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

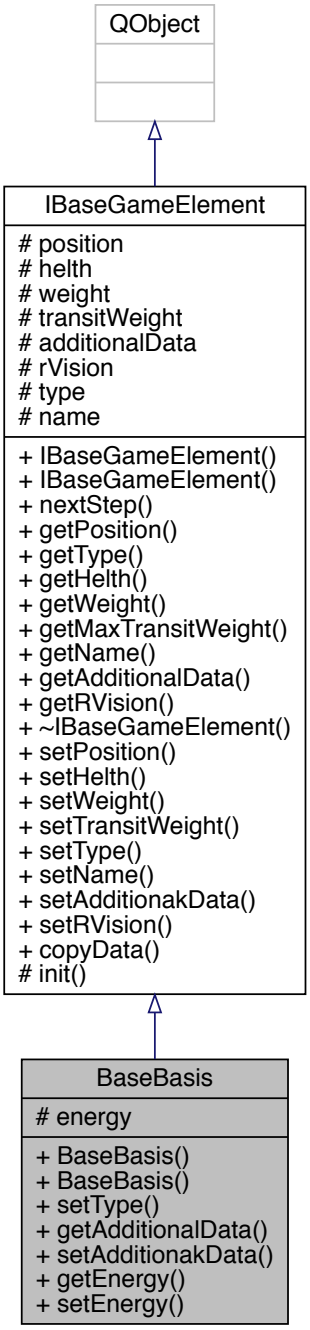
Chapter 2

Class Documentation

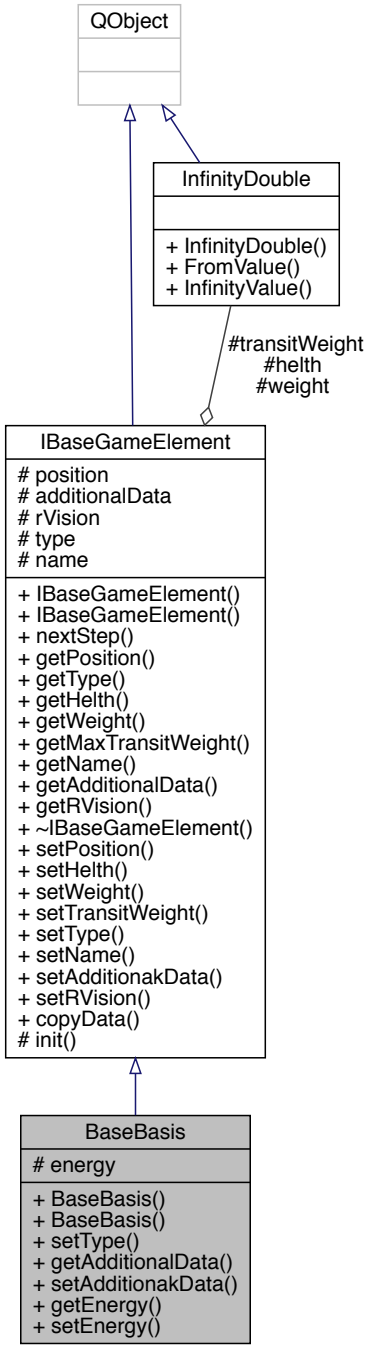
2.1 BaseBasis Class Reference

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

Inheritance diagram for BaseBasis:



Collaboration diagram for BaseBasis:



Public Member Functions

- BaseBasis ()
- BaseBasis (GameElementData &data)
- virtual void setType (int value) override
- virtual QByteArray * getAdditionalData () const
- virtual void setAdditionakData (QByteArray *data) override

- virtual int getEnergy () const
- virtual void setEnergy (int _enery)
- virtual void nextStep ()
- virtual QVector3D * getPosition () const
- virtual int getType () const
- virtual InfinityDouble * getHelth () const
- virtual InfinityDouble * getWeight () const
- virtual InfinityDouble * getMaxTransitWeight () const
- virtual QString getName () const
- virtual int getRVision () const
- 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)
- void copyData (GameElementData &out)

Protected Member Functions

- virtual void init (GameElementData &data)

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() [1/2]

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

Definition at line 4 of file basebasis.cpp.

References eBasis, IBaseGameElement::nextStep(), and IBaseGameElement::type.

```
00004                     : IBaseGameElement() {
00005   this->type = eBasis;
00006   helth = InfinityDouble::InfinityValue();
00007   weight = InfinityDouble::InfinityValue();
00008   transitWeight = InfinityDouble::InfinityValue();
00009 }
```

Here is the call graph for this function:

2.1.2.2 BaseBasis() [2/2]

```
BaseBasis::BaseBasis (
            GameElementData & data )
```

Definition at line 11 of file basebasis.cpp.

References eBasis, IBaseGameElement::nextStep(), and IBaseGameElement::type.

```
00011                                       : IBaseGameElement() {
00012   init(data);
00013   this->type = eBasis;
00014   helth = InfinityDouble::InfinityValue();
00015   weight = InfinityDouble::InfinityValue();
00016   transitWeight = InfinityDouble::InfinityValue();
00017 }
```

Here is the call graph for this function:

2.1.3 Member Function Documentation

2.1.3.1 copyData()

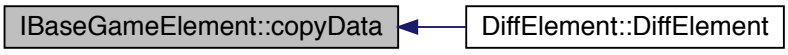
```
void IBaseGameElement::copyData (
            GameElementData & out )  [inherited]
```

Definition at line 27 of file ibasegameelement.cpp.

Referenced by DiffElement::DiffElement().

```
00027                                                   {
00028   out.position = new QVector3D(*position);
00029   out.helth = new InfinityDouble(*helth);
00030   out.weight = new InfinityDouble(*weight);
00031   out.transitWeight = new InfinityDouble(*
      transitWeight);
00032   if (additionalData == nullptr)
00033     out.additionalData = nullptr;
00034   else
00035     out.additionalData = new QByteArray(*additionalData);
00036   out.type = type;
00037   out.name = name;
00038   out.rVision = rVision;
00039 }
```

Here is the caller graph for this function:



2.1.3.2 getAdditionalData()

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

Reimplemented from IBaseGameElement.

Definition at line 26 of file basebasis.cpp.

```
00026                                               {
00027   additionalData->clear();
00028   QDataStream stream(additionalData, QIODevice::WriteOnly);
00029   stream << energy;
00030 }
```

2.1.3.3 getEnergy()

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

Definition at line 16 of file basebasis.h.

References energy.

```
00016 { return energy; }
```

2.1.3.4 getHelth()

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

Definition at line 64 of file ibasegameelement.h.

```
00064 { return helth; }
```

2.1.3.5 getMaxTransitWeight()

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

Definition at line 66 of file ibasegameelement.h.

```
00066 { return transitWeight; }
```

2.1.3.6 getName()

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

Definition at line 67 of file ibasegameelement.h.

```
00067 { return name; }
```

2.1.3.7 getPosition()

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

Definition at line 62 of file ibasegameelement.h.

```
00062 { return position; }
```

2.1.3.8 getRVision()

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

Definition at line 69 of file ibasegameelement.h.

References IBaseGameElement::rVision.

```
00069 { return rVision; }
```

2.1.3.9 getType()

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

Definition at line 63 of file ibasegameelement.h.

References IBaseGameElement::type.

```
00063 { return type; }
```

2.1.3.10 getWeight()

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

Definition at line 65 of file ibasegameelement.h.

```
00065 { return weight; }
```

2.1.3.11 init()

```
void IBaseGameElement::init (
            GameElementData & data )  [protected], [virtual], [inherited]
```

Definition at line 41 of file ibasegameelement.cpp.

```
00041                                                {
00042   setHelth(new InfinityDouble(*data.helth));
00043   setWeight(new InfinityDouble(*data.weight));
00044   setTransitWeight(new InfinityDouble(*data.
      transitWeight));
00045   setPosition(new QVector3D(*data.position));
00046   setName(data.name);
00047   setAdditionakData(new QByteArray(*data.additionalData));
00048   setRVision(data.rVision);
00049   setType(data.type);
00050 }
```

2.1.3.12 nextStep()

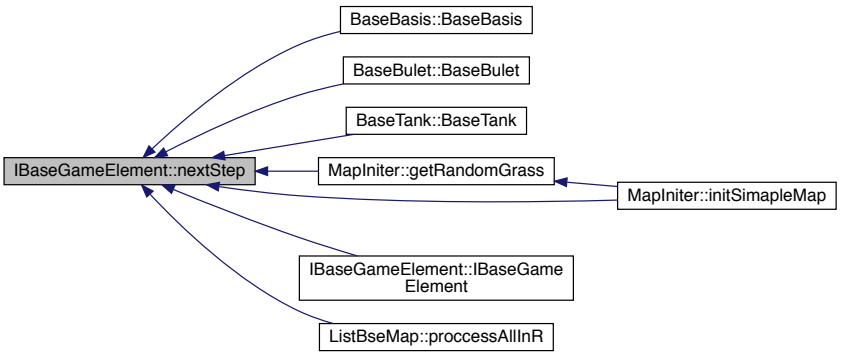
```
virtual void IBaseGameElement::nextStep ( )  [inline], [virtual], [inherited]
```

Definition at line 60 of file ibasegameelement.h.

Referenced by BaseBasis(), BaseBulet::BaseBulet(), BaseTank::BaseTank(), MapIniter::getRandomGrass(), IBaseGameElement::IBaseGameElement(), MapIniter::initSimapleMap(), and ListBseMap::proccessAllInR().

```
00060 {};
```

Here is the caller graph for this function:



2.1.3.13 setAdditionakData()

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

Reimplemented from IBaseGameElement.

Definition at line 32 of file basebasis.cpp.

```
00032                                                {
00033   QDataStream stream(*data);
00034   stream >> energy;
00035   IBaseGameElement::setAdditionakData(data);
00036 }
```

2.1.3.14 setEnergy()

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

Definition at line 17 of file basebasis.h.

References energy.

```
00017 { this->energy = _enery; }
```

2.1.3.15 setHelth()

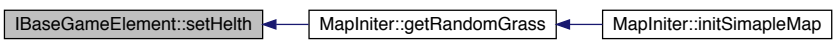
```
virtual void IBaseGameElement::setHelth (
            InfinityDouble * value )  [inline], [virtual], [inherited]
```

Definition at line 87 of file ibasegameelement.h.

Referenced by MapIniter::getRandomGrass().

```
00087 { this->helth = value; }
```

Here is the caller graph for this function:



2.1.3.16 setName()

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

Definition at line 93 of file ibasegameelement.h.

```
00093 { this->name = name; }
```

2.1.3.17 setPosition()

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

Definition at line 86 of file ibasegameelement.h.

```
00086 { this->position = value; }
```

2.1.3.18 setRVision()

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

Definition at line 97 of file ibasegameelement.h.

References IBaseGameElement::rVision.

```
00097 { rVision = _rVison; }
```

2.1.3.19 setTransitWeight()

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

Definition at line 89 of file ibasegameelement.h.

Referenced by MapIniter::getRandomGrass().

```
00089                                                   {
00090     this->transitWeight = value;
00091   }
```

Here is the caller graph for this function:

2.1.3.20 setType()

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

Reimplemented from IBaseGameElement.

Definition at line 19 of file basebasis.cpp.

References eBasis.

```
00019                                {
00020   if (value != ((int)eBasis)) {
00021     Q_ASSERT_X(false, "game logic", "Wrong type for basis");
00022   }
00023   IBaseGameElement::setType(value);
00024 }
```

2.1.3.21 setWeight()

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

Definition at line 88 of file ibasegameelement.h.

```
00088 { this->weight = value; }
```

2.1.4 Member Data Documentation

2.1.4.1 additionalData

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

Definition at line 109 of file ibasegameelement.h.

2.1.4.2 energy

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

Definition at line 20 of file basebasis.h.

Referenced by getEnergy(), and setEnergy().

2.1.4.3 helth

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

Definition at line 106 of file ibasegameelement.h.

2.1.4.4 name

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

Definition at line 113 of file ibasegameelement.h.

2.1.4.5 position

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

Definition at line 105 of file ibasegameelement.h.

2.1.4.6 rVision

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

Definition at line 110 of file ibasegameelement.h.

Referenced by IBaseGameElement::getRVision(), and IBaseGameElement::setRVision().

2.1.4.7 transitWeight

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

Definition at line 108 of file ibasegameelement.h.

2.1.4.8 type

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

Definition at line 112 of file ibasegameelement.h.

Referenced by BaseBasis(), BaseBulet::BaseBulet(), BaseTank::BaseTank(), IBaseGameElement::getType(), and IBaseGameElement::setType().

2.1.4.9 weight

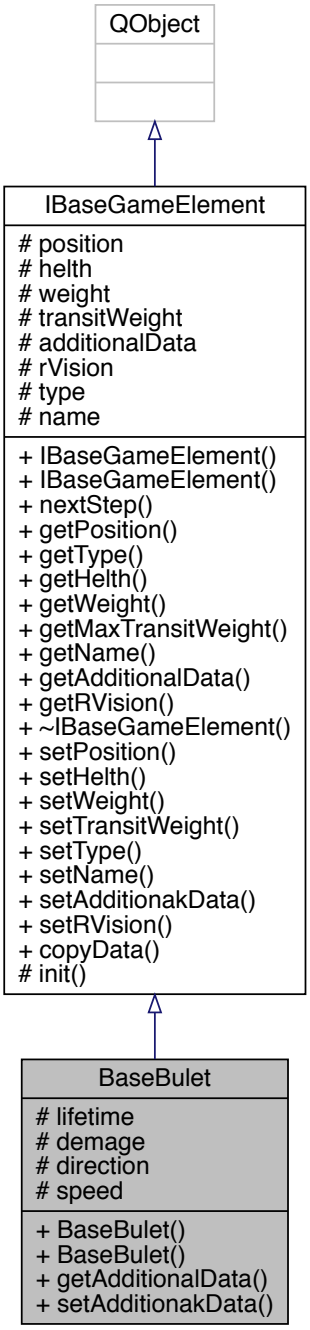
```
InfinityDouble* IBaseGameElement::weight = nullptr  [protected], [inherited]
```

Definition at line 107 of file ibasegameelement.h.

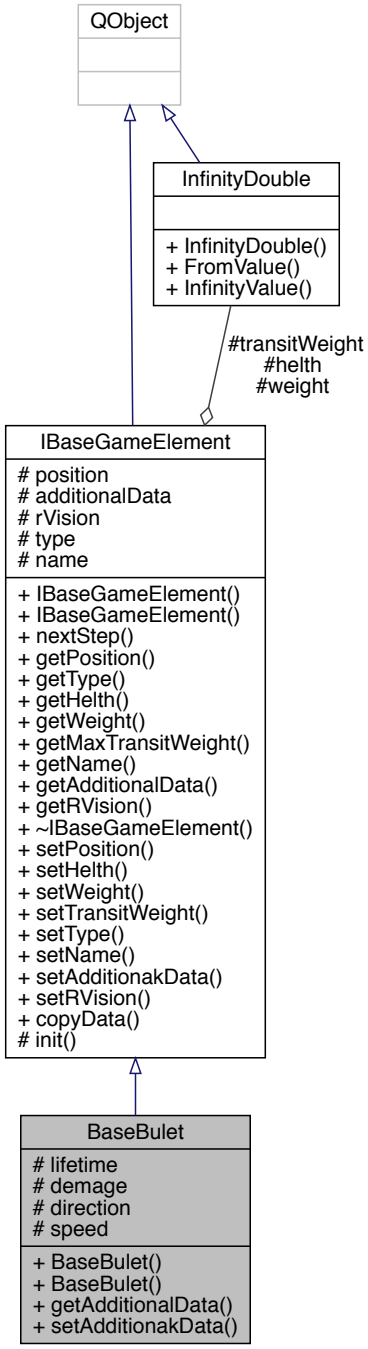
2.2 BaseBulet Class Reference

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

Inheritance diagram for BaseBulet:



Collaboration diagram for BaseBulet:



Public Member Functions

- BaseBulet ()
- BaseBulet (GameElementData &data)
- QByteArray * getAdditionalData () const
- void setAdditionakData (QByteArray *data) override
- virtual void nextStep ()

- virtual QVector3D * getPosition () const
- virtual int getType () const
- virtual InfinityDouble * getHelth () const
- virtual InfinityDouble * getWeight () const
- virtual InfinityDouble * getMaxTransitWeight () const
- virtual QString getName () const
- virtual int getRVision () const
- 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)
- void copyData (GameElementData &out)

Protected Member Functions

- virtual void init (GameElementData &data)

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() [1/2]

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

Definition at line 3 of file basebulet.cpp.

References eBullet, IBaseGameElement::nextStep(), and IBaseGameElement::type.

```
00003                     : IBaseGameElement() {
00004   this->type = eBullet;
00005 }
```

Here is the call graph for this function:

2.2.2.2 BaseBulet() [2/2]

```
BaseBulet::BaseBulet (
            GameElementData & data )
```

Definition at line 7 of file basebulet.cpp.

References eBullet, IBaseGameElement::nextStep(), and IBaseGameElement::type.

```
00007                                       : IBaseGameElement() {
00008   init(data);
00009   this->type = eBullet;
00010 }
```

Here is the call graph for this function:

2.2.3 Member Function Documentation

2.2.3.1 copyData()

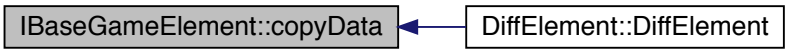
```
void IBaseGameElement::copyData (
            GameElementData & out )  [inherited]
```

Definition at line 27 of file ibasegameelement.cpp.

Referenced by DiffElement::DiffElement().

```
00027                                                  {
00028   out.position = new QVector3D(*position);
00029   out.helth = new InfinityDouble(*helth);
00030   out.weight = new InfinityDouble(*weight);
00031   out.transitWeight = new InfinityDouble(*
      transitWeight);
00032   if (additionalData == nullptr)
00033     out.additionalData = nullptr;
00034   else
00035     out.additionalData = new QByteArray(*additionalData);
00036   out.type = type;
00037   out.name = name;
00038   out.rVision = rVision;
00039 }
```

Here is the caller graph for this function:



2.2.3.2 getAdditionalData()

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

Reimplemented from IBaseGameElement.

Definition at line 12 of file basebulet.cpp.

```
00012 {}
```

2.2.3.3 getHelth()

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

Definition at line 64 of file ibasegameelement.h.

```
00064 { return helth; }
```

2.2.3.4 getMaxTransitWeight()

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

Definition at line 66 of file ibasegameelement.h.

```
00066 { return transitWeight; }
```

2.2.3.5 getName()

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

Definition at line 67 of file ibasegameelement.h.

```
00067 { return name; }
```

2.2.3.6 getPosition()

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

Definition at line 62 of file ibasegameelement.h.

```
00062 { return position; }
```

2.2.3.7 getRVision()

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

Definition at line 69 of file ibasegameelement.h.

References IBaseGameElement::rVision.

```
00069 { return rVision; }
```

2.2.3.8 getType()

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

Definition at line 63 of file ibasegameelement.h.

References IBaseGameElement::type.

```
00063 { return type; }
```

2.2.3.9 getWeight()

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

Definition at line 65 of file ibasegameelement.h.

```
00065 { return weight; }
```

2.2.3.10 init()

```
void IBaseGameElement::init (
            GameElementData & data )  [protected], [virtual], [inherited]
```

Definition at line 41 of file ibasegameelement.cpp.

```
00041                                                {
00042   setHelth(new InfinityDouble(*data.helth));
00043   setWeight(new InfinityDouble(*data.weight));
00044   setTransitWeight(new InfinityDouble(*data.
      transitWeight));
00045   setPosition(new QVector3D(*data.position));
00046   setName(data.name);
00047   setAdditionakData(new QByteArray(*data.additionalData));
00048   setRVision(data.rVision);
00049   setType(data.type);
00050 }
```

2.2.3.11 nextStep()

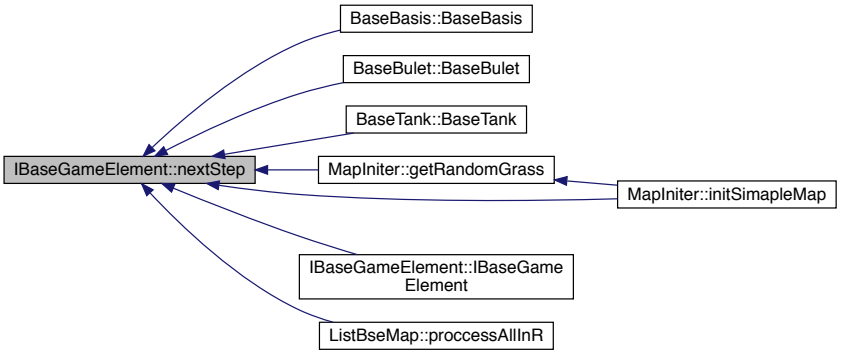
```
virtual void IBaseGameElement::nextStep ( )  [inline], [virtual], [inherited]
```

Definition at line 60 of file ibasegameelement.h.

Referenced by BaseBasis::BaseBasis(), BaseBulet(), BaseTank::BaseTank(), MapIniter::getRandomGrass(), IBaseGameElement::IBaseGameElement(), MapIniter::initSimapleMap(), and ListBseMap::proccessAllInR().

```
00060 {};
```

Here is the caller graph for this function:



2.2.3.12 setAdditionakData()

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

Reimplemented from IBaseGameElement.

Definition at line 14 of file basebulet.cpp.

```
00014 {}
```

2.2.3.13 setHelth()

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

Definition at line 87 of file ibasegameelement.h.

Referenced by MapIniter::getRandomGrass().

```
00087 { this->helth = value; }
```

Here is the caller graph for this function:

2.2.3.14 setName()

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

Definition at line 93 of file ibasegameelement.h.

```
00093 { this->name = name; }
```

2.2.3.15 setPosition()

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

Definition at line 86 of file ibasegameelement.h.

```
00086 { this->position = value; }
```

2.2.3.16 setRVision()

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

Definition at line 97 of file ibasegameelement.h.

References IBaseGameElement::rVision.

```
00097 { rVision = _rVison; }
```

2.2.3.17 setTransitWeight()

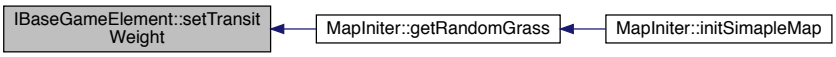
```
virtual void IBaseGameElement::setTransitWeight (
            InfinityDouble * value )  [inline], [virtual], [inherited]
```

Definition at line 89 of file ibasegameelement.h.

Referenced by MapIniter::getRandomGrass().

```
00089                                                    {
00090     this->transitWeight = value;
00091   }
```

Here is the caller graph for this function:



2.2.3.18 setType()

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

Reimplemented in BaseTank, and BaseBasis.

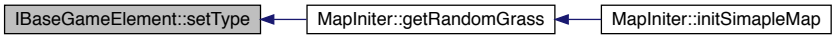
Definition at line 92 of file ibasegameelement.h.

References IBaseGameElement::type.

Referenced by MapIniter::getRandomGrass().

```
00092 { this->type = value; }
```

Here is the caller graph for this function:



2.2.3.19 setWeight()

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

Definition at line 88 of file ibasegameelement.h.

```
00088 { this->weight = value; }
```

2.2.4 Member Data Documentation

2.2.4.1 additionalData

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

Definition at line 109 of file ibasegameelement.h.

2.2.4.2 demage

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

Definition at line 13 of file basebulet.h.

2.2.4.3 direction

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

Definition at line 14 of file basebulet.h.

2.2.4.4 helth

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

Definition at line 106 of file ibasegameelement.h.

2.2.4.5 lifetime

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

Definition at line 12 of file basebulet.h.

2.2.4.6 name

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

Definition at line 113 of file ibasegameelement.h.

2.2.4.7 position

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

Definition at line 105 of file ibasegameelement.h.

2.2.4.8 rVision

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

Definition at line 110 of file ibasegameelement.h.

Referenced by IBaseGameElement::getRVision(), and IBaseGameElement::setRVision().

2.2.4.9 speed

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

Definition at line 15 of file basebulet.h.

2.2.4.10 transitWeight

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

Definition at line 108 of file ibasegameelement.h.

2.2.4.11 type

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

Definition at line 112 of file ibasegameelement.h.

Referenced by BaseBasis::BaseBasis(), BaseBulet(), BaseTank::BaseTank(), IBaseGameElement::getType(), and IBaseGameElement::setType().

2.2.4.12 weight

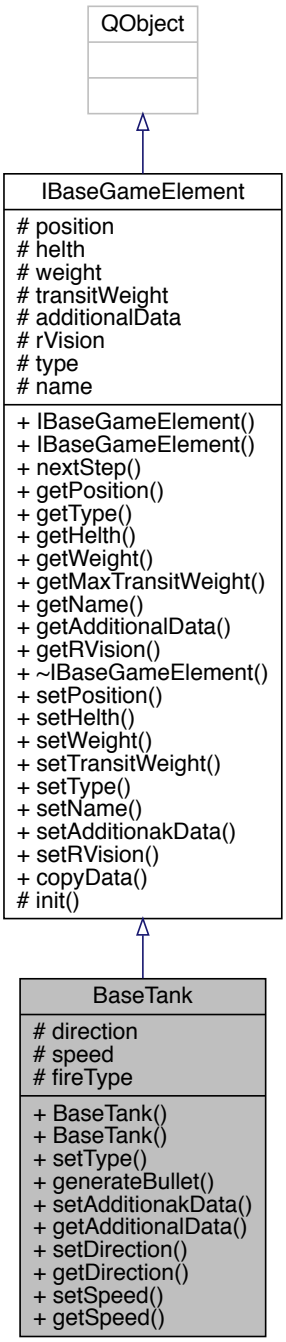
```
InfinityDouble* IBaseGameElement::weight = nullptr  [protected], [inherited]
```

Definition at line 107 of file ibasegameelement.h.

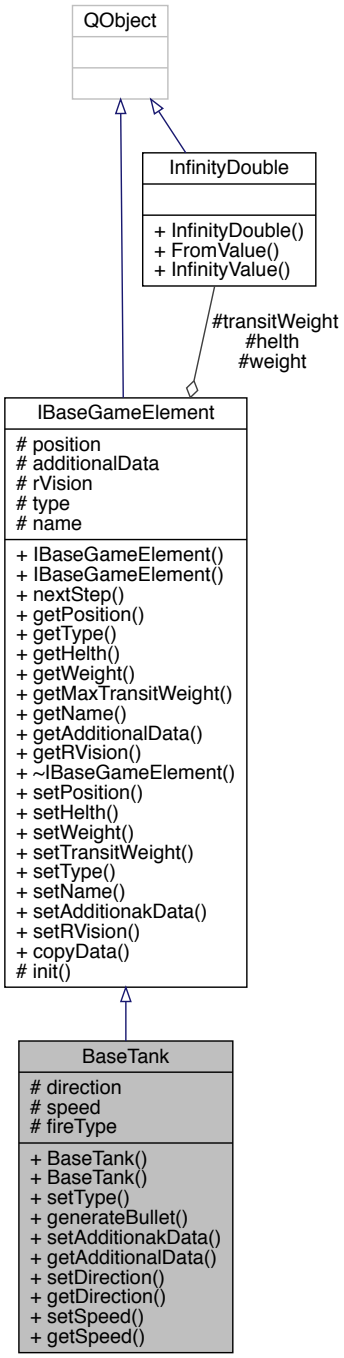
2.3 BaseTank Class Reference

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

Inheritance diagram for BaseTank:



Collaboration diagram for BaseTank:



Public Member Functions

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

- virtual QByteArray * getAdditionalData () const
- virtual void setDirection (double _direction)
- virtual double getDirection () const
- virtual void setSpeed (double _speed)
- virtual double getSpeed () const
- virtual void nextStep ()
- virtual QVector3D * getPosition () const
- virtual int getType () const
- virtual InfinityDouble * getHelth () const
- virtual InfinityDouble * getWeight () const
- virtual InfinityDouble * getMaxTransitWeight () const
- virtual QString getName () const
- virtual int getRVision () const
- 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)
- void copyData (GameElementData &out)

Protected Member Functions

- virtual void init (GameElementData &data)

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() [1/2]

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

Definition at line 4 of file basetank.cpp.

References eSimpleTank, IBaseGameElement::nextStep(), and IBaseGameElement::type.

```
00004                    : IBaseGameElement() {
00005   this->type = eSimpleTank;
00006 }
```

Here is the call graph for this function:

2.3.2.2 BaseTank() [2/2]

```
BaseTank::BaseTank (
            GameElementData & data )
```

Definition at line 8 of file basetank.cpp.

References eSimpleTank, IBaseGameElement::nextStep(), and IBaseGameElement::type.

```
00008                                       : IBaseGameElement() {
00009   init(data);
00010   this->type = eSimpleTank;
00011 }
```

Here is the call graph for this function:

2.3.3 Member Function Documentation

2.3.3.1 copyData()

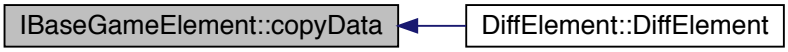
```
void IBaseGameElement::copyData (
            GameElementData & out )  [inherited]
```

Definition at line 27 of file ibasegameelement.cpp.

Referenced by DiffElement::DiffElement().

```
00027                                                 {
00028   out.position = new QVector3D(*position);
00029   out.helth = new InfinityDouble(*helth);
00030   out.weight = new InfinityDouble(*weight);
00031   out.transitWeight = new InfinityDouble(*
      transitWeight);
00032   if (additionalData == nullptr)
00033     out.additionalData = nullptr;
00034   else
00035     out.additionalData = new QByteArray(*additionalData);
00036   out.type = type;
00037   out.name = name;
00038   out.rVision = rVision;
00039 }
```

Here is the caller graph for this function:



2.3.3.2 generateBullet()

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

Definition at line 20 of file basetank.cpp.

```
00020 {}
```

2.3.3.3 getAdditionalData()

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

Reimplemented from IBaseGameElement.

Definition at line 28 of file basetank.cpp.

```
00028                                           {
00029   this->additionalData->clear();
00030   QDataStream stream(this->additionalData, QIODevice::WriteOnly);
00031   stream << direction << speed << fireType;
00032   return additionalData;
00033 }
```

2.3.3.4 getDirection()

```
virtual double BaseTank::getDirection ( ) const  [inline], [virtual]
```

Definition at line 21 of file basetank.h.

References direction.

```
00021 { return direction; }
```

2.3.3.5 getHelth()

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

Definition at line 64 of file ibasegameelement.h.

```
00064 { return helth; }
```

2.3.3.6 getMaxTransitWeight()

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

Definition at line 66 of file ibasegameelement.h.

```
00066 { return transitWeight; }
```

2.3.3.7 getName()

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

Definition at line 67 of file ibasegameelement.h.

```
00067 { return name; }
```

2.3.3.8 getPosition()

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

Definition at line 62 of file ibasegameelement.h.

```
00062 { return position; }
```

2.3.3.9 getRVision()

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

Definition at line 69 of file ibasegameelement.h.

References IBaseGameElement::rVision.

```
00069 { return rVision; }
```

2.3.3.10 getSpeed()

```
virtual double BaseTank::getSpeed ( ) const  [inline], [virtual]
```

Definition at line 24 of file basetank.h.

References speed.

```
00024 { return speed; }
```

2.3.3.11 getType()

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

Definition at line 63 of file ibasegameelement.h.

References IBaseGameElement::type.

```
00063 { return type; }
```

2.3.3.12 getWeight()

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

Definition at line 65 of file ibasegameelement.h.

```
00065 { return weight; }
```

2.3.3.13 init()

```
void IBaseGameElement::init (
            GameElementData & data )  [protected], [virtual], [inherited]
```

Definition at line 41 of file ibasegameelement.cpp.

```
00041                                                {
00042   setHelth(new InfinityDouble(*data.helth));
00043   setWeight(new InfinityDouble(*data.weight));
00044   setTransitWeight(new InfinityDouble(*data.
      transitWeight));
00045   setPosition(new QVector3D(*data.position));
00046   setName(data.name);
00047   setAdditionakData(new QByteArray(*data.additionalData));
00048   setRVision(data.rVision);
00049   setType(data.type);
00050 }
```

2.3.3.14 nextStep()

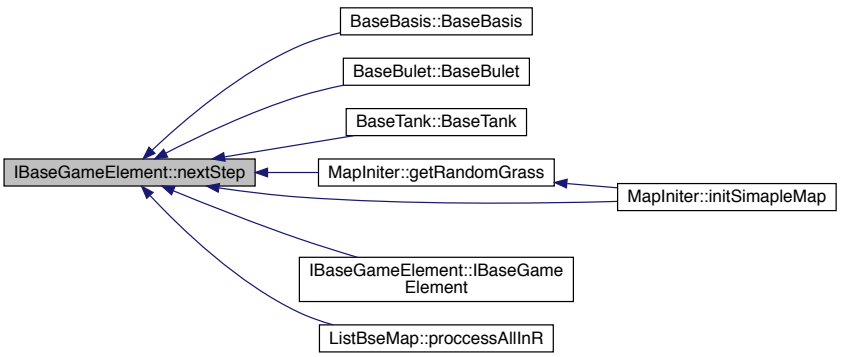
```
virtual void IBaseGameElement::nextStep ( )  [inline], [virtual], [inherited]
```

Definition at line 60 of file ibasegameelement.h.

Referenced by BaseBasis::BaseBasis(), BaseBulet::BaseBulet(), BaseTank(), MapIniter::getRandomGrass(), IBaseGameElement::IBaseGameElement(), MapIniter::initSimapleMap(), and ListBseMap::proccessAllInR().

```
00060 {};
```

Here is the caller graph for this function:



2.3.3.15 setAdditionakData()

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

Reimplemented from IBaseGameElement.

Definition at line 22 of file basetank.cpp.

```
00022                                                {
00023   IBaseGameElement::setAdditionakData(data);
00024   QDataStream stream(*data);
00025   stream >> direction >> speed >> fireType;
00026 }
```

2.3.3.16 setDirection()

```
void BaseTank::setDirection (
            double _direction )  [virtual]
```

Definition at line 35 of file basetank.cpp.

References direction.

```
00035                                            {
00036   direction = _direction;
00037   getAdditionalData();
00038 }
```

2.3.3.17 setHelth()

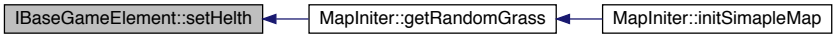
```
virtual void IBaseGameElement::setHelth (
            InfinityDouble * value )  [inline], [virtual], [inherited]
```

Definition at line 87 of file ibasegameelement.h.

Referenced by MapIniter::getRandomGrass().

```
00087 { this->helth = value; }
```

Here is the caller graph for this function:



2.3.3.18 setName()

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

Definition at line 93 of file ibasegameelement.h.

```
00093 { this->name = name; }
```

2.3.3.19 setPosition()

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

Definition at line 86 of file ibasegameelement.h.

```
00086 { this->position = value; }
```

2.3.3.20 setRVision()

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

Definition at line 97 of file ibasegameelement.h.

References IBaseGameElement::rVision.

```
00097 { rVision = _rVison; }
```

2.3.3.21 setSpeed()

```
void BaseTank::setSpeed (
            double _speed )  [virtual]
```

Definition at line 40 of file basetank.cpp.

References speed.

```
00040                                   {
00041   speed = _speed;
00042   getAdditionalData();
00043 }
```

2.3.3.22 setTransitWeight()

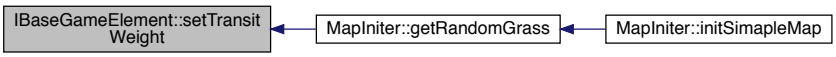
```
virtual void IBaseGameElement::setTransitWeight (
            InfinityDouble * value )  [inline], [virtual], [inherited]
```

Definition at line 89 of file ibasegameelement.h.

Referenced by MapIniter::getRandomGrass().

```
00089                                                    {
00090     this->transitWeight = value;
00091   }
```

Here is the caller graph for this function:



2.3.3.23 setType()

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

Reimplemented from IBaseGameElement.

Definition at line 13 of file basetank.cpp.

References eSimpleTank.

```
00013                                {
00014   if (value != ((int)eSimpleTank)) {
00015     Q_ASSERT_X(false, "game logic", "Wrong type for tank");
00016   }
00017   IBaseGameElement::setType(value);
00018 }
```

2.3.3.24 setWeight()

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

Definition at line 88 of file ibasegameelement.h.

```
00088 { this->weight = value; }
```

2.3.4 Member Data Documentation

2.3.4.1 additionalData

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

Definition at line 109 of file ibasegameelement.h.

2.3.4.2 direction

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

Definition at line 27 of file basetank.h.

Referenced by getDirection(), and setDirection().

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 35 of file basetank.h.

2.3.4.4 helth

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

Definition at line 106 of file ibasegameelement.h.

2.3.4.5 name

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

Definition at line 113 of file ibasegameelement.h.

2.3.4.6 position

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

Definition at line 105 of file ibasegameelement.h.

2.3.4.7 rVision

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

Definition at line 110 of file ibasegameelement.h.

Referenced by IBaseGameElement::getRVision(), and IBaseGameElement::setRVision().

2.3.4.8 speed

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

Definition at line 28 of file basetank.h.

Referenced by getSpeed(), and setSpeed().

2.3.4.9 transitWeight

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

Definition at line 108 of file ibasegameelement.h.

2.3.4.10 type

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

Definition at line 112 of file ibasegameelement.h.

Referenced by BaseBasis::BaseBasis(), BaseBulet::BaseBulet(), BaseTank(), IBaseGameElement::getType(), and IBaseGameElement::setType().

2.3.4.11 weight

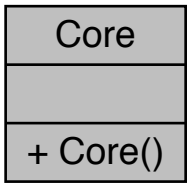
```
InfinityDouble* IBaseGameElement::weight = nullptr  [protected], [inherited]
```

Definition at line 107 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 ( )
```

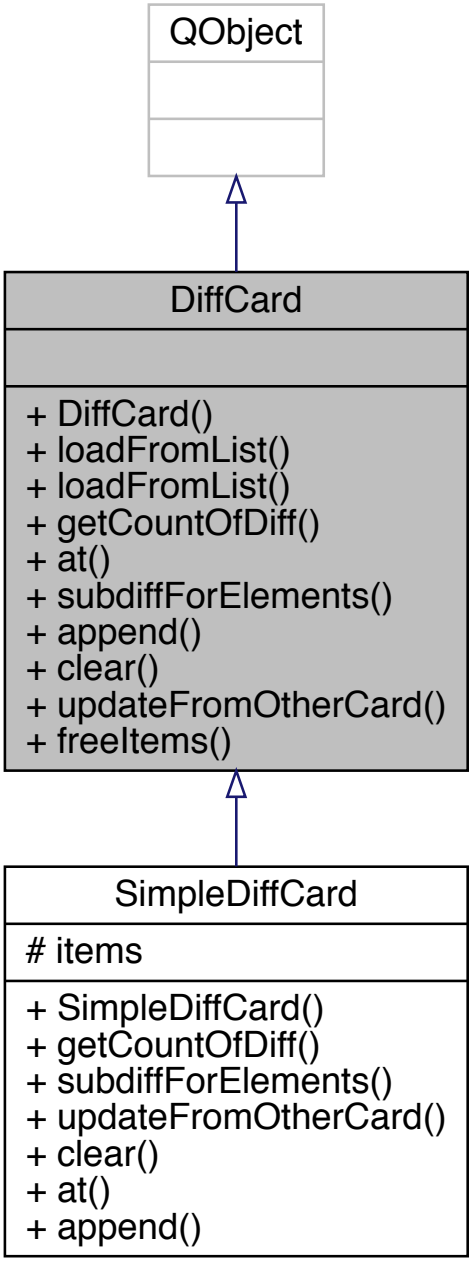
Definition at line 4 of file core.cpp.

```
00005 {
00006 }
```

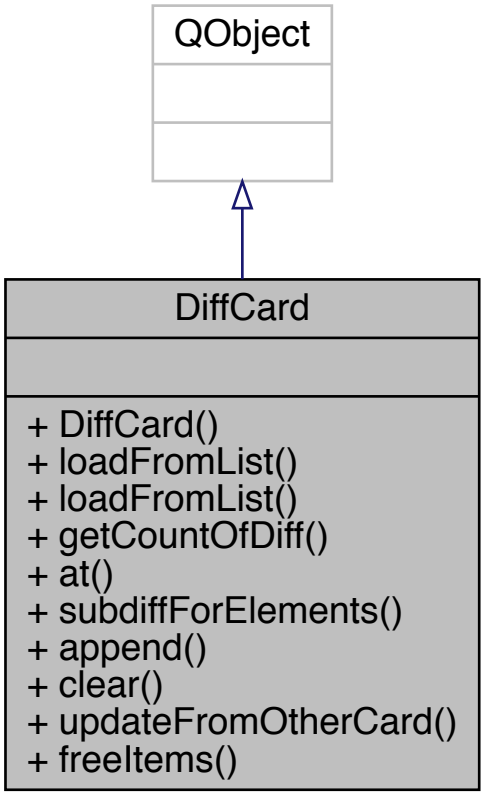
2.5 DiffCard Class Reference

```
#include "diffcard.h"
```

Inheritance diagram for DiffCard:



Collaboration diagram for DiffCard:



Public Member Functions

- DiffCard (QObject *parent=0)
- virtual void loadFromList (QList< DiffElement *> &newItems)
- virtual void loadFromList (QList< DiffElement *> *newItems)
- virtual int getCountOfDiff ()=0
- virtual DiffElement * at (int i)=0
- virtual DiffCard * subdiffForElements (QList< IBaseGameElement *> items)=0
- virtual void append (DiffElement *diff)=0
- virtual void clear ()=0
- virtual void updateFromOtherCard (DiffCard *card)=0
- virtual void freeItems ()

  freeItems simular to claer but also call destructor for diff elemenst

2.5.1 Detailed Description

Definition at line 9 of file diffcard.h.

2.5.2 Constructor & Destructor Documentation

2.5.2.1 DiffCard()

```
DiffCard::DiffCard (
            QObject * parent = 0 )  [explicit]
```

Definition at line 3 of file diffcard.cpp.

```
00003 : QObject(parent) {}
```

2.5.3 Member Function Documentation

2.5.3.1 append()

```
virtual void DiffCard::append (
            DiffElement * diff )  [pure virtual]
```

Implemented in SimpleDiffCard.

2.5.3.2 at()

```
virtual DiffElement* DiffCard::at (
            int i )  [pure virtual]
```

Implemented in SimpleDiffCard.

Referenced by ListBseMap::updateFromDiff().

Here is the caller graph for this function:

2.5.3.3 clear()

```
virtual void DiffCard::clear ( )  [pure virtual]
```

Implemented in SimpleDiffCard.

2.5.3.4 freeItems()

```
void DiffCard::freeItems ( )  [virtual]
```

freeItems simular to claer but also call destructor for diff elemenst

Definition at line 14 of file diffcard.cpp.

```
00014                           {
00015   for (int i = 0; i < getCountOfDiff(); i++) {
00016     delete at(i);
00017   }
00018   clear();
00019 }
```

2.5.3.5 getCountOfDiff()

```
virtual int DiffCard::getCountOfDiff ( )  [pure virtual]
```

Implemented in SimpleDiffCard.

Referenced by ListBseMap::updateFromDiff().

Here is the caller graph for this function:

2.5.3.6 loadFromList() [1/2]

```
void DiffCard::loadFromList (
            QList< DiffElement *> & newItems )  [virtual]
```

Definition at line 5 of file diffcard.cpp.

```
00005                                                       {
00006   foreach (auto i, newItems) { append(i); }
00007 }
```

2.5.3.7 loadFromList() [2/2]

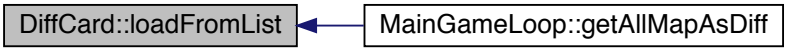
```
void DiffCard::loadFromList (
            QList< DiffElement *> * newItems )  [virtual]
```

Definition at line 9 of file diffcard.cpp.

Referenced by MainGameLoop::getAllMapAsDiff().

