_TEXTURE_2D);
00061
00062   return tex;
00063 }
00064
00065 void Field::initRes() {
00066   loadTexture2(":/res/grass.png", grass);
00067 }
00068
00069 void Field::drawGrass(float x, float y) {
00070   //  glEnable(GL_TEXTURE_2D);
00071   //  glBindTexture(GL_TEXTURE_2D, grass);
00072
00073   //  glBegin(GL_QUADS);
00074   //  glTexCoord2f(0, 0);
00075   //  glVertex3f(-1, -1, -1);
00076   //  glTexCoord2f(1, 0);
00077   //  glVertex3f(1, -1, -1);
00078   //  glTexCoord2f(1, 1);
00079   //  glVertex3f(1, 1, -1);
00080   //  glTexCoord2f(0, 1);
00081   //  glVertex3f(-1, 1, -1);
00082   // .....
00083   //  glDisable(GL_TEXTURE_2D);
00084   float size = scaleK;
```

```
00085   glColor3f(0.560, 0.956, 0.258);
00086   glBegin(GL_QUADS);
00087   glVertex2f(x * this->scaleK, y * scaleK);
00088   glVertex2f(x * this->scaleK, y * scaleK + size);
00089   glVertex2f(x * this->scaleK + size, y * scaleK + size);
00090   glVertex2f(x * this->scaleK + size, y * scaleK);
00091   glEnd();
00092 }
```

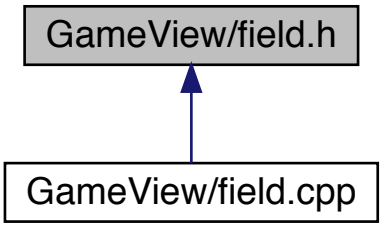
3.49 GameView/field.h File Reference

```
#include <QGLWidget>
#include <ibasegameelement.h>
```

Include dependency graph for field.h:



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



Classes

- class Field

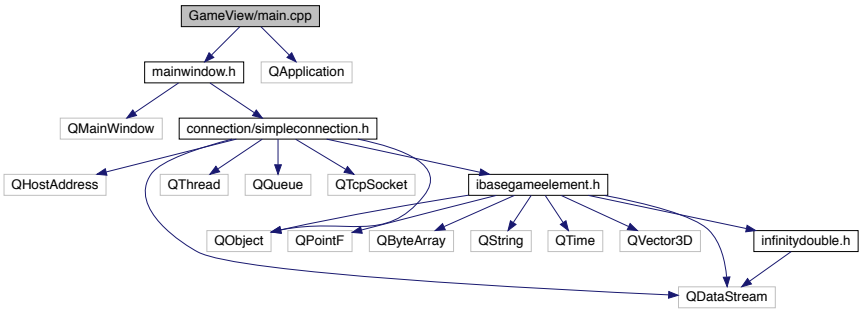
3.50 field.h

```
00001 #ifndef FIELD_H
00002 #define FIELD_H
00003 #include <QGLWidget>
00004 #include <ibasegameelement.h>
00005
00006 class Field : public QGLWidget {
00007   Q_OBJECT
00008  public:
00009   Field(QWidget* parent = 0);
00010   ~Field();
00011
00012  public slots:
00013   void onDiffReceive(QList<DiffElement*>* diff);
00014   // QGLWidget interface
00015  protected:
00016   void initializeGL() Q_DECL_OVERRIDE;
00017   void resizeGL(int w, int h) Q_DECL_OVERRIDE;
00018   void paintGL() Q_DECL_OVERRIDE;
00019   QList<IBaseGameElement*>* fieldState;
00020   void initRes();
00021
00022   void drawGrass(float x, float y);
00023
00024   GLuint grass;
00025
00026   float scaleK = 0.005;
00027 };
00028
00029 #endif  // FIELD_H
```

3.51 GameView/main.cpp File Reference

```
#include "mainwindow.h"
#include <QApplication>
```

Include dependency graph for main.cpp:



Functions

- int main (int argc, char *argv[])

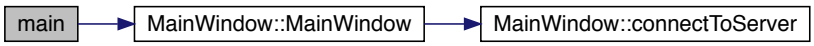
3.51.1 Function Documentation

3.51.1.1 main()

```
int main (
              int argc,
              char * argv[] )
```

Definition at line 4 of file main.cpp.

Here is the call graph for this function:



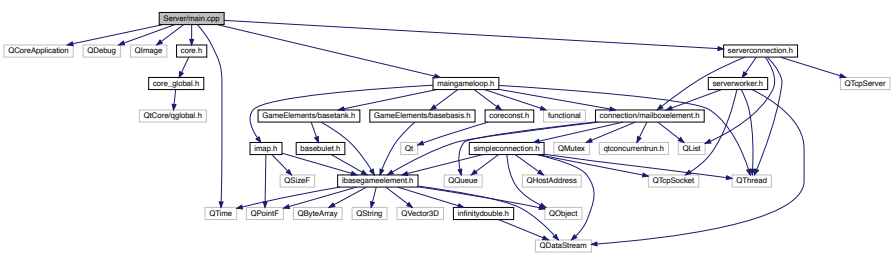
3.52 main.cpp

```
00001 #include "mainwindow.h"
00002 #include <QApplication>
00003
00004 int main(int argc, char* argv[]) {
00005   QApplication a(argc, argv);
00006
00007   qSetMessagePattern(
00008       "%{time boot} :: [%{type}::%{appname} in %{file}:%{line} th:%{threadid}] "
00009       "message:: "
00010       "%{message} "
00011       "%{backtrace [separator=\"\n\t\"]}");
00012   //  qSetMessagePattern("%{time boot} :: %{message}");
00013   MainWindow w;
00014   w.show();
00015
00016   return a.exec();
00017 }
```

3.53 Server/main.cpp File Reference

