Fractale

Généré par Doxygen 1.8.9.1

Vendredi 11 Mars 2016 14 :58 :50

Table des matières

Chapitre 1

Index hiérarchique

1.1 Hiérarchie des classes

Cette liste d'héritage est classée approximativement par ordre alphabétique :

Chapitre 2

Index des classes

2.1 Liste des classes

Liste des classes, structures, unions et interfaces avec une brève description :

Chapitre 3

Index des fichiers

3.1 Liste des fichiers

Liste de tous les fichiers avec une brève description :

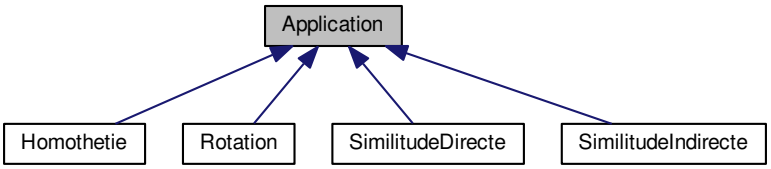
Chapitre 4

Documentation des classes

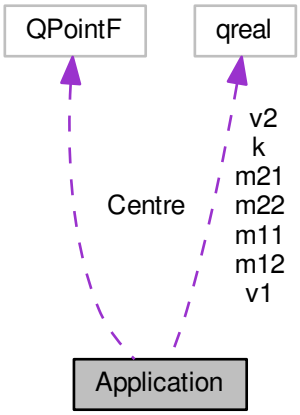
4.1 Référence de la classe Application

`#include "application.h"`

Graphe d'héritage de Application :



Graphe de collaboration de Application :



Fonctions membres publiques

- Application ()
  Application : :Application Constructeur par default de Application.
- void setm11 (qreal m)
  Application : :setm11.
- void setm12 (qreal m)
  Application : :setm12.
- void setm21 (qreal m)
  Application : :setm21.
- void setm22 (qreal m)
  Application : :setm22.
- void setv1 (qreal m)
  Application : :setv1.
- void setv2 (qreal m)
  Application : :setv2.
- void setCentre (QPointF P)
  Application : :setCentre.
- void setK (qreal K)
  Application : :setK.
- qreal getm11 () const
  Application : :getm11.
- qreal getm12 () const
  Application : :getm12.
- qreal getm21 () const
  Application : :getm21.
- qreal getm22 () const
  Application : :getm22.
- qreal getv1 () const
  Application : :getv1.
- qreal getv2 () const
  Application : :getv2.
- QPointF getCentre () const
  Application : :getCentre.
- qreal getK () const
  Application : :getK.
- qreal getTheta () const
  Application : :getTheta.
- bool isHomothetie () const
  Application : :isHomothetie.
- bool isRotation () const
  Application : :isRotation.
- QPointF DoForQPointF (QPointF const &P) const
  Application : :DoForQPointF.
- Forme DoForForme (Forme const &F) const
  Application : :DoForForme.
- QList< Forme > DoForEns (const QList< Forme > &EnsForme) const
  Application : :DoForEns.

Attributs protégés

- qreal k
  k Rapport de l'application
- qreal m11
  Décrit la matrice de rotation
  $$\begin{pmatrix} \mathbf{m_{1,1}} & \mathbf{m_{1,2}} \\ \mathbf{m_{2,1}} & \mathbf{m_{2,2}} \end{pmatrix} = \begin{pmatrix} \mathbf{cos}(\theta) & -\mathbf{sin}(\theta) \\ \mathbf{sin}(\theta) & \mathbf{cos}(\theta) \end{pmatrix} \mathbf{ou} \begin{pmatrix} \mathbf{cos}(\theta) & \mathbf{sin}(\theta) \\ \mathbf{sin}(\theta) & -\mathbf{cos}(\theta) \end{pmatrix}$$
  .
- qreal m12
  Décrit la matrice de rotation
  $$\begin{pmatrix} \mathbf{m_{1,1}} & \mathbf{m_{1,2}} \\ \mathbf{m_{2,1}} & \mathbf{m_{2,2}} \end{pmatrix} = \begin{pmatrix} \mathbf{cos}(\theta) & -\mathbf{sin}(\theta) \\ \mathbf{sin}(\theta) & \mathbf{cos}(\theta) \end{pmatrix} \mathbf{ou} \begin{pmatrix} \mathbf{cos}(\theta) & \mathbf{sin}(\theta) \\ \mathbf{sin}(\theta) & -\mathbf{cos}(\theta) \end{pmatrix}$$
  .
- qreal m21
  Décrit la matrice de rotation
  $$\begin{pmatrix} \mathbf{m_{1,1}} & \mathbf{m_{1,2}} \\ \mathbf{m_{2,1}} & \mathbf{m_{2,2}} \end{pmatrix} = \begin{pmatrix} \mathbf{cos}(\theta) & -\mathbf{sin}(\theta) \\ \mathbf{sin}(\theta) & \mathbf{cos}(\theta) \end{pmatrix} \mathbf{ou} \begin{pmatrix} \mathbf{cos}(\theta) & \mathbf{sin}(\theta) \\ \mathbf{sin}(\theta) & -\mathbf{cos}(\theta) \end{pmatrix}$$

.