```
00009                                                   {
00010   for (int i = 0; i < newItems->size(); i++)
00011     append(newItems->at(i));
00012 }
```

Here is the caller graph for this function:



2.5.3.8 subdiffForElements()

```
virtual DiffCard* DiffCard::subdiffForElements (
            QList< IBaseGameElement *> items )  [pure virtual]
```

Implemented in SimpleDiffCard.

2.5.3.9 updateFromOtherCard()

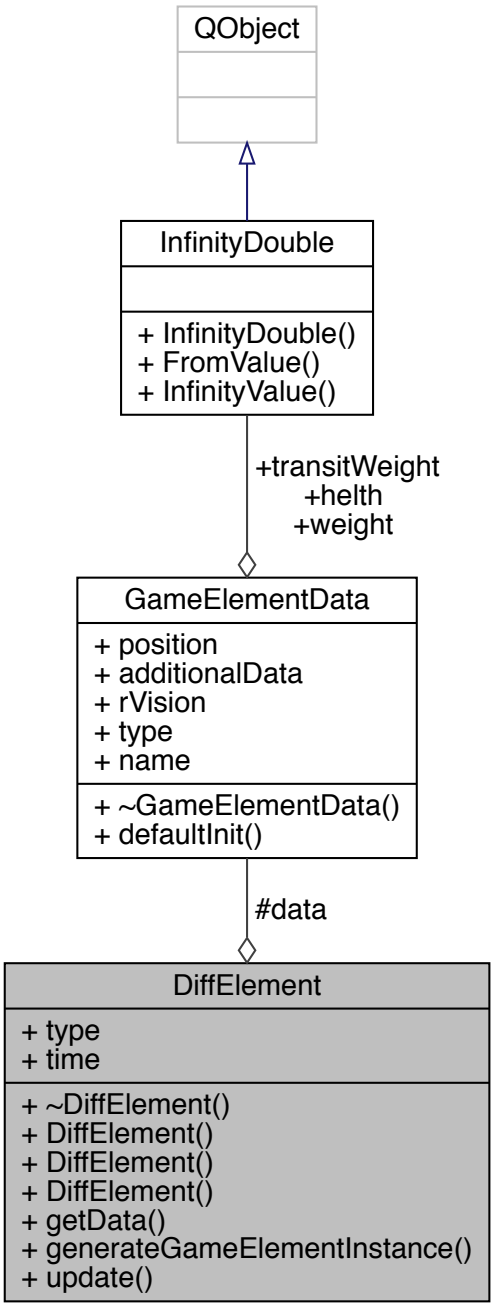
```
virtual void DiffCard::updateFromOtherCard (
            DiffCard * card )  [pure virtual]
```

Implemented in SimpleDiffCard.

2.6 DiffElement Class Reference

`#include "diffelement.h"`

Collaboration diagram for DiffElement:



Public Member Functions

- ~DiffElement ()

- DiffElement ()
- DiffElement (DiffElement &element)
- DiffElement (eDiffType type, IBaseGameElement *data)
- GameElementData * getData ()
- IBaseGameElement * generateGameElementInstance ()
- void update (DiffElement *element)

Public Attributes

- eDiffType type
- uint64_t time

Protected Attributes

- GameElementData * data

Friends

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

2.6.1 Detailed Description

Definition at line 10 of file diffelement.h.

2.6.2 Constructor & Destructor Documentation

2.6.2.1 ~DiffElement()

```
DiffElement::~DiffElement ( )
```

Definition at line 3 of file diffelement.cpp.

References data.

```
00003                           {
00004   delete data;
00005 }
```

2.6.2.2 DiffElement() [1/3]

```
DiffElement::DiffElement ( )
```

Definition at line 7 of file diffelement.cpp.

References data, eEmpty, and type.

```
00007                           {
00008   data = new GameElementData();
00009   type = eEmpty;
00010   time = 0;
00011 }
```

2.6.2.3 DiffElement() [2/3]

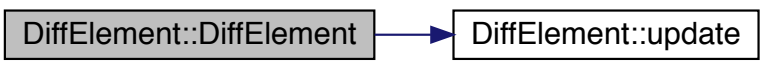
```
DiffElement::DiffElement (
            DiffElement & element )
```

Definition at line 13 of file diffelement.cpp.

References data, eEmpty, type, and update().

```
00013                                                {
00014   data = new GameElementData();
00015   type = eEmpty;
00016   time = 0;
00017   this->update(&element);
00018 }
```

Here is the call graph for this function:



2.6.2.4 DiffElement() [3/3]

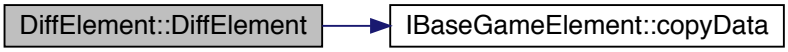
```
DiffElement::DiffElement (
            eDiffType type,
            IBaseGameElement * data )
```

Definition at line 20 of file diffelement.cpp.

References IBaseGameElement::copyData(), data, and type.

```
00020                                                                   {
00021   this->type = type;
00022   this->data = new GameElementData();
00023   data->copyData(*this->data);
00024   time = 0;
00025 }
```

Here is the call graph for this function:



2.6.3 Member Function Documentation

2.6.3.1 generateGameElementInstance()

```
IBaseGameElement * DiffElement::generateGameElementInstance ( )
```

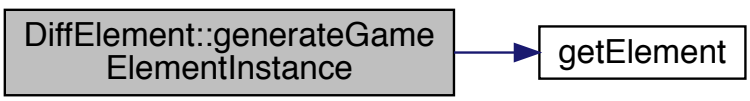
Definition at line 27 of file diffelement.cpp.

References data, and getElement().

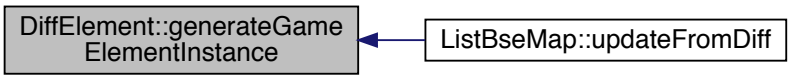
Referenced by ListBseMap::updateFromDiff().

```
00027                                                   {
00028   return getElement(*data);
00029 }
```

Here is the call graph for this function:



Here is the caller graph for this function:



2.6.3.2 getData()

```
GameElementData* DiffElement::getData ( )  [inline]
```

Definition at line 31 of file diffelement.h.

References data.

```
00031 { return data; }
```

2.6.3.3 update()

```
void DiffElement::update (
            DiffElement * element )
```

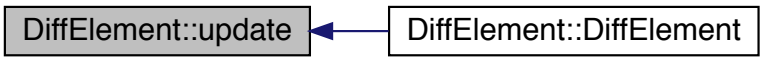
Definition at line 31 of file diffelement.cpp.

References type.

Referenced by DiffElement().

```
00031                                              {
00032   this->type = element->type;
00033   this->time = element->time;
00034   this->data->position = new QVector3D(*element->getData()->
      position);
00035   this->data->helth = new InfinityDouble(*element->
      getData()->helth);
00036   this->data->weight = new InfinityDouble(*element->
      getData()->weight);
00037   this->data->transitWeight =
00038       new InfinityDouble(*element->getData()->transitWeight);
00039   this->data->additionalData =
00040       new QByteArray(*element->getData()->additionalData);
00041   this->data->type = element->getData()->type;
00042   this->data->name = element->getData()->name;
00043   this->data->rVision = element->getData()->rVision;
00044 }
```

Here is the caller graph for this function:



2.6.4 Friends And Related Function Documentation

2.6.4.1 operator<<

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

Definition at line 16 of file diffelement.h.

```
00017                                                       {
00018     stream << ((int)myclass.type);
00019     return stream << (*myclass.data);
00020   }
```

2.6.4.2 operator>>

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

Definition at line 21 of file diffelement.h.

```
00021                                                             {
00022     int type;
00023     myclass.data->defaultInit();
00024     stream >> type >> (*myclass.data);
00025     myclass.type = (eDiffType)type;
00026     return stream;
00027   }
```

2.6.5 Member Data Documentation

2.6.5.1 data

```
GameElementData* DiffElement::data  [protected]
```

Definition at line 37 of file diffelement.h.

Referenced by DiffElement(), generateGameElementInstance(), getData(), and ~DiffElement().

2.6.5.2 time

```
uint64_t DiffElement::time
```

Definition at line 29 of file diffelement.h.

2.6.5.3 type

```
eDiffType DiffElement::type
```

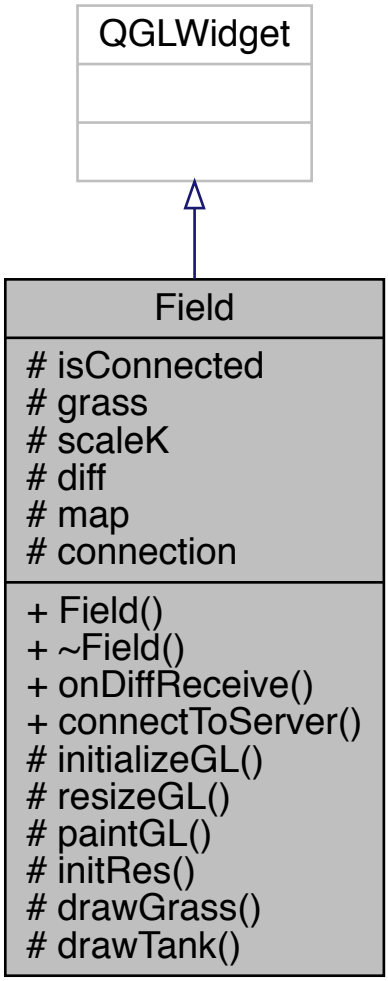
Definition at line 28 of file diffelement.h.

Referenced by DiffElement(), update(), and ListBseMap::updateFromDiff().

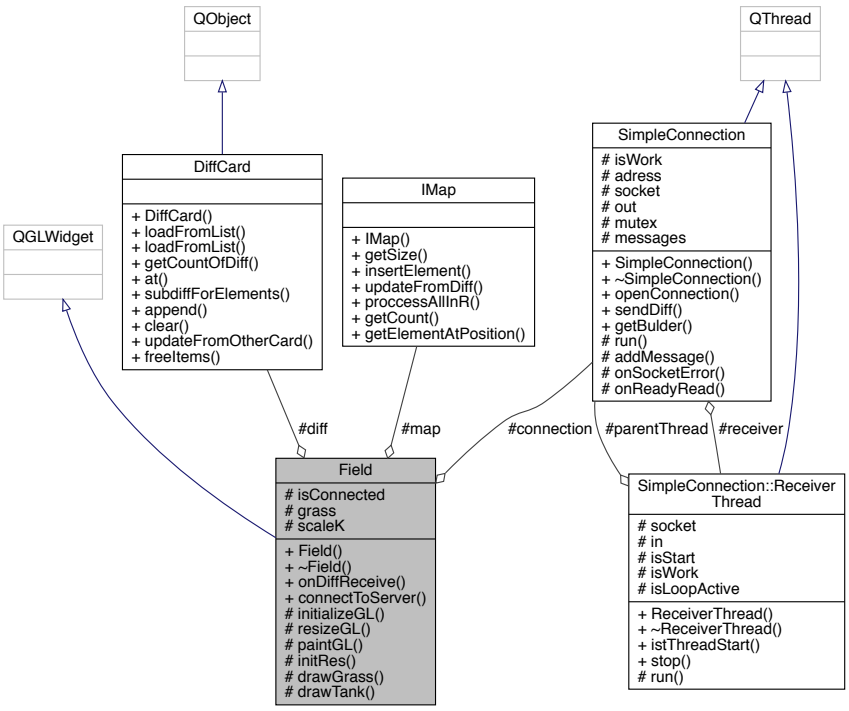
2.7 Field Class Reference

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

Inheritance diagram for Field:



Collaboration diagram for Field:



Public Slots

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

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)
- void drawTank (float x, float y)

Protected Attributes

- bool isConnected = false
- GLuint grass
- float scaleK = 0.02
- DiffCard * diff
- IMap * map
- SimpleConnection connection

2.7.1 Detailed Description

Definition at line 12 of file field.h.

2.7.2 Constructor & Destructor Documentation

2.7.2.1 Field()

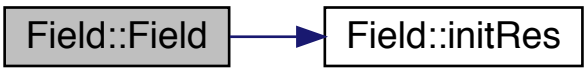
```
Field::Field (
            QWidget * parent = 0 )
```

Definition at line 7 of file field.cpp.

References initRes().

```
00008     : QGLWidget(parent), connection(QHostAddress::LocalHost, this) {
00009   diff = new SimpleDiffCard();
00010   map = new ListBseMap();
00011   initRes();
00012 }
```

Here is the call graph for this function:



2.7.2.2 ~Field()

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

Definition at line 14 of file field.cpp.

References diff, and map.

```
00014              {
00015   delete diff;
00016   delete map;
00017 }
```

2.7.3 Member Function Documentation

2.7.3.1 connectToServer

```
void Field::connectToServer ( )  [slot]
```

Definition at line 116 of file field.cpp.

```
00116                             {
00117   if (isConnected)
00118     return;
00119   connection.openConnection();
00120   connect(&connection, SIGNAL(onDiffReceive(QList<DiffElement*>*)), this,
00121           SLOT(onDiffReceive(QList<DiffElement*>*)));
00122   connection.getBulder()->asFirstMessage(
      eConnectionType::eWatcher)->build();
00123   isConnected = true;
00124 }
```

2.7.3.2 drawGrass()

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

Definition at line 81 of file field.cpp.

```
00081                                     {
00082   //  glEnable(GL_TEXTURE_2D);
00083   //  glBindTexture(GL_TEXTURE_2D, grass);
00084
00085   //  glBegin(GL_QUADS);
00086   //  glTexCoord2f(0, 0);
00087   //  glVertex3f(-1, -1, -1);
00088   //  glTexCoord2f(1, 0);
00089   //  glVertex3f(1, -1, -1);
00090   //  glTexCoord2f(1, 1);
00091   //  glVertex3f(1, 1, -1);
00092   //  glTexCoord2f(0, 1);
00093   //  glVertex3f(-1, 1, -1);
00094   // .....
00095   //  glDisable(GL_TEXTURE_2D);
00096   float size = scaleK;
00097   glColor3f(0.560, 0.956, 0.258);
00098   glBegin(GL_QUADS);
00099   glVertex2f(x * this->scaleK, y * scaleK);
00100   glVertex2f(x * this->scaleK, y * scaleK + size);
00101   glVertex2f(x * this->scaleK + size, y * scaleK + size);
00102   glVertex2f(x * this->scaleK + size, y * scaleK);
00103   glEnd();
00104 }
```

2.7.3.3 drawTank()

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

Definition at line 106 of file field.cpp.

```
00106                                       {
00107   float size = scaleK;
00108   glColor3f(0.6862745098, 0.1215686275, 0.4156862745);
00109   glBegin(GL_QUADS);
00110   glVertex2f(x * this->scaleK, y * scaleK);
00111   glVertex2f(x * this->scaleK, y * scaleK + size * 2);
00112   glVertex2f(x * this->scaleK + size * 2, y * scaleK + size * 2);
00113   glVertex2f(x * this->scaleK + size * 2, y * scaleK);
00114   glEnd();
00115 }
```

2.7.3.4 initializeGL()

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

Definition at line 29 of file field.cpp.

```
00029 {}
```

2.7.3.5 initRes()

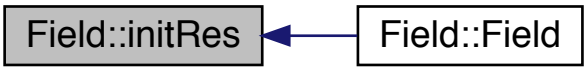
```
void Field::initRes ( )  [protected]
```

Definition at line 77 of file field.cpp.

Referenced by Field().

```
00077                       {
00078   loadTexture2((char*)":/res/grass.png", grass);
00079 }
```

Here is the caller graph for this function:



2.7.3.6 onDiffReceive

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

Definition at line 19 of file field.cpp.

```
00019                                                   {
00020   this->diff->loadFromList(diff);
00021   map->updateFromDiff(this->diff);
00022   this->diff->clear();
00023   delete diff;
00024   update();
00025
00026   QThread::msleep(100);
00027   this->connection.getBulder()->updateWatcher()->
      build();
00028 }
```

2.7.3.7 paintGL()

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

Definition at line 33 of file field.cpp.

```
00033                     {
00034   glClearColor(0.1f, 0.1f, 0.1f, 1.0f);
00035   glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
00036   //  glTranslatef(-1, -1, 0);
00037   for (int i = 0; i < map->getCount(); i++) {
00038     auto currentElement = map->getElementAtPosition(i);
00039     if (currentElement->getType() == (int)eBaseGameElementType::eGrass) {
00040       drawGrass(currentElement->getPosition()->x(),
00041                 currentElement->getPosition()->y());
00042     }
00043     if (currentElement->getType() == (int)eBaseGameElementType::eSimpleTank
      ) {
00044       drawTank(currentElement->getPosition()->x(),
00045                currentElement->getPosition()->y());
00046     }
00047   }
00048   //  glTranslatef(1, 1, 0);
00049   //  glEnd();
00050
00051   //  glDisable(GL_TEXTURE_2D);
00052 }
```

2.7.3.8 resizeGL()

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

Definition at line 31 of file field.cpp.

```
00031 {}
```

2.7.4 Member Data Documentation

2.7.4.1 connection

```
SimpleConnection Field::connection  [protected]
```

Definition at line 42 of file field.h.

2.7.4.2 diff

```
DiffCard* Field::diff  [protected]
```

Definition at line 39 of file field.h.

Referenced by ~Field().

2.7.4.3 grass

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

Definition at line 35 of file field.h.

2.7.4.4 isConnected

```
bool Field::isConnected = false  [protected]
```

Definition at line 24 of file field.h.

2.7.4.5 map

```
IMap* Field::map  [protected]
```

Definition at line 40 of file field.h.

Referenced by ~Field().

2.7.4.6 scaleK

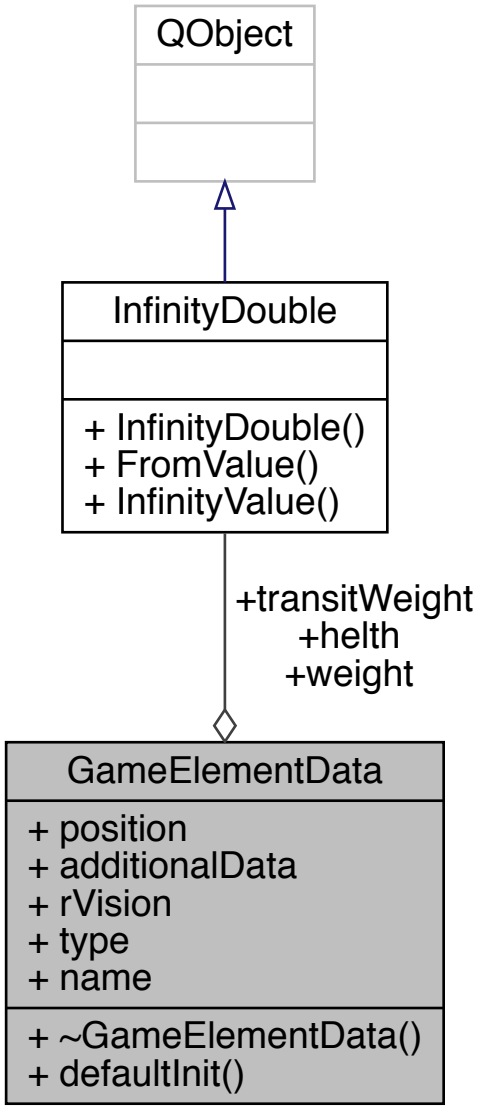
```
float Field::scaleK = 0.02  [protected]
```

Definition at line 37 of file field.h.

2.8 GameElementData Struct Reference

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

Collaboration diagram for GameElementData:



Public Member Functions

- ~GameElementData ()
- void defaultInit ()

Public Attributes

- QVector3D * position = nullptr
- InfinityDouble * helth = nullptr
- InfinityDouble * weight = nullptr
- InfinityDouble * transitWeight = nullptr
- QByteArray * additionalData = nullptr
- qint32 rVision = 1
- qint32 type = -1
- QString name = ""

2.8.1 Detailed Description

Definition at line 16 of file ibasegameelement.h.

2.8.2 Constructor & Destructor Documentation

2.8.2.1 ~GameElementData()

```
GameElementData::~GameElementData ( )  [inline]
```

Definition at line 26 of file ibasegameelement.h.

```
00026                       {
00027     delete position;
00028     delete helth;
00029     delete weight;
00030     delete transitWeight;
00031     delete additionalData;
00032   }
```

2.8.3 Member Function Documentation

2.8.3.1 defaultInit()

```
void GameElementData::defaultInit ( )  [inline]
```

Definition at line 34 of file ibasegameelement.h.

```
00034                       {
00035     position = new QVector3D(0, 0, 0);
00036     helth = InfinityDouble::FromValue(1);
00037     weight = InfinityDouble::FromValue(0);
00038     transitWeight = InfinityDouble::FromValue(0);
00039     additionalData = new QByteArray();
00040   }
```

2.8.4 Member Data Documentation

2.8.4.1 additionalData

```
QByteArray* GameElementData::additionalData = nullptr
```

Definition at line 21 of file ibasegameelement.h.

2.8.4.2 helth

```
InfinityDouble* GameElementData::helth = nullptr
```

Definition at line 18 of file ibasegameelement.h.

2.8.4.3 name

```
QString GameElementData::name = ""
```

Definition at line 24 of file ibasegameelement.h.

2.8.4.4 position

```
QVector3D* GameElementData::position = nullptr
```

Definition at line 17 of file ibasegameelement.h.

2.8.4.5 rVision

```
qint32 GameElementData::rVision = 1
```

Definition at line 22 of file ibasegameelement.h.

2.8.4.6 transitWeight

```
InfinityDouble* GameElementData::transitWeight = nullptr
```

Definition at line 20 of file ibasegameelement.h.

2.8.4.7 type

```
qint32 GameElementData::type = -1
```

Definition at line 23 of file ibasegameelement.h.

2.8.4.8 weight

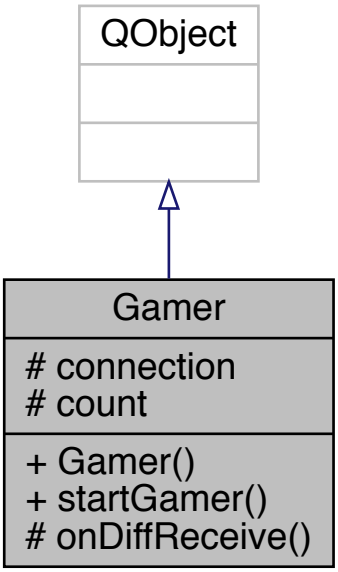
```
InfinityDouble* GameElementData::weight = nullptr
```

Definition at line 19 of file ibasegameelement.h.

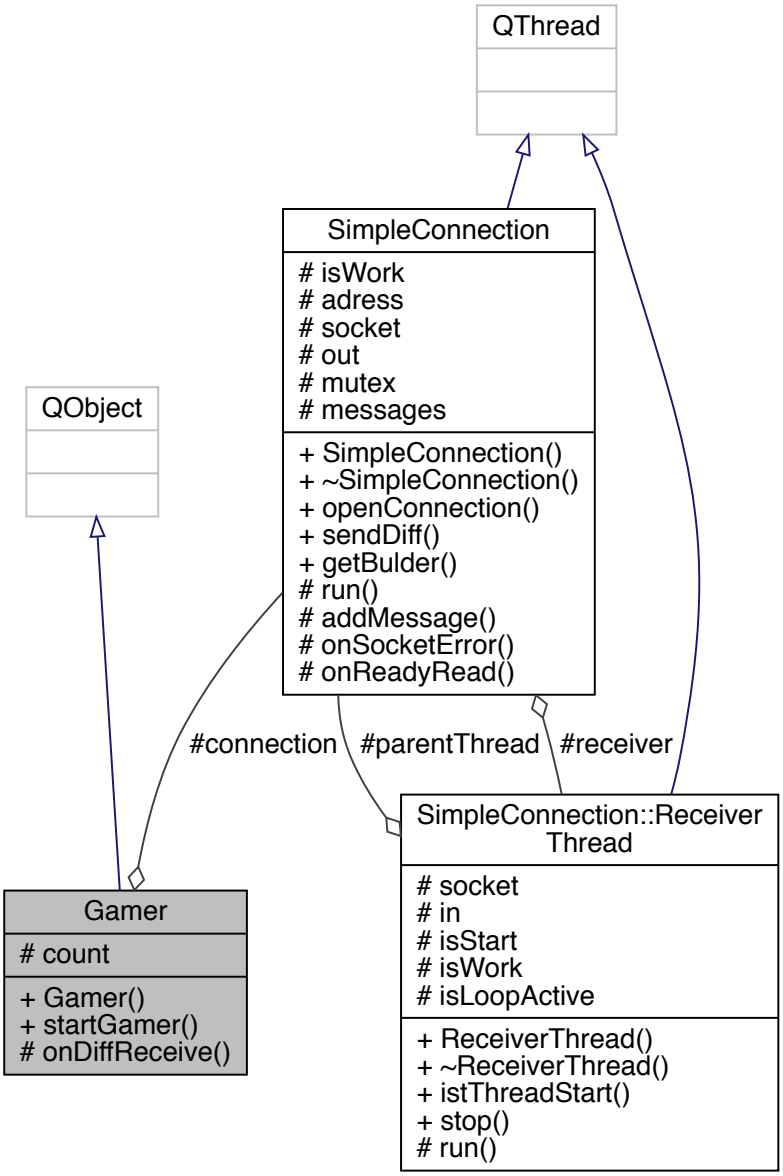
2.9 Gamer Class Reference

```
#include "gamer.h"
```

Inheritance diagram for Gamer:



Collaboration diagram for Gamer:



Public Member Functions

- Gamer (QObject *parent=0)
- void startGamer ()

Protected Slots

- void onDiffReceive (QList< DiffElement *> *)

Protected Attributes

- SimpleConnection connection
- int count = 0

2.9.1 Detailed Description

Definition at line 8 of file gamer.h.

2.9.2 Constructor & Destructor Documentation

2.9.2.1 Gamer()

```
Gamer::Gamer (
            QObject * parent = 0 )  [explicit]
```

Definition at line 4 of file gamer.cpp.

```
00005     : QObject(parent), connection(QHostAddress::LocalHost, this) {}
```

2.9.3 Member Function Documentation

2.9.3.1 onDiffReceive

```
void Gamer::onDiffReceive (
            QList< DiffElement *> * diffList )  [protected], [slot]
```

Definition at line 14 of file gamer.cpp.

Referenced by main().

```
00014                                                    {
00015   if (count > 10)
00016     return;
00017   BaseTank* newElement = new BaseTank();
00018   newElement->setPosition(new QVector3D(0, 0, 0));
00019   newElement->setDirection(0.2);
00020   newElement->setSpeed(10);
00021   QList<IBaseGameElement*>* items = new QList<IBaseGameElement*>();
00022   items->append(newElement);
00023   connection.getBulder()->addNewItem(items)->build();
00024   count++;
00025 }
```

Here is the caller graph for this function:

2.9.3.2 startGamer()

```
void Gamer::startGamer ( )
```

Definition at line 7 of file gamer.cpp.

Referenced by main().

```
00007                       {
00008   connection.openConnection();
00009   connect(&connection, SIGNAL(onDiffReceive(QList<DiffElement*>*)), this,
00010           SLOT(onDiffReceive(QList<DiffElement*>*)));
00011   connection.getBulder()->asFirstMessage(
      eConnectionType::eGamer)->build();
00012 }
```

Here is the caller graph for this function:

2.9.4 Member Data Documentation

2.9.4.1 connection

```
SimpleConnection Gamer::connection  [protected]
```

Definition at line 19 of file gamer.h.

2.9.4.2 count

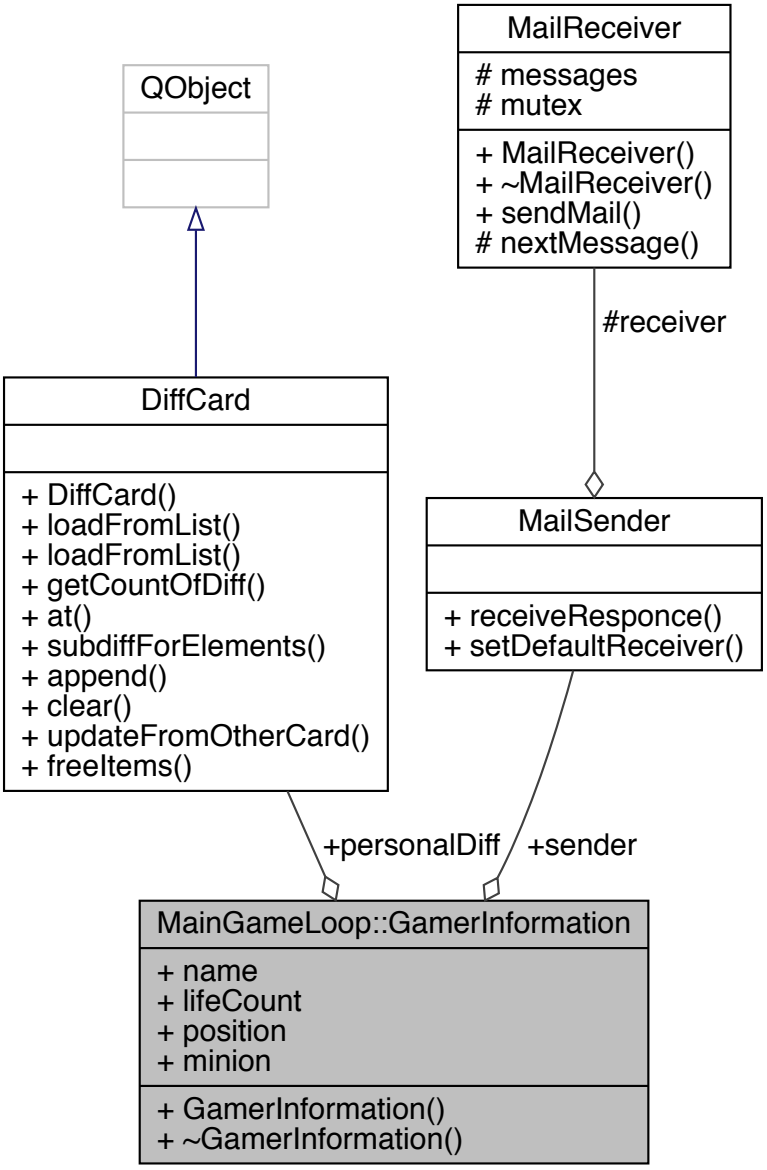
```
int Gamer::count = 0  [protected]
```

Definition at line 20 of file gamer.h.

2.10 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
- DiffCard * personalDiff

2.10.1 Detailed Description

Definition at line 45 of file maingameloop.h.

2.10.2 Constructor & Destructor Documentation

2.10.2.1 GamerInformation()

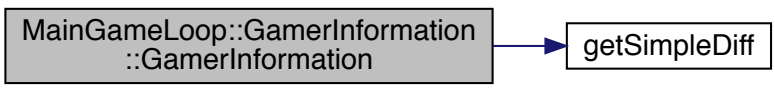
```
MainGameLoop::GamerInformation::GamerInformation (
            IMap * map )
```

Definition at line 148 of file maingameloop.cpp.

References getSimpleDiff(), and personalDiff.

```
00148                                                   {
00149   name = "";
00150   lifeCount = limits::defaultBasisEnergy;
00151   QSizeF* size = map->getSize();
00152   float rand1 = ((float)(rand())) / ((float)(RAND_MAX));
00153   float rand2 = ((float)(rand())) / ((float)(RAND_MAX));
00154   position = QVector3D(size->width() * rand1, size->height() * rand2, 0);
00155   personalDiff = getSimpleDiff();
00156 }
```

Here is the call graph for this function:



2.10.2.2 ~GamerInformation()

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

Definition at line 158 of file maingameloop.cpp.

References personalDiff.

```
00158                                                {
00159   delete personalDiff;
00160 }
```

2.10.3 Member Data Documentation

2.10.3.1 lifeCount

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

Definition at line 48 of file maingameloop.h.

2.10.3.2 minion

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

Definition at line 51 of file maingameloop.h.

2.10.3.3 name

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

Definition at line 47 of file maingameloop.h.

2.10.3.4 personalDiff

```
DiffCard* MainGameLoop::GamerInformation::personalDiff
```

Definition at line 52 of file maingameloop.h.

Referenced by GamerInformation(), and ~GamerInformation().

2.10.3.5 position

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

Definition at line 49 of file maingameloop.h.

2.10.3.6 sender

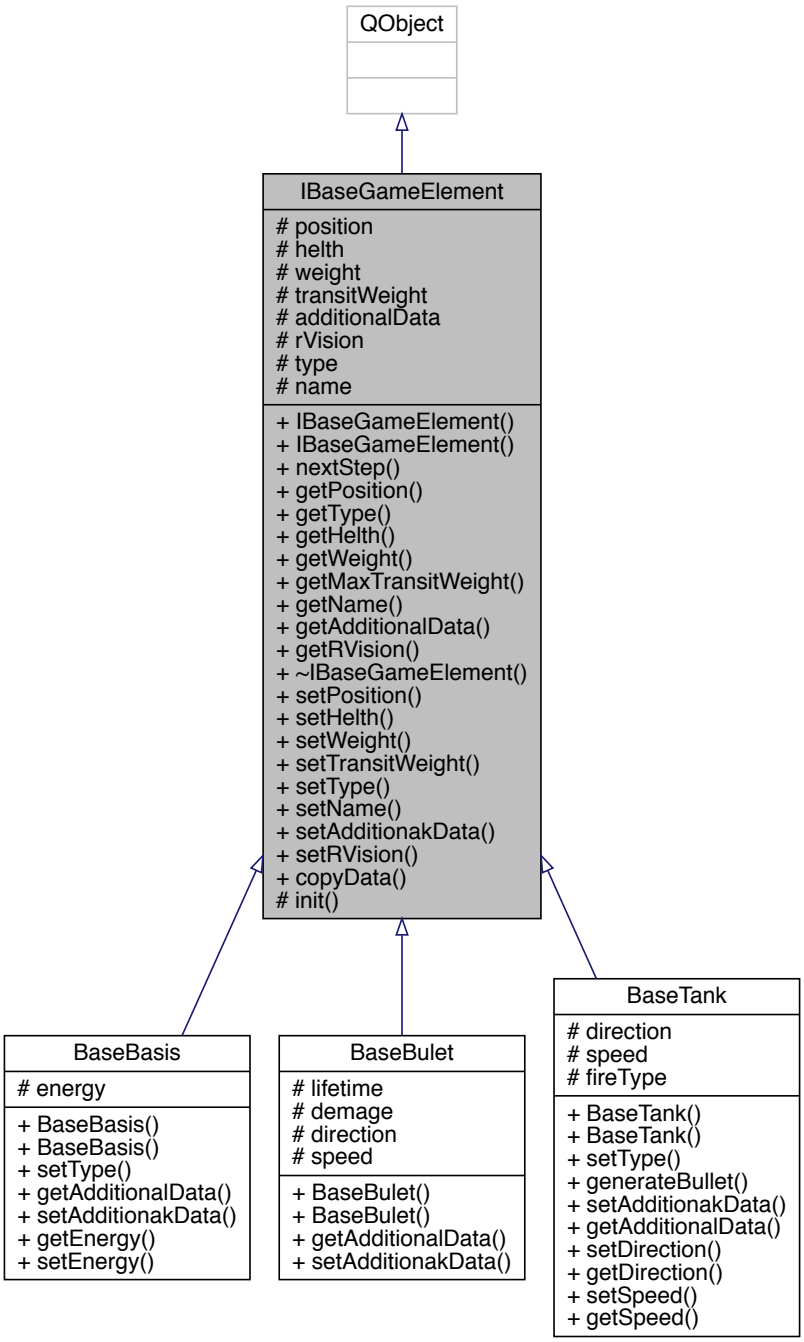
```
MailSender* MainGameLoop::GamerInformation::sender
```

Definition at line 50 of file maingameloop.h.

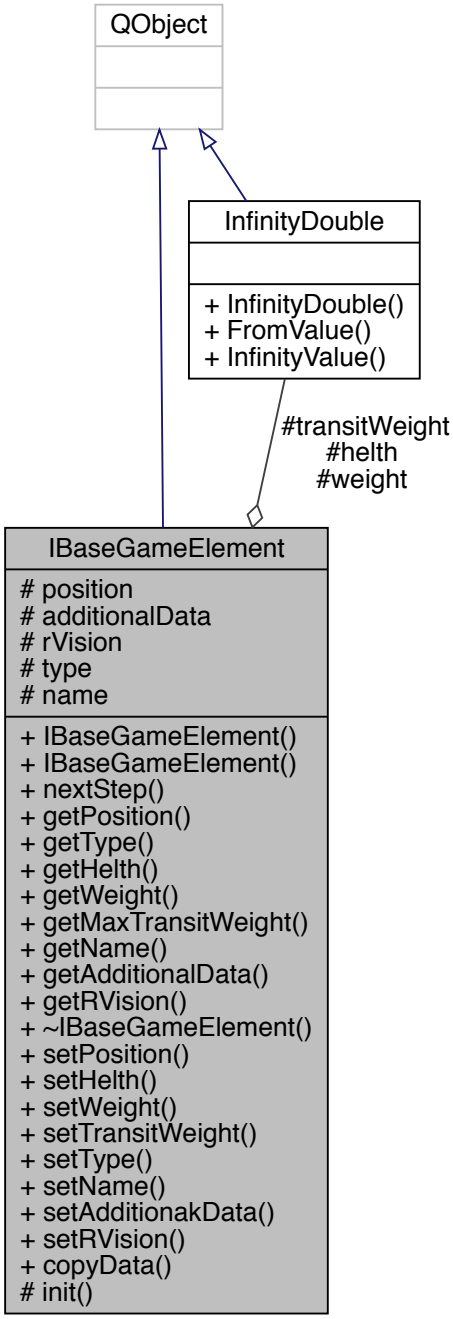
2.11 IBaseGameElement Class Reference

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

Inheritance diagram for IBaseGameElement:



Collaboration diagram for IBaseGameElement:



Public Member Functions

- IBaseGameElement ()
- IBaseGameElement (GameElementData &data)
- virtual void nextStep ()
- virtual QVector3D * getPosition () const
- virtual int getType () const

- virtual InfinityDouble * getHelth () const
- virtual InfinityDouble * getWeight () const
- virtual InfinityDouble * getMaxTransitWeight () const
- virtual QString getName () const
- virtual QByteArray * getAdditionalData () const
- virtual int getRVision () const
- 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)
- void copyData (GameElementData &out)

Protected Member Functions

- virtual void init (GameElementData &data)

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)

2.11.1 Detailed Description

Definition at line 46 of file ibasegameelement.h.

2.11.2 Constructor & Destructor Documentation

2.11.2.1 IBaseGameElement() [1/2]

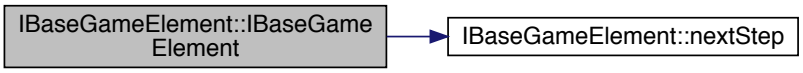
```
IBaseGameElement::IBaseGameElement ( )  [inline]
```

Definition at line 49 of file ibasegameelement.h.

References nextStep().

```
00049                       {
00050     helth = InfinityDouble::FromValue(1);
00051     weight = InfinityDouble::FromValue(0);
00052     transitWeight = InfinityDouble::FromValue(0);
00053     position = new QVector3D(0, 0, 0);
00054     name = "";
00055     additionalData = new QByteArray();
00056   }
```

Here is the call graph for this function:



2.11.2.2 IBaseGameElement() [2/2]

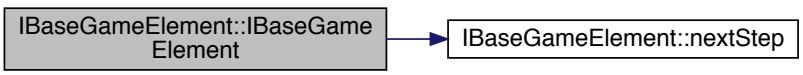
```
IBaseGameElement::IBaseGameElement (
            GameElementData & data )  [inline]
```

Definition at line 58 of file ibasegameelement.h.

References nextStep().

```
00058 { init(data); }
```

Here is the call graph for this function:



2.11.2.3 ~IBaseGameElement()

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

Definition at line 79 of file ibasegameelement.h.

```
00079                               {
00080     delete helth;
00081     delete weight;
00082     delete transitWeight;
00083     delete position;
00084   }
```

2.11.3 Member Function Documentation

2.11.3.1 copyData()

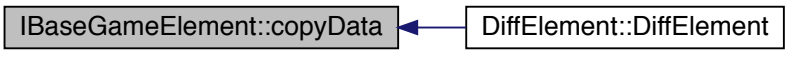
```
void IBaseGameElement::copyData (
             GameElementData & out )
```

Definition at line 27 of file ibasegameelement.cpp.

Referenced by DiffElement::DiffElement().

```
00027                                                    {
00028   out.position = new QVector3D(*position);
00029   out.helth = new InfinityDouble(*helth);
00030   out.weight = new InfinityDouble(*weight);
00031   out.transitWeight = new InfinityDouble(*
      transitWeight);
00032   if (additionalData == nullptr)
00033     out.additionalData = nullptr;
00034   else
00035     out.additionalData = new QByteArray(*additionalData);
00036   out.type = type;
00037   out.name = name;
00038   out.rVision = rVision;
00039 }
```

Here is the caller graph for this function:



2.11.3.2 getAdditionalData()

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

Reimplemented in BaseBulet, BaseTank, and BaseBasis.

Definition at line 68 of file ibasegameelement.h.

```
00068 { return additionalData; }
```

2.11.3.3 getHelth()

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

Definition at line 64 of file ibasegameelement.h.

```
00064 { return helth; }
```

2.11.3.4 getMaxTransitWeight()

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

Definition at line 66 of file ibasegameelement.h.

```
00066 { return transitWeight; }
```

2.11.3.5 getName()

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

Definition at line 67 of file ibasegameelement.h.

```
00067 { return name; }
```

2.11.3.6 getPosition()

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

Definition at line 62 of file ibasegameelement.h.

```
00062 { return position; }
```

2.11.3.7 getRVision()

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

Definition at line 69 of file ibasegameelement.h.

References rVision.

```
00069 { return rVision; }
```

2.11.3.8 getType()

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

Definition at line 63 of file ibasegameelement.h.

References type.

```
00063 { return type; }
```

2.11.3.9 getWeight()

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

Definition at line 65 of file ibasegameelement.h.

```
00065 { return weight; }
```

2.11.3.10 init()

```
void IBaseGameElement::init (
            GameElementData & data )  [protected], [virtual]
```

Definition at line 41 of file ibasegameelement.cpp.

```
00041                                                {
00042   setHelth(new InfinityDouble(*data.helth));
00043   setWeight(new InfinityDouble(*data.weight));
00044   setTransitWeight(new InfinityDouble(*data.
      transitWeight));
00045   setPosition(new QVector3D(*data.position));
00046   setName(data.name);
00047   setAdditionakData(new QByteArray(*data.additionalData));
00048   setRVision(data.rVision);
00049   setType(data.type);
00050 }
```

2.11.3.11 nextStep()

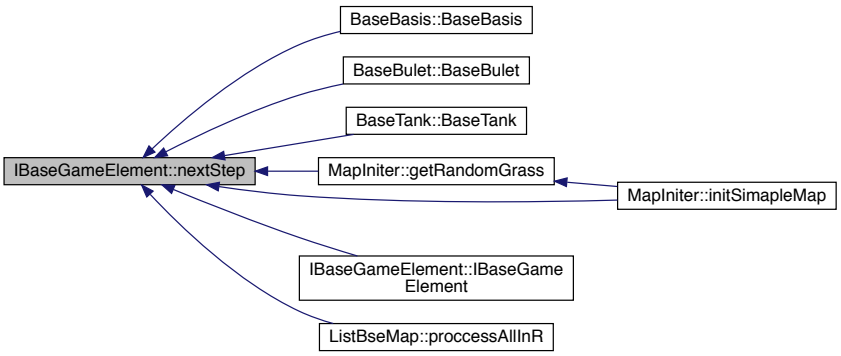
```
virtual void IBaseGameElement::nextStep ( )  [inline], [virtual]
```

Definition at line 60 of file ibasegameelement.h.

Referenced by BaseBasis::BaseBasis(), BaseBulet::BaseBulet(), BaseTank::BaseTank(), MapIniter::getRandomGrass(), IBaseGameElement(), MapIniter::initSimapleMap(), and ListBseMap::proccessAllInR().

```
00060 {};
```

Here is the caller graph for this function:



2.11.3.12 setAdditionakData()

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

Reimplemented in BaseBulet, BaseTank, and BaseBasis.

Definition at line 94 of file ibasegameelement.h.

```
00094                                                {
00095     this->additionalData = data;
00096   }
```

2.11.3.13 setHelth()

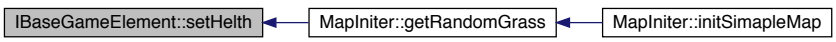
```
virtual void IBaseGameElement::setHelth (
            InfinityDouble * value )  [inline], [virtual]
```

Definition at line 87 of file ibasegameelement.h.

Referenced by MapIniter::getRandomGrass().

```
00087 { this->helth = value; }
```

Here is the caller graph for this function:



2.11.3.14 setName()

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

Definition at line 93 of file ibasegameelement.h.

```
00093 { this->name = name; }
```

2.11.3.15 setPosition()

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

Definition at line 86 of file ibasegameelement.h.

```
00086 { this->position = value; }
```

2.11.3.16 setRVision()

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

Definition at line 97 of file ibasegameelement.h.

References rVision.

```
00097 { rVision = _rVison; }
```

2.11.3.17 setTransitWeight()

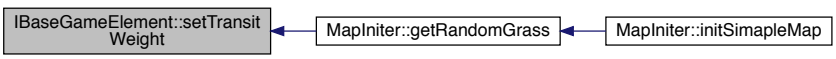
```
virtual void IBaseGameElement::setTransitWeight (
            InfinityDouble * value )  [inline], [virtual]
```

Definition at line 89 of file ibasegameelement.h.

Referenced by MapIniter::getRandomGrass().