```
#include <QCoreApplication>
#include <QDebug>
#include <QImage>
#include <core.h>
#include <QTime>
#include "serverconnection.h"
#include "maingameloop.h"
```

Include dependency graph for main.cpp:



Functions

- void onServerError (serverError error)
- void initMainLooper ()
- void initServer ()
- int main (int argc, char ∗argv[ ])

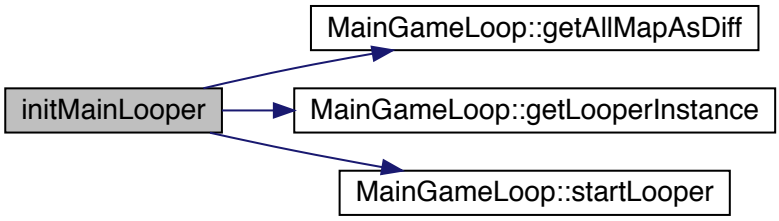
3.53.1 Function Documentation

3.53.1.1 initMainLooper()

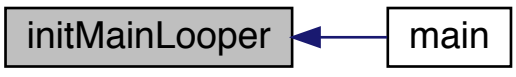
```
void initMainLooper ( )
```

Definition at line 15 of file main.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

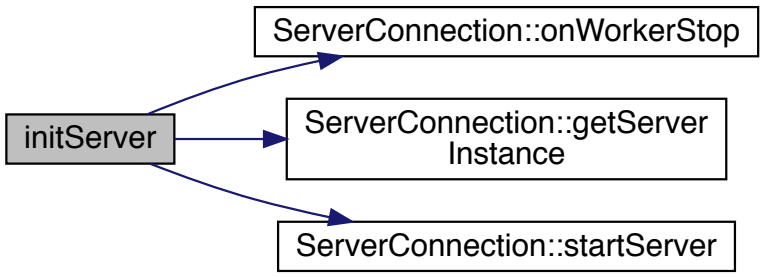


3.53.1.2 initServer()

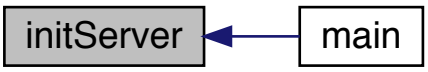
```
void initServer ( )
```

Definition at line 20 of file main.cpp.

Here is the call graph for this function:



Here is the caller graph for this function:

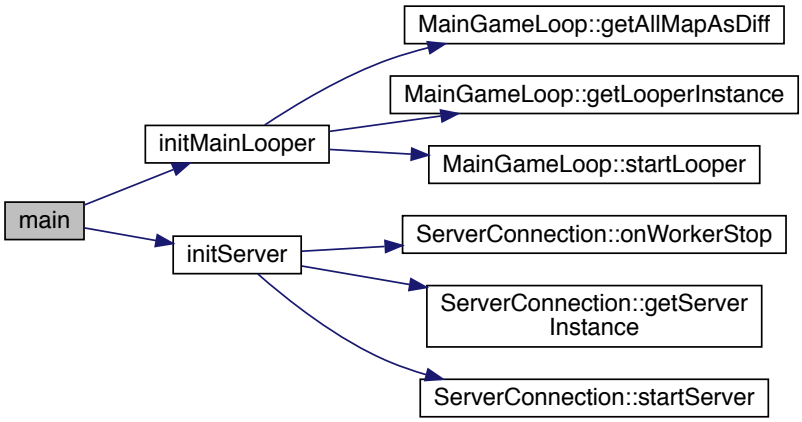


3.53.1.3 main()

```
int main (
            int argc,
            char * argv[] )
```

Definition at line 27 of file main.cpp.

Here is the call graph for this function:



3.53.1.4 onServerError()

```
void onServerError (
            serverError error )
```

Definition at line 11 of file main.cpp.

3.54 main.cpp

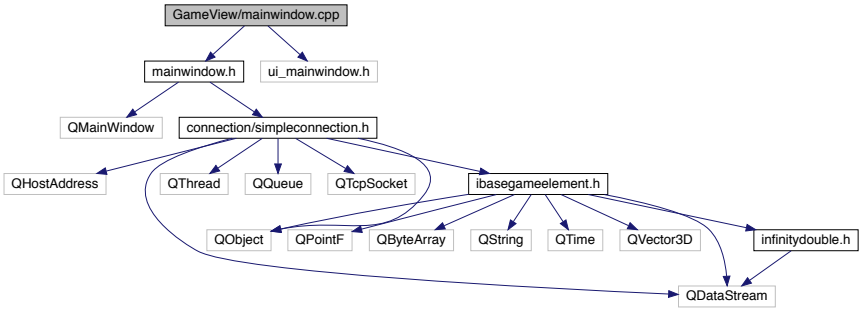
```
00001 #include <QCoreApplication>
00002 #include <QDebug>
00003 #include <QImage>
00004 #include <core.h>
00005 #include <QDebug>
00006 #include <QTime>
00007
00008 #include "serverconnection.h"
00009 #include "maingameloop.h"
00010
00011 void onServerError(serverError error) {
00012   qDebug() << "some bad error :: " << error;
00013 }
00014
00015 void initMainLooper() {
00016   MainGameLoop* mainLooper = MainGameLoop::
      getLooperInstance();
00017   mainLooper->startLooper();
00018 }
00019
00020 void initServer() {
00021   ServerConnection* server = ServerConnection
      ::getServerInstance();
00022   QObject::connect(server, &ServerConnection::onServerError, &onServerError);
00023   server->setDefaultReceiver(MainGameLoop::getLooperInstance());
00024   server->startServer();
00025 }
00026
00027 int main(int argc, char* argv[]) {
00028   QCoreApplication a(argc, argv);
00029   qSetMessagePattern(
```