```
00089                                                  {
00090     this->transitWeight = value;
00091   }
```

Here is the caller graph for this function:



2.11.3.18 setType()

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

Reimplemented in BaseTank, and BaseBasis.

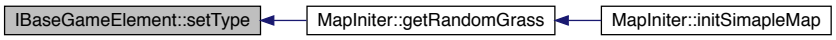
Definition at line 92 of file ibasegameelement.h.

References type.

Referenced by MapIniter::getRandomGrass().

```
00092 { this->type = value; }
```

Here is the caller graph for this function:



2.11.3.19 setWeight()

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

Definition at line 88 of file ibasegameelement.h.

```
00088 { this->weight = value; }
```

2.11.4 Friends And Related Function Documentation

2.11.4.1 operator<<

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

Definition at line 71 of file ibasegameelement.h.

```
00072                                                                {
00073     return stream << (*myclass.position) << (*myclass.helth)
00074                   << (*myclass.weight) << (*myclass.transitWeight)
00075                   << (*myclass.additionalData) << myclass.type << myclass.
      name
00076                   << myclass.rVision;
00077   }
```

2.11.5 Member Data Documentation

2.11.5.1 additionalData

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

Definition at line 109 of file ibasegameelement.h.

2.11.5.2 helth

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

Definition at line 106 of file ibasegameelement.h.

2.11.5.3 name

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

Definition at line 113 of file ibasegameelement.h.

2.11.5.4 position

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

Definition at line 105 of file ibasegameelement.h.

2.11.5.5 rVision

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

Definition at line 110 of file ibasegameelement.h.

Referenced by getRVision(), and setRVision().

2.11.5.6 transitWeight

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

Definition at line 108 of file ibasegameelement.h.

2.11.5.7 type

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

Definition at line 112 of file ibasegameelement.h.

Referenced by BaseBasis::BaseBasis(), BaseBulet::BaseBulet(), BaseTank::BaseTank(), getType(), and setType().

2.11.5.8 weight

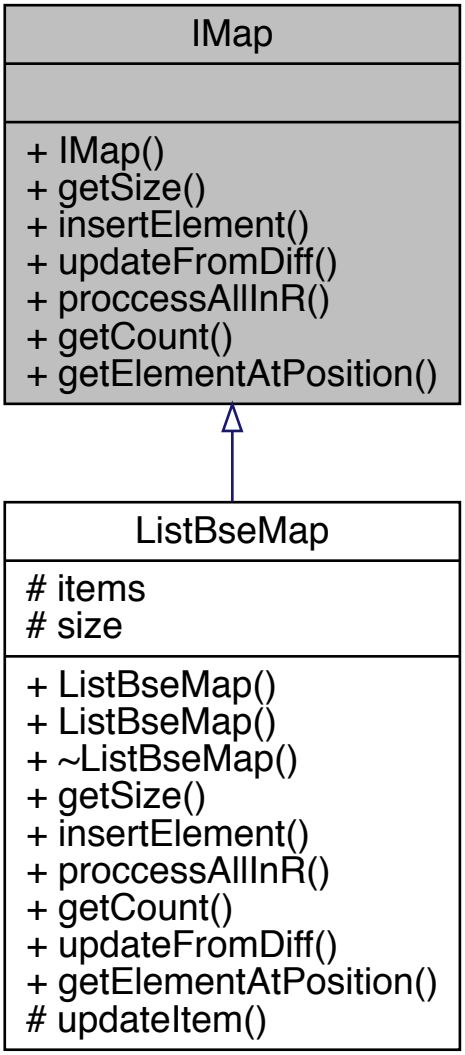
```
InfinityDouble* IBaseGameElement::weight = nullptr  [protected]
```

Definition at line 107 of file ibasegameelement.h.

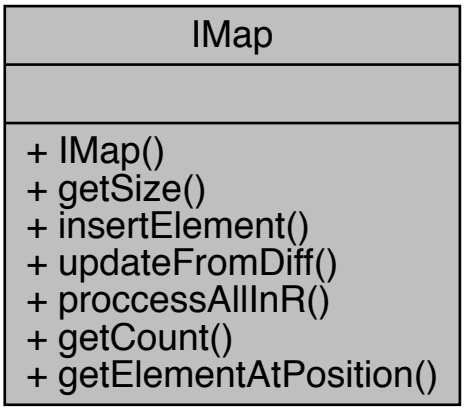
2.12 IMap Class Reference

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

Inheritance diagram for IMap:



Collaboration diagram for IMap:



Public Member Functions

- IMap ()
- virtual QSizeF ∗ getSize ()=0
- virtual void insertElement (IBaseGameElement ∗element)=0
- virtual void updateFromDiff (DiffCard ∗diff)=0
- virtual void proccessAllInR (IBaseGameElement ∗element, double r, mapOperator)=0
- virtual int getCount ()=0
- virtual IBaseGameElement ∗ getElementAtPosition (int pos)=0

2.12.1 Detailed Description

Definition at line 11 of file imap.h.

2.12.2 Constructor & Destructor Documentation

2.12.2.1 IMap()

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

Definition at line 4 of file imap.cpp.

```
00005 {
00006
00007 }
```

2.12.3 Member Function Documentation

2.12.3.1 getCount()

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

Implemented in ListBseMap.

2.12.3.2 getElementAtPosition()

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

Implemented in ListBseMap.

2.12.3.3 getSize()

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

Implemented in ListBseMap.

2.12.3.4 insertElement()

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

Implemented in ListBseMap.

2.12.3.5 proccessAllInR()

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

Implemented in ListBseMap.

2.12.3.6 updateFromDiff()

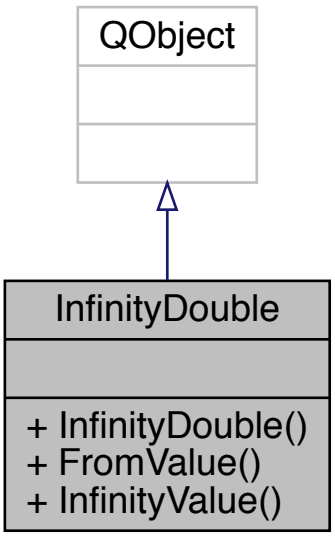
```
virtual void IMap::updateFromDiff (
            DiffCard * diff )  [pure virtual]
```

Implemented in ListBseMap.

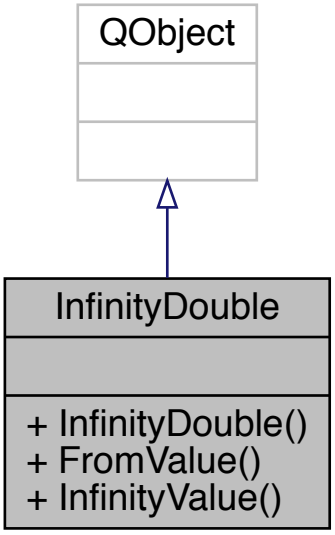
2.13 InfinityDouble Class Reference

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

Inheritance diagram for InfinityDouble:



Collaboration diagram for InfinityDouble:



Public Member Functions

- InfinityDouble (InfinityDouble &id)

Static Public Member Functions

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

Friends

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

2.13.1 Detailed Description

Definition at line 6 of file infinitydouble.h.

2.13.2 Constructor & Destructor Documentation

2.13.2.1 InfinityDouble()

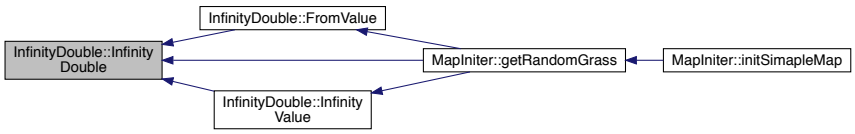
```
InfinityDouble::InfinityDouble (
            InfinityDouble & id )
```

Definition at line 3 of file infinitydouble.cpp.

Referenced by FromValue(), MapIniter::getRandomGrass(), and InfinityValue().

```
00003                                                {
00004   this->w = id.w;
00005   this->isInfinity = id.isInfinity;
00006 }
```

Here is the caller graph for this function:



2.13.3 Member Function Documentation

2.13.3.1 FromValue()

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

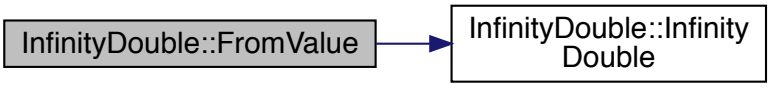
Definition at line 10 of file infinitydouble.cpp.

References InfinityDouble().

Referenced by MapIniter::getRandomGrass().

```
00010                                                {
00011   InfinityDouble* result = new InfinityDouble();
00012   result->w = w;
00013   result->isInfinity = false;
00014   return result;
00015 }
```

Here is the call graph for this function:



Here is the caller graph for this function:

2.13.3.2 InfinityValue()

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

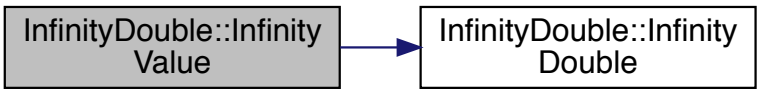
Definition at line 17 of file infinitydouble.cpp.

References InfinityDouble().

Referenced by MapIniter::getRandomGrass().

```
00017                                          {
00018   InfinityDouble* result = new InfinityDouble();
00019   result->isInfinity = true;
00020   return result;
00021 }
```

Here is the call graph for this function:



Here is the caller graph for this function:

2.13.4 Friends And Related Function Documentation

2.13.4.1 operator<<

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

Definition at line 9 of file infinitydouble.h.

```
00010                                                                    {
00011     return stream << myclass.isInfinity << myclass.w;
00012   }
```

2.13.4.2 operator>>

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

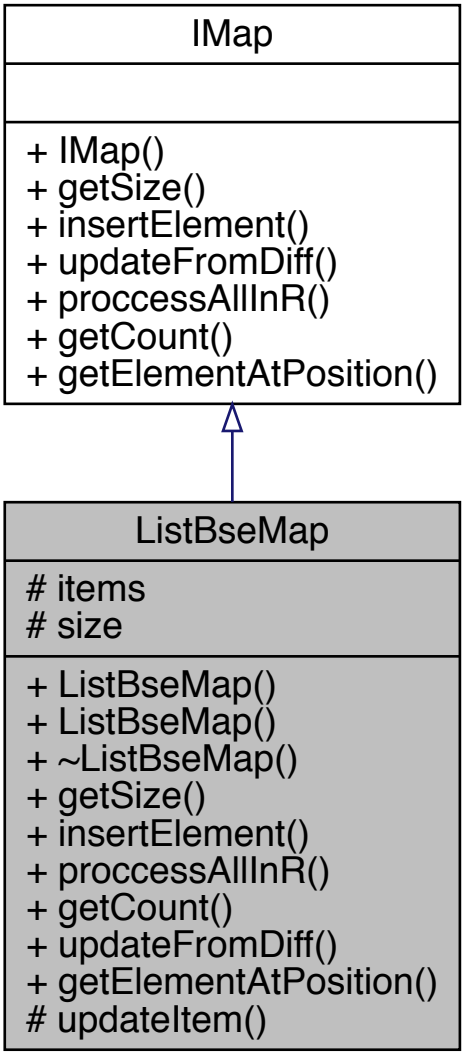
Definition at line 14 of file infinitydouble.h.

```
00014                                                                    {
00015     return stream >> myclass.isInfinity >> myclass.w;
00016   }
```

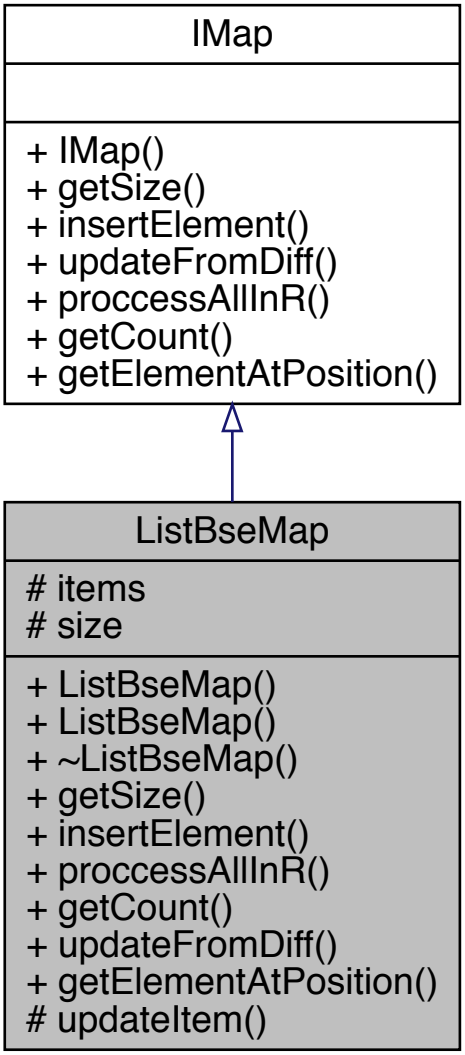
2.14 ListBseMap Class Reference

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

Inheritance diagram for ListBseMap:



Collaboration diagram for ListBseMap:



Public Member Functions

- ListBseMap (double width, double heigth)
- ListBseMap ()
- ~ListBseMap ()
- virtual QSizeF * getSize ()
- virtual void insertElement (IBaseGameElement *element)
- virtual void proccessAllInR (IBaseGameElement *element, double r, mapOperator op)
- virtual int getCount ()
- virtual void updateFromDiff (DiffCard *diff)
- IBaseGameElement * getElementAtPosition (int pos)

Protected Member Functions

- void updateItem (IBaseGameElement *gameEleement, bool isReplace=true)

  updateItem update items in map

Protected Attributes

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

2.14.1 Detailed Description

Definition at line 7 of file listbsemap.h.

2.14.2 Constructor & Destructor Documentation

2.14.2.1 ListBseMap() [1/2]

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

Definition at line 4 of file listbsemap.cpp.

```
00004                                                   {
00005   size = new QSizeF(width, heigth);
00006   items = new QList<IBaseGameElement*>();
00007 }
```

2.14.2.2 ListBseMap() [2/2]

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

Definition at line 9 of file listbsemap.cpp.

```
00009                       {
00010   items = new QList<IBaseGameElement*>();
00011 }
```

2.14.2.3 ~ListBseMap()

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

Definition at line 13 of file listbsemap.cpp.

```
00013                         {
00014   delete items;
00015   delete size;
00016 }
```

2.14.3 Member Function Documentation

2.14.3.1 getCount()

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

Implements IMap.

Definition at line 36 of file listbsemap.cpp.

```
00036                         {
00037   return items->size();
00038 }
```

2.14.3.2 getElementAtPosition()

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

Implements IMap.

Definition at line 58 of file listbsemap.cpp.

```
00058                                                   {
00059   return items->at(pos);
00060 }
```

2.14.3.3 getSize()

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

Implements IMap.

Definition at line 18 of file listbsemap.cpp.

```
00018                             {
00019   return size;
00020 }
```

2.14.3.4 insertElement()

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

Implements IMap.

Definition at line 22 of file listbsemap.cpp.

```
00022                                                 {
00023   items->append(element);
00024 }
```

2.14.3.5 proccessAllInR()

```
void ListBseMap::proccessAllInR (
            IBaseGameElement * element,
            double r,
            mapOperator op )  [virtual]
```

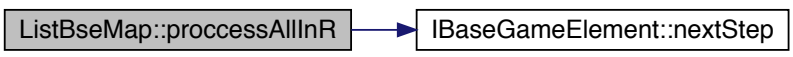
Implements IMap.

Definition at line 26 of file listbsemap.cpp.

References IBaseGameElement::nextStep().

```
00028                                                 {
00029   for (int i = 0; i < items->size(); i++) {
00030     IBaseGameElement* el = items->at(i);
00031     if (element != el && distanceBetweenElement(el, element))
00032       op(el);
00033   }
00034 }
```

Here is the call graph for this function:



2.14.3.6 updateFromDiff()

```
void ListBseMap::updateFromDiff (
            DiffCard * diff )  [virtual]
```

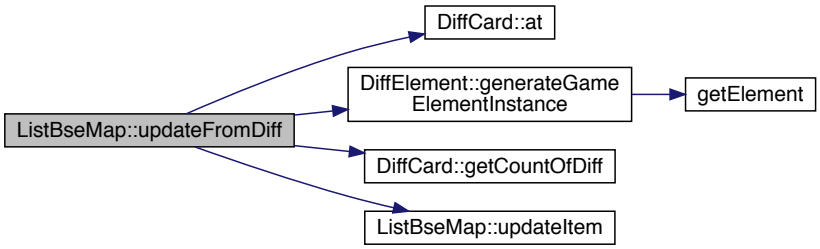
Implements IMap.

Definition at line 40 of file listbsemap.cpp.

References DiffCard::at(), eChange, eDeleted, eEmpty, DiffElement::generateGameElementInstance(), DiffCard::getCountOfDiff(), DiffElement::type, and updateItem().

```
00040                                                {
00041   for (int i = 0; i < diff->getCountOfDiff(); i++) {
00042     switch (diff->at(i)->type) {
00043       case eNew:
00044         items->append(diff->at(i)->generateGameElementInstance());
00045         break;
00046       case eChange:
00047         updateItem(diff->at(i)->generateGameElementInstance());
00048         break;
00049       case eDeleted:
00050         updateItem(diff->at(i)->generateGameElementInstance(), false
      );
00051         break;
00052       case eEmpty:
00053         break;
00054     };
00055   }
00056 }
```

Here is the call graph for this function:



2.14.3.7 updateItem()

```
void ListBseMap::updateItem (
            IBaseGameElement * gameEleement,
            bool isReplace = true )  [protected]
```

updateItem update items in map

Parameters

gameEleement	element with information for update
isReplace	if true than replace old item with . Otherway delete from map list

Definition at line 62 of file listbsemap.cpp.

Referenced by updateFromDiff().

```
00062                                                                                {
00063   for (int i = 0; i < items->size(); i++) {
00064     if ((items->at(i)->getName()) == (gameEleement->getName())) {
00065       delete items->at(i);
00066       items->removeAt(i);
00067       if (isReplace)
00068         items->append(gameEleement);
00069       else
00070         delete gameEleement;
00071     }
00072   }
00073 }
```

Here is the caller graph for this function:

2.14.4 Member Data Documentation

2.14.4.1 items

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

Definition at line 24 of file listbsemap.h.

2.14.4.2 size

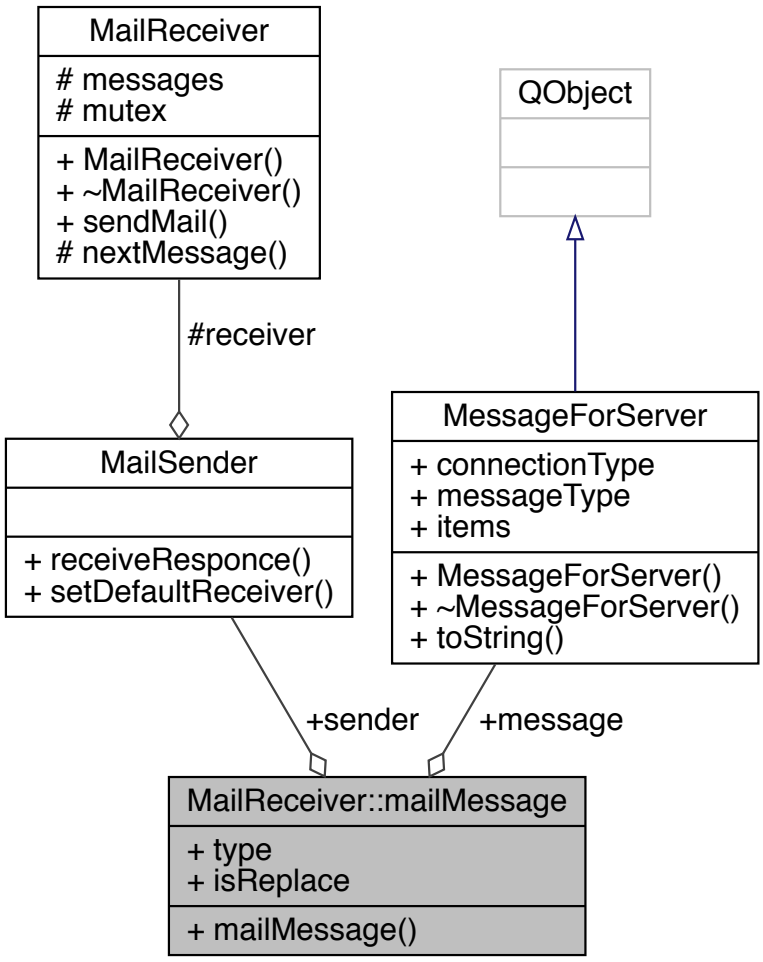
```
QSizeF* ListBseMap::size  [protected]
```

Definition at line 25 of file listbsemap.h.

2.15 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)

2.15.1 Detailed Description

Definition at line 27 of file mailboxelement.h.

2.15.2 Constructor & Destructor Documentation

2.15.2.1 mailMessage()

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

Definition at line 53 of file mailboxelement.cpp.

References isReplace, message, sender, and type.

```
00056                                                          {
00057   this->message = message;
00058   this->sender = sender;
00059   this->type = type;
00060   this->isReplace = isReplace;
00061 }
```

2.15.3 Friends And Related Function Documentation

2.15.3.1 operator<<

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

Definition at line 37 of file mailboxelement.h.

```
00037                                                                  {
00038       QDebugStateSaver saver(debug);
00039
00040       if (c.message != nullptr) {
00041         debug.nospace() << "mailMessage {message:" << (*c.message) << "}";
00042       } else {
00043         debug.nospace() << "mailMessage {message:empty}";
00044       }
00045
00046       return debug;
00047     }
```

2.15.4 Member Data Documentation

2.15.4.1 isReplace

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

Definition at line 32 of file mailboxelement.h.

Referenced by mailMessage().

2.15.4.2 message

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

Definition at line 29 of file mailboxelement.h.

Referenced by mailMessage().

2.15.4.3 sender

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

Definition at line 30 of file mailboxelement.h.

Referenced by mailMessage().

2.15.4.4 type

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

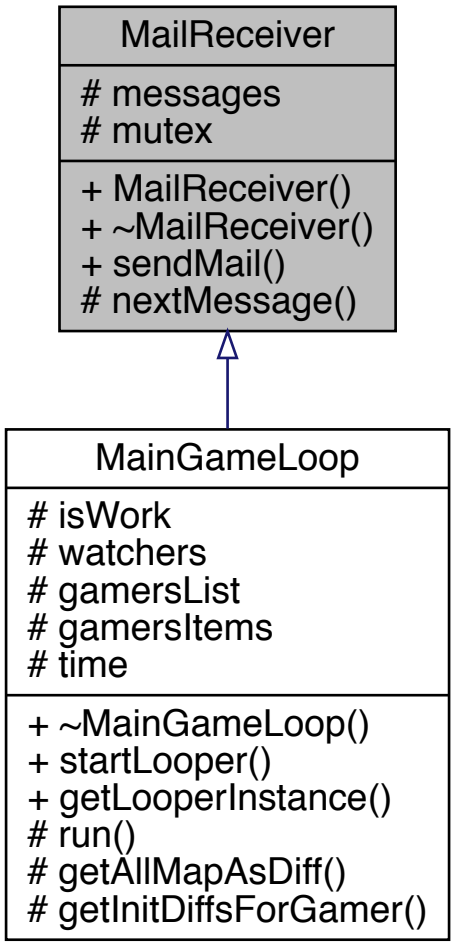
Definition at line 31 of file mailboxelement.h.

Referenced by mailMessage().

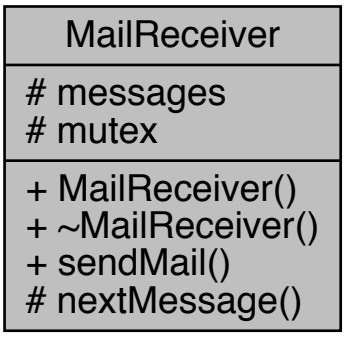
2.16 MailReceiver Class Reference

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

Inheritance diagram for MailReceiver:



Collaboration diagram for MailReceiver:



Classes

- class mailMessage

Public Member Functions

- MailReceiver ()
- ~MailReceiver ()
- 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

2.16.1 Detailed Description

Definition at line 17 of file mailboxelement.h.

2.16.2 Constructor & Destructor Documentation

2.16.2.1 MailReceiver()

```
MailReceiver::MailReceiver ( )
```

Definition at line 3 of file mailboxelement.cpp.

```
00003                             {
00004   mutex = new QMutex();
00005 }
```

2.16.2.2 ~MailReceiver()

```
MailReceiver::~MailReceiver ( )
```

Definition at line 7 of file mailboxelement.cpp.

```
00007                              {
00008   delete mutex;
00009 }
```

2.16.3 Member Function Documentation

2.16.3.1 nextMessage()

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

Definition at line 42 of file mailboxelement.cpp.

```
00042                                                  {
00043   mutex->lock();
00044   mailMessage* result = nullptr;
00045   if (messages.length() > 0) {
00046     qInfo() << "enqueue message";
00047     result = messages.takeFirst();
00048   }
00049   mutex->unlock();
00050   return result;
00051 }
```

2.16.3.2 sendMail()

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

Definition at line 11 of file mailboxelement.cpp.

```
00014                                           {
00015   run(QThreadPool::globalInstance(), [=] {
00016     qInfo() << "MailReceiver::sendMail - befor mutex";
00017     mutex->lock();
00018     qInfo() << "MailReceiver::sendMail - in mutex";
00019     qInfo() << "MailReceiver :: adding new message item in to queue";
00020     mailMessage* msg;
00021     if (isReplace && messages.length() != 0) {
00022       for (int i = 0; i < messages.length(); i++) {
00023         msg = messages.at(i);
00024         if (msg->sender == sender && msg->type == type) {
00025           messages.removeAt(i);
00026           messages.insert(i, new mailMessage(message, sender, type, isReplace));
00027           break;
00028         }
00029         if (i == messages.length() - 1) {
00030           messages.push_back(new mailMessage(message, sender, type, isReplace));
00031         }
00032       }
00033     } else
00034       messages.push_back(new mailMessage(message, sender, type, isReplace));
00035     mailMessage* test = messages.last();
00036     qInfo() << "getting new message in queue :: " << (*messages.last());
00037     mutex->unlock();
00038     qInfo() << "MailReceiver::sendMail - after mutex";
00039   });
00040 }
```

2.16.4 Member Data Documentation

2.16.4.1 messages

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

Definition at line 49 of file mailboxelement.h.

2.16.4.2 mutex

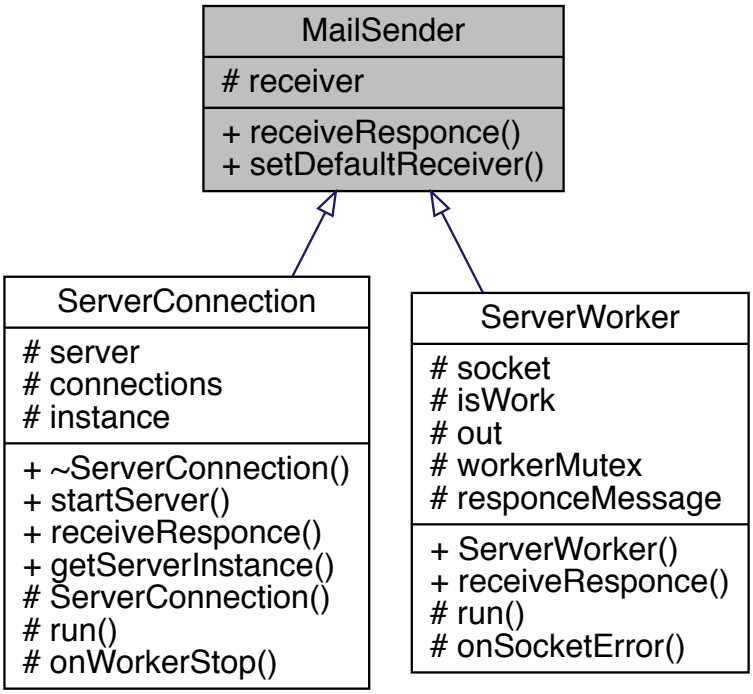
```
QMutex* MailReceiver::mutex  [protected]
```

Definition at line 50 of file mailboxelement.h.

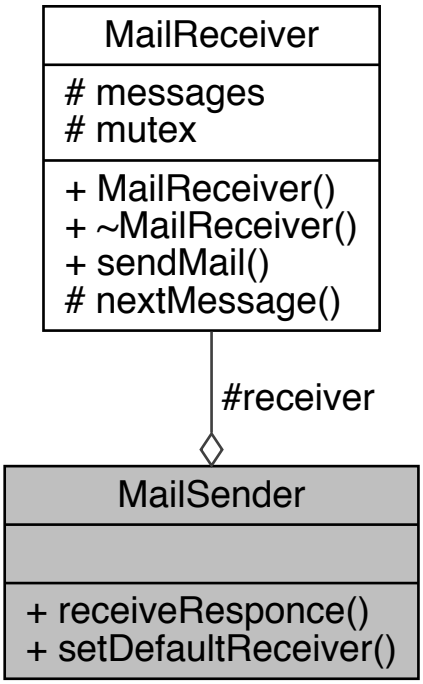
2.17 MailSender Class Reference

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

Inheritance diagram for MailSender:



Collaboration diagram for MailSender:



Public Member Functions

- virtual void receiveResponce (DiffCard ∗diff, MessageForServer ∗message)=0
- virtual void setDefaultReceiver (MailReceiver ∗receiver)

Protected Attributes

- MailReceiver ∗ receiver

2.17.1 Detailed Description

Definition at line 55 of file mailboxelement.h.

2.17.2 Member Function Documentation

2.17.2.1 receiveResponce()

```
virtual void MailSender::receiveResponce (
            DiffCard * diff,
            MessageForServer * message )  [pure virtual]
```

Implemented in ServerWorker, and ServerConnection.

2.17.2.2 setDefaultReceiver()

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

Definition at line 58 of file mailboxelement.h.

References receiver.

```
00058                                                    {
00059     this->receiver = receiver;
00060   }
```

2.17.3 Member Data Documentation

2.17.3.1 receiver

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

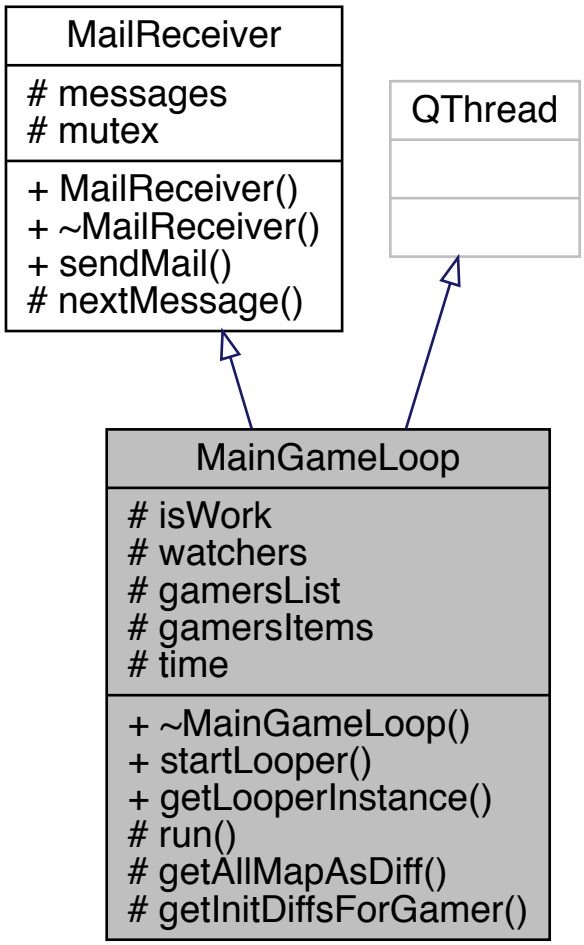
Definition at line 63 of file mailboxelement.h.

Referenced by setDefaultReceiver().

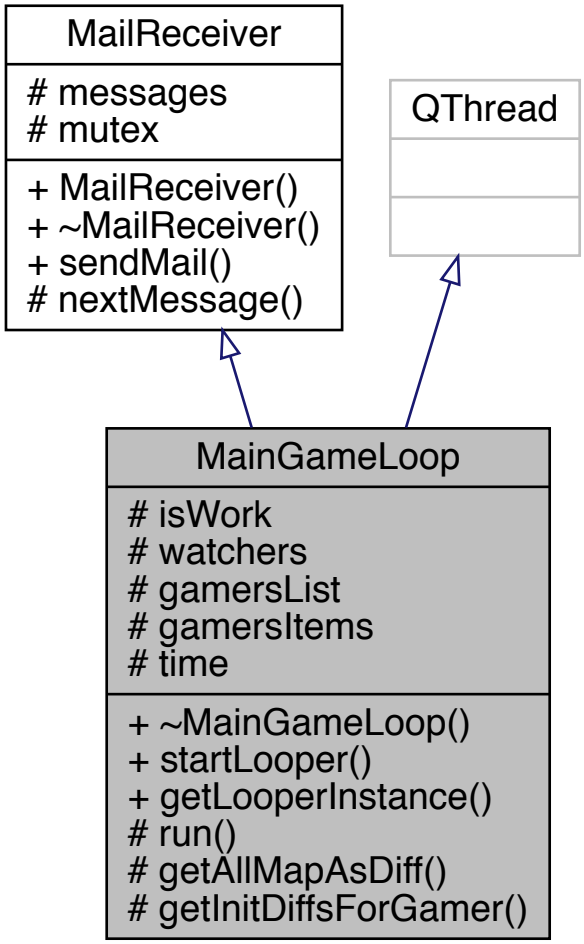
2.18 MainGameLoop Class Reference

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

Inheritance diagram for MainGameLoop:



Collaboration diagram for MainGameLoop:



Classes

- class GamerInformation
- class WathcerInformation

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 ()
- DiffCard * getAllMapAsDiff ()
- QList< DiffElement * > * getInitDiffsForGamer (BaseBasis *basis)
- virtual mailMessage * nextMessage ()

Protected Attributes

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

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

  gamersItems
- uint64_t time
- QQueue< mailMessage * > messages
- QMutex * mutex

2.18.1 Detailed Description

Definition at line 19 of file maingameloop.h.

2.18.2 Constructor & Destructor Documentation

2.18.2.1 ~MainGameLoop()

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

Definition at line 135 of file maingameloop.cpp.

```
00135                               {
00136   delete map;
00137   for (int i = 0; i < gamersItems.size(); i++) {
00138     delete gamersItems.at(i);
00139   }
00140 }
```

2.18.3 Member Function Documentation

2.18.3.1 getAllMapAsDiff()

```
DiffCard * MainGameLoop::getAllMapAsDiff ( )  [protected]
```

Definition at line 126 of file maingameloop.cpp.

References DiffCard::loadFromList().

```
00126                                       {
00127   DiffCard* result = new SimpleDiffCard();
00128   DiffElement* item;
00129   for (int i = 0; i < map->getCount(); i++)
00130     result->append(
00131         new DiffElement(eDiffType::eNew, map->
      getElementAtPosition(i)));
00132   return result;
00133 }
```

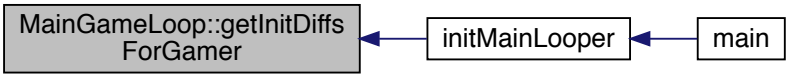
Here is the call graph for this function:

2.18.3.2 getInitDiffsForGamer()

```
QList<DiffElement*>* MainGameLoop::getInitDiffsForGamer (
            BaseBasis * basis )  [protected]
```

Referenced by initMainLooper().

Here is the caller graph for this function:



2.18.3.3 getLooperInstance()

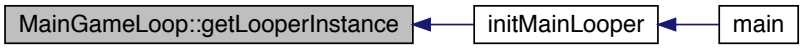
```
static MainGameLoop* MainGameLoop::getLooperInstance ( )  [inline], [static]
```

Definition at line 22 of file maingameloop.h.

Referenced by initMainLooper().

```
00022                                                {
00023     if (instance == nullptr) {
00024       instance = new MainGameLoop();
00025     }
00026     return instance;
00027   }
```

Here is the caller graph for this function:



2.18.3.4 nextMessage()

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

Definition at line 42 of file mailboxelement.cpp.

```
00042                                                {
00043   mutex->lock();
00044   mailMessage* result = nullptr;
00045   if (messages.length() > 0) {
00046     qInfo() << "enqueue message";
00047     result = messages.takeFirst();
00048   }
00049   mutex->unlock();
00050   return result;
00051 }
```

2.18.3.5 run()

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

Definition at line 100 of file maingameloop.cpp.

References isWork.

```
00100                       {
00101   mailMessage* msg;
00102   qInfo() << "starting main loop";
00103   while (isWork) {
00104     // process incoming messages
00105     while ((msg = nextMessage()) != nullptr) {
00106       qInfo() << TAG << "receive new messahe " << (*msg);
00107       auto fun = getProccessorForMessage(msg->message->connectionType);
00108       (this->*fun)(msg, msg->sender);
00109     }
00110
00111     foreach (auto items, gamersItems) {
00112       for (int i = 0; i < items->size(); i++) {
00113         //        switch ((eBaseGameElementType)items->at(i)->getType())
00114       }
00115     }
00116     for (int i = 0; i < watchers.size(); i++) {
00117       WathcerInformation* watcher = watchers.at(i);
00118       watcher->personalDiff->updateFromOtherCard(stepDiff);
00119     }
00120     stepDiff->freeItems();
00121
00122     msleep(100);
00123   }
00124 }
```

2.18.3.6 sendMail()

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

Definition at line 11 of file mailboxelement.cpp.

```
00014                                           {
00015   run(QThreadPool::globalInstance(), [=] {
00016     qInfo() << "MailReceiver::sendMail - befor mutex";
00017     mutex->lock();
00018     qInfo() << "MailReceiver::sendMail - in mutex";
00019     qInfo() << "MailReceiver :: adding new message item in to queue";
00020     mailMessage* msg;
00021     if (isReplace && messages.length() != 0) {
00022       for (int i = 0; i < messages.length(); i++) {
00023         msg = messages.at(i);
00024         if (msg->sender == sender && msg->type == type) {
00025           messages.removeAt(i);
00026           messages.insert(i, new mailMessage(message, sender, type, isReplace));
00027           break;
00028         }
00029         if (i == messages.length() - 1) {
00030           messages.push_back(new mailMessage(message, sender, type, isReplace));
00031         }
00032       }
00033     } else
00034       messages.push_back(new mailMessage(message, sender, type, isReplace));
00035     mailMessage* test = messages.last();
00036     qInfo() << "getting new message in queue :: " << (*messages.last());
00037     mutex->unlock();
00038     qInfo() << "MailReceiver::sendMail - after mutex";
00039   });
00040 }
```

2.18.3.7 startLooper()

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

Definition at line 142 of file maingameloop.cpp.

Referenced by initMainLooper().

```
00142                                {
00143   this->start();
00144 }
```

Here is the caller graph for this function:

2.18.4 Member Data Documentation

2.18.4.1 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 79 of file maingameloop.h.

2.18.4.2 gamersList

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

gamersList

list of client as gamers

Definition at line 73 of file maingameloop.h.

2.18.4.3 isWork

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

Definition at line 65 of file maingameloop.h.

Referenced by run().

2.18.4.4 messages

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

Definition at line 49 of file mailboxelement.h.

2.18.4.5 mutex

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

Definition at line 50 of file mailboxelement.h.

2.18.4.6 time

```
uint64_t MainGameLoop::time  [protected]
```

Definition at line 81 of file maingameloop.h.

2.18.4.7 watchers

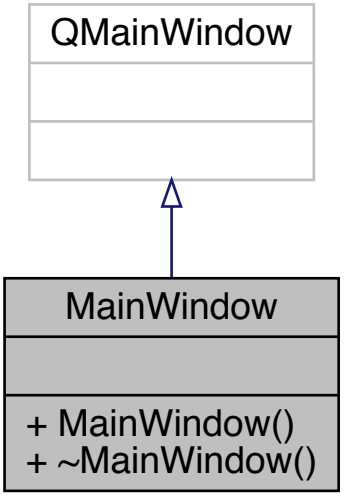
```
QList<WathcerInformation*> MainGameLoop::watchers  [protected]
```

Definition at line 68 of file maingameloop.h.

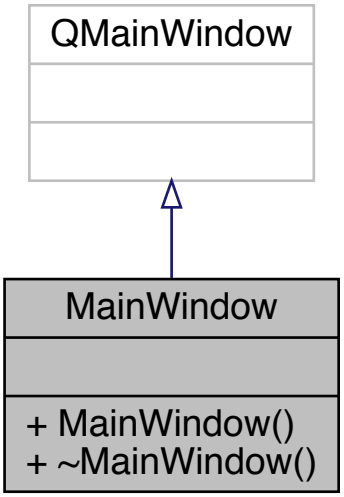
2.19 MainWindow Class Reference

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

Inheritance diagram for MainWindow:



Collaboration diagram for MainWindow:



Public Member Functions

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

2.19.1 Detailed Description

Definition at line 11 of file mainwindow.h.

2.19.2 Constructor & Destructor Documentation

2.19.2.1 MainWindow()

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

Definition at line 4 of file mainwindow.cpp.

Referenced by main().

```
00005     : QMainWindow(parent), ui(new Ui::MainWindow) {
00006   ui->setupUi(this);
00007   ui->gameField->connectToServer();
00008 }
```

Here is the caller graph for this function:

2.19.2.2 ~MainWindow()

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

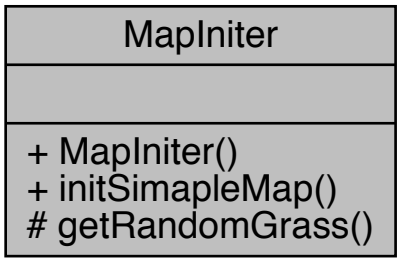
Definition at line 10 of file mainwindow.cpp.

```
00010                       {
00011   delete ui;
00012 }
```

2.20 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)

2.20.1 Detailed Description

Definition at line 8 of file mapiniter.h.

2.20.2 Constructor & Destructor Documentation

2.20.2.1 MapIniter()

`MapIniter::MapIniter ( )`

Definition at line 3 of file mapiniter.cpp.

```
00003 {}
```

2.20.3 Member Function Documentation

2.20.3.1 getRandomGrass()

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

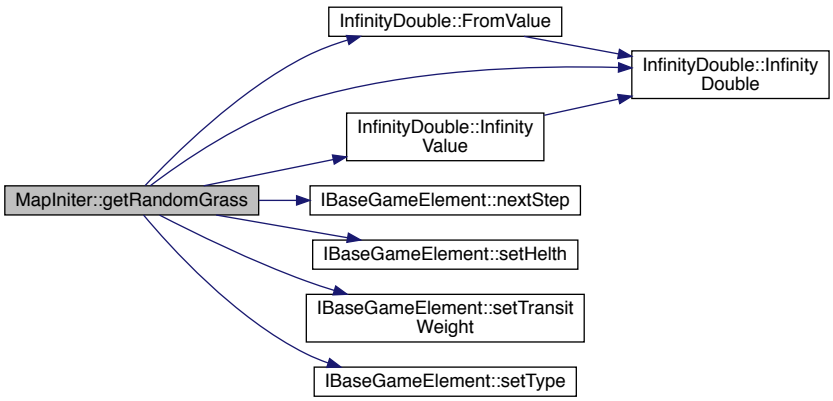
Definition at line 17 of file mapiniter.cpp.

References eGrass, InfinityDouble::FromValue(), InfinityDouble::InfinityDouble(), InfinityDouble::InfinityValue(), IBaseGameElement::nextStep(), IBaseGameElement::setHelth(), IBaseGameElement::setTransitWeight(), and IBaseGameElement::setType().

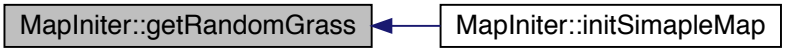
Referenced by initSimapleMap().

```
00017                                                                                {
00018   IBaseGameElement* result = new IBaseGameElement();
00019   result->setPosition(new QVector3D(
00020       (double)rand() / (double)RAND_MAX * maxWidth - maxWidth / 2,
00021       (double)rand() / (double)RAND_MAX * maxHeigth - maxHeigth / 2, 0));
00022   result->setHelth(InfinityDouble::InfinityValue());
00023   result->setTransitWeight(InfinityDouble::FromValue(0));
00024   result->setType(eBaseGameElementType::eGrass);
00025   return result;
00026 }
```

Here is the call graph for this function:



Here is the caller graph for this function:



2.20.3.2 initSimapleMap()

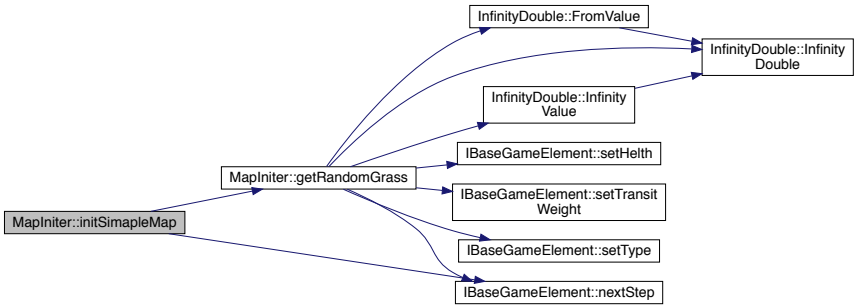
```
IMap * MapIniter::initSimapleMap ( )
```

Definition at line 5 of file mapiniter.cpp.

References getRandomGrass(), and IBaseGameElement::nextStep().

```
00005                                {
00006   const int mapW = 100;
00007   const int mapH = 100;
00008
00009   IMap* result = new ListBseMap(mapW, mapH);
00010   for (int i = 0; i < 400; i++) {
00011     IBaseGameElement* item = getRandomGrass(mapW, mapH);
00012     result->insertElement(item);
00013   }
00014   return result;
00015 }
```

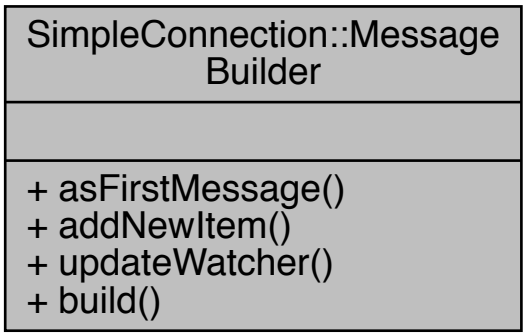
Here is the call graph for this function:



2.21 SimpleConnection::MessageBuilder Class Reference

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

Collaboration diagram for SimpleConnection::MessageBuilder:



Public Member Functions

- MessageBuilder * asFirstMessage (eConnectionType type)
- MessageBuilder * addNewItem (QList< IBaseGameElement *> *newEleements)
- MessageBuilder * updateWatcher ()
- void build ()

Friends

- class SimpleConnection

2.21.1 Detailed Description

Definition at line 77 of file simpleconnection.h.

2.21.2 Member Function Documentation

2.21.2.1 addNewItem()

```
SimpleConnection::MessageBuilder * SimpleConnection::MessageBuilder::addNewItem (
            QList< IBaseGameElement *> * newEleements )
```

Definition at line 92 of file simpleconnection.cpp.

```
00093                                                {
00094   this->message->connectionType = eConnectionType::eGamer;
00095   this->message->messageType = eInsertNewItem;
00096   this->message->items->append(*newEleements);
00097   return this;
00098 }
```

2.21.2.2 asFirstMessage()

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

Definition at line 86 of file simpleconnection.cpp.

References eFirstMessae, and MessageForServer::messageType.

```
00086                                                                    {
00087   this->message->connectionType = type;
00088   this->message->messageType = eFirstMessae;
00089   return this;
00090 }
```

2.21.2.3 build()

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

Definition at line 107 of file simpleconnection.cpp.

```
00107                                          {
00108   parent->addMessage(this);
00109 }
```

2.21.2.4 updateWatcher()

```
SimpleConnection::MessageBuilder * SimpleConnection::MessageBuilder::updateWatcher ( )
```

Definition at line 101 of file simpleconnection.cpp.

```
00101                                              {
00102   this->message->connectionType = eConnectionType::eWatcher;
00103   this->message->messageType = eGetUpdateMessage;
00104   return this;
00105 }
```

2.21.3 Friends And Related Function Documentation

2.21.3.1 SimpleConnection

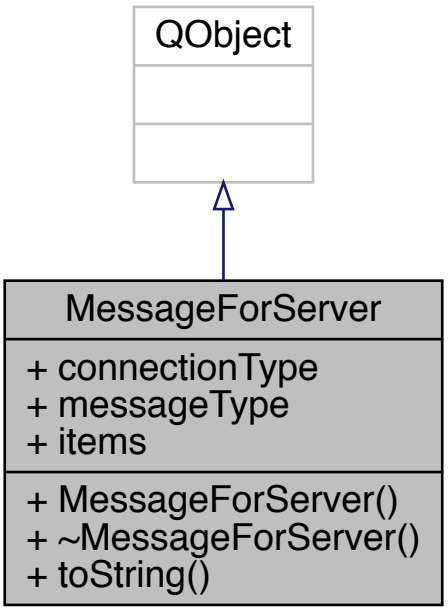
```
friend class SimpleConnection  [friend]
```

Definition at line 78 of file simpleconnection.h.

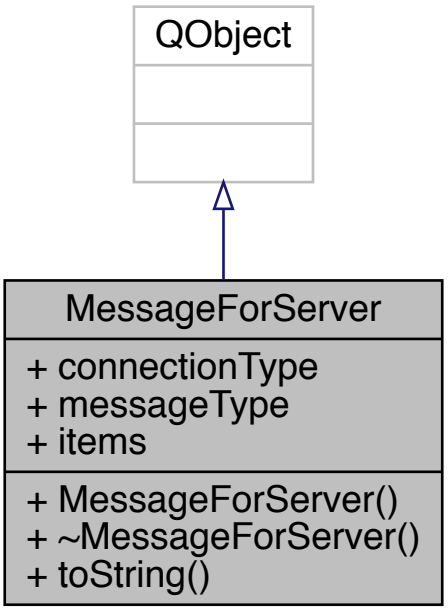
2.22 MessageForServer Class Reference

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

Inheritance diagram for MessageForServer:



Collaboration diagram for MessageForServer:



Public Member Functions

- MessageForServer ()
- ~MessageForServer ()
- QString toString ()

Public Attributes

- eConnectionType connectionType
- eMessageType messageType
- QList< IBaseGameElement * > * items

Friends

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

2.22.1 Detailed Description

Definition at line 21 of file simpleconnection.h.

2.22.2 Constructor & Destructor Documentation

2.22.2.1 MessageForServer()

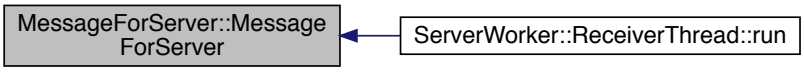
```
MessageForServer::MessageForServer ( )  [inline]
```

Definition at line 29 of file simpleconnection.h.

Referenced by ServerWorker::ReceiverThread::run().

```
00029 { items = new QList<IBaseGameElement*>(); }
```

Here is the caller graph for this function:



2.22.2.2 ~MessageForServer()

```
MessageForServer::~MessageForServer ( )  [inline]
```

Definition at line 31 of file simpleconnection.h.

```
00031 { delete items; }
```

2.22.3 Member Function Documentation

2.22.3.1 toString()

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

Definition at line 115 of file simpleconnection.cpp.

```
00115                                    {
00116   return "MessageForServer :: connectionType = " + stringify(
      connectionType) +
00117          "; messageType = " + stringify(messageType);
00118 }
```

2.22.4 Friends And Related Function Documentation

2.22.4.1 operator<< [1/2]

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

Definition at line 33 of file simpleconnection.h.

```
00034                                                                  {
00035     int itemsSize = myclass.items->size();
00036     QDataStream& result = stream << ((int)myclass.connectionType)
00037                                  << ((int)myclass.messageType) << itemsSize;
00038     for (int i = 0; i < myclass.items->size(); i++)
00039       result << (*myclass.items->at(i));
00040     return result;
00041   }
```

2.22.4.2 operator<< [2/2]

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

Definition at line 59 of file simpleconnection.h.

```
00059                                                                 {
00060     QDebugStateSaver saver(debug);
00061     debug.nospace() << c.toString();
00062
00063     return debug;
00064   }
```

2.22.4.3 operator>>

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

Definition at line 42 of file simpleconnection.h.

```
00043                                                                 {
00044     int con;
00045     int msg;
00046     int countOfNewItems = 0;
00047     QDataStream& result = (stream >> con >> msg >> countOfNewItems);
00048     myclass.connectionType = (eConnectionType)con;
00049     myclass.messageType = (eMessageType)msg;
00050     for (int i = 0; i < countOfNewItems; i++) {
00051       GameElementData item;
00052       item.defaultInit();
00053       result >> item;
00054       myclass.items->append(getElement(item));
00055     }
00056     return result;
00057   }
```

2.22.5 Member Data Documentation

2.22.5.1 connectionType

```
eConnectionType MessageForServer::connectionType
```

Definition at line 25 of file simpleconnection.h.

2.22.5.2 items

```
QList<IBaseGameElement*>* MessageForServer::items
```

Definition at line 27 of file simpleconnection.h.

Referenced by ServerWorker::ReceiverThread::run().

2.22.5.3 messageType

```
eMessageType MessageForServer::messageType
```

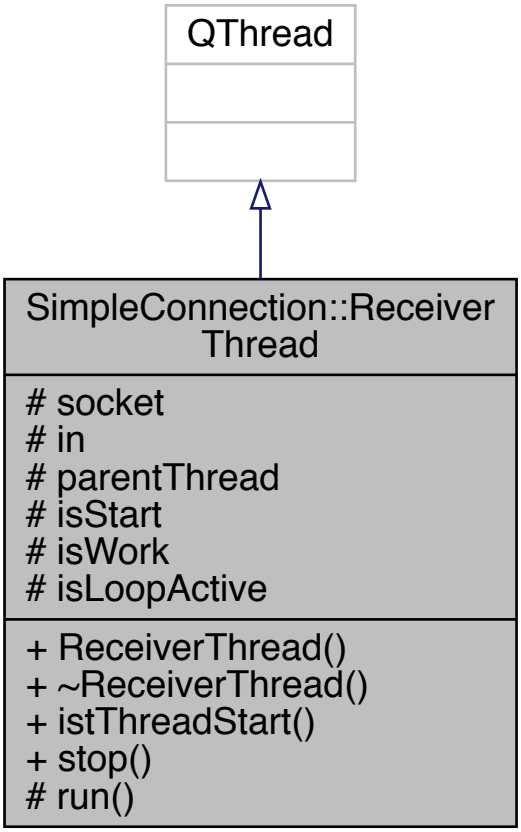
Definition at line 26 of file simpleconnection.h.

Referenced by SimpleConnection::MessageBuilder::asFirstMessage().

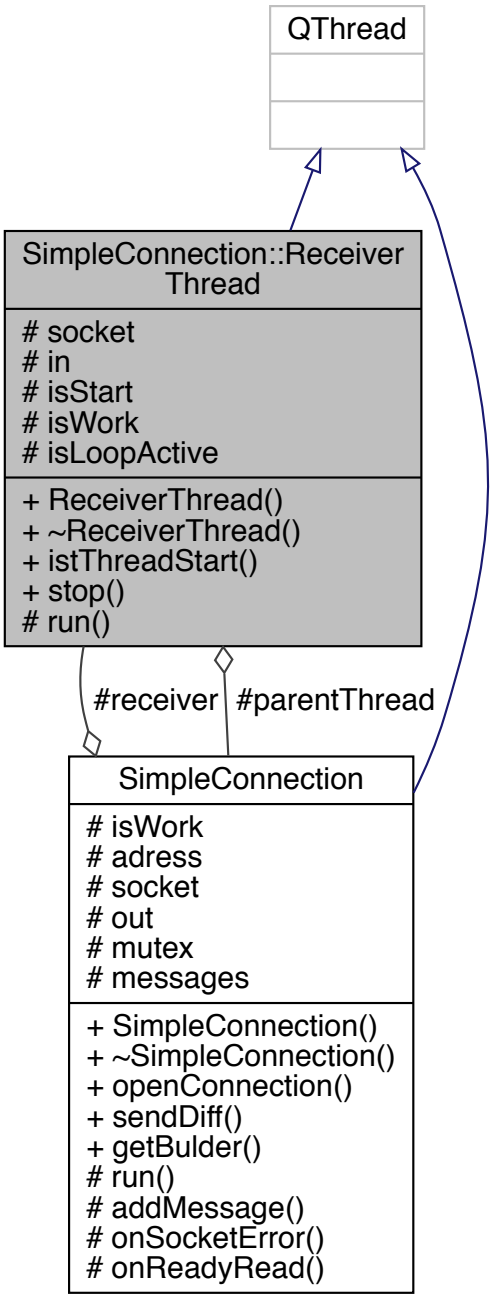
2.23 SimpleConnection::ReceiverThread Class Reference

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

Inheritance diagram for SimpleConnection::ReceiverThread:



Collaboration diagram for SimpleConnection::ReceiverThread:



Public Member Functions

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

Protected Member Functions

- void run () Q_DECL_OVERRIDE

Protected Attributes

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

2.23.1 Detailed Description

Definition at line 118 of file simpleconnection.h.

2.23.2 Constructor & Destructor Documentation

2.23.2.1 ReceiverThread()

```
SimpleConnection::ReceiverThread::ReceiverThread (
            QTcpSocket * socket,
            SimpleConnection * parentThread,
            QObject * parent = 0 )  [inline]
```

Definition at line 120 of file simpleconnection.h.

References parentThread.

```
00122                                       {
00123       this->socket = socket;
00124       this->parentThread = parentThread;
00125       in = new QDataStream(socket);
00126       in->setVersion(QDataStream::Qt_5_7);
00127     }
```

2.23.2.2 ~ReceiverThread()

```
virtual SimpleConnection::ReceiverThread::~ReceiverThread ( )  [inline], [virtual]
```

Definition at line 129 of file simpleconnection.h.

```
00129 { delete in; }
```

2.23.3 Member Function Documentation

2.23.3.1 istThreadStart()

```
bool SimpleConnection::ReceiverThread::istThreadStart ( )
```

Definition at line 140 of file simpleconnection.cpp.

```
00140                                                 {
00141   return isStart;
00142   msleep(100);
00143 }
```

2.23.3.2 run()

```
void SimpleConnection::ReceiverThread::run ( )  [inline], [protected]
```

Definition at line 144 of file simpleconnection.h.

```
00144                                 {
00145       qInfo() << "starting message receiver loop";
00146
00147       int diffsLenth;
00148       MessageForServer sendedMessage;
00149       while (true) {
00150         isLoopActive = true;
00151         if (!isWork)
00152           break;
00153         isStart = true;
00154         bool waitForReadyReadResult = socket->waitForReadyRead(-1);
00155
00156         qInfo() << "starting reading some response";
00157         (*in) >> sendedMessage;
00158         (*in) >> diffsLenth;
00159         QList<DiffElement*>* result = new QList<DiffElement*>();
00160         for (int i = 0; i < diffsLenth; i++) {
00161           DiffElement* newItem = new DiffElement();
00162           (*in) >> (*newItem);
00163           result->append(newItem);
00164         }
00165         parentThread->sendDiff(result);
00166         qInfo() << "something read from server";
00167       }
00168       isLoopActive = false;
00169     }
```

2.23.3.3 stop()

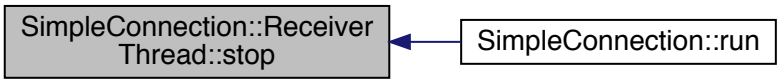
```
void SimpleConnection::ReceiverThread::stop ( )
```

Definition at line 145 of file simpleconnection.cpp.

Referenced by SimpleConnection::run().

```
00145                                          {
00146   isWork = false;
00147   while (isLoopActive) {
00148     msleep(100);
00149   }
00150 }
```

Here is the caller graph for this function:



2.23.4 Member Data Documentation

2.23.4.1 in

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

Definition at line 136 of file simpleconnection.h.

2.23.4.2 isLoopActive

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

Definition at line 140 of file simpleconnection.h.

2.23.4.3 isStart

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

Definition at line 138 of file simpleconnection.h.

2.23.4.4 isWork

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

Definition at line 139 of file simpleconnection.h.

2.23.4.5 parentThread

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

Definition at line 137 of file simpleconnection.h.

Referenced by ReceiverThread().

2.23.4.6 socket

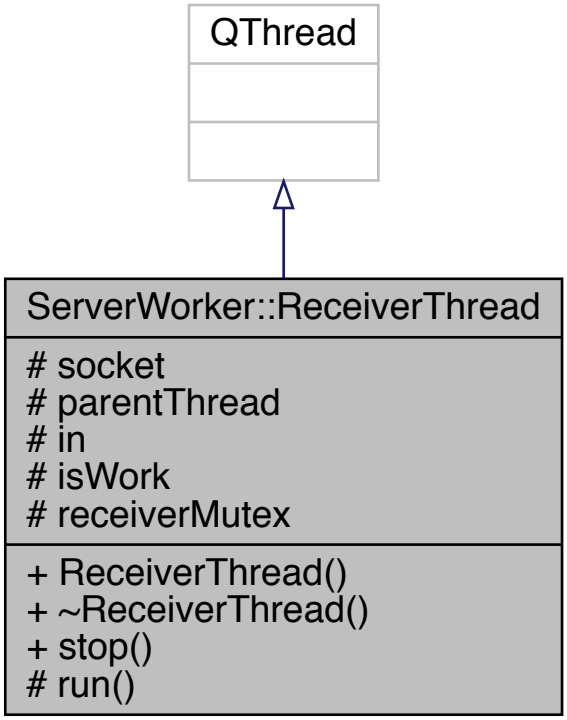
```
QTcpSocket* SimpleConnection::ReceiverThread::socket  [protected]
```

Definition at line 135 of file simpleconnection.h.

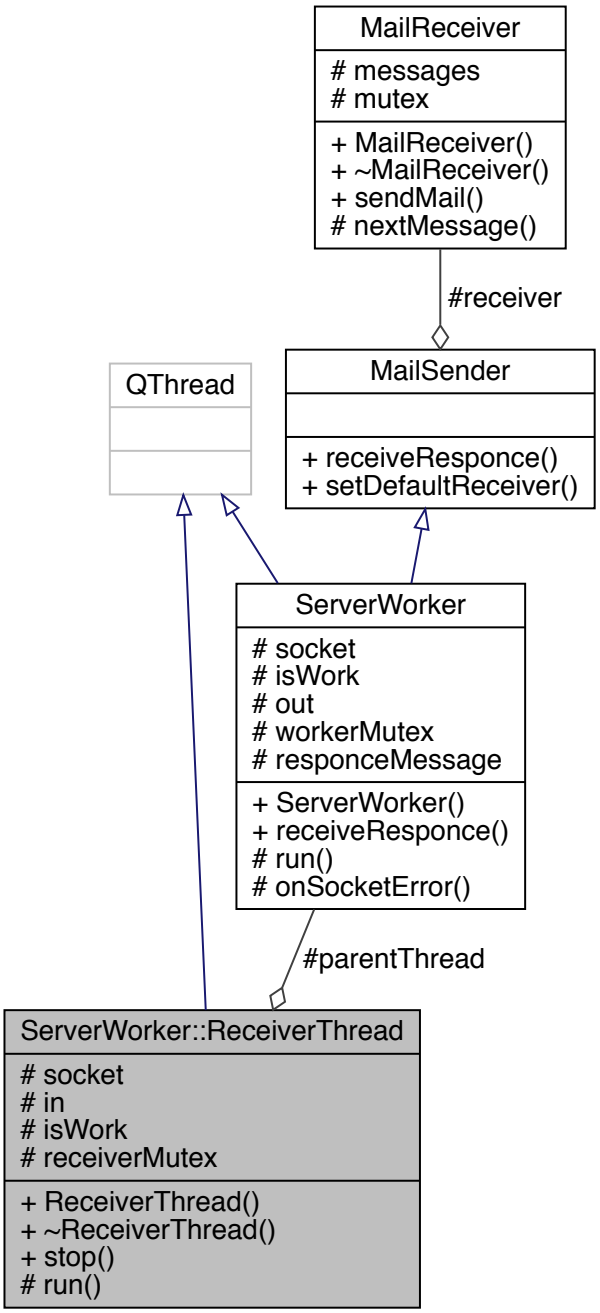
2.24 ServerWorker::ReceiverThread Class Reference

```
#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

2.24.1 Detailed Description

Definition at line 44 of file serverworker.h.

2.24.2 Constructor & Destructor Documentation

2.24.2.1 ReceiverThread()

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

Definition at line 66 of file serverworker.cpp.

References parentThread.

```
00067                                                       {
00068     this->socket = socket;
00069     this->parentThread = parent;
00070     this->start();
00071 }
```

2.24.2.2 ~ReceiverThread()

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

Definition at line 73 of file serverworker.cpp.

```
00073                                          {
00074     delete in;
00075 }
```

2.24.3 Member Function Documentation

2.24.3.1 run()

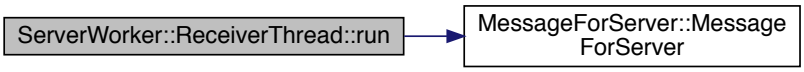
```
void ServerWorker::ReceiverThread::run ( )  [protected]
```

Definition at line 83 of file serverworker.cpp.

References isWork, MessageForServer::items, and MessageForServer::MessageForServer().

```
00083                                    {
00084   in = new QDataStream(socket);
00085   in->setVersion(QDataStream::Qt_5_7);
00086
00087   qInfo() << socket->peerAddress() << ":" << socket->peerPort()
00088           << "starting receiving thread";
00089
00090   while (true) {
00091     receiverMutex.lock();
00092     socket->waitForReadyRead(-1);
00093     if (socket->state() == QTcpSocket::UnconnectedState ||
00094         socket->state() == QAbstractSocket::ClosingState)
00095       isWork = false;
00096     if (!isWork) {
00097       qInfo() << socket->peerAddress() << ":" << socket->peerPort()
00098               << "stoping receiver loop";
00099       break;
00100     }
00101     qInfo() << socket->peerAddress() << ":" << socket->peerPort()
00102             << "ready read from client";
00103     MessageForServer* newMessage = new MessageForServer();
00104     (*in) >> (*newMessage);
00105     qInfo() << socket->peerAddress() << ":" << socket->peerPort()
00106             << "get new message on server :: " << (*newMessage).toString();
00107     parentThread->receiver->sendMail(newMessage,
      parentThread,
00108                                      newMessage->messageType);
00109     receiverMutex.unlock();
00110   }
00111 }
```

Here is the call graph for this function:



2.24.3.2 stop()

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

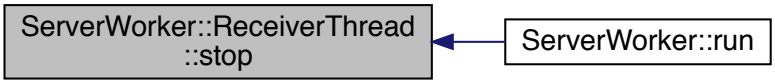
Definition at line 77 of file serverworker.cpp.

References isWork.

Referenced by ServerWorker::run().

```
00077                                       {
00078     receiverMutex.lock();
00079     isWork = false;
00080     receiverMutex.unlock();
00081 }
```

Here is the caller graph for this function:



2.24.4 Member Data Documentation

2.24.4.1 in

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

Definition at line 55 of file serverworker.h.

2.24.4.2 isWork

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

Definition at line 56 of file serverworker.h.

Referenced by run(), and stop().

2.24.4.3 parentThread

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

Definition at line 54 of file serverworker.h.

Referenced by ReceiverThread().

2.24.4.4 receiverMutex

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

Definition at line 57 of file serverworker.h.

2.24.4.5 socket

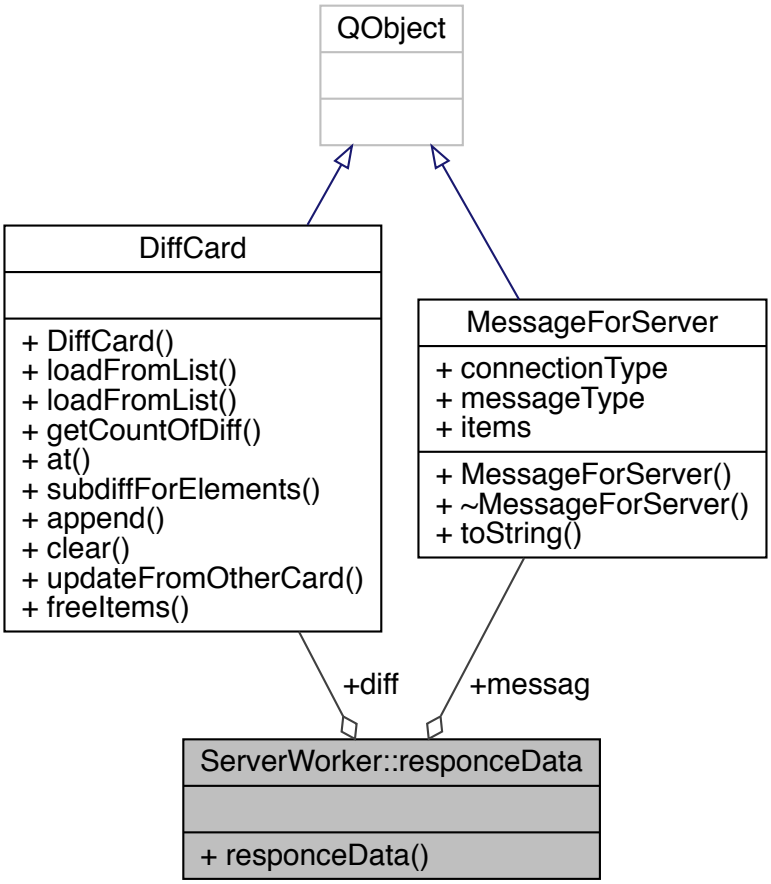
```
QTcpSocket* ServerWorker::ReceiverThread::socket  [protected]
```

Definition at line 53 of file serverworker.h.

2.25 ServerWorker::responceData Struct Reference

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

Collaboration diagram for ServerWorker::responceData:



Public Member Functions

- responceData (DiffCard *_diff, MessageForServer *_messag)

Public Attributes

- DiffCard * diff
- MessageForServer * messag

2.25.1 Detailed Description

Definition at line 27 of file serverworker.h.

2.25.2 Constructor & Destructor Documentation

2.25.2.1 responceData()

```
ServerWorker::responceData::responceData (
            DiffCard * _diff,
            MessageForServer * _messag )  [inline]
```

Definition at line 29 of file serverworker.h.

References diff, and messag.

```
00029                                                          {
00030       this->diff = _diff;
00031       this->messag = _messag;
00032     }
```

2.25.3 Member Data Documentation

2.25.3.1 diff

```
DiffCard* ServerWorker::responceData::diff
```

Definition at line 34 of file serverworker.h.

Referenced by responceData().

2.25.3.2 messag

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

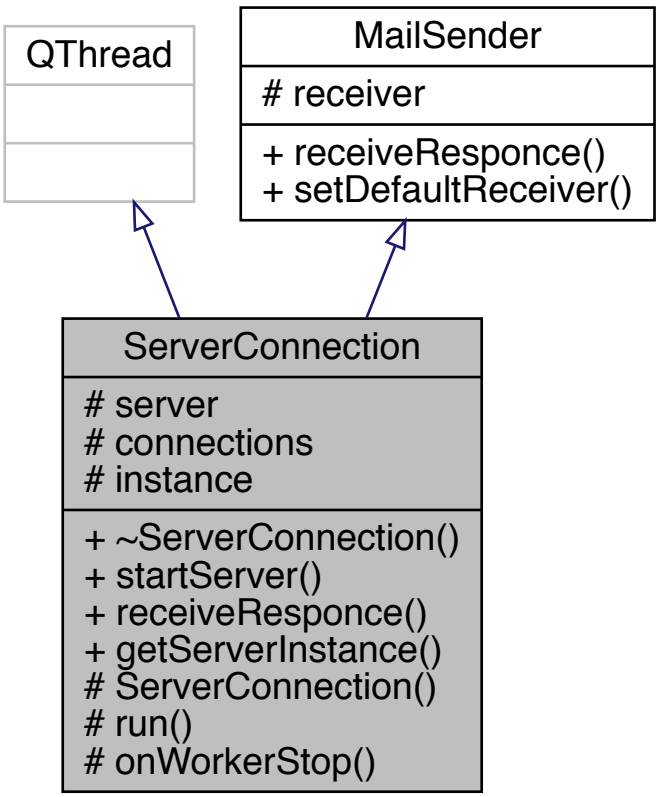
Definition at line 35 of file serverworker.h.

Referenced by responceData().

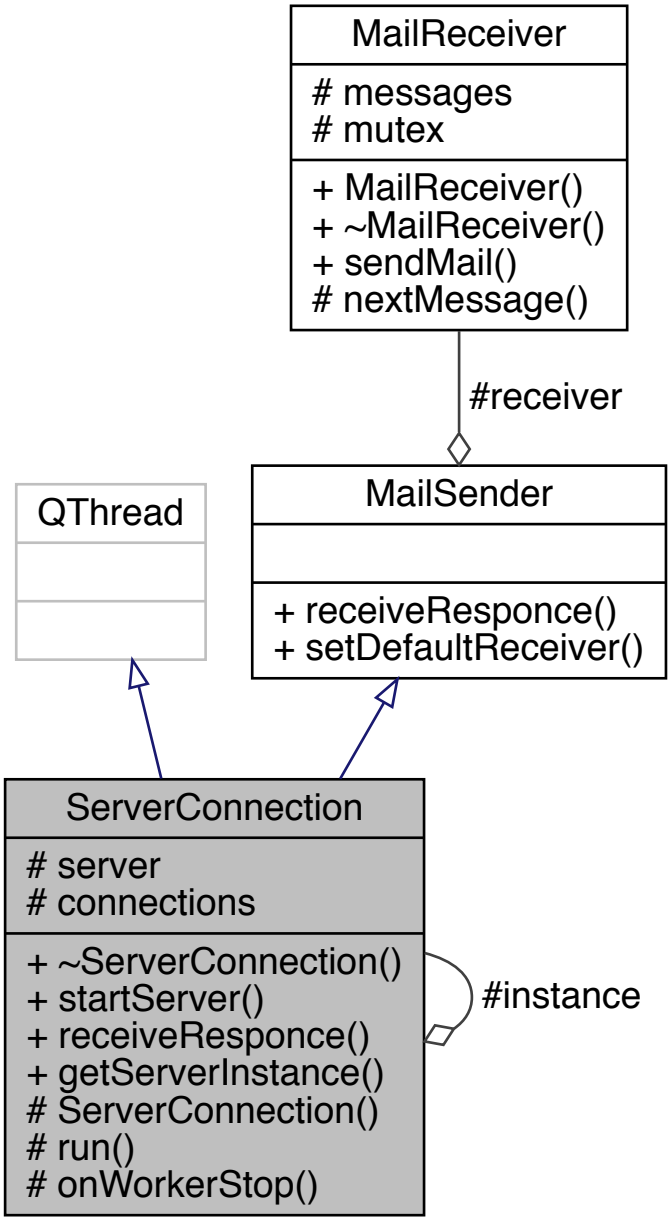
2.26 ServerConnection Class Reference

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

Inheritance diagram for ServerConnection:



Collaboration diagram for ServerConnection:



Signals

- void onServerError (serverError error)

Public Member Functions

- ~ServerConnection ()

- void startServer ()
- void receiveResponce (DiffCard *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

2.26.1 Detailed Description

Definition at line 12 of file serverconnection.h.

2.26.2 Constructor & Destructor Documentation

2.26.2.1 ~ServerConnection()

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

Definition at line 6 of file serverconnection.cpp.

```
00006                                    {
00007   for (int i = 0; i < connections->size(); i++)
00008     delete connections->at(i);
00009   delete connections;
00010 }
```

2.26.2.2 ServerConnection()

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

Definition at line 12 of file serverconnection.cpp.

```
00012                                    {
00013   connections = new QList<ServerWorker*>();
00014 }
```

2.26.3 Member Function Documentation

2.26.3.1 getServerInstance()

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

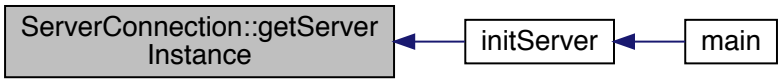
Definition at line 18 of file serverconnection.h.

References instance.

Referenced by initServer().

```
00018                                                    {
00019     if (instance == nullptr)
00020       instance = new ServerConnection();
00021     return instance;
00022   }
```

Here is the caller graph for this function:



2.26.3.2 onServerError

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

2.26.3.3 onWorkerStop

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

Definition at line 16 of file serverconnection.cpp.

Referenced by initServer().

```
00016                                                   {
00017   connections->removeAll(worker);
00018   delete worker;
00019 }
```

Here is the caller graph for this function:

2.26.3.4 receiveResponce()

```
void ServerConnection::receiveResponce (
            DiffCard * diff,
            MessageForServer * message )  [virtual]
```

Implements MailSender.

Definition at line 54 of file serverconnection.cpp.

```
00055                                                   {}
```

2.26.3.5 run()

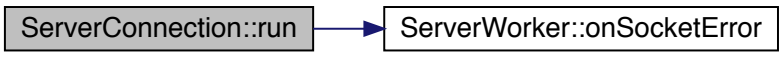
```
void ServerConnection::run ( )  [protected]
```

Definition at line 27 of file serverconnection.cpp.

References ServerWorker::onSocketError().

```
00027                           {
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(-1);
00037     //    isAppareNewConnection = server->hasPendingConnections();
00038     if (isAppareNewConnection) {
00039       QTcpSocket* newConnection = server->nextPendingConnection();
00040       if (newConnection == nullptr)
00041         continue;
00042       qInfo() << "appare new connection  :: "
00043               << newConnection->peerAddress().toString() << ":"
00044               << newConnection->peerPort();
00045       ServerWorker* newWorker = new ServerWorker(newConnection);
00046       newWorker->setDefaultReceiver(this->receiver);
00047       connections->append(newWorker);
00048       newWorker->start();
00049     }
00050     msleep(30);
00051   }
00052 }
```

Here is the call graph for this function:



2.26.3.6 setDefaultReceiver()

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

Definition at line 58 of file mailboxelement.h.

References MailSender::receiver.

```
00058                                                   {
00059     this->receiver = receiver;
00060   }
```

2.26.3.7 startServer()

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

Definition at line 21 of file serverconnection.cpp.

Referenced by initServer().

```
00021                                  {
00022   if (this->isRunning())
00023     return;
00024   this->start(Priority::NormalPriority);
00025 }
```

Here is the caller graph for this function:

2.26.4 Member Data Documentation

2.26.4.1 connections

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

Definition at line 32 of file serverconnection.h.

2.26.4.2 instance

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

Definition at line 29 of file serverconnection.h.

Referenced by getServerInstance().

2.26.4.3 receiver

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

Definition at line 63 of file mailboxelement.h.

Referenced by MailSender::setDefaultReceiver().

2.26.4.4 server

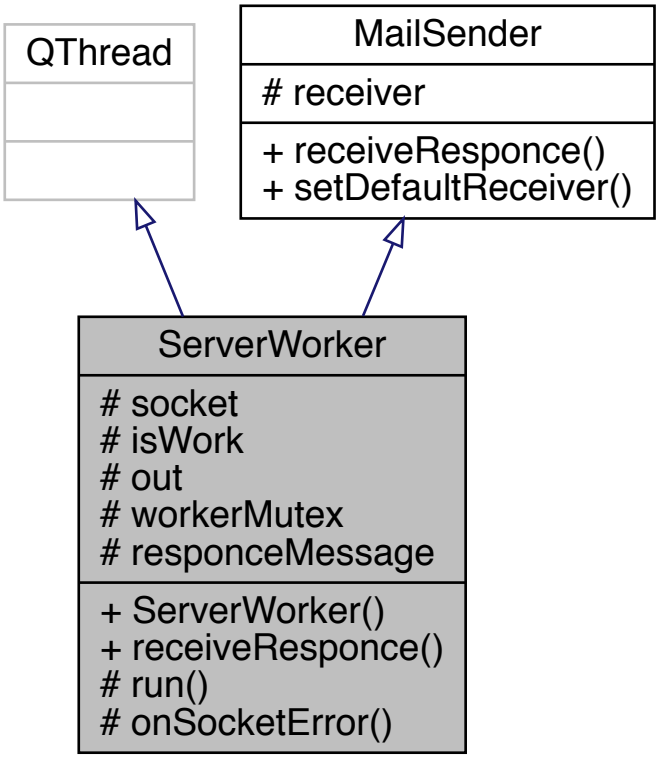
```
QTcpServer* ServerConnection::server  [protected]
```

Definition at line 31 of file serverconnection.h.

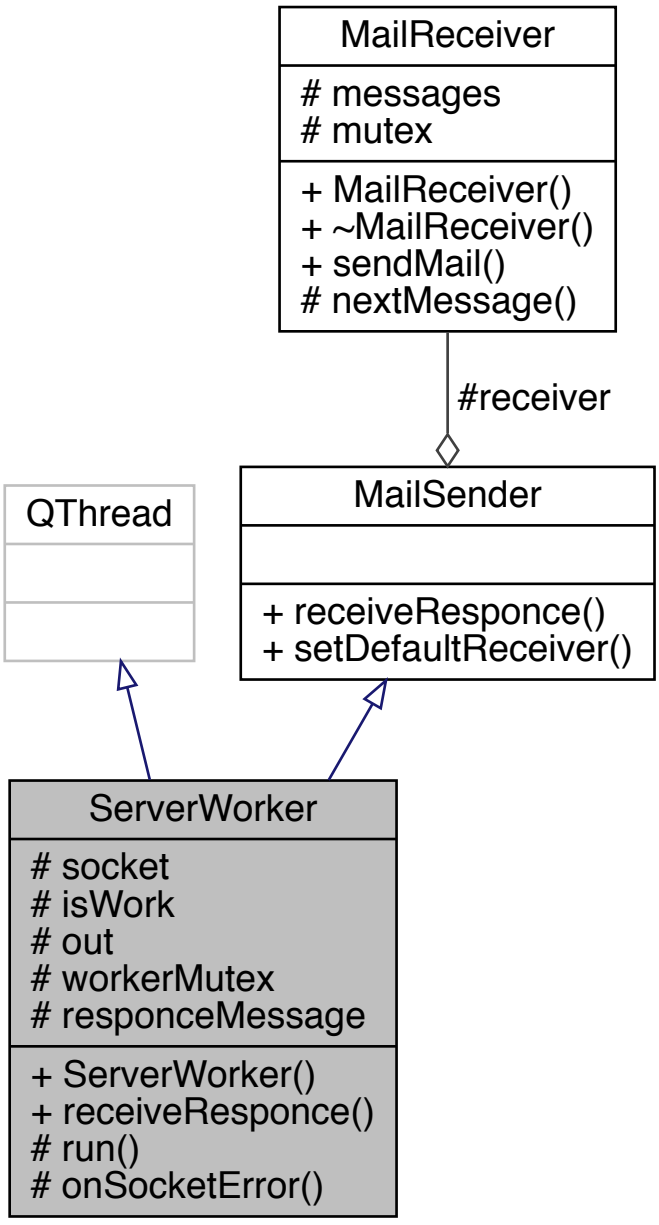
2.27 ServerWorker Class Reference

```
#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 (DiffCard ∗diff, MessageForServer ∗message)
- virtual void setDefaultReceiver (MailReceiver ∗receiver)

Protected Slots

- void onSocketError (QAbstractSocket::SocketError error)

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

2.27.1 Detailed Description

Definition at line 10 of file serverworker.h.

2.27.2 Constructor & Destructor Documentation

2.27.2.1 ServerWorker()

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

Definition at line 6 of file serverworker.cpp.

```
00006                                           {
00007   this->socket = socket;
00008   connect(socket, SIGNAL(error(QAbstractSocket::SocketError)), this,
00009           SLOT(onSocketError(QAbstractSocket::SocketError)));
00010 }
```

2.27.3 Member Function Documentation

2.27.3.1 onSocketError

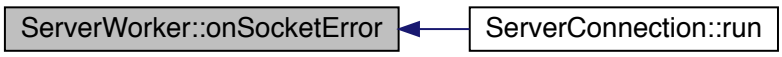
```
void ServerWorker::onSocketError (
            QAbstractSocket::SocketError error )  [protected], [slot]
```

Definition at line 48 of file serverworker.cpp.

Referenced by ServerConnection::run().

```
00048                                                                   {
00049   qCritical() << socket->peerAddress() << ":" << socket->peerPort()
00050               << "error occured :: " << socket->errorString();
00051 }
```

Here is the caller graph for this function:



2.27.3.2 onStop

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

2.27.3.3 receiveResponce()

```
void ServerWorker::receiveResponce (
            DiffCard * diff,
            MessageForServer * message )  [virtual]
```

Implements MailSender.

Definition at line 53 of file serverworker.cpp.

```
00053                                                                                   {
00054   qInfo() << "getting responce for client from map loop";
00055
00056   QtConcurrent::run(QThreadPool::globalInstance(), [=] {
00057     qInfo() << "ServerWorker::receiveResponce - befor mutex";
00058     workerMutex.lock();
00059     qInfo() << "ServerWorker::receiveResponce - in mutex";
00060     responceMessage.push_back(responceData(diff, message));
00061     workerMutex.unlock();
00062     qInfo() << "ServerWorker::receiveResponce - after mutex";
00063   });
00064 }
```

2.27.3.4 run()

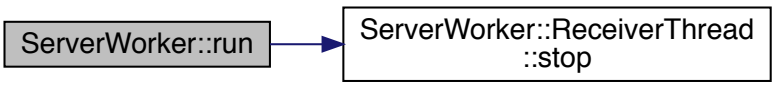
```
void ServerWorker::run ( )  [protected]
```

Definition at line 12 of file serverworker.cpp.

References isWork, and ServerWorker::ReceiverThread::stop().

```
00012                       {
00013   out = new QDataStream(socket);
00014   out->setVersion(QDataStream::Qt_5_7);
00015
00016   ReceiverThread* th = new ReceiverThread(this, socket);
00017
00018   while (this->isWork) {
00019     workerMutex.lock();
00020     while (responceMessage.length() > 0) {
00021       responceData data = responceMessage.takeFirst();
00022       qInfo() << socket->peerAddress() << ":" << socket->peerPort()
00023               << "getting some message for sending";
00024       out->startTransaction();
00025       (*out) << (*data.messag);
00026       (*out) << data.diff->getCountOfDiff();
00027       for (int i = 0; i < data.diff->getCountOfDiff(); i++)
00028         (*out) << (*data.diff->at(i));
00029       out->commitTransaction();
00030       socket->flush();
00031       qInfo() << socket->peerAddress() << ":" << socket->peerPort()
00032               << "send  response to client";
00033       for (int i = 0; i < data.diff->getCountOfDiff(); i++)
00034         delete data.diff->at(i);
00035       data.diff->clear();
00036       delete data.diff;
00037       delete data.messag;
00038     }
00039     msleep(100);
00040     workerMutex.unlock();
00041   }
00042   th->stop();
00043   delete th;
00044   qInfo() << "worker has stoped";
00045   emit onStop(this);
00046 }
```

Here is the call graph for this function:



2.27.3.5 setDefaultReceiver()

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

Definition at line 58 of file mailboxelement.h.

References MailSender::receiver.