```
00030       "%{time boot} :: [%{type}::%{appname} in %{file}:%{line} th:%{threadid}] "
00031       "message:: "
00032       "%{message} "
00033       "%{backtrace [separator=\"\n\t\"]}");
00034
00035   QTime midnight(0, 0, 0);
00036   qsrand(midnight.secsTo(QTime::currentTime()));
00037
00038   initMainLooper();
00039
00040   initServer();
00041
00042   return a.exec();
00043 }
```

3.55 GameView/mainwindow.cpp File Reference

```
#include "mainwindow.h"
#include "ui_mainwindow.h"
```

Include dependency graph for mainwindow.cpp:



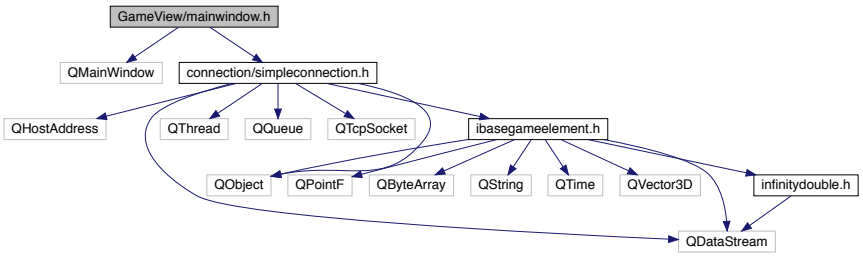
3.56 mainwindow.cpp

```
00001 #include "mainwindow.h"
00002 #include "ui_mainwindow.h"
00003
00004 MainWindow::MainWindow(QWidget* parent)
00005     : QMainWindow(parent),
00006       ui(new Ui::MainWindow),
00007       connection(QHostAddress::LocalHost, this) {
00008   ui->setupUi(this);
00009   connectToServer();
00010 }
00011
00012 MainWindow::~MainWindow() {
00013   delete ui;
00014 }
00015
00016 void MainWindow::connectToServer() {
00017   connection.openConnection();
00018   connection.getBulder()->asFirstMessage(eConnectionType::eWatcher)->build();
00019   connect(&connection, SIGNAL(onDiffReceive(QList<DiffElement*>*)),
00020           ui->gameField, SLOT(onDiffReceive(QList<DiffElement*>*)));
00021 }
```

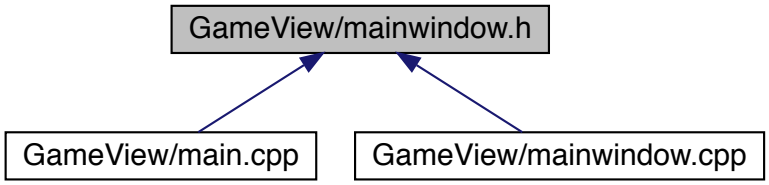
3.57 GameView/mainwindow.h File Reference

```
#include <QMainWindow>
#include "connection/simpleconnection.h"
```

Include dependency graph for mainwindow.h:



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



Classes

- class MainWindow

Namespaces

- Ui

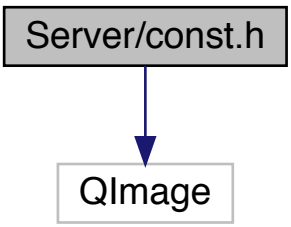
3.58 mainwindow.h

```
00001 #ifndef MAINWINDOW_H
00002 #define MAINWINDOW_H
00003
00004 #include <QMainWindow>
00005 #include "connection/simpleconnection.h"
00006
00007 namespace Ui {
00008 class MainWindow;
00009 }
00010
00011 class MainWindow : public QMainWindow {
00012   Q_OBJECT
```

```
00013
00014  public:
00015   explicit MainWindow(QWidget* parent = 0);
00016   ~MainWindow();
00017
00018  protected:
00019   void connectToServer();
00020   SimpleConnection connection;
00021
00022  private:
00023   Ui::MainWindow* ui;
00024 };
00025
00026 #endif  // MAINWINDOW_H
```

3.59 Server/const.h File Reference

`#include <QImage>`
Include dependency graph for const.h:



Namespaces

- staticBockTypes

Variables

- int staticBockTypes::grass = 1

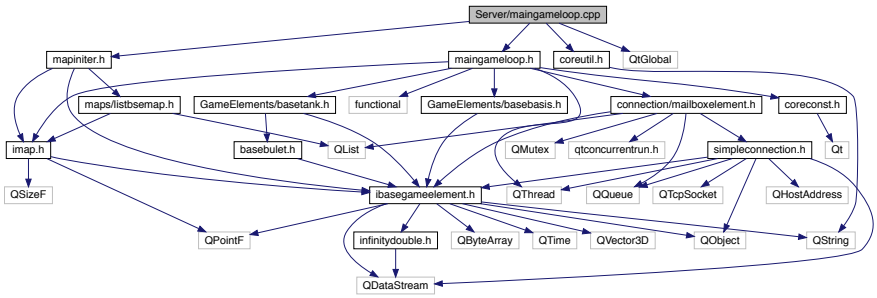
3.60 const.h

```
00001 #ifndef CONST_H
00002 #define CONST_H
00003
00004 #include <QImage>
00005
00006 namespace staticBockTypes {
00007 int grass = 1;
00008 }
00009
00010
00011
00012 #endif // CONST_H
```

3.61 Server/maingameloop.cpp File Reference