```
00058                                                   {
00059     this->receiver = receiver;
00060   }
```

2.27.4 Friends And Related Function Documentation

2.27.4.1 receiveRespnceThread

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

Definition at line 12 of file serverworker.h.

2.27.5 Member Data Documentation

2.27.5.1 isWork

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

Definition at line 39 of file serverworker.h.

Referenced by run().

2.27.5.2 out

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

Definition at line 40 of file serverworker.h.

2.27.5.3 receiver

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

Definition at line 63 of file mailboxelement.h.

Referenced by MailSender::setDefaultReceiver().

2.27.5.4 responceMessage

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

Definition at line 42 of file serverworker.h.

2.27.5.5 socket

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

Definition at line 38 of file serverworker.h.

2.27.5.6 workerMutex

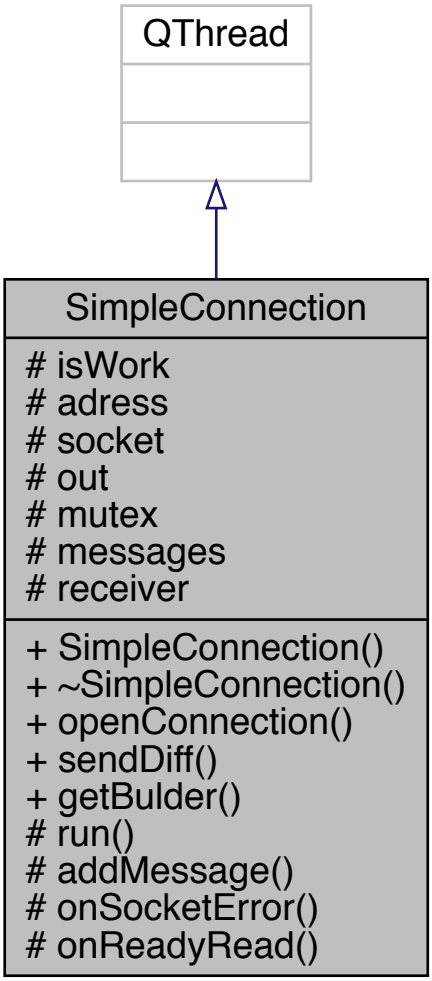
```
QMutex ServerWorker::workerMutex  [protected]
```

Definition at line 41 of file serverworker.h.

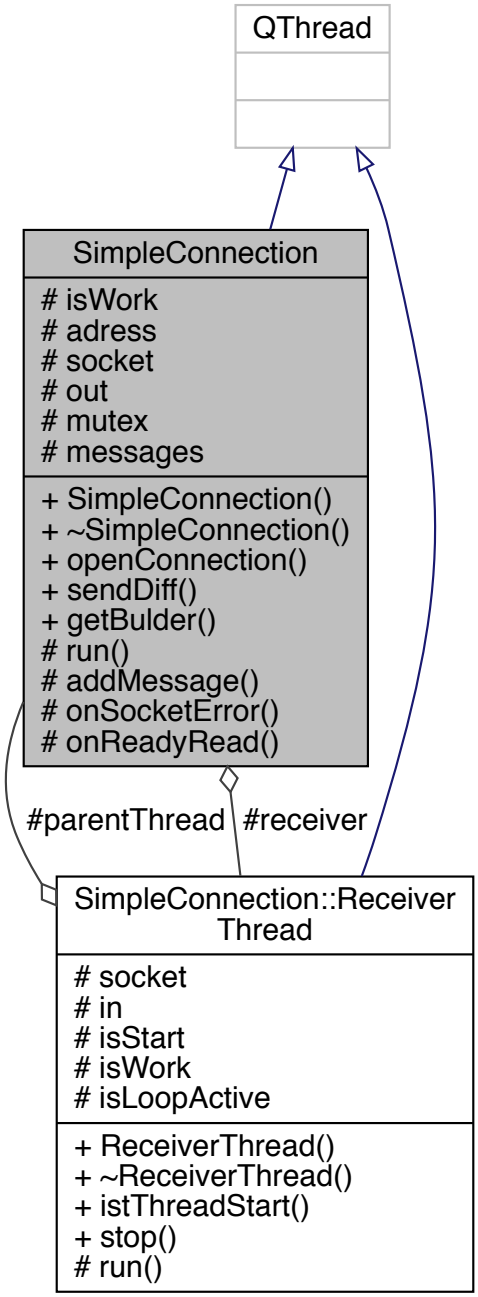
2.28 SimpleConnection Class Reference

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

Inheritance diagram for SimpleConnection:



Collaboration diagram for SimpleConnection:



Classes

- class MessageBuilder
- class ReceiverThread

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)
- void onReadyRead ()

Protected Member Functions

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

Protected Attributes

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

2.28.1 Detailed Description

Definition at line 69 of file simpleconnection.h.

2.28.2 Constructor & Destructor Documentation

2.28.2.1 SimpleConnection()

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

Definition at line 5 of file simpleconnection.cpp.

```
00006     : QThread(parent) {
00007   this->adress = adress;
00008 }
```

2.28.2.2 ~SimpleConnection()

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

Definition at line 10 of file simpleconnection.cpp.

```
00010                                    {
00011   delete socket;
00012   delete out;
00013 }
```

2.28.3 Member Function Documentation

2.28.3.1 addMessage()

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

Definition at line 73 of file simpleconnection.cpp.

```
00073                                                    {
00074   mutex.lock();
00075   this->messages.push_back(message->message);
00076   delete message;
00077   mutex.unlock();
00078 }
```

2.28.3.2 getBulder()

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

Definition at line 30 of file simpleconnection.cpp.

```
00030                                                {
00031   return new MessageBuilder(this);
00032 }
```

2.28.3.3 onDiffReceive

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

2.28.3.4 onReadyRead

```
void SimpleConnection::onReadyRead ( )  [protected], [slot]
```

Definition at line 40 of file simpleconnection.cpp.

```
00040                                   {
00041   qInfo() << "onReadyRead";
00042 }
```

2.28.3.5 onSocketError

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

Definition at line 34 of file simpleconnection.cpp.

References isWork.

```
00034                                                                   {
00035   qCritical() << err;
00036   isWork = false;
00037   // TODO add something
00038 }
```

2.28.3.6 openConnection()

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

Definition at line 15 of file simpleconnection.cpp.

```
00015                                     {
00016   socket = new QTcpSocket();
00017   connect(socket, SIGNAL(error(QAbstractSocket::SocketError)), this,
00018           SLOT(onSocketError(QAbstractSocket::SocketError)),
00019           Qt::DirectConnection);
00020   //  connect(socket, SIGNAL(readyRead()), this, SLOT(onReadyRead()),
00021   //          Qt::DirectConnection);
00022   out = new QDataStream();
00023   out->setDevice(socket);
00024   out->setVersion(QDataStream::Qt_5_7);
00025   receiver = new ReceiverThread(socket, this);
00026   receiver->start();
00027   this->start();
00028 }
```

2.28.3.7 run()

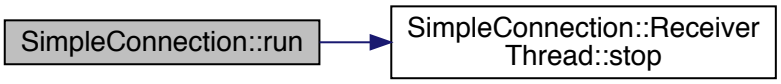
```
void SimpleConnection::run ( )  [protected]
```

Definition at line 44 of file simpleconnection.cpp.

References isWork, receiver, and SimpleConnection::ReceiverThread::stop().

```
00044                           {
00045   qInfo() << "starring connection thread. Opening socket connection.........";
00046   socket->connectToHost(adress, DefaultServerParams::port);
00047   isWork = socket->waitForConnected();
00048   qDebug() << "Connect to server on " << socket->peerAddress().toString() << ":"
00049            << socket->peerPort();
00050   qDebug() << "owen port :: " << socket->localPort();
00051
00052   while (!receiver->istThreadStart())
00053     qInfo() << "whaiting for receiver loop start........";
00054
00055   qInfo() << "starting sending to server message loop";
00056   while (isWork) {
00057     if (messages.length() > 0) {
00058       MessageForServer* newMessage = messages.takeFirst();
00059       out->startTransaction();
00060       (*out) << (*newMessage);
00061       out->commitTransaction();
00062       socket->flush();
00063       qInfo() << "sending some message for server";
00064       continue;
00065     } else {
00066       msleep(100);
00067     }
00068   }
00069   receiver->stop();
00070   delete receiver;
00071 }
```

Here is the call graph for this function:



2.28.3.8 sendDiff()

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

Definition at line 111 of file simpleconnection.cpp.

```
00111                                                    {
00112   emit onDiffReceive(diffs);
00113 }
```

2.28.4 Member Data Documentation

2.28.4.1 adress

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

Definition at line 108 of file simpleconnection.h.

2.28.4.2 isWork

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

Definition at line 106 of file simpleconnection.h.

Referenced by onSocketError(), and run().

2.28.4.3 messages

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

Definition at line 114 of file simpleconnection.h.

2.28.4.4 mutex

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

Definition at line 113 of file simpleconnection.h.

2.28.4.5 out

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

Definition at line 111 of file simpleconnection.h.

2.28.4.6 receiver

```
ReceiverThread* SimpleConnection::receiver  [protected]
```

Definition at line 171 of file simpleconnection.h.

Referenced by run().

2.28.4.7 socket

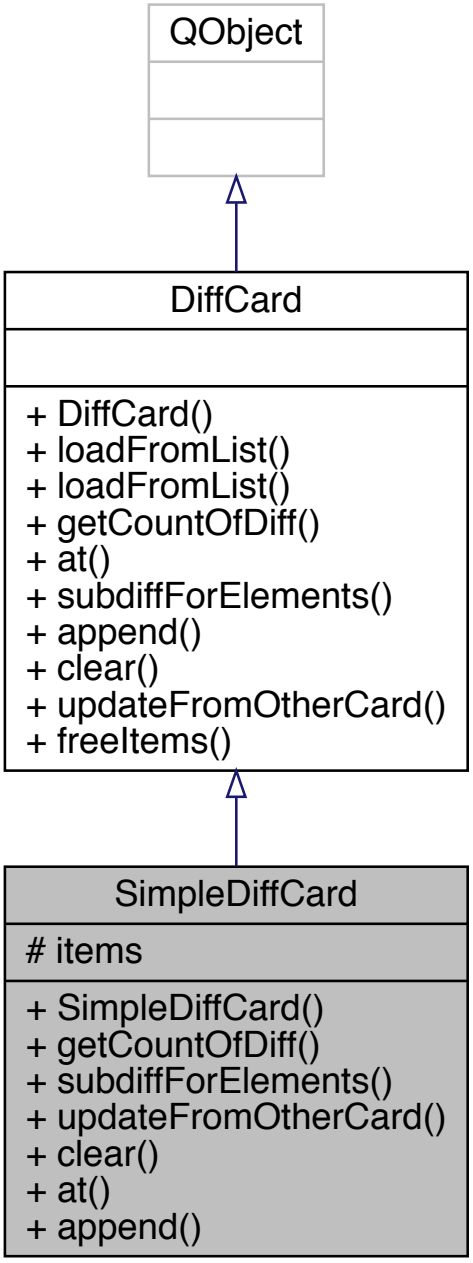
```
QTcpSocket* SimpleConnection::socket  [protected]
```

Definition at line 109 of file simpleconnection.h.

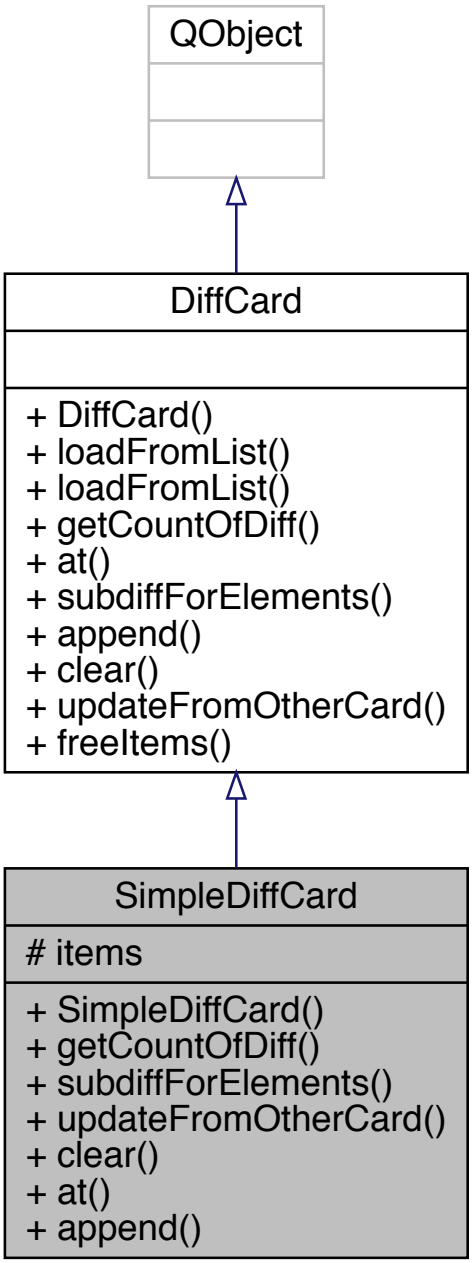
2.29 SimpleDiffCard Class Reference

```
#include "simplediffcard.h"
```

Inheritance diagram for SimpleDiffCard:



Collaboration diagram for SimpleDiffCard:



Public Member Functions

- SimpleDiffCard ()
- virtual int getCountOfDiff ()
- virtual DiffCard * subdiffForElements (QList< IBaseGameElement *> items)
- virtual void updateFromOtherCard (DiffCard *card)
- virtual void clear ()

- virtual DiffElement * at (int i)
- virtual void append (DiffElement *diff)
- virtual void loadFromList (QList< DiffElement *> &newItems)
- virtual void loadFromList (QList< DiffElement *> *newItems)
- virtual void freeItems ()

  freeItems simular to claer but also call destructor for diff elemenst

Protected Attributes

- QList< DiffElement * > items

2.29.1 Detailed Description

Definition at line 8 of file simplediffcard.h.

2.29.2 Constructor & Destructor Documentation

2.29.2.1 SimpleDiffCard()

```
SimpleDiffCard::SimpleDiffCard ( )
```

Definition at line 3 of file simplediffcard.cpp.

```
00003 {}
```

2.29.3 Member Function Documentation

2.29.3.1 append()

```
virtual void SimpleDiffCard::append (
            DiffElement * diff )  [inline], [virtual]
```

Implements DiffCard.

Definition at line 18 of file simplediffcard.h.

```
00018 { items.append(diff); }
```

2.29.3.2 at()

```
virtual DiffElement* SimpleDiffCard::at (
            int i )  [inline], [virtual]
```

Implements DiffCard.

Definition at line 17 of file simplediffcard.h.

```
00017 { return items.at(i); }
```

2.29.3.3 clear()

```
virtual void SimpleDiffCard::clear ( )  [inline], [virtual]
```

Implements DiffCard.

Definition at line 16 of file simplediffcard.h.

```
00016 { items.clear(); }
```

2.29.3.4 freeItems()

```
void DiffCard::freeItems ( )  [virtual], [inherited]
```

freeItems simular to claer but also call destructor for diff elemenst

Definition at line 14 of file diffcard.cpp.

```
00014                           {
00015   for (int i = 0; i < getCountOfDiff(); i++) {
00016     delete at(i);
00017   }
00018   clear();
00019 }
```

2.29.3.5 getCountOfDiff()

```
int SimpleDiffCard::getCountOfDiff ( )  [virtual]
```

Implements DiffCard.

Definition at line 5 of file simplediffcard.cpp.

```
00005                                   {
00006   return items.size();
00007 }
```

2.29.3.6 loadFromList() [1/2]

```
void DiffCard::loadFromList (
            QList< DiffElement *> & newItems )  [virtual], [inherited]
```

Definition at line 5 of file diffcard.cpp.

```
00005                                                   {
00006   foreach (auto i, newItems) { append(i); }
00007 }
```

2.29.3.7 loadFromList() [2/2]

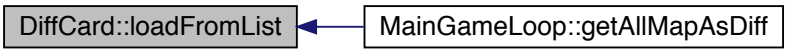
```
void DiffCard::loadFromList (
            QList< DiffElement *> * newItems )  [virtual], [inherited]
```

Definition at line 9 of file diffcard.cpp.

Referenced by MainGameLoop::getAllMapAsDiff().

```
00009                                                   {
00010   for (int i = 0; i < newItems->size(); i++)
00011     append(newItems->at(i));
00012 }
```

Here is the caller graph for this function:



2.29.3.8 subdiffForElements()

```
DiffCard * SimpleDiffCard::subdiffForElements (
            QList< IBaseGameElement *> items )  [virtual]
```

Implements DiffCard.

Definition at line 9 of file simplediffcard.cpp.

```
00009 {}
```

2.29.3.9 updateFromOtherCard()

```
void SimpleDiffCard::updateFromOtherCard (
            DiffCard * card )  [virtual]
```

Implements DiffCard.

Definition at line 13 of file simplediffcard.cpp.

```
00013                                                    {
00014   for (int i = 0; i < card->getCountOfDiff(); i++) {
00015     for (int j = 0; j < items.size(); j++) {
00016       if (card->at(i)->getData()->name == items.at(j)->getData()->name) {
00017         switch (card->at(i)->type) {
00018           case eNew:
00019           case eChange:
00020             items.at(j)->update(card->at(i));
00021             break;
00022           case eDeleted:
00023             delete items.at(j);
00024             items.removeAt(j);
00025             break;
00026           case eEmpty:
00027             break;
00028         }
00029         break;
00030       }
00031       if (j == items.size() - 1) {
00032         items.append(new DiffElement(*card->at(i)));
00033       }
00034     }
00035   }
00036 }
```

2.29.4 Member Data Documentation

2.29.4.1 items

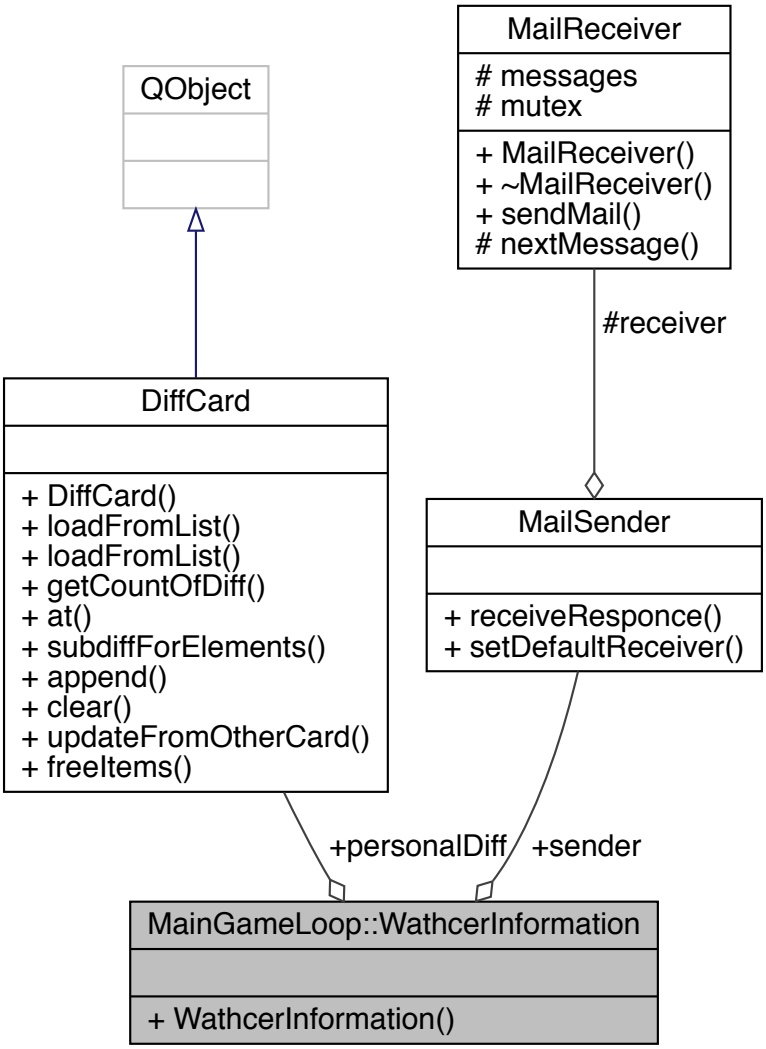
```
QList<DiffElement*> SimpleDiffCard::items  [protected]
```

Definition at line 21 of file simplediffcard.h.

2.30 MainGameLoop::WathcerInformation Class Reference

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

Collaboration diagram for MainGameLoop::WathcerInformation:



Public Member Functions

- WathcerInformation ()

Public Attributes

- MailSender * sender
- DiffCard * personalDiff

2.30.1 Detailed Description

Definition at line 57 of file maingameloop.h.

2.30.2 Constructor & Destructor Documentation

2.30.2.1 WathcerInformation()

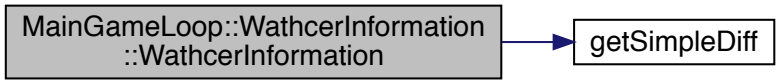
```
MainGameLoop::WathcerInformation::WathcerInformation ( )  [inline]
```

Definition at line 61 of file maingameloop.h.

References getSimpleDiff(), and personalDiff.

```
00061 { personalDiff = getSimpleDiff(); }
```

Here is the call graph for this function:



2.30.3 Member Data Documentation

2.30.3.1 personalDiff

```
DiffCard* MainGameLoop::WathcerInformation::personalDiff
```

Definition at line 60 of file maingameloop.h.

Referenced by WathcerInformation().

2.30.3.2 sender

```
MailSender* MainGameLoop::WathcerInformation::sender
```

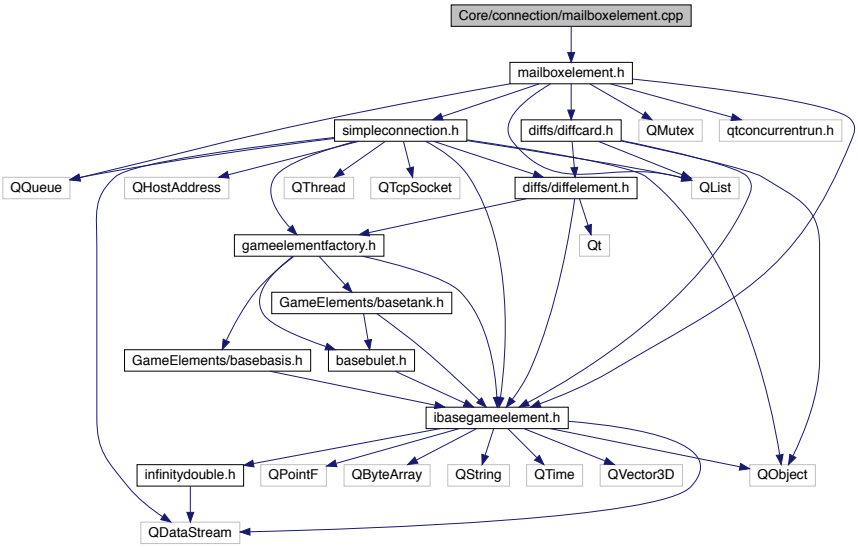
Definition at line 59 of file maingameloop.h.

Chapter 3

File Documentation

3.1 Core/connection/mailboxelement.cpp File Reference

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



3.2 mailboxelement.cpp

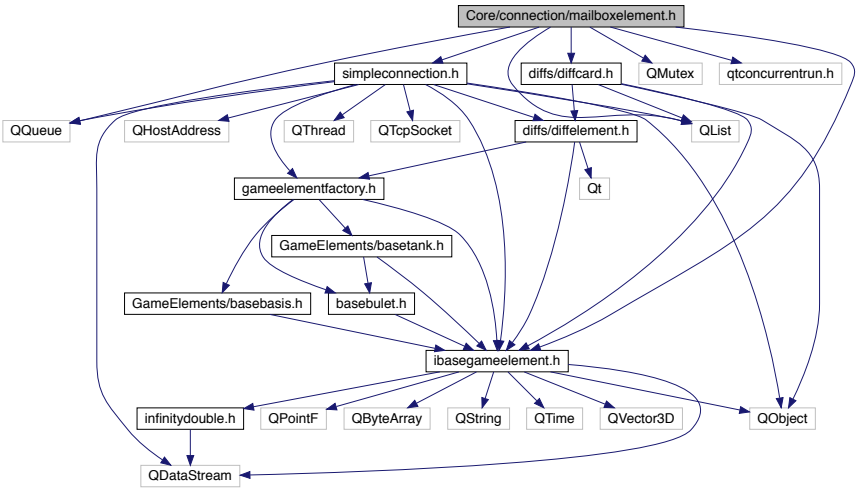
```
00001 #include "mailboxelement.h"
00002
00003 MailReceiver::MailReceiver() {
00004   mutex = new QMutex();
00005 }
00006
00007 MailReceiver::~MailReceiver() {
00008   delete mutex;
00009 }
00010
00011 void MailReceiver::sendMail(MessageForServer* message,
00012                             MailSender* sender,
00013                             int type,
```

```
00014                             bool isReplace) {
00015   run(QThreadPool::globalInstance(), [=] {
00016     qInfo() << "MailReceiver::sendMail - befor mutex";
00017     mutex->lock();
00018     qInfo() << "MailReceiver::sendMail - in mutex";
00019     qInfo() << "MailReceiver :: adding new message item in to queue";
00020     mailMessage* msg;
00021     if (isReplace && messages.length() != 0) {
00022       for (int i = 0; i < messages.length(); i++) {
00023         msg = messages.at(i);
00024         if (msg->sender == sender && msg->type == type) {
00025           messages.removeAt(i);
00026           messages.insert(i, new mailMessage(message, sender, type, isReplace));
00027           break;
00028         }
00029         if (i == messages.length() - 1) {
00030           messages.push_back(new mailMessage(message, sender, type, isReplace));
00031         }
00032       }
00033     } else
00034       messages.push_back(new mailMessage(message, sender, type, isReplace));
00035     mailMessage* test = messages.last();
00036     qInfo() << "getting new message in queue :: " << (*messages.last());
00037     mutex->unlock();
00038     qInfo() << "MailReceiver::sendMail - after mutex";
00039   });
00040 }
00041
00042 MailReceiver::mailMessage* MailReceiver::
     nextMessage() {
00043   mutex->lock();
00044   mailMessage* result = nullptr;
00045   if (messages.length() > 0) {
00046     qInfo() << "enqueue message";
00047     result = messages.takeFirst();
00048   }
00049   mutex->unlock();
00050   return result;
00051 }
00052
00053 MailReceiver::mailMessage::mailMessage(
     MessageForServer* message,
00054                                        MailSender* sender,
00055                                        int type,
00056                                        bool isReplace) {
00057   this->message = message;
00058   this->sender = sender;
00059   this->type = type;
00060   this->isReplace = isReplace;
00061 }
```

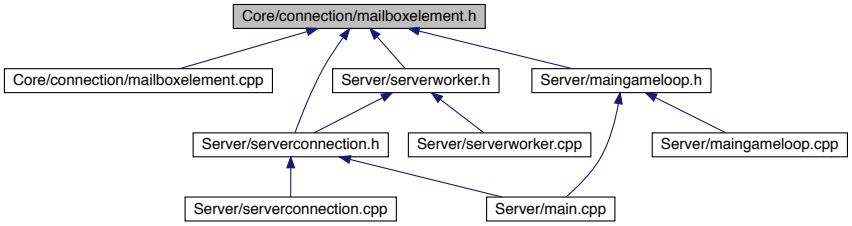
3.3 Core/connection/mailboxelement.h File Reference

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

3.4 mailboxelement.h

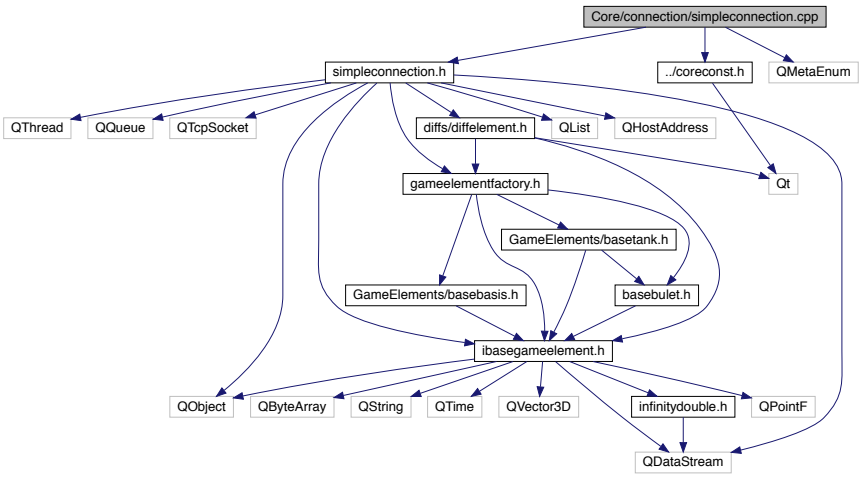
```
00001 #ifndef MAILBOXELEMENT_H
00002 #define MAILBOXELEMENT_H
00003
00004 #include "simpleconnection.h"
00005 #include "../ibasegameelement.h"
00006 #include <QList>
00007 #include <QMutex>
00008 #include <QQueue>
00009 #include <qtconcurrentrun.h>
00010 #include "diffs/diffcard.h"
00011
00012 using namespace QtConcurrent;
00013
00014 class MailReceiver;
00015 class MailSender;
```

```
00016
00017 class MailReceiver {
00018  public:
00019   MailReceiver();
00020   ~MailReceiver();
00021   virtual void sendMail(MessageForServer* message,
00022                         MailSender* sender,
00023                         int type,
00024                         bool isReplace = true);
00025
00026  protected:
00027   class mailMessage {
00028    public:
00029     MessageForServer* message;
00030     MailSender* sender;
00031     int type;
00032     bool isReplace;
00033     mailMessage(MessageForServer* message,
00034                 MailSender* sender,
00035                 int type,
00036                 bool isReplace = true);
00037     friend QDebug operator<<(QDebug debug, const
      mailMessage& c) {
00038       QDebugStateSaver saver(debug);
00039
00040       if (c.message != nullptr) {
00041         debug.nospace() << "mailMessage {message:" << (*c.
      message) << "}";
00042       } else {
00043         debug.nospace() << "mailMessage {message:empty}";
00044       }
00045
00046       return debug;
00047     }
00048   };
00049   QQueue<mailMessage*> messages;
00050   QMutex* mutex;
00051
00052   virtual mailMessage* nextMessage();
00053 };
00054
00055 class MailSender {
00056  public:
00057   virtual void receiveResponce(DiffCard* diff,
      MessageForServer* message) = 0;
00058   virtual void setDefaultReceiver(MailReceiver* receiver) {
00059     this->receiver = receiver;
00060   }
00061
00062  protected:
00063   MailReceiver* receiver;
00064 };
00065
00066 #endif  // MAILBOXELEMENT_H
```

3.5 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)

3.5.1 Function Documentation

3.5.1.1 stringify() [1/2]

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

Definition at line 120 of file simpleconnection.cpp.

```
00120                                       {
00121   switch (e) {
00122     case eGamer:
00123       return "eGamer";
00124     case eWatcher:
00125       return "eWatcher";
00126   }
00127   return "";
00128 }
```

3.5.1.2 stringify() [2/2]

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

Definition at line 130 of file simpleconnection.cpp.

```
00130                                   {
00131   switch (e) {
00132     case eFirstMessae:
00133       return "eFirstMessae";
00134     case eGetUpdateMessage:
00135       return "eGetUpdateMessage";
00136   }
00137   return "";
00138 }
```

3.6 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   socket = new QTcpSocket();
00017   connect(socket, SIGNAL(error(QAbstractSocket::SocketError)), this,
00018           SLOT(onSocketError(QAbstractSocket::SocketError)),
00019           Qt::DirectConnection);
00020   //  connect(socket, SIGNAL(readyRead()), this, SLOT(onReadyRead()),
00021   //          Qt::DirectConnection);
00022   out = new QDataStream();
00023   out->setDevice(socket);
00024   out->setVersion(QDataStream::Qt_5_7);
00025   receiver = new ReceiverThread(socket, this);
00026   receiver->start();
00027   this->start();
00028 }
00029
00030 SimpleConnection::MessageBuilder*
      SimpleConnection::getBulder() {
00031   return new MessageBuilder(this);
00032 }
00033
00034 void SimpleConnection::onSocketError(QAbstractSocket::SocketError err) {
00035   qCritical() << err;
00036   isWork = false;
00037   // TODO add something
00038 }
00039
00040 void SimpleConnection::onReadyRead() {
00041   qInfo() << "onReadyRead";
00042 }
00043
00044 void SimpleConnection::run() {
00045   qInfo() << "starring connection thread. Opening socket connection.........";
00046   socket->connectToHost(adress, DefaultServerParams::port);
00047   isWork = socket->waitForConnected();
00048   qDebug() << "Connect to server on " << socket->peerAddress().toString() << ":"
00049            << socket->peerPort();
00050   qDebug() << "owen port :: " << socket->localPort();
00051
00052   while (!receiver->istThreadStart())
00053     qInfo() << "whaiting for receiver loop start........";
00054
00055   qInfo() << "starting sending to server message loop";
00056   while (isWork) {
00057     if (messages.length() > 0) {
```

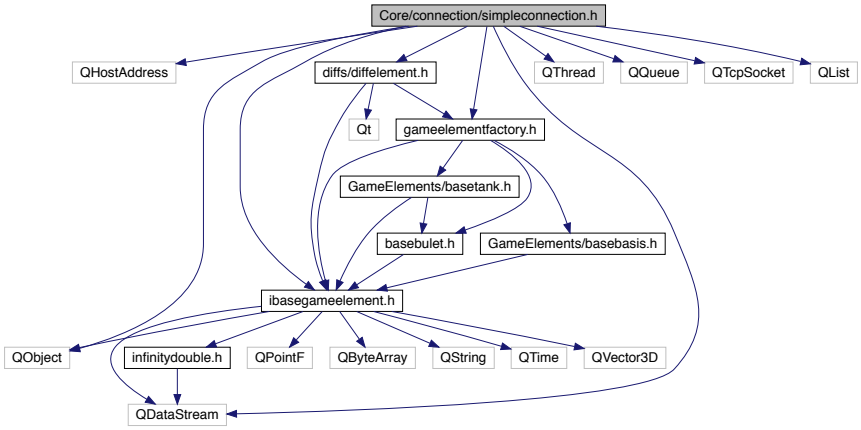
```
00058       MessageForServer* newMessage = messages.takeFirst();
00059       out->startTransaction();
00060       (*out) << (*newMessage);
00061       out->commitTransaction();
00062       socket->flush();
00063       qInfo() << "sending some message for server";
00064       continue;
00065     } else {
00066       msleep(100);
00067     }
00068   }
00069   receiver->stop();
00070   delete receiver;
00071 }
00072
00073 void SimpleConnection::addMessage(
      MessageBuilder* message) {
00074   mutex.lock();
00075   this->messages.push_back(message->message);
00076   delete message;
00077   mutex.unlock();
00078 }
00079
00080 SimpleConnection::MessageBuilder::MessageBuilder(
      SimpleConnection* sender) {
00081   this->parent = sender;
00082   this->message = new MessageForServer();
00083 }
00084
00085 SimpleConnection::MessageBuilder*
00086 SimpleConnection::MessageBuilder::
      asFirstMessage(eConnectionType type) {
00087   this->message->connectionType = type;
00088   this->message->messageType = eFirstMessae;
00089   return this;
00090 }
00091
00092 SimpleConnection::MessageBuilder*
      SimpleConnection::MessageBuilder::addNewItem(
00093     QList<IBaseGameElement*>* newEleements) {
00094   this->message->connectionType = eConnectionType::eGamer;
00095   this->message->messageType = eInsertNewItem;
00096   this->message->items->append(*newEleements);
00097   return this;
00098 }
00099
00100 SimpleConnection::MessageBuilder*
00101 SimpleConnection::MessageBuilder::updateWatcher() {
00102   this->message->connectionType = eConnectionType::eWatcher;
00103   this->message->messageType = eGetUpdateMessage;
00104   return this;
00105 }
00106
00107 void SimpleConnection::MessageBuilder::build() {
00108   parent->addMessage(this);
00109 }
00110
00111 void SimpleConnection::sendDiff(QList<DiffElement*>* diffs) {
00112   emit onDiffReceive(diffs);
00113 }
00114
00115 QString MessageForServer::toString() {
00116   return "MessageForServer :: connectionType = " + stringify(connectionType) +
00117          "; messageType = " + stringify(messageType);
00118 }
00119
00120 QString stringify(eConnectionType e) {
00121   switch (e) {
00122     case eGamer:
00123       return "eGamer";
00124     case eWatcher:
00125       return "eWatcher";
00126   }
00127   return "";
00128 }
00129
00130 QString stringify(eMessageType e) {
00131   switch (e) {
00132     case eFirstMessae:
00133       return "eFirstMessae";
00134     case eGetUpdateMessage:
00135       return "eGetUpdateMessage";
00136   }
00137   return "";
00138 }
00139
00140 bool SimpleConnection::ReceiverThread::istThreadStart() {
```

```
00141   return isStart;
00142   msleep(100);
00143 }
00144
00145 void SimpleConnection::ReceiverThread::stop() {
00146   isWork = false;
00147   while (isLoopActive) {
00148     msleep(100);
00149   }
00150 }
```

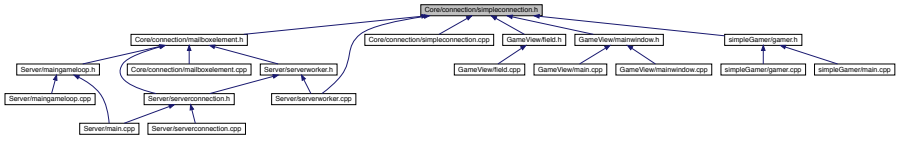
3.7 Core/connection/simpleconnection.h File Reference

```
#include <QHostAddress>
#include <QObject>
#include <QThread>
#include <QQueue>
#include <QTcpSocket>
#include <QDataStream>
#include <ibasegameelement.h>
#include "diffs/diffelement.h"
#include <QList>
#include "gameelementfactory.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::ReceiverThread

Enumerations

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

Functions

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

3.7.1 Enumeration Type Documentation

3.7.1.1 eConnectionType

```
enum eConnectionType
```

Enumerator

eGamer	
eWatcher	

Definition at line 15 of file simpleconnection.h.

```
00015 { eGamer, eWatcher };
```

3.7.1.2 eMessageType

```
enum eMessageType
```

Enumerator

eFirstMessae	
eGetUpdateMessage	
eInsertNewItem	

Definition at line 16 of file simpleconnection.h.

```
00016 { eFirstMessae, eGetUpdateMessage, eInsertNewItem };
```

3.7.2 Function Documentation

3.7.2.1 stringify() [1/2]

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

Definition at line 120 of file simpleconnection.cpp.

```
00120                                       {
00121   switch (e) {
00122     case eGamer:
00123       return "eGamer";
00124     case eWatcher:
00125       return "eWatcher";
00126   }
00127   return "";
00128 }
```

3.7.2.2 stringify() [2/2]

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

Definition at line 130 of file simpleconnection.cpp.

```
00130                                       {
00131   switch (e) {
00132     case eFirstMessae:
00133       return "eFirstMessae";
00134     case eGetUpdateMessage:
00135       return "eGetUpdateMessage";
00136   }
00137   return "";
00138 }
```

3.8 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 #include "diffs/diffelement.h"
00012 #include <QList>
00013 #include "gameelementfactory.h"
00014
00015 enum eConnectionType { eGamer, eWatcher };
00016 enum eMessageType { eFirstMessae,
      eGetUpdateMessage, eInsertNewItem };
00017
00018 QString stringify(eConnectionType e);
00019 QString stringify(eMessageType e);
00020
00021 class MessageForServer : public QObject {
00022   Q_OBJECT
00023
00024  public:
00025   eConnectionType connectionType;
00026   eMessageType messageType;
00027   QList<IBaseGameElement*>* items;
00028
```

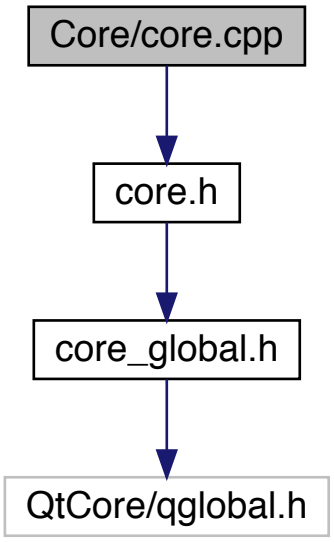
```
00029   MessageForServer() { items = new QList<IBaseGameElement*>(); }
00030
00031   ~MessageForServer() { delete items; }
00032
00033   friend QDataStream& operator<<(QDataStream&
      stream,
00034                                  const MessageForServer&
      myclass) {
00035     int itemsSize = myclass.items->size();
00036     QDataStream& result = stream << ((int)myclass.
      connectionType)
00037                                  << ((int)myclass.messageType) <<
      itemsSize;
00038     for (int i = 0; i < myclass.items->size(); i++)
00039       result << (*myclass.items->at(i));
00040     return result;
00041   }
00042   friend QDataStream& operator>>(QDataStream&
      stream,
00043                                  MessageForServer& myclass) {
00044     int con;
00045     int msg;
00046     int countOfNewItems = 0;
00047     QDataStream& result = (stream >> con >> msg >>
      countOfNewItems);
00048     myclass.connectionType = (eConnectionType)
      con;
00049     myclass.messageType = (eMessageType)msg;
00050     for (int i = 0; i < countOfNewItems; i++) {
00051       GameElementData item;
00052       item.defaultInit();
00053       result >> item;
00054       myclass.items->append(getElement(item));
00055     }
00056     return result;
00057   }
00058
00059   friend QDebug operator<<(QDebug debug,
      MessageForServer& c) {
00060     QDebugStateSaver saver(debug);
00061     debug.nospace() << c.toString();
00062
00063     return debug;
00064   }
00065
00066   QString toString();
00067 };
00068
00069 class SimpleConnection : public QThread {
00070   Q_OBJECT
00071  public:
00072   explicit SimpleConnection(QHostAddress adress,
      QObject* parent = 0);
00073   virtual ~SimpleConnection();
00074   void openConnection();
00075   void sendDiff(QList<DiffElement*>* diffs);
00076
00077   class MessageBuilder {
00078     friend class SimpleConnection;
00079
00080     MessageBuilder(SimpleConnection* sender);
00081
00082    public:
00083     MessageBuilder* asFirstMessage(
      eConnectionType type);
00084     MessageBuilder* addNewItem(QList<
      IBaseGameElement*>* newEleements);
00085     MessageBuilder* updateWatcher();
00086     void build();
00087
00088    private:
00089     MessageForServer* message;
00090     SimpleConnection* parent;
00091   };
00092
00093   MessageBuilder* getBulder();
00094
00095  signals:
00096   void onDiffReceive(QList<DiffElement*>*
      diffs);
00097
00098  public slots:
00099
00100  protected slots:
00101   void onSocketError(QAbstractSocket::
      SocketError error);
00102   void onReadyRead();
```

```
00103
00104   // QThread interface
00105  protected:
00106   volatile bool isWork = true;
00107   void run() Q_DECL_OVERRIDE;
00108   QHostAddress adress;
00109   QTcpSocket* socket;
00110
00111   QDataStream* out;
00112
00113   QMutex mutex;
00114   QQueue<MessageForServer*> messages;
00115
00116   void addMessage(MessageBuilder* messages);
00117
00118   class ReceiverThread : public QThread {
00119    public:
00120     ReceiverThread(QTcpSocket* socket,
00121                    SimpleConnection* parentThread,
00122                    QObject* parent = 0) {
00123       this->socket = socket;
00124       this->parentThread = parentThread;
00125       in = new QDataStream(socket);
00126       in->setVersion(QDataStream::Qt_5_7);
00127     }
00128
00129     virtual ~ReceiverThread() { delete in; }
00130
00131     bool istThreadStart();
00132     void stop();
00133
00134    protected:
00135     QTcpSocket* socket;
00136     QDataStream* in;
00137     SimpleConnection* parentThread;
00138     volatile bool isStart = false;
00139     volatile bool isWork = true;
00140     volatile bool isLoopActive = false;
00141
00142     // QThread interface
00143    protected:
00144     void run() Q_DECL_OVERRIDE {
00145       qInfo() << "starting message receiver loop";
00146
00147       int diffsLenth;
00148       MessageForServer sendedMessage;
00149       while (true) {
00150         isLoopActive = true;
00151         if (!isWork)
00152           break;
00153         isStart = true;
00154         bool waitForReadyReadResult = socket->
    waitForReadyRead(-1);
00155
00156         qInfo() << "starting reading some response";
00157         (*in) >> sendedMessage;
00158         (*in) >> diffsLenth;
00159         QList<DiffElement*>* result = new QList<
    DiffElement*>();
00160         for (int i = 0; i < diffsLenth; i++) {
00161           DiffElement* newItem = new DiffElement();
00162           (*in) >> (*newItem);
00163           result->append(newItem);
00164         }
00165         parentThread->sendDiff(result);
00166         qInfo() << "something read from server";
00167       }
00168       isLoopActive = false;
00169     }
00170   };
00171   ReceiverThread* receiver;
00172 };
00173
00174 #endif  // SIMPLECONNECTION_H
```

3.9 Core/core.cpp File Reference

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

Include dependency graph for core.cpp:



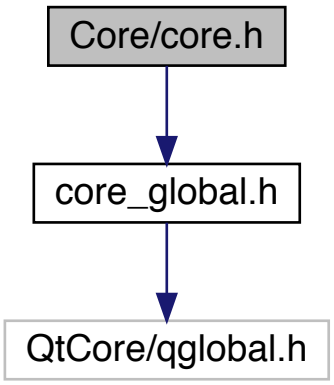
3.10 core.cpp

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

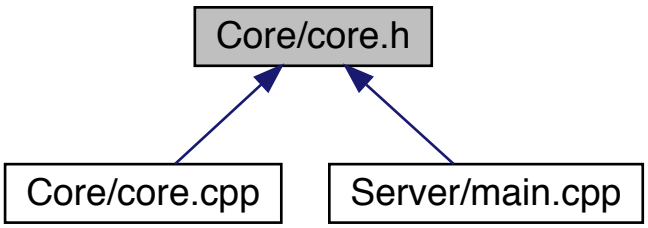
3.11 Core/core.h File Reference

```
#include "core_global.h"
```

Include dependency graph for core.h:



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



Classes

- class Core

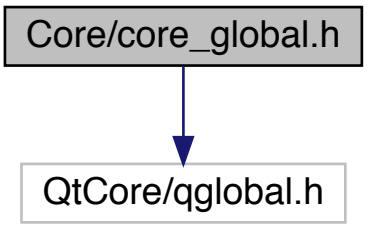
3.12 core.h

```
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
```

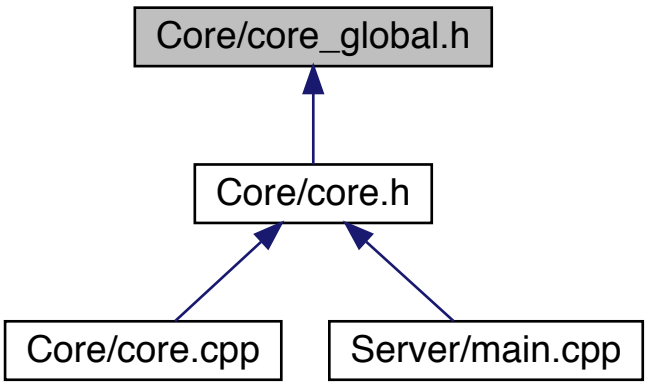
3.13 Core/core_global.h File Reference

```
#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

3.13.1 Macro Definition Documentation

3.13.1.1 CORESHARED_EXPORT

```
#define CORESHARED_EXPORT Q_DECL_IMPORT
```

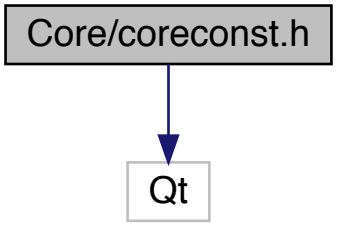
Definition at line 9 of file core_global.h.

3.14 core_global.h

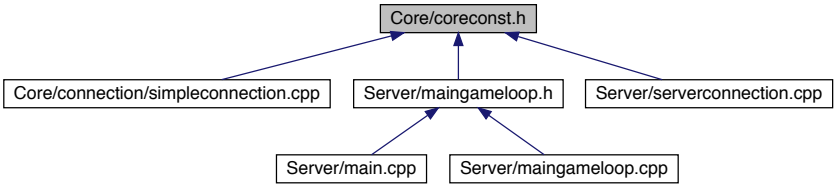
```
00001 #ifndef CORE_GLOBAL_H
00002 #define CORE_GLOBAL_H
00003
00004 #include <QtCore/qglobal.h>
00005
00006 #if defined(CORE_LIBRARY)
00007 #  define CORESHARED_EXPORT Q_DECL_EXPORT
00008 #else
00009 #  define CORESHARED_EXPORT Q_DECL_IMPORT
00010 #endif
00011
00012 #endif // CORE_GLOBAL_H
```

3.15 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

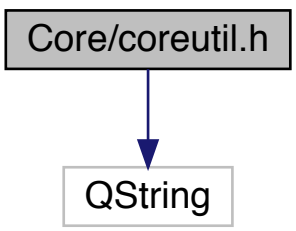
- const int DefaultServerParams::port = 23856
- const uint64_t limits::maxCountOfItems = 18446744073709551615
- const uint64_t limits::defaultBasisEnergy = 1000
- const uint64_t limits::maxRadiusVisionOfBasis = 100

3.16 coreconst.h

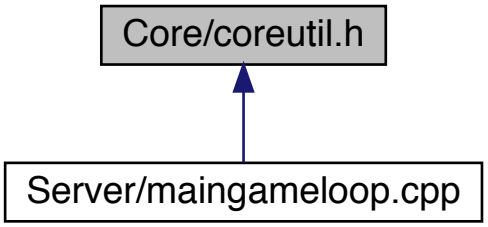
```
00001 #ifndef CONST_H
00002 #define CONST_H
00003
00004 #include <Qt>
00005
00006 namespace DefaultServerParams {
00007 const int port = 23856;
00008 }
00009
00010 namespace limits {
00011 const uint64_t maxCountOfItems = 18446744073709551615;
00012 const uint64_t defaultBasisEnergy = 1000;
00013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

3.17 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)

3.17.1 Function Documentation

3.17.1.1 codingNum()

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

Definition at line 16 of file coreutil.h.

```
00016                                       {
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 }
```

3.17.2 Variable Documentation

3.17.2.1 alphabetSize

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

Definition at line 14 of file coreutil.h.