```
#include "maingameloop.h"
#include "mapiniter.h"
#include <QtGlobal>
#include <coreutil.h>
```

Include dependency graph for maingameloop.cpp:



Variables

- QString TAG = "MainGameLoop"

3.61.1 Variable Documentation

3.61.1.1 TAG

```
QString TAG = "MainGameLoop"
```

Definition at line 6 of file maingameloop.cpp.

3.62 maingameloop.cpp

```
00001 #include "maingameloop.h"
00002 #include "mapiniter.h"
00003 #include <QtGlobal>
00004 #include <coreutil.h>
00005
00006 QString TAG = "MainGameLoop";
00007
00008 MainGameLoop::MainGameLoop() {
00009   map = MapIniter().initSimapleMap();
00010 }
00011 void MainGameLoop::proccessGamerMessage(
      MailReceiver::mailMessage* msg,
00012                                         MailSender* receiver) {
00013   GamerInformation* currentGamer = nullptr;
00014   bool isAlredyExist = false;
00015   for (int i = 0; i < gamersList.size(); i++) {
00016     if (gamersList.at(i)->sender == receiver) {
00017       isAlredyExist = true;
00018       currentGamer = gamersList.at(i);
```

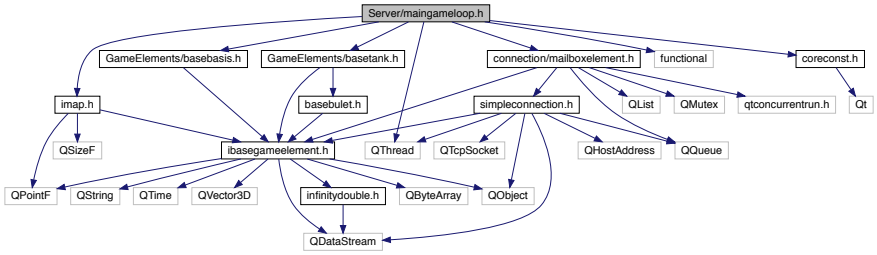
```
00019       break;
00020     }
00021   }
00022   if (!isAlredyExist) {
00023     currentGamer->name = codingNum(gamersList.size());
00024     currentGamer->sender = receiver;
00025     gamersList.append(currentGamer);
00026
00027     gamersItems.append(new QList<IBaseGameElement*>());
00028
00029     BaseBasis* basis = new BaseBasis();
00030     basis->setPosition(&currentGamer->position);
00031     basis->setEnergy(currentGamer->lifeCount);
00032     map->insertElement(basis, basis->getPosition());
00033
00034     gamersItems.last()->append(basis);
00035     currentGamer->minion = gamersItems.last();
00036   }
00037 }
00038
00039 void MainGameLoop::proccessWatcherMessage(
      MailReceiver::mailMessage* msg,
00040                                           MailSender* receiver) {
00041   if (!watchers.contains(receiver))
00042     watchers.append(receiver);
00043   QList<DiffElement*>* diff;
00044   switch (msg->message->messageType) {
00045     case eFirstMessae:
00046       diff = getAllMapAsDiff();
00047       receiver->receiveResponce(diff, msg->message);
00048       break;
00049     case eGetUpdateMessage:
00050       Q_ASSERT_X(2 * 2 != 4, "implementation", "not implemented yet");
00051       break;
00052     default:
00053       receiver->receiveResponce(diff, msg->message);
00054   }
00055 }
00056
00057 MainGameLoop::messageProccessor
      MainGameLoop::getProccessorForMessage(
00058     eConnectionType type) {
00059   switch (type) {
00060     case eGamer:
00061       return &MainGameLoop::proccessGamerMessage;
00062     case eWatcher:
00063       return &MainGameLoop::proccessWatcherMessage;
00064   }
00065   Q_ASSERT_X(2 * 2 != 4, "missing branch ",
00066              "missing brunch for eConnectionType");
00067 }
00068
00069 void MainGameLoop::run() {
00070   mailMessage* msg;
00071   qInfo() << "starting main loop";
00072   while (isWork) {
00073     while ((msg = nextMessage()) != nullptr) {
00074       qInfo() << TAG << "receive new messahe " << (*msg);
00075       auto fun = getProccessorForMessage(msg->message->connectionType);
00076       (this->*fun)(msg, msg->sender);
00077     }
00078     msleep(100);
00079   }
00080 }
00081
00082 QList<DiffElement*>* MainGameLoop::getAllMapAsDiff() {
00083   QList<DiffElement*>* result = new QList<DiffElement*>();
00084   DiffElement* item;
00085   for (int i = 0; i < map->getCount(); i++)
00086     result->append(
00087         new DiffElement(eDiffType::eNew, map->getElementAtPosition(i)));
00088   return result;
00089 }
00090
00091 MainGameLoop::~MainGameLoop() {
00092   delete map;
00093   for (int i = 0; i < gamersItems.size(); i++) {
00094     delete gamersItems.at(i);
00095   }
00096 }
00097
00098 void MainGameLoop::startLooper() {
00099   this->start();
00100 }
00101
00102 MainGameLoop* MainGameLoop::instance = nullptr;
00103
```

```
00104 MainGameLoop::GamerInformation::
      GamerInformation(IMap* map) {
00105   name = "";
00106   lifeCount = limits::defaultBasisEnergy;
00107   QSizeF* size = map->getSize();
00108   float rand1 = static_cast<float>(rand()) / static_cast<float>(RAND_MAX);
00109   float rand2 = static_cast<float>(rand()) / static_cast<float>(RAND_MAX);
00110   position = QVector3D(size->width() * rand1, size->height() * rand2, 0);
00111
00112   personalDiff = new QList<DiffElement*>();
00113 }
00114
00115 MainGameLoop::GamerInformation::~
      GamerInformation() {
00116   delete personalDiff;
00117 }
```

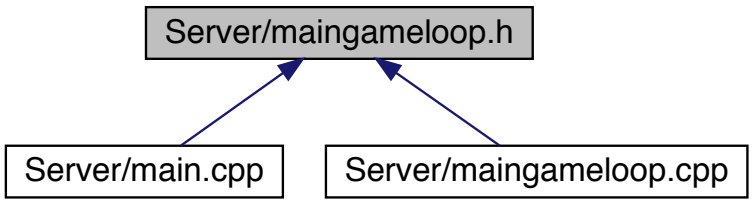
3.63 Server/maingameloop.h File Reference

```
#include "imap.h"
#include "connection/mailboxelement.h"
#include <QThread>
#include <functional>
#include "GameElements/basetank.h"
#include "coreconst.h"
#include "GameElements/basebasis.h"
```

Include dependency graph for maingameloop.h:



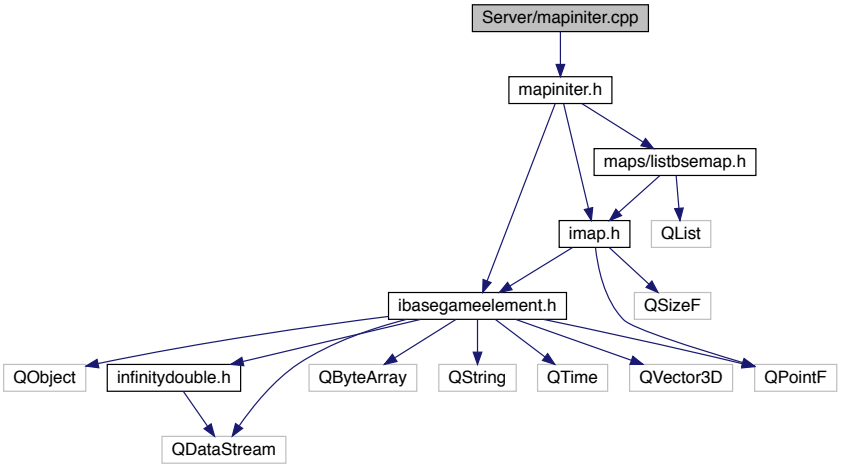
This graph shows which files directly or indirectly include this file:



Classes

- class MainGameLoop
- class MainGameLoop::GamerInformation

Include dependency graph for mapiniter.cpp:



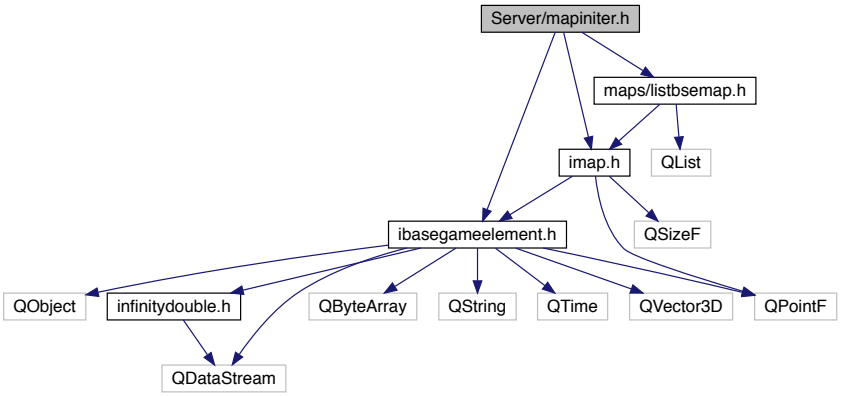
3.66 mapiniter.cpp

```
00001 #include "mapiniter.h"
00002
00003 MapIniter::MapIniter() {}
00004
00005 IMap* MapIniter::initSimapleMap() {
00006   const int mapW = 1000;
00007   const int mapH = 1000;
00008
00009   IMap* result = new ListBseMap(mapW, mapH);
00010   for (int i = 0; i < 1000; i++) {
00011     IBaseGameElement* item = getRandomGrass(mapW
      , mapH);
00012     result->insertElement(item, *item->getPosition());
00013   }
00014   return result;
00015 }
00016
00017 IBaseGameElement* MapIniter::getRandomGrass(double
      maxWidth, double maxHeigth) {
00018   IBaseGameElement* result = new IBaseGameElement();
00019   result->setPosition(
00020       new QVector3D((double)rand() / (double)RAND_MAX * maxWidth,
00021                     (double)rand() / (double)RAND_MAX * maxHeigth, 0));
00022   result->setHelth(InfinityDouble::InfinityValue
      ());
00023   result->setTransitWeight(InfinityDouble
      ::FromValue(0));
00024   result->setType(eBaseGameElementType::
      eGrass);
00025   return result;
00026 }
```

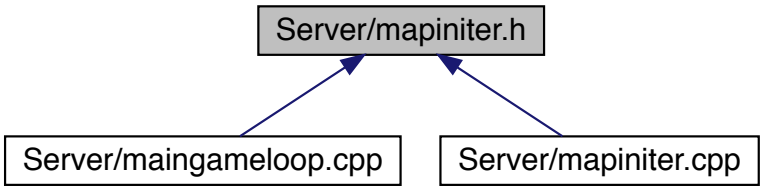
3.67 Server/mapiniter.h File Reference