3.17.2.2 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.

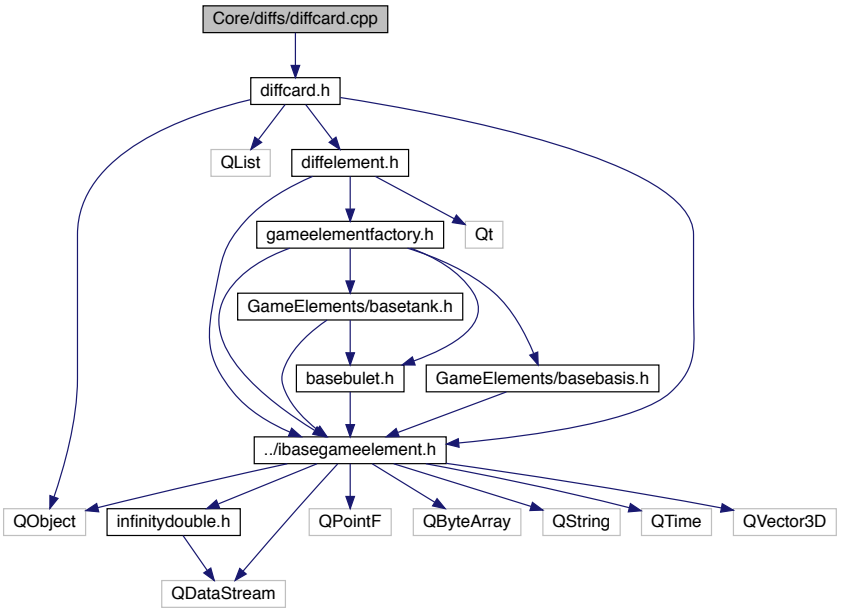
3.18 coreutil.h

```
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
```

3.19 Core/diffs/diffcard.cpp File Reference

```
#include "diffcard.h"
```

Include dependency graph for diffcard.cpp:



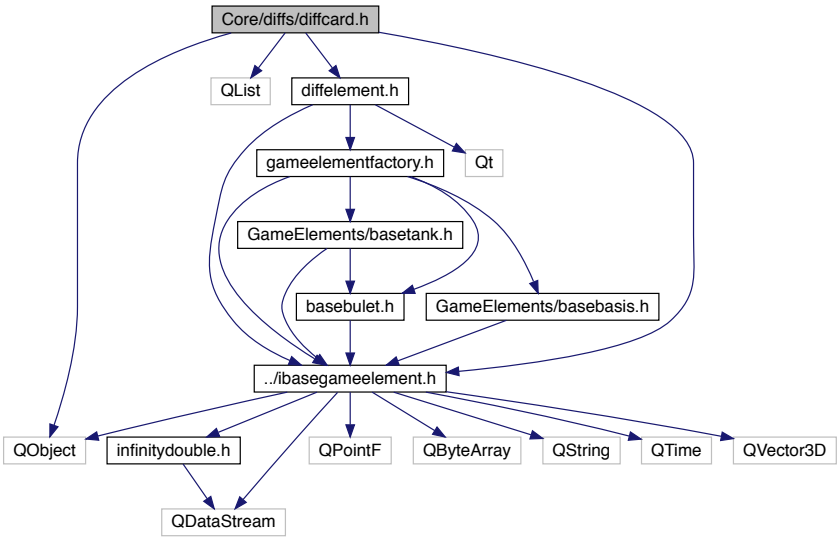
3.20 diffcard.cpp

```
00001 #include "diffcard.h"
00002
00003 DiffCard::DiffCard(QObject* parent) : QObject(parent) {}
00004
00005 void DiffCard::loadFromList(QList<DiffElement*>& newItems) {
00006   foreach (auto i, newItems) { append(i); }
00007 }
00008
00009 void DiffCard::loadFromList(QList<DiffElement*>* newItems) {
00010   for (int i = 0; i < newItems->size(); i++)
00011     append(newItems->at(i));
00012 }
00013
00014 void DiffCard::freeItems() {
00015   for (int i = 0; i < getCountOfDiff(); i++) {
00016     delete at(i);
00017   }
00018   clear();
00019 }
```

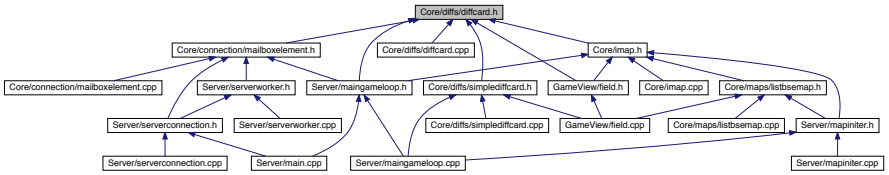
3.21 Core/diffs/diffcard.h File Reference

```
#include <QObject>
#include <QList>
#include "diffelement.h"
```

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



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



Classes

- class DiffCard

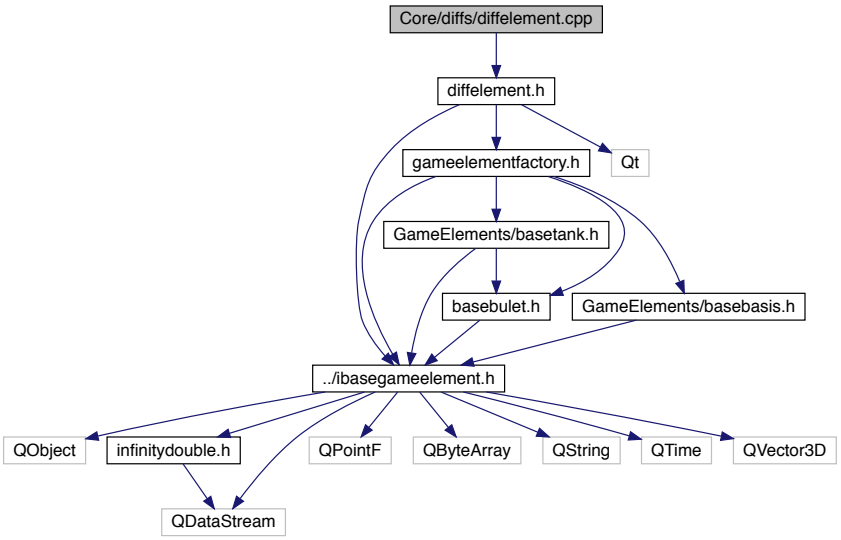
3.22 diffcard.h

```
00001 #ifndef DIFFCARD_H
00002 #define DIFFCARD_H
00003
00004 #include <QObject>
00005 #include <QList>
00006 #include "diffelement.h"
00007 #include "../ibasegameelement.h"
00008
00009 class DiffCard : public QObject {
00010   Q_OBJECT
00011  public:
00012   explicit DiffCard(QObject* parent = 0);
00013
00014   virtual void loadFromList(QList<DiffElement*>&
      newItems);
00015   virtual void loadFromList(QList<DiffElement*>*
      newItems);
00016   virtual int getCountOfDiff() = 0;
```

```
00017   virtual DiffElement* at(int i) = 0;
00018   virtual DiffCard* subdiffForElements(QList<
      IBaseGameElement*> items) = 0;
00019   virtual void append(DiffElement* diff) = 0;
00020   virtual void clear() = 0;
00021   virtual void updateFromOtherCard(DiffCard* card) = 0;
00022
00023   /**
00024    * @brief freeItems simular to claer but also call destructor for diff
00025    * elemenst
00026    */
00027   virtual void freeItems();
00028
00029  signals:
00030
00031  public slots:
00032 };
00033
00034 #endif  // DIFFCARD_H
```

3.23 Core/diffs/diffelement.cpp File Reference

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



3.24 diffelement.cpp

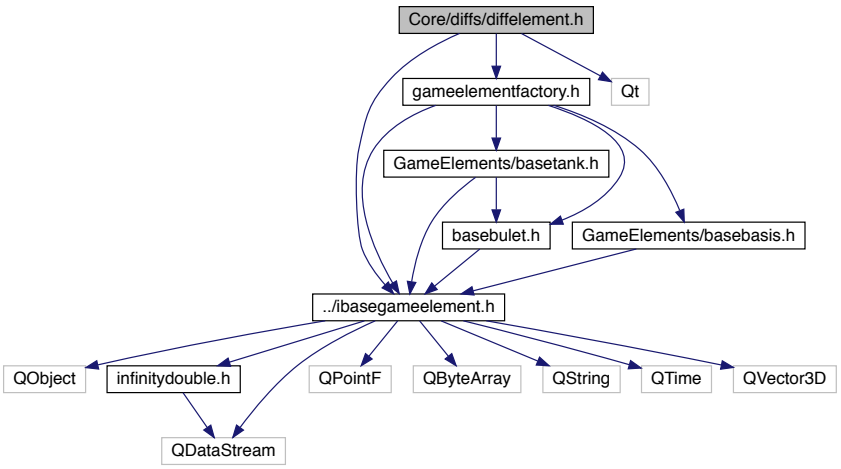
```
00001 #include "diffelement.h"
00002
00003 DiffElement::~DiffElement() {
00004   delete data;
00005 }
00006
00007 DiffElement::DiffElement() {
00008   data = new GameElementData();
00009   type = eEmpty;
00010   time = 0;
00011 }
00012
00013 DiffElement::DiffElement(DiffElement& element) {
00014   data = new GameElementData();
00015   type = eEmpty;
```

```
00016   time = 0;
00017   this->update(&element);
00018 }
00019
00020 DiffElement::DiffElement(eDiffType type,
      IBaseGameElement* data) {
00021   this->type = type;
00022   this->data = new GameElementData();
00023   data->copyData(*this->data);
00024   time = 0;
00025 }
00026
00027 IBaseGameElement* DiffElement::
      generateGameElementInstance() {
00028   return getElement(*data);
00029 }
00030
00031 void DiffElement::update(DiffElement* element) {
00032   this->type = element->type;
00033   this->time = element->time;
00034   this->data->position = new QVector3D(*element->getData()->position);
00035   this->data->helth = new InfinityDouble(*element->getData()->helth);
00036   this->data->weight = new InfinityDouble(*element->getData()->weight);
00037   this->data->transitWeight =
00038       new InfinityDouble(*element->getData()->transitWeight);
00039   this->data->additionalData =
00040       new QByteArray(*element->getData()->additionalData);
00041   this->data->type = element->getData()->type;
00042   this->data->name = element->getData()->name;
00043   this->data->rVision = element->getData()->rVision;
00044 }
```

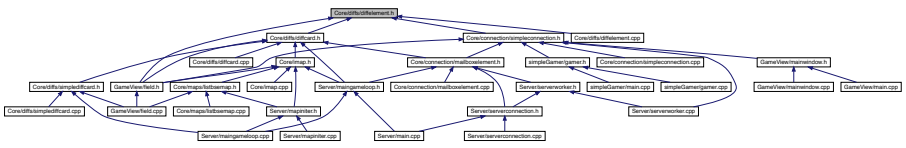
3.25 Core/diffs/diffelement.h File Reference

```
#include "../ibasegameelement.h"
#include "gameelementfactory.h"
#include <Qt>
```

Include dependency graph for diffelement.h:



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



Classes

- class DiffElement

Enumerations

- enum eDiffType { eNew, eChange, eDeleted, eEmpty }

3.25.1 Enumeration Type Documentation

3.25.1.1 eDiffType

```
enum eDiffType
```

Enumerator

eNew	
eChange	
eDeleted	
eEmpty	

Definition at line 8 of file diffelement.h.

```
00008 { eNew, eChange, eDeleted, eEmpty };
```

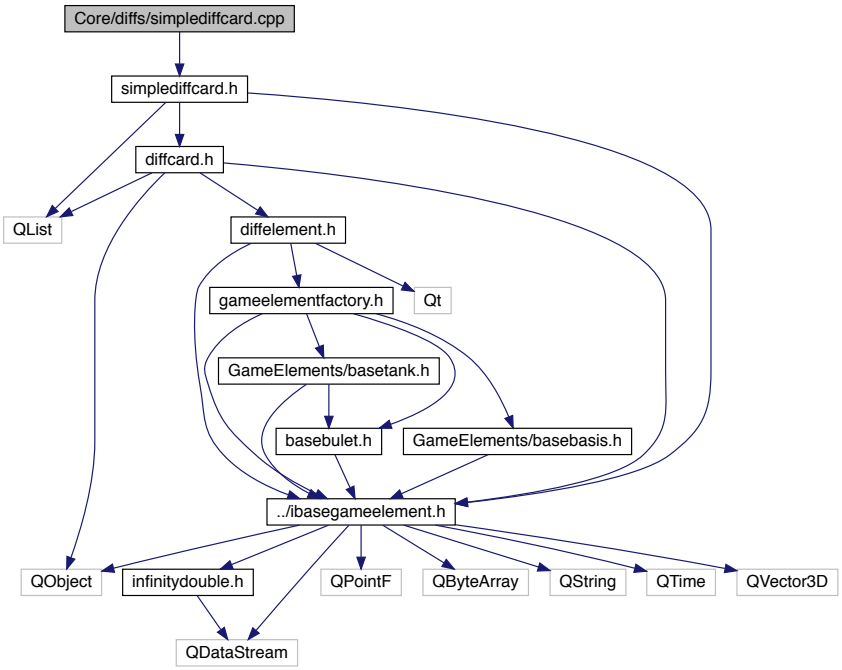
3.26 diffelement.h

```
00001 #ifndef DIFFELEMENT_H
00002 #define DIFFELEMENT_H
00003
00004 #include "../ibasegameelement.h"
00005 #include "gameelementfactory.h"
00006 #include <Qt>
00007
00008 enum eDiffType { eNew, eChange, eDeleted,
      eEmpty };
00009
00010 class DiffElement {
00011  public:
00012   ~DiffElement();
00013   DiffElement();
00014   DiffElement(DiffElement& element);
00015   DiffElement(eDiffType type, IBaseGameElement* data);
00016   friend QDataStream& operator<<(QDataStream&
      stream,
00017                                  const DiffElement& myclass) {
00018     stream << ((int)myclass.type);
00019     return stream << (*myclass.data);
00020   }
00021   friend QDataStream& operator>>(QDataStream&
      stream, DiffElement& myclass) {
00022     int type;
00023     myclass.data->defaultInit();
00024     stream >> type >> (*myclass.data);
00025     myclass.type = (eDiffType)type;
```

```
00026     return stream;
00027   }
00028   eDiffType type;
00029   uint64_t time;
00030
00031   inline GameElementData* getData() { return
      data; }
00032   IBaseGameElement* generateGameElementInstance();
00033
00034   void update(DiffElement* element);
00035
00036  protected:
00037   GameElementData* data;
00038 };
00039
00040 #endif  // DIFFELEMENT_H
```

3.27 Core/diffs/simplediffcard.cpp File Reference

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



3.28 simplediffcard.cpp

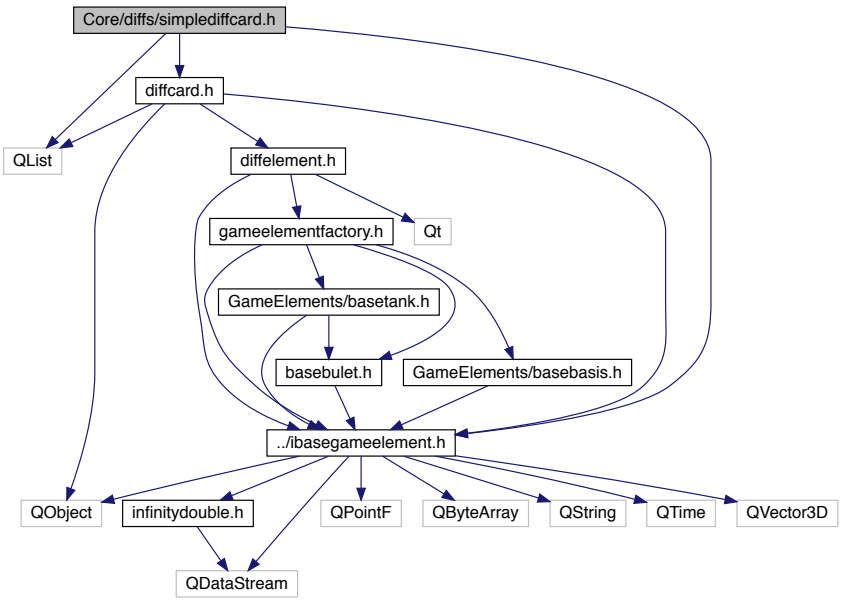
```
00001 #include "simplediffcard.h"
00002
00003 SimpleDiffCard::SimpleDiffCard() {}
00004
00005 int SimpleDiffCard::getCountOfDiff() {
00006   return items.size();
00007 }
00008
00009 DiffCard* SimpleDiffCard::subdiffForElements(QList<IBaseGameElement*> items
      ) {}
00010
00011 // enum eDiffType { eNew, eChange, eDeleted, eEmpty };
```

```
00012
00013 void SimpleDiffCard::updateFromOtherCard(DiffCard* card) {
00014   for (int i = 0; i < card->getCountOfDiff(); i++) {
00015     for (int j = 0; j < items.size(); j++) {
00016       if (card->at(i)->getData()->name == items.at(j)->getData()->name) {
00017         switch (card->at(i)->type) {
00018           case eNew:
00019           case eChange:
00020             items.at(j)->update(card->at(i));
00021             break;
00022           case eDeleted:
00023             delete items.at(j);
00024             items.removeAt(j);
00025             break;
00026           case eEmpty:
00027             break;
00028         }
00029         break;
00030       }
00031       if (j == items.size() - 1) {
00032         items.append(new DiffElement(*card->at(i)));
00033       }
00034     }
00035   }
00036 }
```

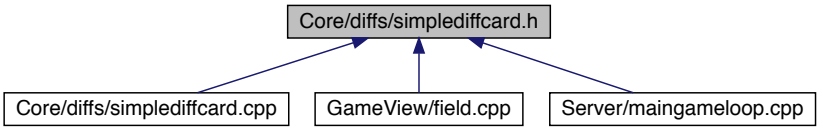
3.29 Core/diffs/simplediffcard.h File Reference

```
#include "diffcard.h"
#include <QList>
#include "../ibasegameelement.h"
```

Include dependency graph for simplediffcard.h:



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



Classes

- class SimpleDiffCard

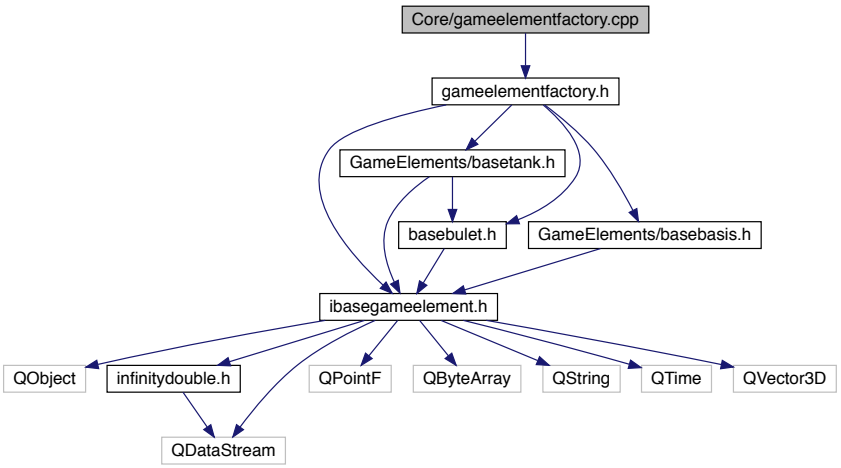
3.30 simplediffcard.h

```
00001 #ifndef SIMPLEDIFFCARD_H
00002 #define SIMPLEDIFFCARD_H
00003
00004 #include "diffcard.h"
00005 #include <QList>
00006 #include "../ibasegameelement.h"
00007
00008 class SimpleDiffCard : public DiffCard {
00009  public:
00010   SimpleDiffCard();
00011
00012  public:
00013   virtual int getCountOfDiff();
00014   virtual DiffCard* subdiffForElements(QList<
      IBaseGameElement*> items);
00015   virtual void updateFromOtherCard(DiffCard* card);
00016   virtual void clear() { items.clear(); }
00017   virtual DiffElement* at(int i) { return items.at(i); }
00018   virtual void append(DiffElement* diff) { items.append(diff); }
00019
00020  protected:
00021   QList<DiffElement*> items;
00022 };
00023
00024 #endif  // SIMPLEDIFFCARD_H
```

3.31 Core/gameelementfactory.cpp File Reference

```
#include "gameelementfactory.h"
```

Include dependency graph for gameelementfactory.cpp:



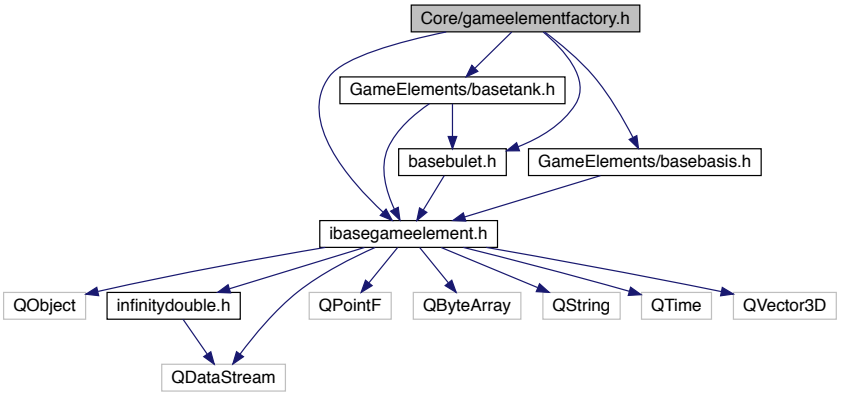
3.32 gameelementfactory.cpp

```
00001 #include "gameelementfactory.h"
```

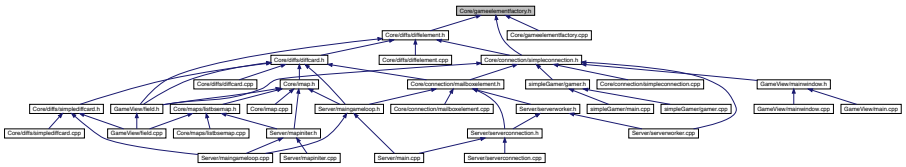
3.33 Core/gameelementfactory.h File Reference

```
#include "ibasegameelement.h"
#include "GameElements/basetank.h"
#include "GameElements/basebulet.h"
#include "GameElements/basebasis.h"
```

Include dependency graph for gameelementfactory.h:



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



Functions

- IBaseGameElement * getElement (GameElementData &data)

3.33.1 Function Documentation

3.33.1.1 getElement()

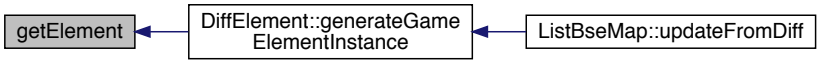
```
IBaseGameElement* getElement (
            GameElementData & data )  [inline]
```

Definition at line 9 of file gameelementfactory.h.

Referenced by DiffElement::generateGameElementInstance().

```
00009                                                                {
00010   switch (((eBaseGameElementType)data.type)) {
00011     case eGrass:
00012       return new IBaseGameElement(data);
00013     case eSimpleTank:
00014       return new BaseTank(data);
00015     case eBasis:
00016       return new BaseBasis(data);
00017     case eBullet:
00018       return new BaseBulet(data);
00019     default:
00020       return new IBaseGameElement(data);
00021   }
00022 }
```

Here is the caller graph for this function:



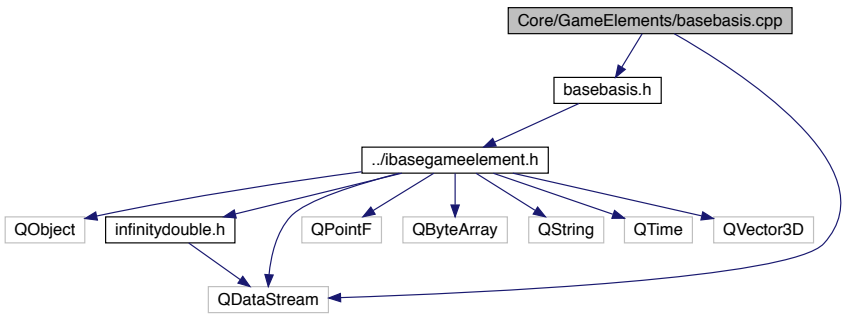
3.34 gameelementfactory.h

```
00001 #ifndef GAMEELEMENTFACTORY_H
00002 #define GAMEELEMENTFACTORY_H
00003
00004 #include "ibasegameelement.h"
00005 #include "GameElements/basetank.h"
00006 #include "GameElements/basebulet.h"
00007 #include "GameElements/basebasis.h"
00008
00009 inline IBaseGameElement* getElement(
      GameElementData& data) {
00010   switch (((eBaseGameElementType)data.type)) {
00011     case eGrass:
00012       return new IBaseGameElement(data);
00013     case eSimpleTank:
00014       return new BaseTank(data);
00015     case eBasis:
00016       return new BaseBasis(data);
00017     case eBullet:
00018       return new BaseBulet(data);
00019     default:
00020       return new IBaseGameElement(data);
00021   }
00022 }
00023
00024 #endif  // GAMEELEMENTFACTORY_H
```

3.35 Core/GameElements/basebasis.cpp File Reference

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

Include dependency graph for basebasis.cpp:



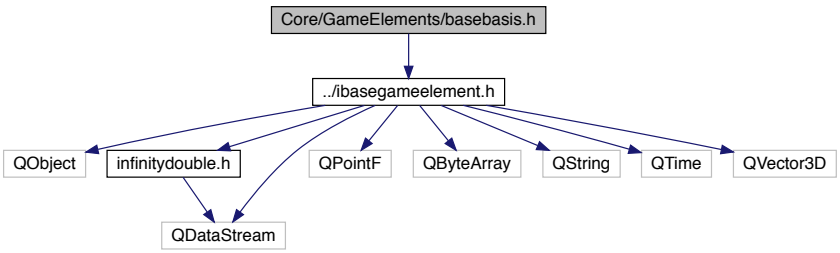
3.36 basebasis.cpp

```
00001 #include "basebasis.h"
00002 #include <QDataStream>
00003
00004 BaseBasis::BaseBasis() : IBaseGameElement() {
00005   this->type = eBasis;
00006   helth = InfinityDouble::InfinityValue();
00007   weight = InfinityDouble::InfinityValue();
00008   transitWeight = InfinityDouble::InfinityValue();
00009 }
00010
00011 BaseBasis::BaseBasis(GameElementData& data) :
      IBaseGameElement() {
00012   init(data);
00013   this->type = eBasis;
00014   helth = InfinityDouble::InfinityValue();
```

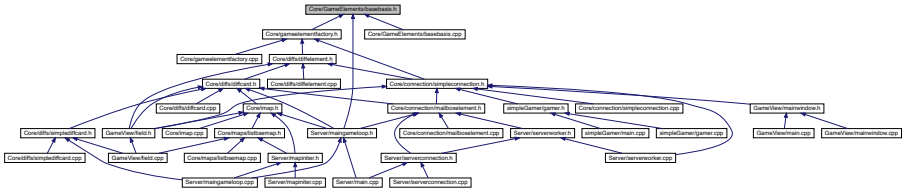
```
00015   weight = InfinityDouble::InfinityValue();
00016   transitWeight = InfinityDouble::InfinityValue();
00017 }
00018
00019 void BaseBasis::setType(int value) {
00020   if (value != ((int)eBasis)) {
00021     Q_ASSERT_X(false, "game logic", "Wrong type for basis");
00022   }
00023   IBaseGameElement::setType(value);
00024 }
00025
00026 QByteArray* BaseBasis::getAdditionalData() const {
00027   additionalData->clear();
00028   QDataStream stream(additionalData, QIODevice::WriteOnly);
00029   stream << energy;
00030 }
00031
00032 void BaseBasis::setAdditionakData(QByteArray* data) {
00033   QDataStream stream(*data);
00034   stream >> energy;
00035   IBaseGameElement::setAdditionakData(data);
00036 }
```

3.37 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

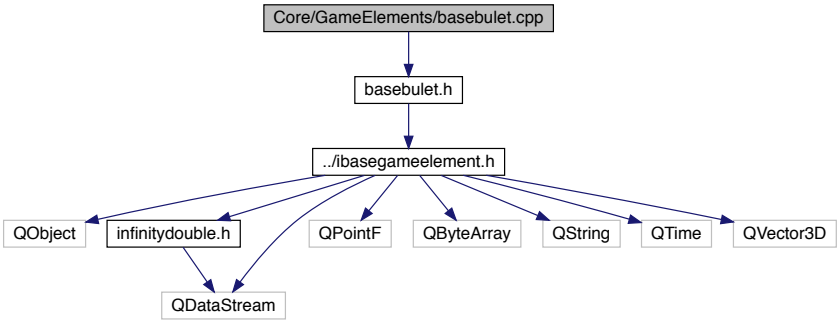
- class BaseBasis

3.38 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   BaseBasis(GameElementData& data);
00010   using IBaseGameElement::getType;
00011   virtual void setType(int value) override;
00012   virtual QByteArray* getAdditionalData() const;
00013   using IBaseGameElement::setAdditionakData;
00014   virtual void setAdditionakData(QByteArray*
      data) override;
00015
00016   virtual int getEnergy() const { return energy; }
00017   virtual void setEnergy(int _enery) { this->energy = _enery; }
00018
00019  protected:
00020   int energy = 100;
00021 };
00022
00023 #endif  // BASEBASIS_H
```

3.39 Core/GameElements/basebulet.cpp File Reference

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



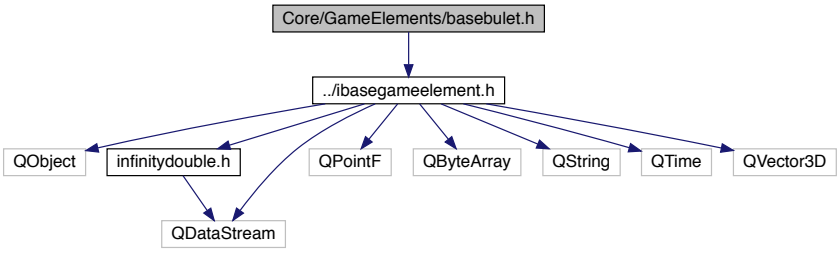
3.40 basebulet.cpp

```
00001 #include "basebulet.h"
00002
00003 BaseBulet::BaseBulet() : IBaseGameElement() {
00004   this->type = eBullet;
00005 }
00006
00007 BaseBulet::BaseBulet(GameElementData& data) :
      IBaseGameElement() {
00008   init(data);
00009   this->type = eBullet;
00010 }
00011
00012 QByteArray* BaseBulet::getAdditionalData() const {}
00013
00014 void BaseBulet::setAdditionakData(QByteArray* data) {}
```

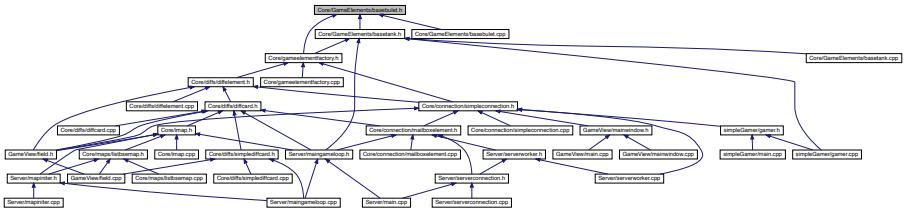
3.41 Core/GameElements/basebulet.h File Reference

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

Include dependency graph for basebulet.h:



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



Classes

- class BaseBulet

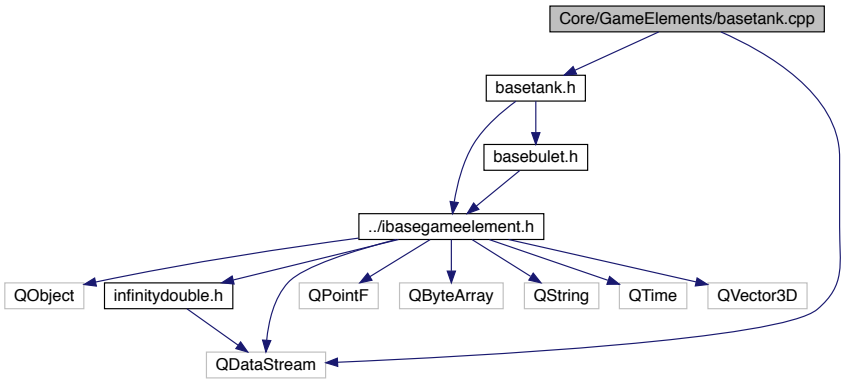
3.42 basebulet.h

```
00001 #ifndef BASEBULET_H
00002 #define BASEBULET_H
00003
00004 #include "../ibasegameelement.h"
00005
00006 class BaseBulet : public IBaseGameElement {
00007  public:
00008   BaseBulet();
00009   BaseBulet(GameElementData& data);
00010
00011  protected:
00012   int lifetime = 0;
00013   double demage = 0;
00014   double direction;
00015   double speed;
00016
00017   // IBaseGameElement interface
00018  public:
00019   QByteArray* getAdditionalData() const;
00020   using IBaseGameElement::setAdditionakData;
00021   void setAdditionakData(QByteArray* data)
      override;
00022 };
00023
00024 #endif  // BASEBULET_H
```

3.43 Core/GameElements/basetank.cpp File Reference

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

Include dependency graph for basetank.cpp:



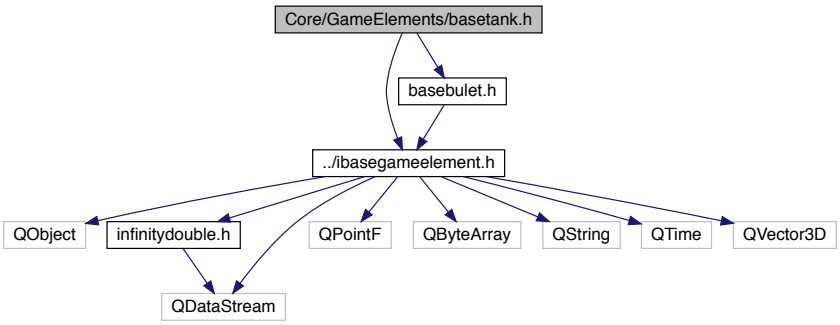
3.44 basetank.cpp

```
00001 #include "basetank.h"
00002 #include <QDataStream>
00003
00004 BaseTank::BaseTank() : IBaseGameElement() {
00005   this->type = eSimpleTank;
00006 }
00007
00008 BaseTank::BaseTank(GameElementData& data) :
      IBaseGameElement() {
00009   init(data);
00010   this->type = eSimpleTank;
00011 }
00012
00013 void BaseTank::setType(int value) {
00014   if (value != ((int)eSimpleTank)) {
00015     Q_ASSERT_X(false, "game logic", "Wrong type for tank");
00016   }
00017   IBaseGameElement::setType(value);
00018 }
00019
00020 BaseBulet BaseTank::generateBullet() const {}
00021
00022 void BaseTank::setAdditionakData(QByteArray* data) {
00023   IBaseGameElement::setAdditionakData(data);
00024   QDataStream stream(*data);
00025   stream >> direction >> speed >> fireType;
00026 }
00027
00028 QByteArray* BaseTank::getAdditionalData() const {
00029   this->additionalData->clear();
00030   QDataStream stream(this->additionalData, QIODevice::WriteOnly);
00031   stream << direction << speed << fireType;
00032   return additionalData;
00033 }
00034
00035 void BaseTank::setDirection(double _direction) {
00036   direction = _direction;
00037   getAdditionalData();
00038 }
00039
00040 void BaseTank::setSpeed(double _speed) {
00041   speed = _speed;
00042   getAdditionalData();
00043 }
```

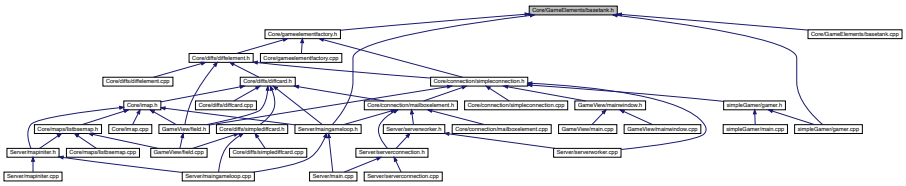
3.45 Core/GameElements/basetank.h File Reference

```
#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

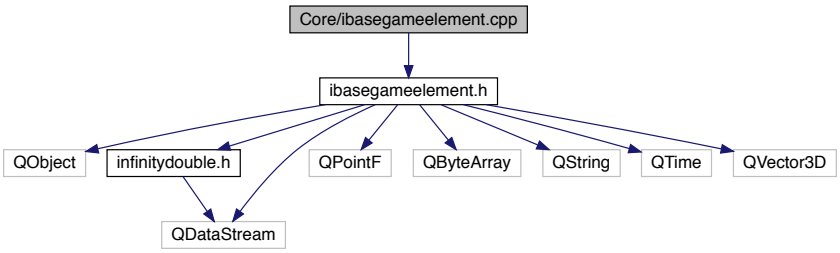
3.46 basetank.h

```
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   BaseTank(GameElementData& data);
00010
00011  public:
00012   using IBaseGameElement::setType;
00013   virtual void setType(int value) override;
00014   virtual BaseBulet generateBullet() const;
00015
00016   using IBaseGameElement::setAdditionakData;
00017   virtual void setAdditionakData(QByteArray*
      data) override;
00018   virtual QByteArray* getAdditionalData() const;
00019
00020   virtual void setDirection(double _direction);
00021   virtual double getDirection() const { return direction; }
00022
```

```
00023   virtual void setSpeed(double _speed);
00024   virtual double getSpeed() const { return speed; }
00025
00026  protected:
00027   double direction = 0;
00028   double speed = 0;
00029   /**
00030    * @brief fireType
00031    * 0  for non fire
00032    * -1 for single fire
00033    * if fireType > 0 then fire will be call evry fireType tik of game
00034    */
00035   int fireType = 0;
00036 };
00037
00038 #endif  // BASETANK_H
```

3.47 Core/ibasegameelement.cpp File Reference

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



Functions

- QDataStream & operator>> (QDataStream &stream, GameElementData &myclass)
- QDataStream & operator<< (QDataStream &stream, GameElementData &myclass)

3.47.1 Function Documentation

3.47.1.1 operator<<()

```
QDataStream& operator<< (
            QDataStream & stream,
            GameElementData & myclass )
```

Definition at line 15 of file ibasegameelement.cpp.

```
00015                                                                   {
00016   stream << (*myclass.position);
00017   stream << (*myclass.helth);
00018   stream << (*myclass.weight);
00019   stream << (*myclass.transitWeight);
00020   stream << (*myclass.additionalData);
00021   stream << myclass.type;
00022   stream << myclass.name;
00023   stream << myclass.rVision;
00024   return stream;
00025 }
```

3.47.1.2 operator>>()

```
QDataStream& operator>> (
             QDataStream & stream,
             GameElementData & myclass )
```

Definition at line 3 of file ibasegameelement.cpp.

```
00003                                                                    {
00004   stream >> (*myclass.position);
00005   stream >> (*myclass.helth);
00006   stream >> (*myclass.weight);
00007   stream >> (*myclass.transitWeight);
00008   stream >> (*myclass.additionalData);
00009   stream >> myclass.type;
00010   stream >> myclass.name;
00011   stream >> myclass.rVision;
00012   return stream;
00013 }
```

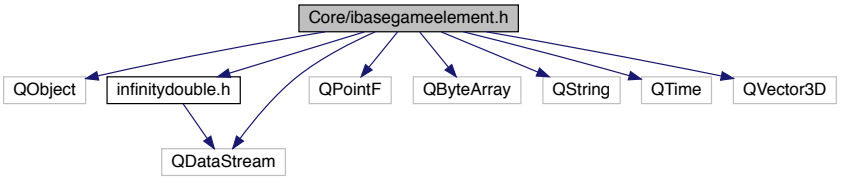
3.48 ibasegameelement.cpp

```
00001 #include "ibasegameelement.h"
00002
00003 QDataStream& operator>>(QDataStream&
      stream, GameElementData& myclass) {
00004   stream >> (*myclass.position);
00005   stream >> (*myclass.helth);
00006   stream >> (*myclass.weight);
00007   stream >> (*myclass.transitWeight);
00008   stream >> (*myclass.additionalData);
00009   stream >> myclass.type;
00010   stream >> myclass.name;
00011   stream >> myclass.rVision;
00012   return stream;
00013 }
00014
00015 QDataStream& operator<<(QDataStream&
      stream, GameElementData& myclass) {
00016   stream << (*myclass.position);
00017   stream << (*myclass.helth);
00018   stream << (*myclass.weight);
00019   stream << (*myclass.transitWeight);
00020   stream << (*myclass.additionalData);
00021   stream << myclass.type;
00022   stream << myclass.name;
00023   stream << myclass.rVision;
00024   return stream;
00025 }
00026
00027 void IBaseGameElement::copyData(
      GameElementData& out) {
00028   out.position = new QVector3D(*position);
00029   out.helth = new InfinityDouble(*helth);
00030   out.weight = new InfinityDouble(*weight);
00031   out.transitWeight = new InfinityDouble(*transitWeight);
00032   if (additionalData == nullptr)
00033     out.additionalData = nullptr;
00034   else
00035     out.additionalData = new QByteArray(*additionalData);
00036   out.type = type;
00037   out.name = name;
00038   out.rVision = rVision;
00039 }
00040
00041 void IBaseGameElement::init(GameElementData& data) {
00042   setHelth(new InfinityDouble(*data.helth));
00043   setWeight(new InfinityDouble(*data.weight));
00044   setTransitWeight(new InfinityDouble(*data.transitWeight));
00045   setPosition(new QVector3D(*data.position));
00046   setName(data.name);
00047   setAdditionakData(new QByteArray(*data.additionalData));
00048   setRVision(data.rVision);
00049   setType(data.type);
00050 }
```

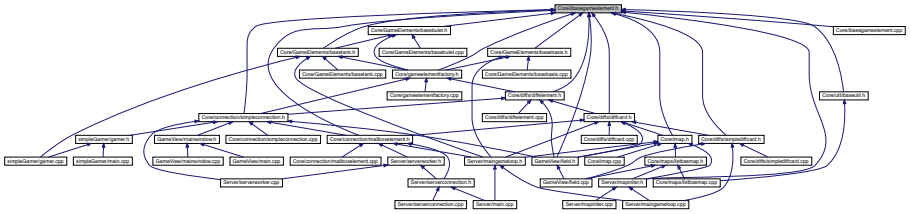
3.49 Core/ibasegameelement.h File Reference

```
#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

- struct GameElementData
- class IBaseGameElement

Enumerations

- enum eBaseGameElementType { eGrass, eSimpleTank, eBasis, eBullet }

Functions

- QDataStream & operator>> (QDataStream &stream, GameElementData &myclass)
- QDataStream & operator<< (QDataStream &stream, GameElementData &myclass)

3.49.1 Enumeration Type Documentation

3.49.1.1 eBaseGameElementType

```
enum eBaseGameElementType
```

Enumerator

eGrass	
eSimpleTank	
eBasis	
eBullet	

Definition at line 13 of file ibasegameelement.h.