```
#include "imap.h"
#include "maps/listbsemap.h"
```

`#include "ibasegameelement.h"`
Include dependency graph for mapiniter.h:



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



Classes

- class MapIniter

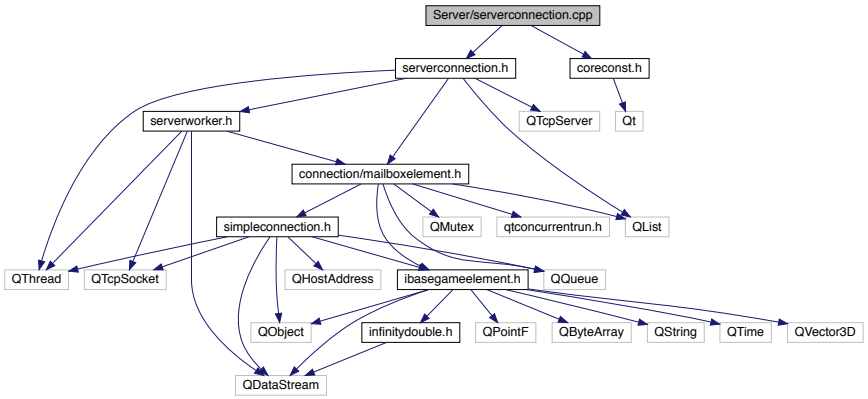
3.68 mapiniter.h

```
00001 #ifndef MAPINITER_H
00002 #define MAPINITER_H
00003
00004 #include "imap.h"
00005 #include "maps/listbsemap.h"
00006 #include "ibasegameelement.h"
00007
00008 class MapIniter
00009 {
00010 public:
00011     MapIniter();
00012     IMap *initSimapleMap();
00013
00014 protected:
00015     IBaseGameElement *getRandomGrass(double maxWidth,double maxHeigth);
00016
00017 };
00018
00019 #endif // MAPINITER_H
```

3.69 Server/serverconnection.cpp File Reference

```
#include "serverconnection.h"
#include "coreconst.h"
```

Include dependency graph for serverconnection.cpp:



3.70 serverconnection.cpp

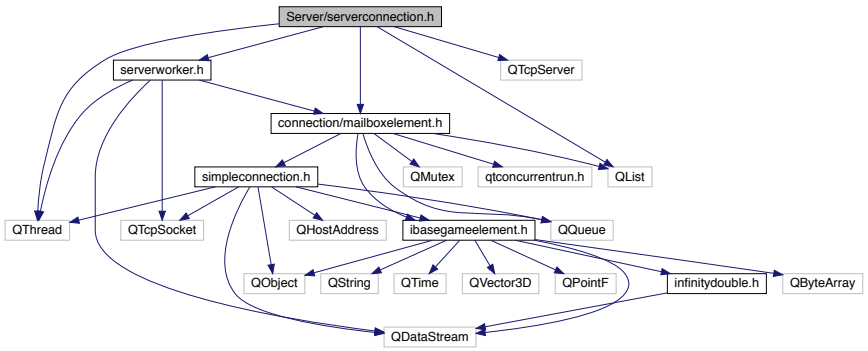
```
00001 #include "serverconnection.h"
00002 #include "coreconst.h"
00003
00004 ServerConnection* ServerConnection::
      instance = nullptr;
00005
00006 ServerConnection::~ServerConnection() {
00007   for (int i = 0; i < connections->size(); i++)
00008     delete connections->at(i);
00009   delete connections;
00010 }
00011
00012 ServerConnection::ServerConnection() {
00013   connections = new QList<ServerWorker*>();
00014 }
00015
00016 void ServerConnection::onWorkerStop(ServerWorker* worker) {
00017   connections->removeAll(worker);
00018   delete worker;
00019 }
00020
00021 void ServerConnection::startServer() {
00022   if (this->isRunning())
00023     return;
00024   this->start(Priority::NormalPriority);
00025 }
00026
00027 void ServerConnection::run() {
00028   server = new QTcpServer();
00029   if (!server->listen(QHostAddress::Any, DefaultServerParams::port)) {
00030     emit onServerError(serverError::canNotStartServer);
00031     return;
00032   }
00033   qDebug() << server->isListening();
00034   bool isAppareNewConnection;
00035   while (true) {
00036     isAppareNewConnection = server->waitForNewConnection(100);
00037     if (isAppareNewConnection) {
00038       QTcpSocket* newConnection = server->nextPendingConnection();
00039       if (newConnection == nullptr)
00040         continue;
00041       qInfo() << "appare new connection  :: "
00042               << newConnection->peerAddress().toString();
00043       ServerWorker* newWorker = new ServerWorker(newConnection);
00044       newWorker->setDefaultReceiver(this->receiver);
00045       connections->append(newWorker);
```

```
00046       newWorker->start();
00047     }
00048   }
00049 }
00050
00051 void ServerConnection::receiveResponce(QList<DiffElement*>* diff,
00052                                        MessageForServer* message) {}
```

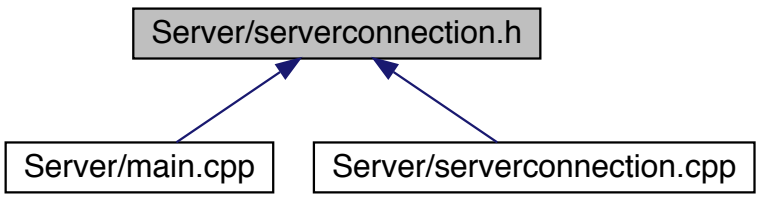
3.71 Server/serverconnection.h File Reference

```
#include <QThread>
#include <QTcpServer>
#include <QList>
#include "serverworker.h"
#include "connection/mailboxelement.h"
```

Include dependency graph for serverconnection.h:



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



Classes

- class ServerConnection

Enumerations

- enum serverError { canNotStartServer }

3.71.1 Enumeration Type Documentation

3.71.1.1 serverError

```
enum serverError
```

Enumerator

canNotStartServer	

Definition at line 10 of file serverconnection.h.

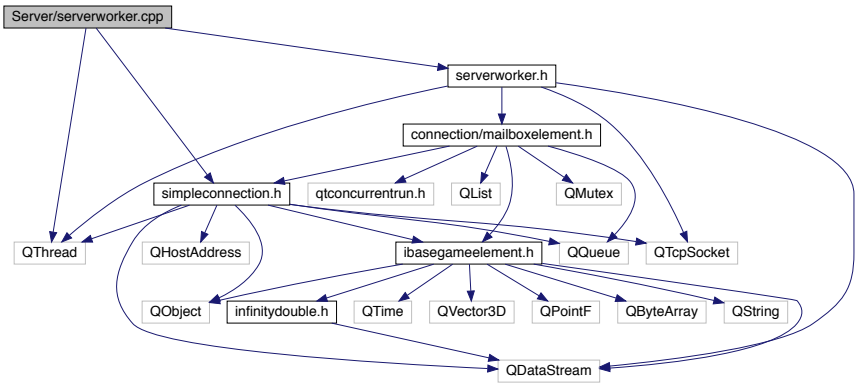
3.72 serverconnection.h

```
00001 #ifndef SERVERCONNECTION_H
00002 #define SERVERCONNECTION_H
00003
00004 #include <QThread>
00005 #include <QTcpServer>
00006 #include <QList>
00007 #include "serverworker.h"
00008 #include "connection/mailboxelement.h"
00009
00010 enum serverError { canNotStartServer };
00011
00012 class ServerConnection : public QThread, public
      MailSender {
00013  public:
00014   Q_OBJECT
00015
00016  public:
00017   ~ServerConnection();
00018   inline static ServerConnection* getServerInstance() {
00019     if (instance == nullptr)
00020       instance = new ServerConnection();
00021     return instance;
00022   }
00023   void startServer();
00024
00025  signals:
00026   void onServerError(serverError error);
00027
00028  protected:
00029   static ServerConnection* instance;
00030   ServerConnection();
00031   QTcpServer* server;
00032   QList<ServerWorker*>* connections;
00033  protected slots:
00034   void onWorkerStop(ServerWorker* worker);
00035
00036   // QThread interface
00037  protected:
00038   void run();
00039
00040   // MailSender interface
00041  public:
00042   void receiveResponce(QList<DiffElement*>*
      diff, MessageForServer* message);
00043 };
00044
00045 #endif  // SERVERCONNECTION_H
```

3.73 Server/serverworker.cpp File Reference

```
#include "serverworker.h"
#include <connection/simpleconnection.h>
```

`#include <QThread>`
Include dependency graph for serverworker.cpp:



Classes

- class receiveRespnceThread

3.74 serverworker.cpp