```
00013 { eGrass, eSimpleTank, eBasis, eBullet };
```

3.49.2 Function Documentation

3.49.2.1 operator<<()

```
QDataStream& operator<< (
             QDataStream & stream,
             GameElementData & myclass )
```

Definition at line 15 of file ibasegameelement.cpp.

```
00015                                                                {
00016   stream << (*myclass.position);
00017   stream << (*myclass.helth);
00018   stream << (*myclass.weight);
00019   stream << (*myclass.transitWeight);
00020   stream << (*myclass.additionalData);
00021   stream << myclass.type;
00022   stream << myclass.name;
00023   stream << myclass.rVision;
00024   return stream;
00025 }
```

3.49.2.2 operator>>()

```
QDataStream& operator>> (
             QDataStream & stream,
             GameElementData & myclass )
```

Definition at line 3 of file ibasegameelement.cpp.

```
00003                                                                {
00004   stream >> (*myclass.position);
00005   stream >> (*myclass.helth);
00006   stream >> (*myclass.weight);
00007   stream >> (*myclass.transitWeight);
00008   stream >> (*myclass.additionalData);
00009   stream >> myclass.type;
00010   stream >> myclass.name;
00011   stream >> myclass.rVision;
00012   return stream;
00013 }
```

3.50 ibasegameelement.h

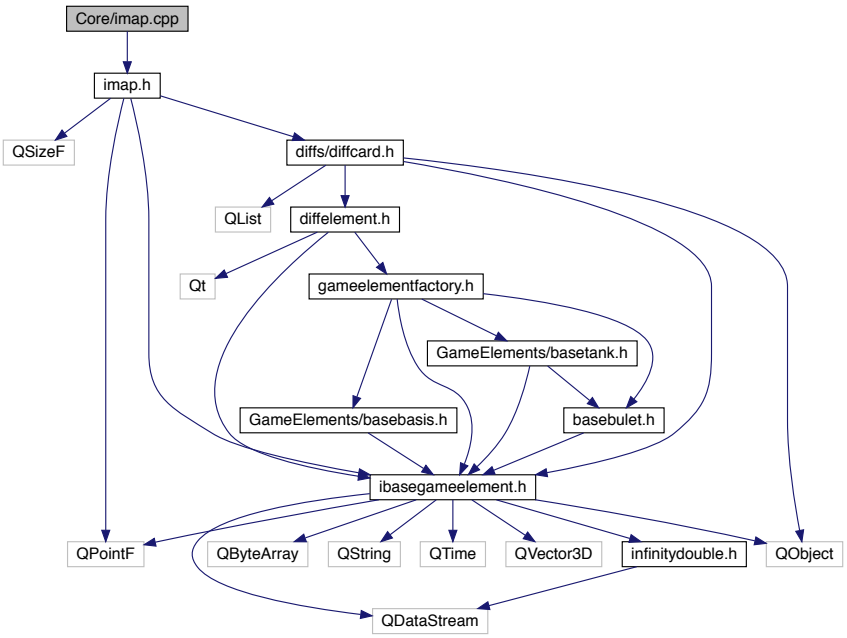
```
00001 #ifndef IBASEGAMEELEMENT_H
00002 #define IBASEGAMEELEMENT_H
00003
00004 #include <QObject>
00005 #include "infinitydouble.h"
00006 #include <QPointF>
00007 #include <QByteArray>
00008 #include <QDataStream>
00009 #include <QString>
00010 #include <QTime>
00011 #include <QVector3D>
00012
00013 enum eBaseGameElementType { eGrass,
      eSimpleTank, eBasis, eBullet };
00014 class IBaseGameElement;
00015
00016 struct GameElementData {
00017   QVector3D* position = nullptr;
00018   InfinityDouble* helth = nullptr;
00019   InfinityDouble* weight = nullptr;
00020   InfinityDouble* transitWeight = nullptr;
00021   QByteArray* additionalData = nullptr;
00022   qint32 rVision = 1;
00023   qint32 type = -1;
00024   QString name = "";
00025
00026   ~GameElementData() {
00027     delete position;
00028     delete helth;
00029     delete weight;
00030     delete transitWeight;
00031     delete additionalData;
00032   }
00033
00034   void defaultInit() {
00035     position = new QVector3D(0, 0, 0);
00036     helth = InfinityDouble::FromValue(1);
00037     weight = InfinityDouble::FromValue(0);
00038     transitWeight = InfinityDouble::FromValue(0);
00039     additionalData = new QByteArray();
00040   }
00041 };
00042
00043 extern QDataStream& operator>>(QDataStream&
      stream, GameElementData& myclass);
00044 extern QDataStream& operator<<(QDataStream&
      stream, GameElementData& myclass);
00045
00046 class IBaseGameElement : public QObject {
00047   Q_OBJECT
00048  public:
00049   IBaseGameElement() {
00050     helth = InfinityDouble::FromValue(1);
00051     weight = InfinityDouble::FromValue(0);
00052     transitWeight = InfinityDouble::FromValue(0);
00053     position = new QVector3D(0, 0, 0);
00054     name = "";
00055     additionalData = new QByteArray();
00056   }
00057
00058   IBaseGameElement(GameElementData&
      data) { init(data); }
00059
00060   virtual void nextStep(){};
00061
00062   virtual QVector3D* getPosition() const { return position; }
00063   virtual int getType() const { return type; }
00064   virtual InfinityDouble* getHelth() const { return helth; }
00065   virtual InfinityDouble* getWeight() const { return weight; }
00066   virtual InfinityDouble* getMaxTransitWeight() const { return
      transitWeight; }
00067   virtual QString getName() const { return name; }
00068   virtual QByteArray* getAdditionalData() const { return
      additionalData; }
00069   virtual int getRVision() const { return rVision; }
00070
00071   friend QDataStream& operator<<(QDataStream&
      stream,
00072                                  const IBaseGameElement&
      myclass) {
00073     return stream << (*myclass.position) << (*myclass.
      helth)
00074                   << (*myclass.weight) << (*myclass.
      transitWeight)
```

```
00075                 << (*myclass.additionalData) <<
      myclass.type << myclass.name
00076                 << myclass.rVision;
00077   }
00078
00079   virtual ~IBaseGameElement() {
00080     delete helth;
00081     delete weight;
00082     delete transitWeight;
00083     delete position;
00084   }
00085
00086   virtual void setPosition(QVector3D* value) { this->position = value; }
00087   virtual void setHelth(InfinityDouble* value) { this->helth = value; }
00088   virtual void setWeight(InfinityDouble* value) { this->weight = value; }
00089   virtual void setTransitWeight(InfinityDouble* value) {
00090     this->transitWeight = value;
00091   }
00092   virtual void setType(int value) { this->type = value; }
00093   virtual void setName(QString name) { this->name = name; }
00094   virtual void setAdditionakData(QByteArray* data) {
00095     this->additionalData = data;
00096   }
00097   virtual void setRVision(int _rVison) { rVision = _rVison; }
00098
00099   void copyData(GameElementData& out);
00100  signals:
00101
00102  public slots:
00103
00104  protected:
00105   QVector3D* position = nullptr;
00106   InfinityDouble* helth = nullptr;
00107   InfinityDouble* weight = nullptr;
00108   InfinityDouble* transitWeight = nullptr;
00109   QByteArray* additionalData = nullptr;
00110   int rVision = 1;
00111
00112   int type = -1;
00113   QString name;
00114
00115   virtual void init(GameElementData& data);
00116 };
00117
00118 #endif  // IBASEGAMEELEMENT_H
```

3.51 Core/imap.cpp File Reference

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

Include dependency graph for imap.cpp:



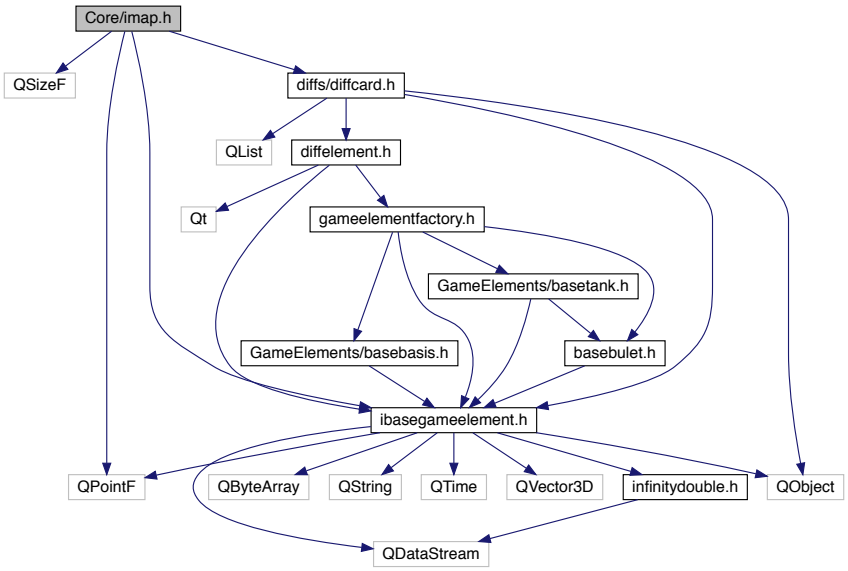
3.52 imap.cpp

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

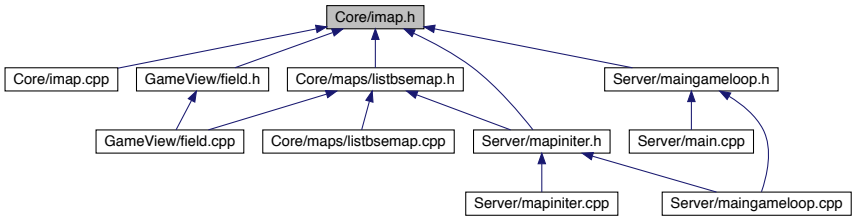
3.53 Core/imap.h File Reference

```
#include <QSizeF>
#include <QPointF>
#include "ibasegameelement.h"
#include "diffs/diffcard.h"
```

Include dependency graph for imap.h:



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



Classes

- class IMap

Typedefs

- typedef std::function< bool(IBaseGameElement *)> mapOperator

3.53.1 Typedef Documentation

3.53.1.1 mapOperator

```
typedef std::function<bool(IBaseGameElement*)> mapOperator
```

Definition at line 9 of file imap.h.

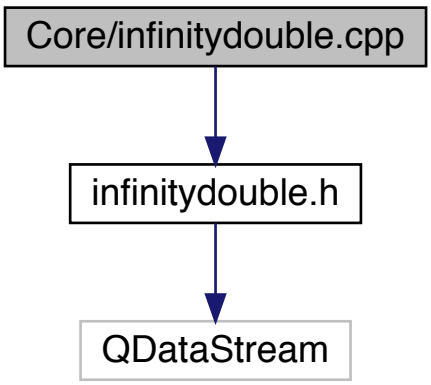
3.54 imap.h

```
00001 #ifndef IMAP_H
00002 #define IMAP_H
00003
00004 #include <QSizeF>
00005 #include <QPointF>
00006 #include "ibasegameelement.h"
00007 #include "diffs/diffcard.h"
00008
00009 typedef std::function<bool(IBaseGameElement*)> mapOperator;
00010
00011 class IMap {
00012  public:
00013   IMap();
00014   virtual QSizeF* getSize() = 0;
00015   virtual void insertElement(IBaseGameElement* element) = 0;
00016   virtual void updateFromDiff(DiffCard* diff) = 0;
00017   virtual void proccessAllInR(IBaseGameElement* element,
00018                               double r,
00019                               mapOperator) = 0;
00020   virtual int getCount() = 0;
00021   virtual IBaseGameElement* getElementAtPosition(int pos) = 0;
00022 };
00023
00024 #endif  // IMAP_H
```

3.55 Core/infinitydouble.cpp File Reference

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

Include dependency graph for infinitydouble.cpp:

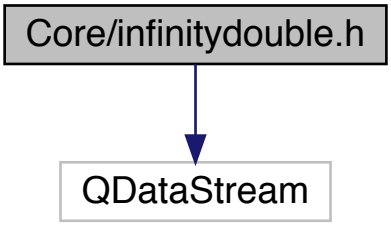


3.56 infinitydouble.cpp

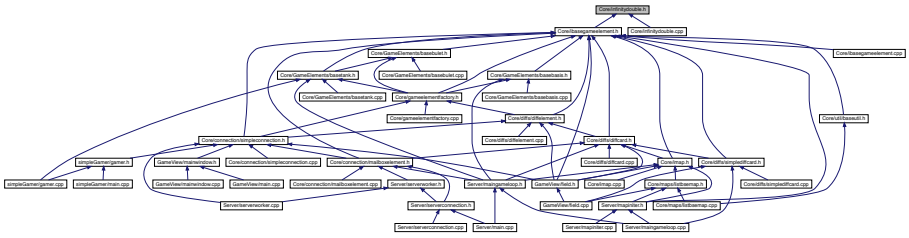
```
00001 #include "infinitydouble.h"
00002
00003 InfinityDouble::InfinityDouble(InfinityDouble& id) {
00004   this->w = id.w;
00005   this->isInfinity = id.isInfinity;
00006 }
00007
00008 InfinityDouble::InfinityDouble() {}
00009
00010 InfinityDouble* InfinityDouble::FromValue(double w) {
00011   InfinityDouble* result = new InfinityDouble();
00012   result->w = w;
00013   result->isInfinity = false;
00014   return result;
00015 }
00016
00017 InfinityDouble* InfinityDouble::
      InfinityValue() {
00018   InfinityDouble* result = new InfinityDouble();
00019   result->isInfinity = true;
00020   return result;
00021 }
```

3.57 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

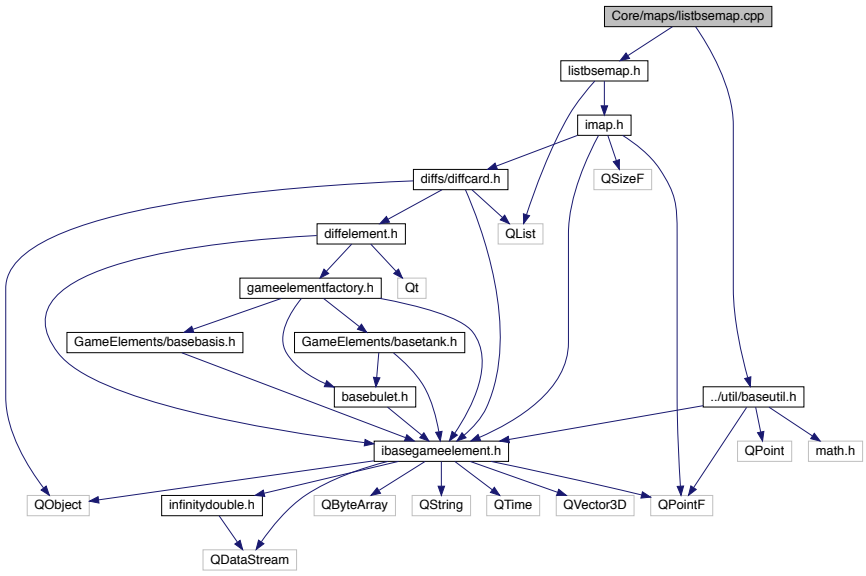
3.58 infinitydouble.h

```
00001 #ifndef WEIGHT_H
00002 #define WEIGHT_H
00003
00004 #include <QDataStream>
00005
00006 class InfinityDouble : public QObject {
00007   Q_OBJECT
00008
00009   friend QDataStream& operator<<(QDataStream&
      stream,
00010                                  const InfinityDouble&
      myclass) {
00011     return stream << myclass.isInfinity << myclass.
      w;
00012   }
00013
00014   friend QDataStream& operator>>(QDataStream&
      stream, InfinityDouble& myclass) {
00015     return stream >> myclass.isInfinity >> myclass.
      w;
00016   }
00017
00018  public:
00019   InfinityDouble(InfinityDouble& id);
00020
00021  private:
00022   InfinityDouble();
00023   double w;
00024   bool isInfinity = false;
00025
00026  public:
00027   static InfinityDouble* FromValue(double w);
00028   static InfinityDouble* InfinityValue();
00029 };
00030
00031 #endif  // WEIGHT_H
```

3.59 Core/maps/listbsemap.cpp File Reference

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

Include dependency graph for listbsemap.cpp:



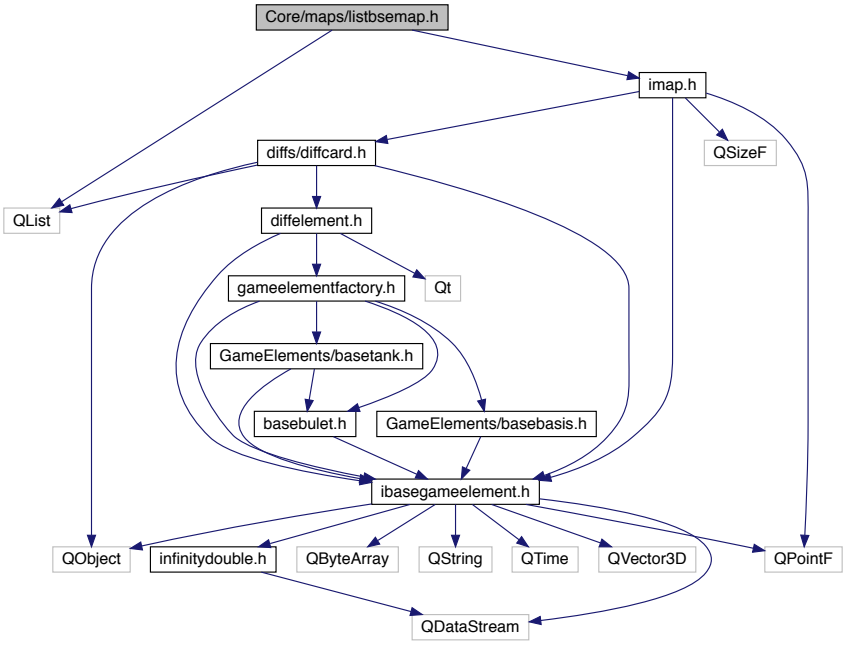
3.60 listbsemap.cpp

```
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   items = new QList<IBaseGameElement*>();
00011 }
00012
00013 ListBseMap::~ListBseMap() {
00014   delete items;
00015   delete size;
00016 }
00017
00018 QSizeF* ListBseMap::getSize() {
00019   return size;
00020 }
00021
00022 void ListBseMap::insertElement(IBaseGameElement* element)
      {
00023   items->append(element);
00024 }
00025
00026 void ListBseMap::proccessAllInR(
      IBaseGameElement* element,
00027                                 double r,
00028                                 mapOperator op) {
00029   for (int i = 0; i < items->size(); i++) {
00030     IBaseGameElement* el = items->at(i);
00031     if (element != el && distanceBetweenElement(el, element))
00032       op(el);
00033   }
00034 }
00035
00036 int ListBseMap::getCount() {
00037   return items->size();
00038 }
00039
00040 void ListBseMap::updateFromDiff(DiffCard* diff) {
00041   for (int i = 0; i < diff->getCountOfDiff(); i++) {
00042     switch (diff->at(i)->type) {
00043       case eNew:
00044         items->append(diff->at(i)->generateGameElementInstance());
00045         break;
00046       case eChange:
00047         updateItem(diff->at(i)->
      generateGameElementInstance());
00048         break;
00049       case eDeleted:
00050         updateItem(diff->at(i)->
      generateGameElementInstance(), false);
00051         break;
00052       case eEmpty:
00053         break;
00054     };
00055   }
00056 }
00057
00058 IBaseGameElement* ListBseMap::
      getElementAtPosition(int pos) {
00059   return items->at(pos);
00060 }
00061
00062 void ListBseMap::updateItem(IBaseGameElement* gameEleement,
      bool isReplace) {
00063   for (int i = 0; i < items->size(); i++) {
00064     if ((items->at(i)->getName()) == (gameEleement->getName())) {
00065       delete items->at(i);
00066       items->removeAt(i);
00067       if (isReplace)
00068         items->append(gameEleement);
00069       else
00070         delete gameEleement;
00071     }
00072   }
00073 }
```

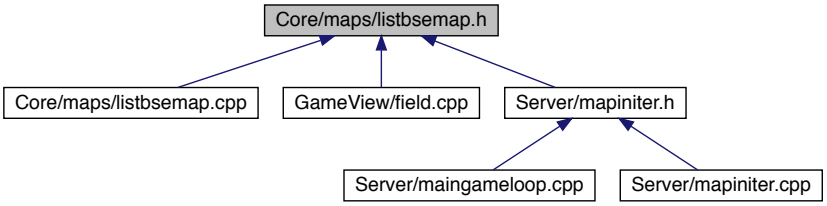
3.61 Core/maps/listbsemap.h File Reference

```
#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

3.62 listbsemap.h

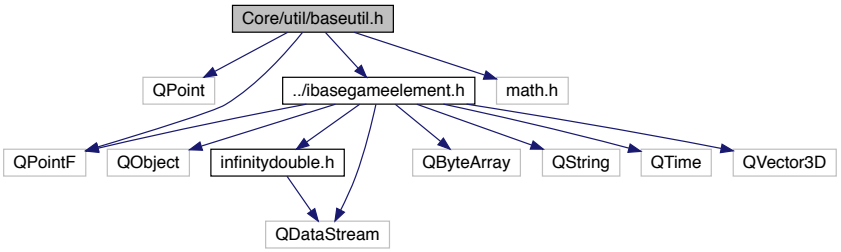
```
00001 #ifndef LISTBSEMAP_H
00002 #define LISTBSEMAP_H
00003
```

```
00004 #include <QList>
00005 #include "imap.h"
00006
00007 class ListBseMap : public IMap {
00008  public:
00009   ListBseMap(double width, double heigth);
00010   ListBseMap();
00011   ~ListBseMap();
00012   // IMap interface
00013  public:
00014   virtual QSizeF* getSize();
00015   virtual void insertElement(IBaseGameElement* element);
00016   virtual void proccessAllInR(IBaseGameElement* element,
00017                               double r,
00018                               mapOperator op);
00019   virtual int getCount();
00020   virtual void updateFromDiff(DiffCard* diff);
00021   IBaseGameElement* getElementAtPosition(int pos);
00022
00023  protected:
00024   QList<IBaseGameElement*>* items;
00025   QSizeF* size;
00026   /**
00027    * @brief updateItem update items in map
00028    * @param gameEleement element with information for update
00029    * @param isReplace if true than replace old item with @gameEleement. Otherway
00030    * delete from map list
00031    */
00032   void updateItem(IBaseGameElement* gameEleement, bool isReplace = true);
00033 };
00034
00035 #endif  // LISTBSEMAP_H
```

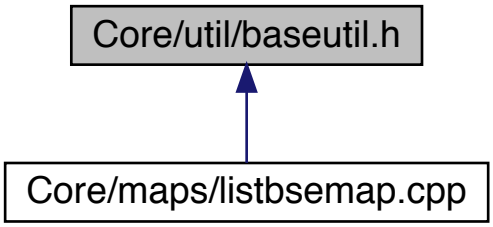
3.63 Core/util/baseutil.h File Reference

```
#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)

3.63.1 Function Documentation

3.63.1.1 distanceBetweenElement()

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

Definition at line 6 of file baseutil.h.

```
00006                                                                    {
00007   if (el1 == NULL || el2 == NULL)
00008     return 0;
00009   if (el1 == el2)
00010     return 0;
00011   const QVector3D* p1 = el1->getPosition();
00012   const 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 }
```

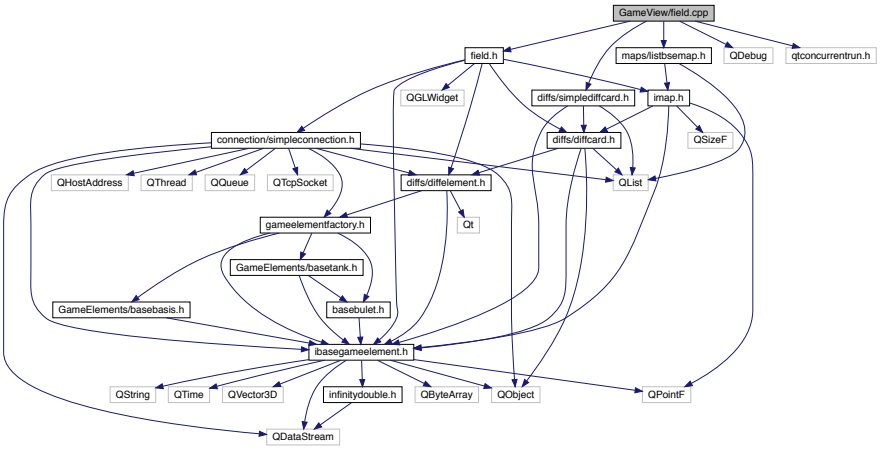
3.64 baseutil.h

```
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   const QVector3D* p1 = el1->getPosition();
00012   const 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 }
```

3.65 GameView/field.cpp File Reference

```
#include "field.h"
#include <QDebug>
#include "diffs/simplediffcard.h"
#include "maps/listbsemap.h"
#include <qtconcurrentrun.h>
```

Include dependency graph for field.cpp:



Functions

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

3.65.1 Function Documentation

3.65.1.1 loadTexture2()

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

Definition at line 54 of file field.cpp.

```
00054                                                   {
00055   glEnable(GL_TEXTURE_2D);  // Enable texturing
00056
00057   glGenTextures(1, &textureID);             // Obtain an id for the texture
00058   glBindTexture(GL_TEXTURE_2D, textureID);  // Set as the current texture
00059
00060   glPixelStorei(GL_UNPACK_ALIGNMENT, 1);
00061   glTexEnvf(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_DECAL);
00062
00063   QImage im(filename);
00064   QImage tex = QGLWidget::convertToGLFormat(im);
00065
00066   glTexImage2D(GL_TEXTURE_2D, 0, GL_RGBA, tex.width(), tex.height(), 0, GL_RGBA,
00067                GL_UNSIGNED_BYTE, tex.bits());
00068
00069   glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
00070   glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
00071
00072   glDisable(GL_TEXTURE_2D);
00073
00074   return tex;
00075 }
```

3.66 field.cpp

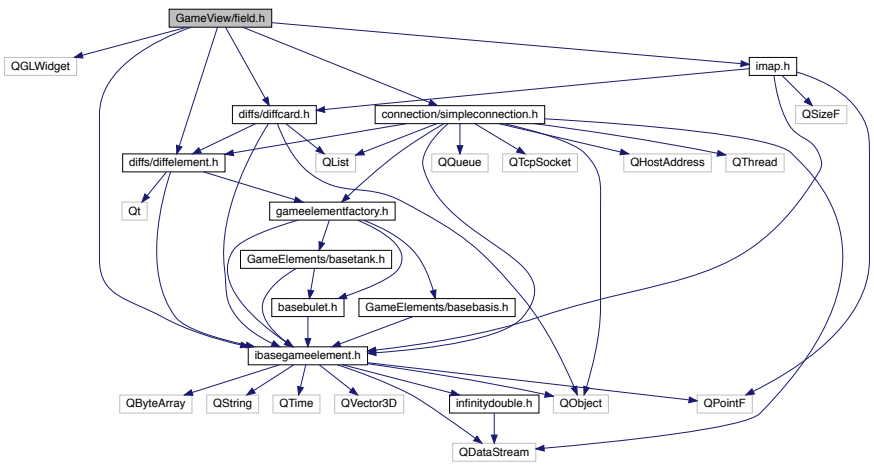
```
00001 #include "field.h"
00002 #include <QDebug>
00003 #include "diffs/simplediffcard.h"
00004 #include "maps/listbsemap.h"
00005 #include <qtconcurrentrun.h>
00006
00007 Field::Field(QWidget* parent)
00008     : QGLWidget(parent), connection(QHostAddress::LocalHost, this) {
00009   diff = new SimpleDiffCard();
00010   map = new ListBseMap();
00011   initRes();
00012 }
00013
00014 Field::~Field() {
00015   delete diff;
00016   delete map;
00017 }
00018
00019 void Field::onDiffReceive(QList<DiffElement*>* diff) {
00020   this->diff->loadFromList(diff);
00021   map->updateFromDiff(this->diff);
00022   this->diff->clear();
00023   delete diff;
00024   update();
00025
00026   QThread::msleep(100);
00027   this->connection.getBulder()->updateWatcher()->build();
00028 }
00029 void Field::initializeGL() {}
00030
00031 void Field::resizeGL(int w, int h) {}
00032
00033 void Field::paintGL() {
00034   glClearColor(0.1f, 0.1f, 0.1f, 1.0f);
00035   glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
00036   //  glTranslatef(-1, -1, 0);
00037   for (int i = 0; i < map->getCount(); i++) {
00038     auto currentElement = map->getElementAtPosition(i);
00039     if (currentElement->getType() == (int)eBaseGameElementType::eGrass) {
00040       drawGrass(currentElement->getPosition()->x(),
00041                 currentElement->getPosition()->y());
00042     }
00043     if (currentElement->getType() == (int)eBaseGameElementType::eSimpleTank) {
00044       drawTank(currentElement->getPosition()->x(),
00045                currentElement->getPosition()->y());
00046     }
00047   }
00048   //  glTranslatef(1, 1, 0);
00049   //  glEnd();
00050
00051   //  glDisable(GL_TEXTURE_2D);
00052 }
00053
00054 QImage loadTexture2(char* filename, GLuint& textureID) {
00055   glEnable(GL_TEXTURE_2D);  // Enable texturing
00056
00057   glGenTextures(1, &textureID);             // Obtain an id for the texture
00058   glBindTexture(GL_TEXTURE_2D, textureID);  // Set as the current texture
00059
00060   glPixelStorei(GL_UNPACK_ALIGNMENT, 1);
00061   glTexEnvf(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_DECAL);
00062
00063   QImage im(filename);
00064   QImage tex = QGLWidget::convertToGLFormat(im);
00065
00066   glTexImage2D(GL_TEXTURE_2D, 0, GL_RGBA, tex.width(), tex.height(), 0, GL_RGBA,
00067                GL_UNSIGNED_BYTE, tex.bits());
00068
00069   glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
00070   glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
00071
00072   glDisable(GL_TEXTURE_2D);
00073
00074   return tex;
00075 }
00076
00077 void Field::initRes() {
00078   loadTexture2((char*)":/res/grass.png", grass);
00079 }
00080
00081 void Field::drawGrass(float x, float y) {
00082   //  glEnable(GL_TEXTURE_2D);
00083   //  glBindTexture(GL_TEXTURE_2D, grass);
00084
```

```
00085   //  glBegin(GL_QUADS);
00086   //  glTexCoord2f(0, 0);
00087   //  glVertex3f(-1, -1, -1);
00088   //  glTexCoord2f(1, 0);
00089   //  glVertex3f(1, -1, -1);
00090   //  glTexCoord2f(1, 1);
00091   //  glVertex3f(1, 1, -1);
00092   //  glTexCoord2f(0, 1);
00093   //  glVertex3f(-1, 1, -1);
00094   // .....
00095   //  glDisable(GL_TEXTURE_2D);
00096   float size = scaleK;
00097   glColor3f(0.560, 0.956, 0.258);
00098   glBegin(GL_QUADS);
00099   glVertex2f(x * this->scaleK, y * scaleK);
00100   glVertex2f(x * this->scaleK, y * scaleK + size);
00101   glVertex2f(x * this->scaleK + size, y * scaleK + size);
00102   glVertex2f(x * this->scaleK + size, y * scaleK);
00103   glEnd();
00104 }
00105
00106 void Field::drawTank(float x, float y) {
00107   float size = scaleK;
00108   glColor3f(0.6862745098, 0.1215686275, 0.4156862745);
00109   glBegin(GL_QUADS);
00110   glVertex2f(x * this->scaleK, y * scaleK);
00111   glVertex2f(x * this->scaleK, y * scaleK + size * 2);
00112   glVertex2f(x * this->scaleK + size * 2, y * scaleK + size * 2);
00113   glVertex2f(x * this->scaleK + size * 2, y * scaleK);
00114   glEnd();
00115 }
00116 void Field::connectToServer() {
00117   if (isConnected)
00118     return;
00119   connection.openConnection();
00120   connect(&connection, SIGNAL(onDiffReceive(QList<DiffElement*>*)), this,
00121           SLOT(onDiffReceive(QList<DiffElement*>*)));
00122   connection.getBulder()->asFirstMessage(eConnectionType::eWatcher)->build();
00123   isConnected = true;
00124 }
```

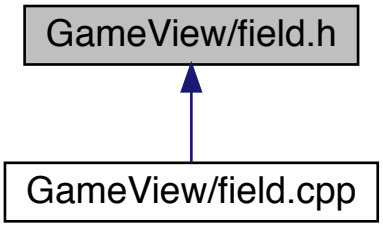
3.67 GameView/field.h File Reference

```
#include <QGLWidget>
#include <ibasegameelement.h>
#include <connection/simpleconnection.h>
#include <diffs/diffelement.h>
#include <diffs/diffcard.h>
#include "imap.h"
```

Include dependency graph for field.h:



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



Classes

- class Field

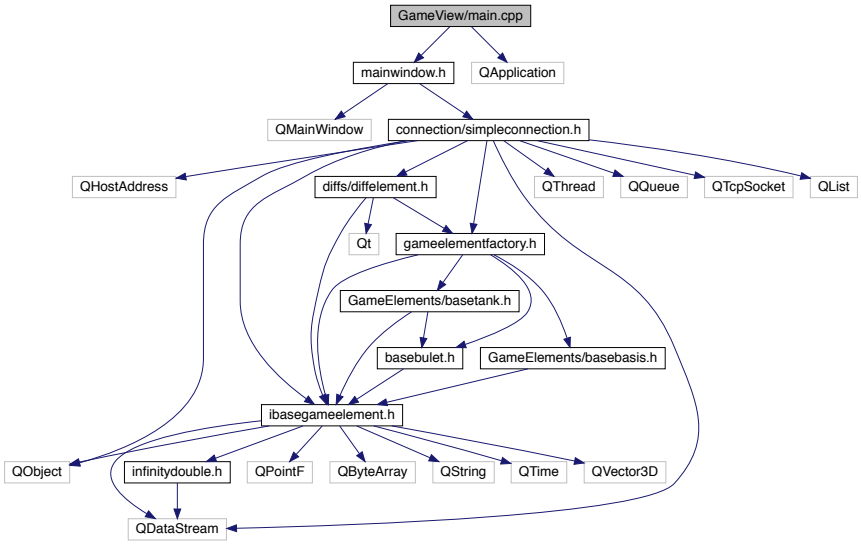
3.68 field.h

```
00001 #ifndef FIELD_H
00002 #define FIELD_H
00003 #include <QGLWidget>
00004 #include <ibasegameelement.h>
00005
00006 #include <connection/simpleconnection.h>
00007
00008 #include <diffs/diffelement.h>
00009 #include <diffs/diffcard.h>
00010 #include "imap.h"
00011
00012 class Field : public QGLWidget {
00013   Q_OBJECT
00014  public:
00015   Field(QWidget* parent = 0);
00016   ~Field();
00017
00018  public slots:
00019   void onDiffReceive(QList<DiffElement*>* diff);
00020
00021   void connectToServer();
00022   // QGLWidget interface
00023  protected:
00024   bool isConnected = false;
00025   void initializeGL() Q_DECL_OVERRIDE;
00026   void resizeGL(int w, int h) Q_DECL_OVERRIDE;
00027   void paintGL() Q_DECL_OVERRIDE;
00028
00029   void initRes();
00030
00031   void drawGrass(float x, float y);
00032
00033   void drawTank(float x, float y);
00034
00035   GLuint grass;
00036
00037   float scaleK = 0.02;
00038
00039   DiffCard* diff;
00040   IMap* map;
00041
00042   SimpleConnection connection;
00043 };
00044
00045 #endif  // FIELD_H
```

3.69 GameView/main.cpp File Reference

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

Include dependency graph for main.cpp:



Functions

- int main (int argc, char ∗argv[ ])

3.69.1 Function Documentation

3.69.1.1 main()

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

Definition at line 4 of file main.cpp.

References MainWindow::MainWindow().

```
00004                                {
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 }
```

Here is the call graph for this function:

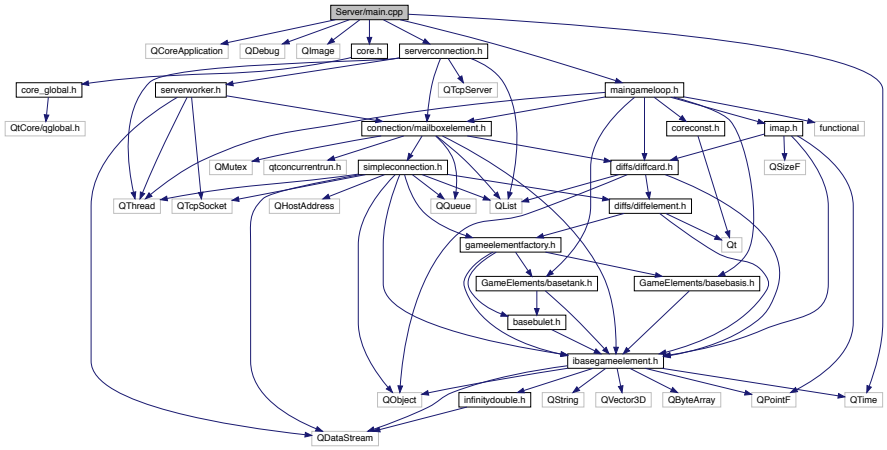
3.70 main.cpp

```
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 }
```

3.71 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[ ])

3.71.1 Function Documentation

3.71.1.1 initMainLooper()

```
void initMainLooper ( )
```

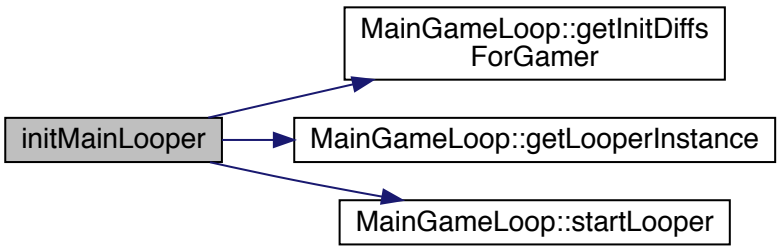
Definition at line 15 of file main.cpp.

References MainGameLoop::getInitDiffsForGamer(), MainGameLoop::getLooperInstance(), and MainGameLoop::startLooper().

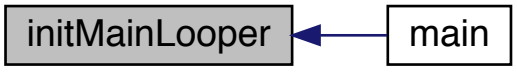
Referenced by main().

```
00015                       {
00016   MainGameLoop* mainLooper = MainGameLoop::getLooperInstance();
00017   mainLooper->startLooper();
00018 }
```

Here is the call graph for this function:



Here is the caller graph for this function:



3.71.1.2 initServer()

```
void initServer ( )
```

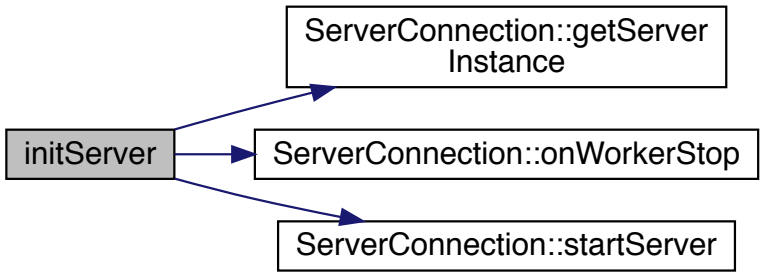
Definition at line 20 of file main.cpp.

References ServerConnection::getServerInstance(), ServerConnection::onWorkerStop(), and ServerConnection::startServer().

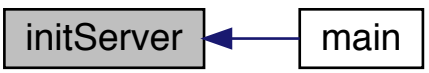
Referenced by main().

```
00020                   {
00021     ServerConnection* server = ServerConnection::getServerInstance
      ();
00022     QObject::connect(server, &ServerConnection::onServerError, &
      onServerError);
00023     server->setDefaultReceiver(MainGameLoop::getLooperInstance
      ());
00024     server->startServer();
00025 }
```

Here is the call graph for this function:



Here is the caller graph for this function:



3.71.1.3 main()

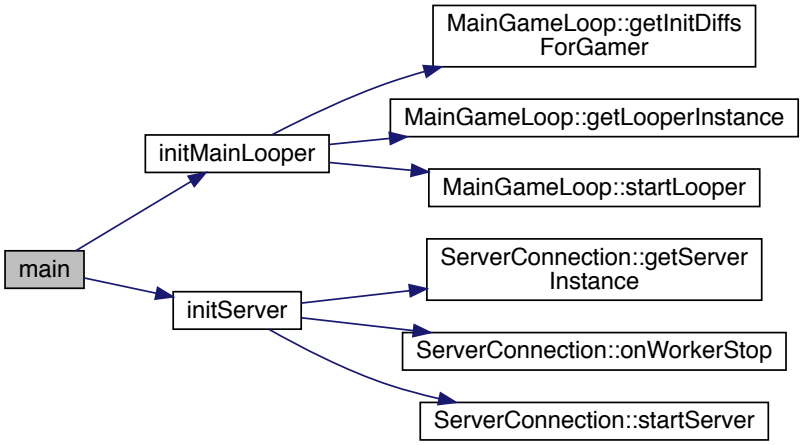
```
int main (
              int argc,
              char * argv[] )
```

Definition at line 27 of file main.cpp.

References initMainLooper(), and initServer().

```
00027                                {
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 }
```

Here is the call graph for this function:



3.71.1.4 onServerError()

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

Definition at line 11 of file main.cpp.

```
00011                                    {
00012   qDebug() << "some bad error :: " << error;
00013 }
```

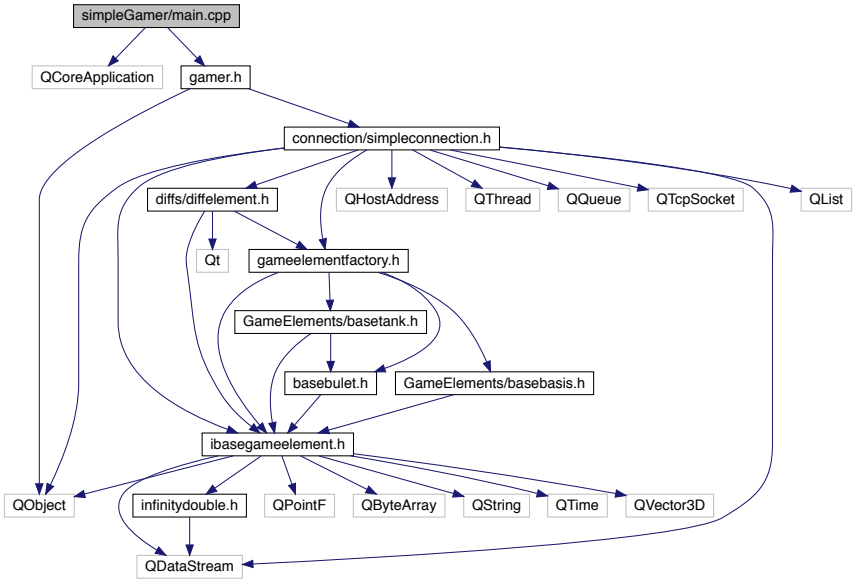
3.72 main.cpp

```
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 }
```

3.73 simpleGamer/main.cpp File Reference

```
#include <QCoreApplication>
#include "gamer.h"
```

Include dependency graph for main.cpp:



Functions

- int main (int argc, char ∗argv[ ])

3.73.1 Function Documentation

3.73.1.1 main()

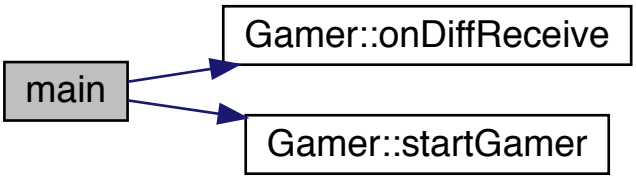
```
int main (
            int argc,
            char * argv[] )
```

Definition at line 4 of file main.cpp.

References Gamer::onDiffReceive(), and Gamer::startGamer().

```
00004                                {
00005   QCoreApplication a(argc, argv);
00006   qSetMessagePattern(
00007       "%{time boot} :: [%{type}::%{appname} in %{file}:%{line} th:%{threadid}] "
00008       "message:: "
00009       "%{message} "
00010       "%{backtrace [separator=\"\n\t\"]}");
00011   Gamer* gamer = new Gamer(&a);
00012   gamer->startGamer();
00013   return a.exec();
00014 }
```

Here is the call graph for this function:



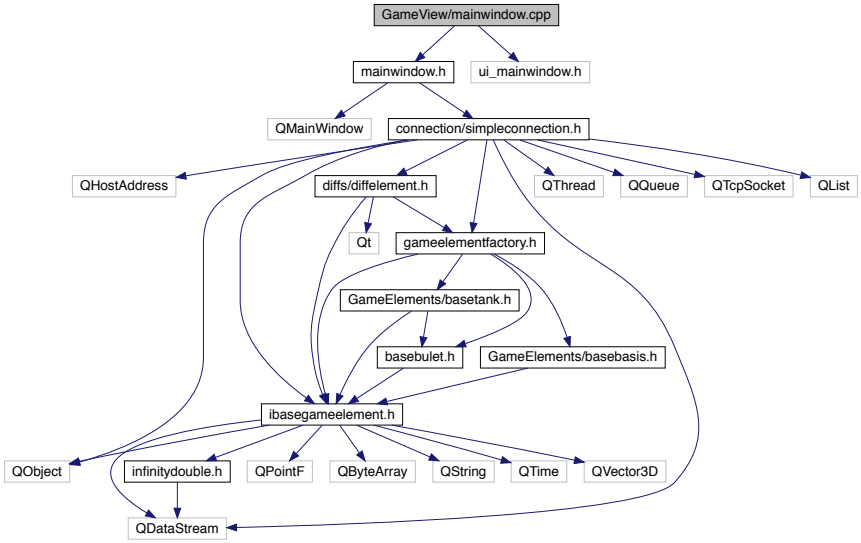
3.74 main.cpp

```
00001 #include <QCoreApplication>
00002 #include "gamer.h"
00003
00004 int main(int argc, char* argv[]) {
00005   QCoreApplication a(argc, argv);
00006   qSetMessagePattern(
00007       "%{time boot} :: [%{type}::%{appname} in %{file}:%{line} th:%{threadid}] "
00008       "message:: "
00009       "%{message} "
00010       "%{backtrace [separator=\"\n\t\"]}");
00011   Gamer* gamer = new Gamer(&a);
00012   gamer->startGamer();
00013   return a.exec();
00014 }
```

3.75 GameView/mainwindow.cpp File Reference

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

Include dependency graph for mainwindow.cpp:



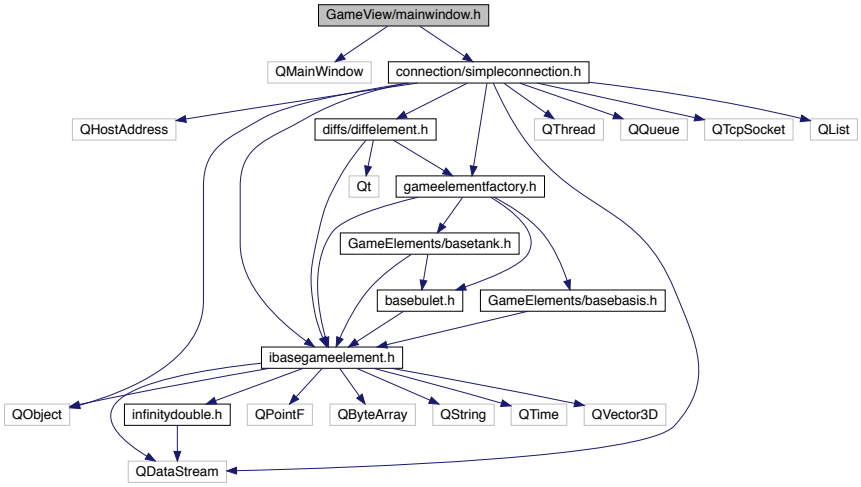
3.76 mainwindow.cpp

```
00001 #include "mainwindow.h"
00002 #include "ui_mainwindow.h"
00003
00004 MainWindow::MainWindow(QWidget* parent)
00005     : QMainWindow(parent), ui(new Ui::MainWindow) {
00006   ui->setupUi(this);
00007   ui->gameField->connectToServer();
00008 }
00009
00010 MainWindow::~MainWindow() {
00011   delete ui;
00012 }
```

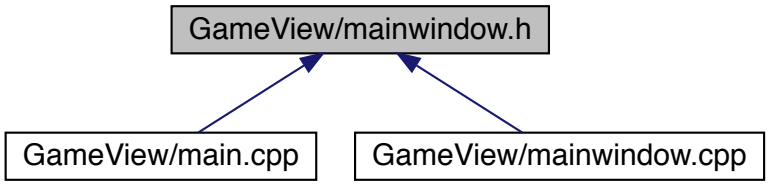
3.77 GameView/mainwindow.h File Reference

```
#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

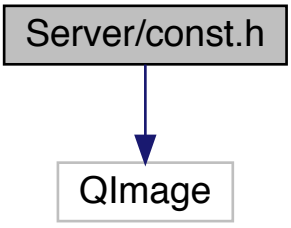
- Ui

3.78 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  private:
00020   Ui::MainWindow* ui;
00021 };
00022
00023 #endif  // MAINWINDOW_H
```

3.79 Server/const.h File Reference

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



Namespaces

- staticBockTypes

Variables

- int staticBockTypes::grass = 1

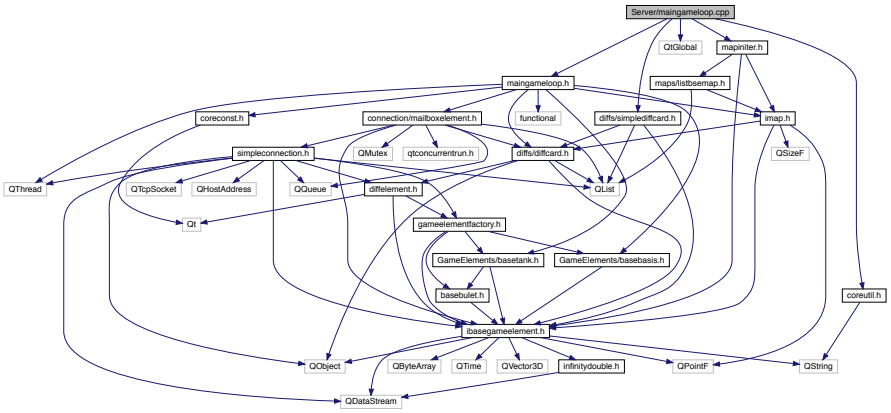
3.80 const.h

```
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
```

3.81 Server/maingameloop.cpp File Reference

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

Include dependency graph for maingameloop.cpp:



Functions

- DiffCard * getSimpleDiff ()

Variables

- QString TAG = "MainGameLoop"

3.81.1 Function Documentation

3.81.1.1 getSimpleDiff()

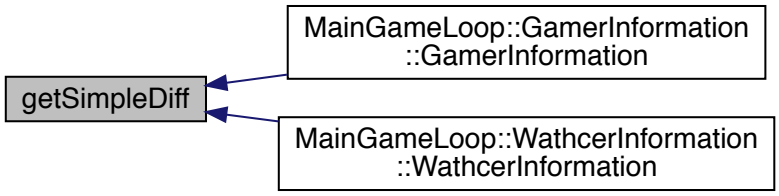
```
DiffCard* getSimpleDiff ( )
```

Definition at line 162 of file maingameloop.cpp.

Referenced by MainGameLoop::GamerInformation::GamerInformation(), and MainGameLoop::WathcerInformation::WathcerInformation().

```
00162                           {
00163   return new SimpleDiffCard();
00164 }
```

Here is the caller graph for this function:



3.81.2 Variable Documentation

3.81.2.1 TAG

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

Definition at line 7 of file maingameloop.cpp.

3.82 maingameloop.cpp

```
00001 #include "maingameloop.h"
00002 #include "mapiniter.h"
00003 #include <QtGlobal>
00004 #include <coreutil.h>
00005 #include "diffs/simplediffcard.h"
00006
00007 QString TAG = "MainGameLoop";
00008
00009 MainGameLoop::MainGameLoop() {
00010   map = MapIniter().initSimapleMap();
00011   time = 1;
00012   stepDiff = getSimpleDiff();
00013 }
00014 void MainGameLoop::proccessGamerMessage(MailReceiver::
      mailMessage* msg,
00015                                         MailSender* receiver) {
00016   GamerInformation* currentGamer = nullptr;
```

```
00017   bool isAlredyExist = false;
00018   for (int i = 0; i < gamersList.size(); i++) {
00019     if (gamersList.at(i)->sender == receiver) {
00020       isAlredyExist = true;
00021       currentGamer = gamersList.at(i);
00022       break;
00023     }
00024   }
00025   if (!isAlredyExist) {
00026     currentGamer = new GamerInformation(map);
00027     currentGamer->name = codingNum(gamersList.size());
00028     currentGamer->sender = receiver;
00029     gamersList.append(currentGamer);
00030
00031     gamersItems.append(new QList<IBaseGameElement*>());
00032
00033     BaseBasis* basis = new BaseBasis();
00034     basis->setPosition(&currentGamer->position);
00035     basis->setEnergy(currentGamer->lifeCount);
00036     map->insertElement(basis);
00037
00038     gamersItems.last()->append(basis);
00039     currentGamer->minion = gamersItems.last();
00040     auto op = [this, currentGamer](IBaseGameElement* element) -> bool {
00041       DiffElement* diff = new DiffElement(eDiffType::eNew, element);
00042       diff->time = this->time;
00043       currentGamer->personalDiff->append(diff);
00044       return false;
00045     };
00046     map->proccessAllInR(basis, basis->getRVision(), op);
00047   } else {
00048     if (msg->message->messageType == eInsertNewItem) {
00049       for (int i = 0; i < msg->message->items->size(); i++) {
00050         map->insertElement(msg->message->items->at(i));
00051         DiffElement* diffItem =
00052             new DiffElement(eDiffType::eNew, msg->message->items->at(i));
00053         diffItem->time = time;
00054         stepDiff->append(diffItem);
00055       }
00056     }
00057   }
00058   auto diffValue = currentGamer->personalDiff;
00059   currentGamer->personalDiff = getSimpleDiff();
00060   receiver->receiveResponce(diffValue, msg->message
      );
00061 }
00062
00063 void MainGameLoop::proccessWatcherMessage(MailReceiver::
      mailMessage* msg,
00064                                           MailSender* receiver) {
00065   WathcerInformation* watcher = nullptr;
00066   for (int i = 0; i < watchers.size() && watcher == nullptr; i++) {
00067     if (watchers.at(i)->sender == receiver)
00068       watcher = watchers.at(i);
00069   }
00070   if (watcher == nullptr) {
00071     watcher = new WathcerInformation();
00072     watcher->sender = receiver;
00073   }
00074   switch (msg->message->messageType) {
00075     case eFirstMessae:
00076       receiver->receiveResponce(getAllMapAsDiff
      (), msg->message);
00077       break;
00078     case eGetUpdateMessage:
00079       receiver->receiveResponce(watcher->personalDiff
      , msg->message);
00080       watcher->personalDiff = getSimpleDiff();
00081       //      receiver->receiveResponce(getAllMapAsDiff(), msg->message);
00082       break;
00083     default:
00084       receiver->receiveResponce(getSimpleDiff
      (), msg->message);
00085   }
00086 }
00087
00088 MainGameLoop::messageProccessor MainGameLoop::getProccessorForMessage(
00089     eConnectionType type) {
00090   switch (type) {
00091     case eGamer:
00092       return &MainGameLoop::proccessGamerMessage;
00093     case eWatcher:
00094       return &MainGameLoop::proccessWatcherMessage;
00095   }
00096   Q_ASSERT_X(2 * 2 != 4, "missing branch ",
00097              "missing brunch for eConnectionType");
00098 }
```

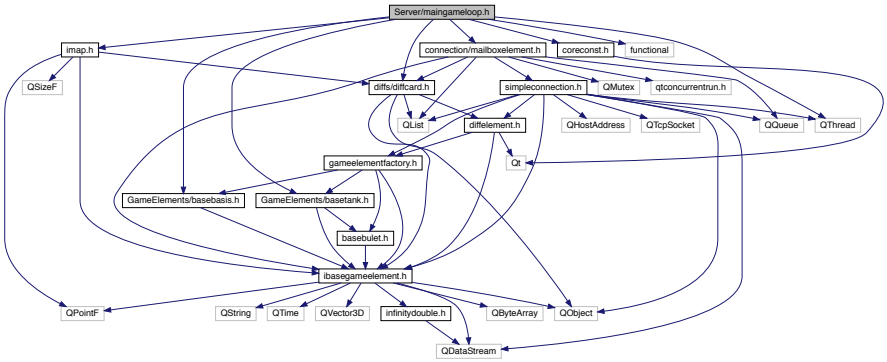
```
00099
00100 void MainGameLoop::run() {
00101   mailMessage* msg;
00102   qInfo() << "starting main loop";
00103   while (isWork) {
00104     // process incoming messages
00105     while ((msg = nextMessage()) != nullptr) {
00106       qInfo() << TAG << "receive new messahe " << (*msg);
00107       auto fun = getProccessorForMessage(msg->message->connectionType);
00108       (this->*fun)(msg, msg->sender);
00109     }
00110
00111     foreach (auto items, gamersItems) {
00112       for (int i = 0; i < items->size(); i++) {
00113         //        switch ((eBaseGameElementType)items->at(i)->getType())
00114       }
00115     }
00116     for (int i = 0; i < watchers.size(); i++) {
00117       WathcerInformation* watcher = watchers.at(i);
00118       watcher->personalDiff->updateFromOtherCard(stepDiff);
00119     }
00120     stepDiff->freeItems();
00121
00122     msleep(100);
00123   }
00124 }
00125
00126 DiffCard* MainGameLoop::getAllMapAsDiff() {
00127   DiffCard* result = new SimpleDiffCard();
00128   DiffElement* item;
00129   for (int i = 0; i < map->getCount(); i++)
00130     result->append(
00131         new DiffElement(eDiffType::eNew, map->getElementAtPosition(i)));
00132   return result;
00133 }
00134
00135 MainGameLoop::~MainGameLoop() {
00136   delete map;
00137   for (int i = 0; i < gamersItems.size(); i++) {
00138     delete gamersItems.at(i);
00139   }
00140 }
00141
00142 void MainGameLoop::startLooper() {
00143   this->start();
00144 }
00145
00146 MainGameLoop* MainGameLoop::instance = nullptr;
00147
00148 MainGameLoop::GamerInformation::
      GamerInformation(IMap* map) {
00149   name = "";
00150   lifeCount = limits::defaultBasisEnergy;
00151   QSizeF* size = map->getSize();
00152   float rand1 = ((float)(rand())) / ((float)(RAND_MAX));
00153   float rand2 = ((float)(rand())) / ((float)(RAND_MAX));
00154   position = QVector3D(size->width() * rand1, size->height() * rand2, 0);
00155   personalDiff = getSimpleDiff();
00156 }
00157
00158 MainGameLoop::GamerInformation::~
      GamerInformation() {
00159   delete personalDiff;
00160 }
00161
00162 DiffCard* getSimpleDiff() {
00163   return new SimpleDiffCard();
00164 }
```

3.83 Server/maingameloop.h File Reference

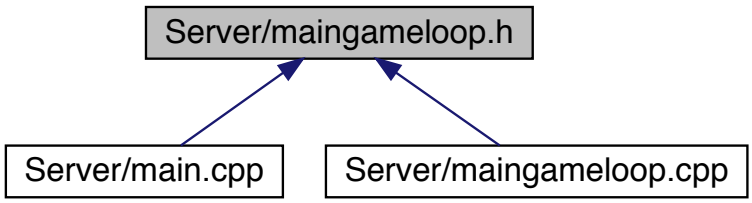
```
#include "imap.h"
#include "connection/mailboxelement.h"
#include <QThread>
#include <functional>
#include "GameElements/basetank.h"
#include "coreconst.h"
#include "GameElements/basebasis.h"
```

```
#include "diffs/diffcard.h"
```

Include dependency graph for maingameloop.h:



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



Classes

- class MainGameLoop
- class MainGameLoop::GamerInformation
- class MainGameLoop::WathcerInformation

Functions

- DiffCard * getSimpleDiff ()

3.83.1 Function Documentation

3.83.1.1 getSimpleDiff()

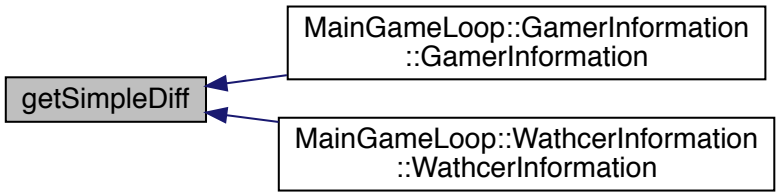
```
DiffCard* getSimpleDiff ( )
```

Definition at line 162 of file maingameloop.cpp.

Referenced by MainGameLoop::GamerInformation::GamerInformation(), and MainGameLoop::WathcerInformation::WathcerInformation().

```
00162                           {
00163   return new SimpleDiffCard();
00164 }
```

Here is the caller graph for this function:



3.84 maingameloop.h

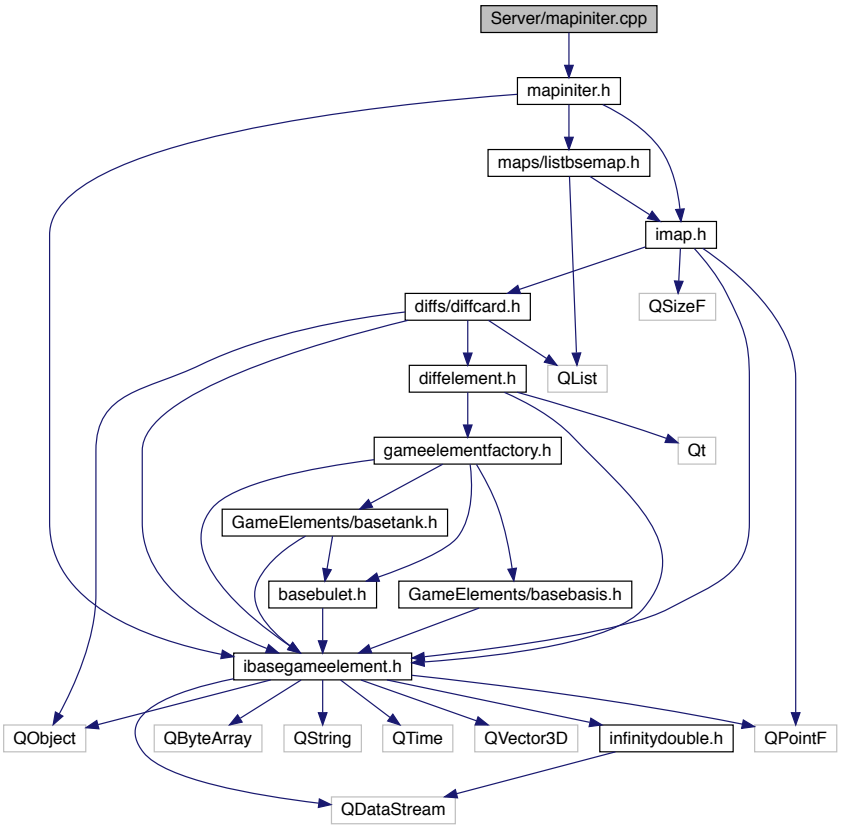
```
00001 #ifndef MAINGAMELOOP_H
00002 #define MAINGAMELOOP_H
00003
00004 #include "imap.h"
00005 #include "connection/mailboxelement.h"
00006 #include <QThread>
00007 #include <functional>
00008 #include "GameElements/basetank.h"
00009 #include "coreconst.h"
00010 #include "GameElements/basebasis.h"
00011 #include "diffs/diffcard.h"
00012
00013 // clever clean, whait for stop all thread
00014
00015 // clean diffs
00016
00017 DiffCard* getSimpleDiff();
00018
00019 class MainGameLoop : public MailReceiver, public
      QThread {
00020  public:
00021   ~MainGameLoop();
00022   inline static MainGameLoop* getLooperInstance() {
00023     if (instance == nullptr) {
00024       instance = new MainGameLoop();
00025     }
00026     return instance;
00027   }
00028
00029   void startLooper();
00030
00031  private:
00032   typedef void (MainGameLoop::*messageProccessor)(MailReceiver::
      mailMessage*,
00033                                                   MailSender*);
00034   static MainGameLoop* instance;
00035   IMap* map;
```

```
00036   MainGameLoop();
00037
00038   void proccessGamerMessage(mailMessage* msg, MailSender* receiver);
00039   void proccessWatcherMessage(mailMessage* msg, MailSender* receiver);
00040   messageProccessor getProccessorForMessage(eConnectionType type);
00041
00042   DiffCard* stepDiff;
00043
00044  protected:
00045   class GamerInformation {
00046    public:
00047     QString name;
00048     uint64_t lifeCount;
00049     QVector3D position;
00050     MailSender* sender;
00051     QList<IBaseGameElement*>* minion;
00052     DiffCard* personalDiff;
00053     GamerInformation(IMap* map);
00054     ~GamerInformation();
00055   };
00056
00057   class WathcerInformation {
00058    public:
00059     MailSender* sender;
00060     DiffCard* personalDiff;
00061     WathcerInformation() { personalDiff =
      getSimpleDiff(); }
00062   };
00063
00064   void run();
00065   bool isWork = true;
00066   DiffCard* getAllMapAsDiff();
00067   QList<DiffElement*>* getInitDiffsForGamer(
      BaseBasis* basis);
00068   QList<WathcerInformation*> watchers;
00069   /**
00070    * list of client as gamers
00071    * @brief gamersList
00072    */
00073   QList<GamerInformation*> gamersList;
00074   /**
00075    * list of gamer object; Each list represent game element of each gamer;
00076    * firs element of each gamer object is basis
00077    * @brief gamersItems
00078    */
00079   QList<QList<IBaseGameElement*>*>
      gamersItems;
00080
00081   uint64_t time;
00082 };
00083
00084 #endif  // MAINGAMELOOP_H
```

3.85 Server/mapiniter.cpp File Reference

```
#include "mapiniter.h"
```

Include dependency graph for mapiniter.cpp:



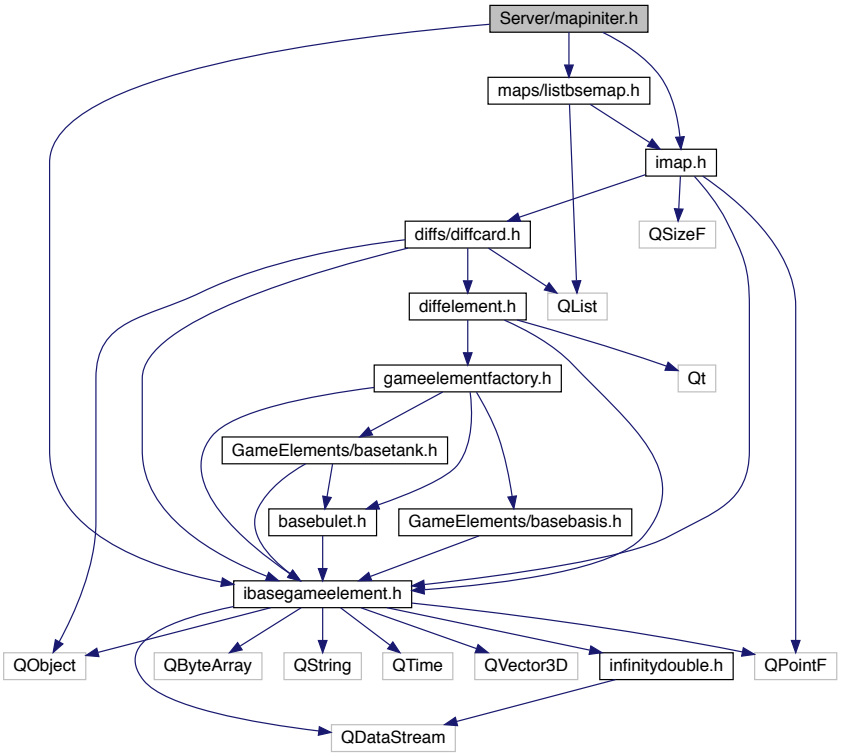
3.86 mapiniter.cpp

```
00001 #include "mapiniter.h"
00002
00003 MapIniter::MapIniter() {}
00004
00005 IMap* MapIniter::initSimapleMap() {
00006   const int mapW = 100;
00007   const int mapH = 100;
00008
00009   IMap* result = new ListBseMap(mapW, mapH);
00010   for (int i = 0; i < 400; i++) {
00011     IBaseGameElement* item = getRandomGrass(mapW
      , mapH);
00012     result->insertElement(item);
00013   }
00014   return result;
00015 }
00016
00017 IBaseGameElement* MapIniter::getRandomGrass(double
      maxWidth, double maxHeigth) {
00018   IBaseGameElement* result = new IBaseGameElement();
00019   result->setPosition(new QVector3D(
00020       (double)rand() / (double)RAND_MAX * maxWidth - maxWidth / 2,
00021       (double)rand() / (double)RAND_MAX * maxHeigth - maxHeigth / 2, 0));
00022   result->setHelth(InfinityDouble::InfinityValue
      ());
00023   result->setTransitWeight(InfinityDouble
      ::FromValue(0));
00024   result->setType(eBaseGameElementType::
      eGrass);
00025   return result;
00026 }
```

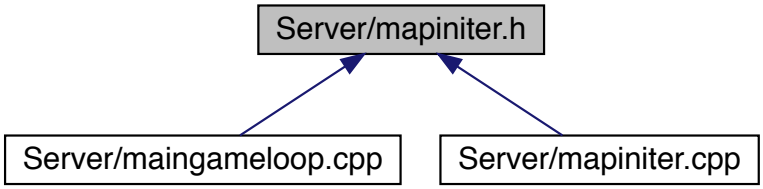
3.87 Server/mapiniter.h File Reference

```
#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

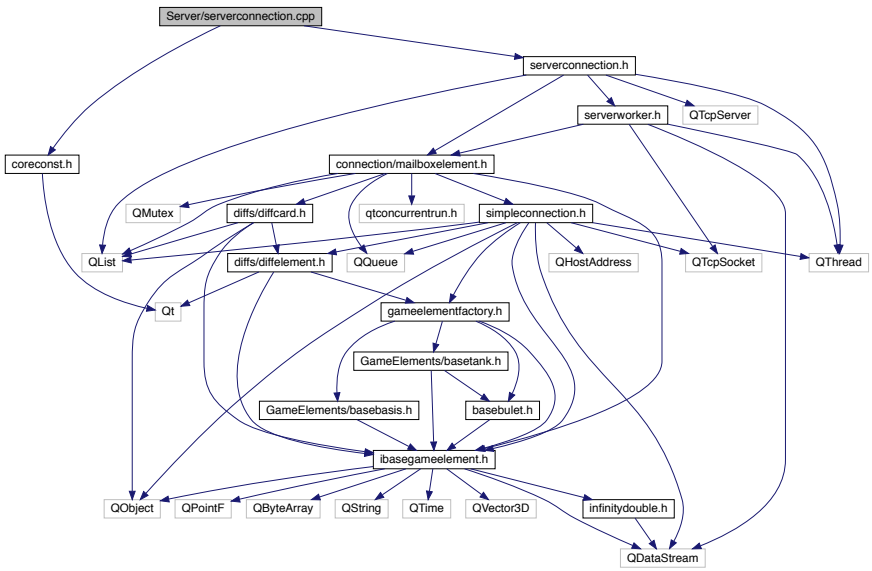
3.88 mapiniter.h

```
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
```

3.89 Server/serverconnection.cpp File Reference

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

Include dependency graph for serverconnection.cpp:



3.90 serverconnection.cpp

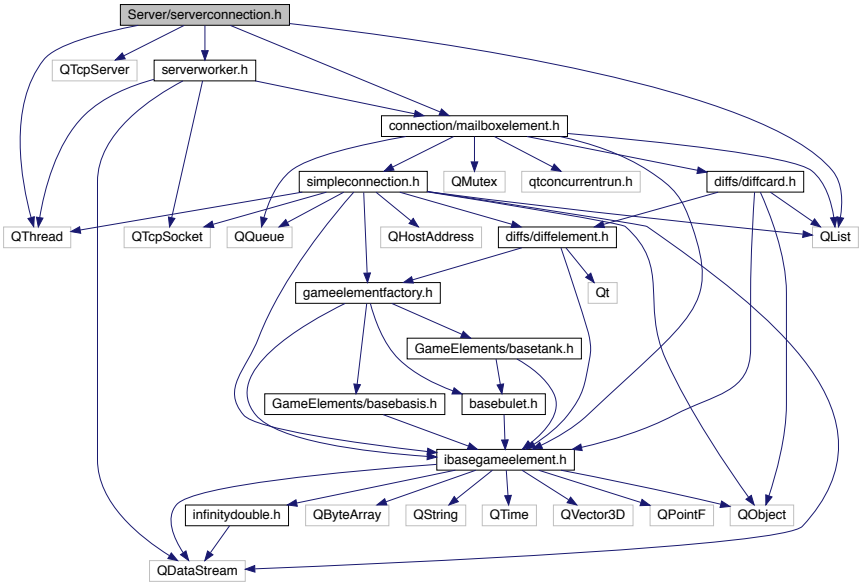
```
00001 #include "serverconnection.h"
00002 #include "coreconst.h"
00003
00004 ServerConnection* ServerConnection::
      instance = nullptr;
00005
00006 ServerConnection::~ServerConnection() {
00007   for (int i = 0; i < connections->size(); i++)
00008     delete connections->at(i);
00009   delete 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(-1);
00037     //    isAppareNewConnection = server->hasPendingConnections();
00038     if (isAppareNewConnection) {
00039       QTcpSocket* newConnection = server->nextPendingConnection();
00040       if (newConnection == nullptr)
00041         continue;
00042       qInfo() << "appare new connection  :: "
00043               << newConnection->peerAddress().toString() << ":"
00044               << newConnection->peerPort();
00045       ServerWorker* newWorker = new ServerWorker(newConnection);
00046       newWorker->setDefaultReceiver(this->receiver);
00047       connections->append(newWorker);
00048       newWorker->start();
00049     }
00050     msleep(30);
00051   }
00052 }
00053
00054 void ServerConnection::receiveResponce(
      DiffCard* diff,
00055                                        MessageForServer* message) {}
```

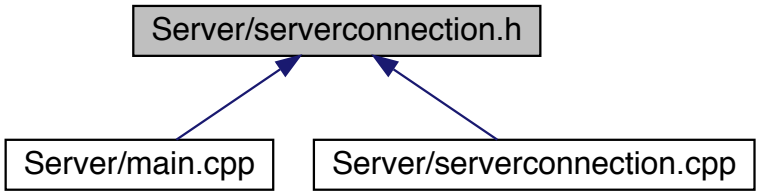
3.91 Server/serverconnection.h File Reference

```
#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 }

3.91.1 Enumeration Type Documentation

3.91.1.1 serverError

```
enum serverError
```

Enumerator

canNotStartServer	

Definition at line 10 of file serverconnection.h.

```
00010 { canNotStartServer };
```

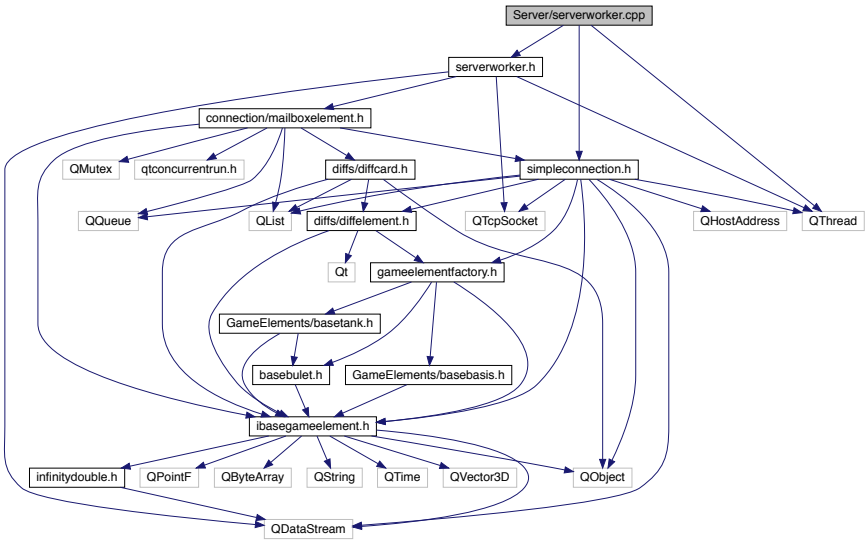
3.92 serverconnection.h

```
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(DiffCard* diff,
      MessageForServer* message);
00043 };
00044
00045 #endif  // SERVERCONNECTION_H
```

3.93 Server/serverworker.cpp File Reference

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

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



3.94 serverworker.cpp

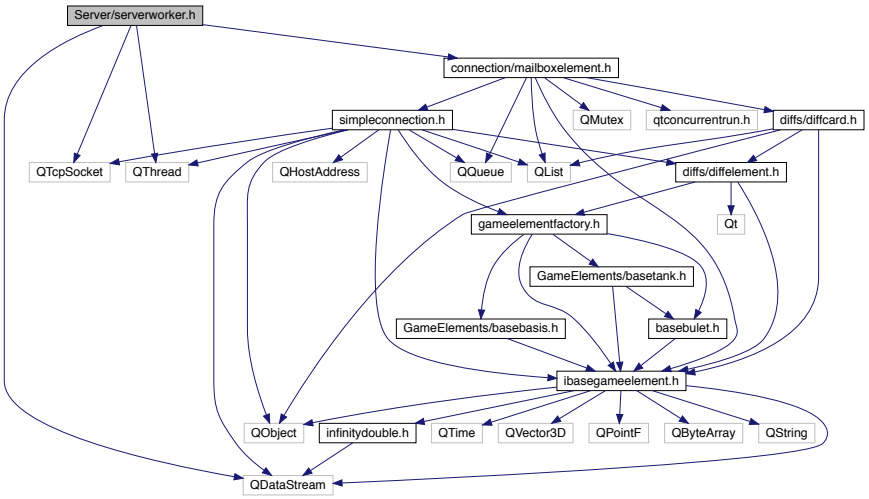
```
00001 #include "serverworker.h"
00002
00003 #include <connection/simpleconnection.h>
00004 #include <QThread>
00005
00006 ServerWorker::ServerWorker(QTcpSocket* socket) {
00007   this->socket = socket;
00008   connect(socket, SIGNAL(error(QAbstractSocket::SocketError)), this,
00009           SLOT(onSocketError(QAbstractSocket::SocketError)));
00010 }
00011
00012 void ServerWorker::run() {
00013   out = new QDataStream(socket);
00014   out->setVersion(QDataStream::Qt_5_7);
00015
00016   ReceiverThread* th = new ReceiverThread(this, socket);
00017
00018   while (this->isWork) {
00019     workerMutex.lock();
00020     while (responceMessage.length() > 0) {
00021       responceData data = responceMessage.takeFirst();
00022       qInfo() << socket->peerAddress() << ":" << socket->peerPort()
00023               << "getting some message for sending";
00024       out->startTransaction();
00025       (*out) << (*data.messag);
00026       (*out) << data.diff->getCountOfDiff();
00027       for (int i = 0; i < data.diff->getCountOfDiff(); i++)
00028         (*out) << (*data.diff->at(i));
00029       out->commitTransaction();
00030       socket->flush();
00031       qInfo() << socket->peerAddress() << ":" << socket->peerPort()
00032               << "send  response to client";
00033       for (int i = 0; i < data.diff->getCountOfDiff(); i++)
00034         delete data.diff->at(i);
00035       data.diff->clear();
00036       delete data.diff;
00037       delete data.messag;
00038     }
00039     msleep(100);
00040     workerMutex.unlock();
00041   }
00042   th->stop();
00043   delete th;
00044   qInfo() << "worker has stoped";
00045   emit onStop(this);
```

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

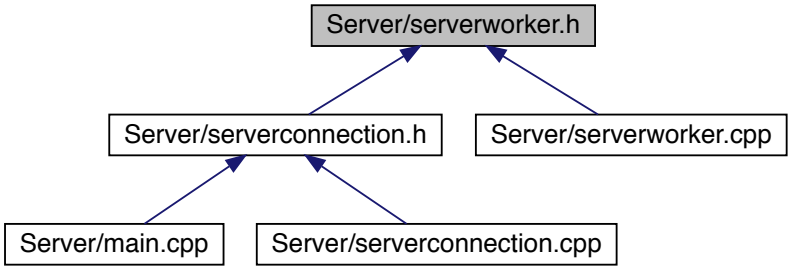
3.95 Server/serverworker.h File Reference

```
#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

3.96 serverworker.h

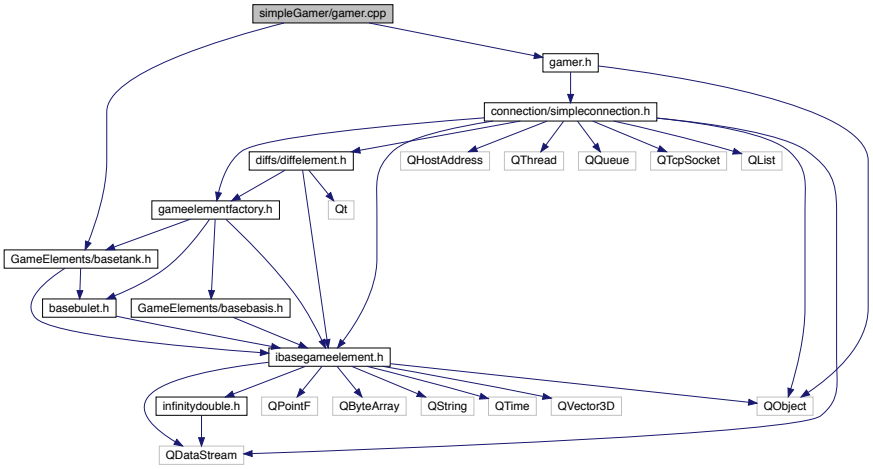
```
00001 #ifndef SERVERWORKER_H
00002 #define SERVERWORKER_H
00003
00004 #include <QTcpSocket>
00005 #include <QThread>
00006 #include <QDataStream>
00007
00008 #include <connection/mailboxelement.h>
00009
00010 class ServerWorker : public QThread, public
```

```
      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 slots:
00024   void onSocketError(QAbstractSocket::
      SocketError error);
00025
00026  protected:
00027   struct responceData {
00028    public:
00029     responceData(DiffCard* _diff, MessageForServer* _messag)
      {
00030       this->diff = _diff;
00031       this->messag = _messag;
00032     }
00033
00034     DiffCard* diff;
00035     MessageForServer* messag;
00036   };
00037
00038   QTcpSocket* socket;
00039   volatile bool isWork = true;
00040   QDataStream* out;
00041   QMutex workerMutex;
00042   QQueue<responceData> responceMessage;
00043
00044   class ReceiverThread : public QThread {
00045    public:
00046     ReceiverThread(ServerWorker* parent, QTcpSocket* socket);
00047     virtual ~ReceiverThread();
00048     void stop();
00049
00050     // QThread interface
00051    protected:
00052     void run();
00053     QTcpSocket* socket;
00054     ServerWorker* parentThread;
00055     QDataStream* in;
00056     bool isWork = true;
00057     QMutex receiverMutex;
00058   };
00059
00060   // MailSender interface
00061  public:
00062   void receiveResponce(DiffCard* diff,
      MessageForServer* message);
00063 };
00064
00065 #endif  // SERVERWORKER_H
```

3.97 simpleGamer/gamer.cpp File Reference

```
#include "gamer.h"
#include "GameElements/basetank.h"
```

Include dependency graph for gamer.cpp:



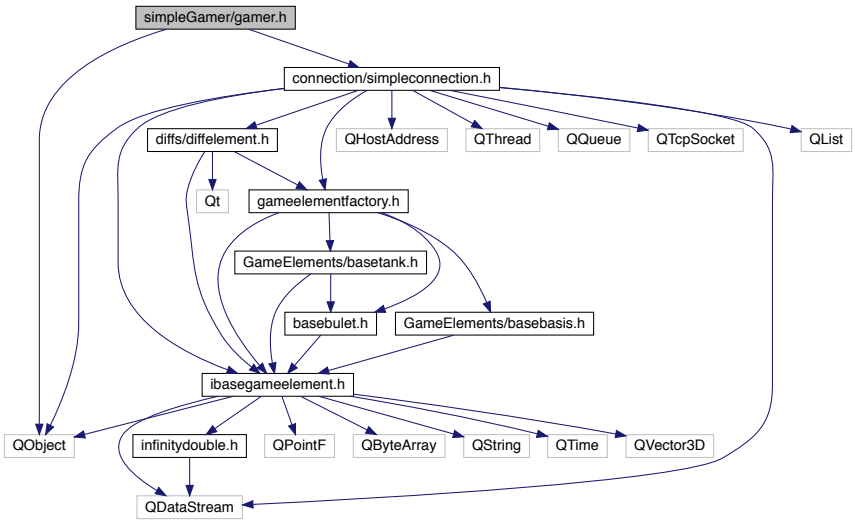
3.98 gamer.cpp

```
00001 #include "gamer.h"
00002 #include "GameElements/basetank.h"
00003
00004 Gamer::Gamer(QObject* parent)
00005     : QObject(parent), connection(QHostAddress::LocalHost, this) {}
00006
00007 void Gamer::startGamer() {
00008   connection.openConnection();
00009   connect(&connection, SIGNAL(onDiffReceive(QList<DiffElement*>*)), this,
00010           SLOT(onDiffReceive(QList<DiffElement*>*)));
00011   connection.getBulder()->asFirstMessage(eConnectionType::eGamer)->build();
00012 }
00013
00014 void Gamer::onDiffReceive(QList<DiffElement*>* diffList) {
00015   if (count > 10)
00016     return;
00017   BaseTank* newElement = new BaseTank();
00018   newElement->setPosition(new QVector3D(0, 0, 0));
00019   newElement->setDirection(0.2);
00020   newElement->setSpeed(10);
00021   QList<IBaseGameElement*>* items = new QList<IBaseGameElement*>();
00022   items->append(newElement);
00023   connection.getBulder()->addNewItem(items)->build();
00024   count++;
00025 }
```

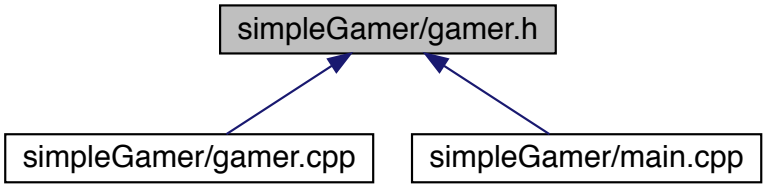
3.99 simpleGamer/gamer.h File Reference

```
#include <QObject>
#include <connection/simpleconnection.h>
```

Include dependency graph for gamer.h:



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



Classes

- class Gamer

3.100 gamer.h

```
00001 #ifndef GAMER_H
00002 #define GAMER_H
00003
00004 #include <QObject>
00005
00006 #include <connection/simpleconnection.h>
00007
00008 class Gamer : public QObject {
00009   Q_OBJECT
00010  public:
00011   explicit Gamer(QObject* parent = 0);
00012   void startGamer();
00013
00014  signals:
```

```
00015
00016  public slots:
00017
00018  protected:
00019   SimpleConnection connection;
00020   int count = 0;
00021
00022  protected slots:
00023   void onDiffReceive(QList<DiffElement*>*);
00024 };
00025
00026 #endif  // GAMER_H
```