```
00001 #include "serverworker.h"
00002
00003 #include <connection/simpleconnection.h>
00004 #include <QThread>
00005
00006 ServerWorker::ServerWorker(QTcpSocket* socket) {
00007   this->socket = socket;
00008 }
00009
00010 void ServerWorker::run() {
00011   out = new QDataStream(socket);
00012   out->setVersion(QDataStream::Qt_5_7);
00013
00014   ReceiverThread* th = new ReceiverThread(this, socket);
00015
00016   while (this->isWork) {
00017     workerMutex.lock();
00018     while (responceMessage.length() > 0) {
00019       responceData data = responceMessage.takeFirst();
00020       qInfo() << "getting some message for sending";
00021       out->startTransaction();
00022       (*out) << (*data.messag);
00023       (*out) << data.diff->length();
00024       for (int i = 0; i < data.diff->length(); i++)
00025         (*out) << (*data.diff->at(i));
00026       out->commitTransaction();
00027       socket->flush();
00028       qInfo() << "send  response to client";
00029       for (int i = 0; i < data.diff->length(); i++) {
00030         delete data.diff->at(i);
00031         data.diff->removeAt(i);
00032         i--;
00033       }
00034       delete data.diff;
00035       delete data.messag;
00036     }
00037     workerMutex.unlock();
00038   }
00039   th->stop();
00040   delete th;
00041   qInfo() << "worker has stoped";
00042   emit onStop(this);
00043 }
00044
```

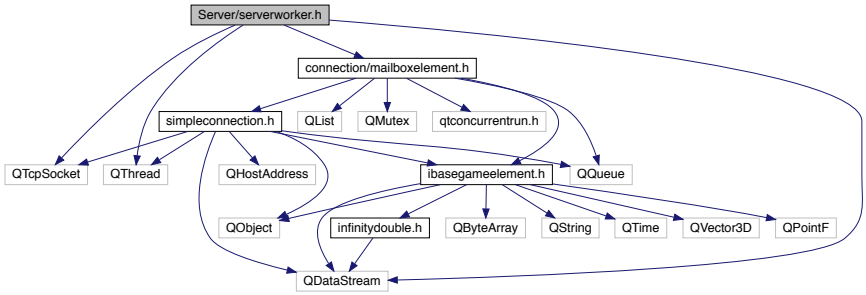
```
00045 class receiveRespnceThread : public QThread {
00046  public:
00047   QMutex* mutex;
00048   QList<DiffElement*>* diff;
00049   MessageForServer* message;
00050   QQueue<ServerWorker::responceData>*
      responceMessage;
00051
00052   // QThread interface
00053  protected:
00054   void run() {
00055     mutex->lock();
00056     responceMessage->push_back(ServerWorker::responceData(diff, message));
00057     mutex->unlock();
00058   }
00059 };
00060
00061 void ServerWorker::receiveResponce(QList<DiffElement*>* diff,
00062                                    MessageForServer* message) {
00063   qInfo() << "getting responce for client from map loop";
00064
00065   //  receiveRespnceThread r;
00066   //  r.mutex = workerMutex;
00067
00068   QtConcurrent::run(QThreadPool::globalInstance(), [=] {
00069     qInfo() << "ServerWorker::receiveResponce - befor mutex";
00070     workerMutex.lock();
00071     qInfo() << "ServerWorker::receiveResponce - in mutex";
00072     responceMessage.push_back(responceData(diff, message));
00073     workerMutex.unlock();
00074     qInfo() << "ServerWorker::receiveResponce - after mutex";
00075   });
00076 }
00077
00078 ServerWorker::ReceiverThread::ReceiverThread(ServerWorker* parent,
00079                                              QTcpSocket* socket) {
00080   this->socket = socket;
00081   this->parentThread = parent;
00082   this->start();
00083 }
00084
00085 ServerWorker::ReceiverThread::~ReceiverThread() {
00086   delete in;
00087 }
00088
00089 void ServerWorker::ReceiverThread::stop() {
00090   receiverMutex.lock();
00091   isWork = false;
00092   receiverMutex.unlock();
00093 }
00094
00095 void ServerWorker::ReceiverThread::run() {
00096   in = new QDataStream(socket);
00097   in->setVersion(QDataStream::Qt_5_7);
00098
00099   qInfo() << "starting receiving thread";
00100
00101   while (true) {
00102     receiverMutex.lock();
00103     socket->waitForReadyRead(-1);
00104     if (socket->state() == QTcpSocket::UnconnectedState ||
00105         socket->state() == QAbstractSocket::ClosingState)
00106       isWork = false;
00107     if (!isWork) {
00108       qInfo() << "stoping receiver loop";
00109       break;
00110     }
00111     qInfo() << "ready read from client";
00112     MessageForServer* newMessage = new MessageForServer();
00113     (*in) >> (*newMessage);
00114     qInfo() << "get new message on server :: " << (*newMessage).toString();
00115     parentThread->receiver->sendMail(newMessage, parentThread,
00116                                      newMessage->messageType);
00117     receiverMutex.unlock();
00118   }
00119 }
```

3.75 Server/serverworker.h File Reference

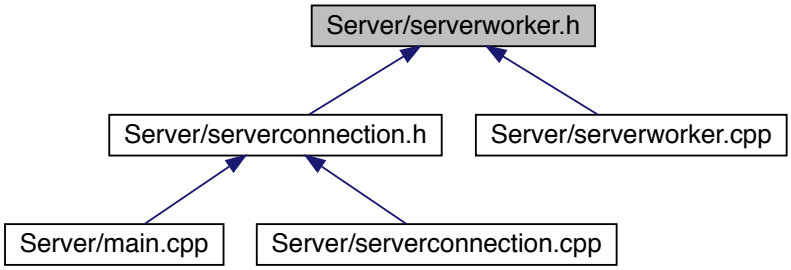
```
#include <QTcpSocket>
#include <QThread>
```

```
#include <QDataStream>
#include <connection/mailboxelement.h>
```

Include dependency graph for serverworker.h:



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



Classes

- class ServerWorker
- struct ServerWorker::responceData
- class ServerWorker::ReceiverThread

3.76 serverworker.h

```
00001 #ifndef SERVERWORKER_H
00002 #define SERVERWORKER_H
00003
00004 #include <QTcpSocket>
00005 #include <QThread>
00006 #include <QDataStream>
00007
00008 #include <connection/mailboxelement.h>
00009
00010 class ServerWorker : public QThread, public
      MailSender {
00011   Q_OBJECT
00012   friend class receiveRespnceThread;
00013
00014  public:
00015   ServerWorker(QTcpSocket* socket);
00016  signals:
```

```
00017   void onStop(ServerWorker* worker);
00018
00019   // QThread interface
00020  protected:
00021   void run();
00022
00023  protected:
00024   struct responceData {
00025    public:
00026     responceData(QList<DiffElement*>* diff,
      MessageForServer* messag) {
00027       this->diff = diff;
00028       this->messag = messag;
00029     }
00030
00031     QList<DiffElement*>* diff;
00032     MessageForServer* messag;
00033   };
00034
00035   QTcpSocket* socket;
00036   volatile bool isWork = true;
00037   QDataStream* out;
00038   QMutex workerMutex;
00039   QQueue<responceData> responceMessage;
00040
00041   class ReceiverThread : public QThread {
00042    public:
00043     ReceiverThread(ServerWorker* parent, QTcpSocket* socket);
00044     virtual ~ReceiverThread();
00045     void stop();
00046
00047     // QThread interface
00048    protected:
00049     void run();
00050     QTcpSocket* socket;
00051     ServerWorker* parentThread;
00052     QDataStream* in;
00053     bool isWork = true;
00054     QMutex receiverMutex;
00055   };
00056
00057   // MailSender interface
00058  public:
00059   void receiveResponce(QList<DiffElement*>*
      diff, MessageForServer* message);
00060 };
00061
00062 #endif  // SERVERWORKER_H
```