— qreal m22
Décrit la matrice de rotation

$$\begin{pmatrix} \mathbf{m_{1,1}} & \mathbf{m_{1,2}} \\ \mathbf{m_{2,1}} & \mathbf{m_{2,2}} \end{pmatrix} = \begin{pmatrix} \mathbf{cos}(\theta) & -\mathbf{sin}(\theta) \\ \mathbf{sin}(\theta) & \mathbf{cos}(\theta) \end{pmatrix} \mathbf{ou} \begin{pmatrix} \mathbf{cos}(\theta) & \mathbf{sin}(\theta) \\ \mathbf{sin}(\theta) & -\mathbf{cos}(\theta) \end{pmatrix}$$

.

— qreal v1
v1 Parametre du vecteur de translation V

$$\mathbf{V} = \begin{pmatrix} \mathbf{v_1} \\ \mathbf{v_2} \end{pmatrix}$$

— qreal v2
v2 Parametre du vecteur de translation V

$$\mathbf{V} = \begin{pmatrix} \mathbf{v_1} \\ \mathbf{v_2} \end{pmatrix}$$

— QPointF Centre
Centre de l'application.

4.1.1 Documentation des constructeurs et destructeur

4.1.1.1 Application : :Application ( )

Application : :Application Constructeur par default de Application.

4.1.2 Documentation des fonctions membres

4.1.2.1 QList< Forme > Application : :DoForEns (const QList< Forme > & *EnsForme*) const

Application : :DoForEns.

Paramètres

EnsForme	

Renvoie

Image de l'application sur chaqune des forme

4.1.2.2 Forme Application : :DoForForme (Forme const & *F*) const

Application : :DoForForme.

Paramètres

F	Forme

Renvoie

Image de l'application sur la forme

4.1.2.3 QPointF Application : :DoForQPointF (QPointF const & *P*) const

Application : :DoForQPointF.

Paramètres

P	

Renvoie

Image de l'application en P

4.1.2.4 QPointF Application : :getCentre ( ) const

Application : :getCentre.

Renvoie

Retourne le centre de l'application

4.1.2.5 qreal Application : :getK ( ) const

Application : :getK.

Renvoie

Retourne le rapport de l'application

4.1.2.6 qreal Application : :getm11 ( ) const

Application : :getm11.

Renvoie

Retourne la valeur de m11

4.1.2.7 qreal Application : :getm12 ( ) const

Application : :getm12.

Renvoie

Retourne la valeur de m12

4.1.2.8 qreal Application : :getm21 ( ) const

Application : :getm21.

Renvoie

Retourne la valeur de m21

4.1.2.9 qreal Application : :getm22 ( ) const

Application : :getm22.

Renvoie

Retourne la valeur de m22

4.1.2.10 qreal Application : :getTheta ( ) const

Application : :getTheta.

Renvoie

Retourne l'angle de rotation de l'application

4.1.2.11 qreal Application : :getv1 ( ) const

Application : :getv1.

Renvoie

Retourne la valeur de v1

4.1.2.12 qreal Application : :getv2 ( ) const

Application : :getv2.

Renvoie

Retourne la valeur de v2

4.1.2.13 bool Application : :isHomothetie ( ) const

Application : :isHomothetie.

Renvoie

Verifie si l'application est une homothetie ?

4.1.2.14 bool Application : :isRotation ( ) const

Application : :isRotation.

Renvoie

Verifie si l'application est une rotation ?

4.1.2.15 void Application : :setCentre (QPointF *P*)

Application : :setCentre.

Paramètres

P	Modifie la valeur du centre

4.1.2.16 void Application : :setK (qreal *K*)

Application : :setK.

Paramètres

K	Modifie la valeur de k

4.1.2.17 void Application : :setm11 (qreal *m*)

Application : :setm11.

Paramètres

m	Modifie la valeur de m11

4.1.2.18 void Application : :setm12 (qreal *m*)

Application : :setm12.

Paramètres

m	Modifie la valeur de m12

4.1.2.19 void Application : :setm21 (qreal *m*)

Application : :setm21.

Paramètres

m	Modifie la valeur de m21

4.1.2.20 void Application : :setm22 (qreal *m*)

Application : :setm22.

Paramètres

m	Modifie la valeur de m22

4.1.2.21 void Application : :setv1 (qreal *m*)

Application : :setv1.

Paramètres

m	Modifie la valeur de v1

4.1.2.22 void Application : :setv2 (qreal *m*)

Application : :setv2.

Paramètres

m	Modifie la valeur de v2

4.1.3 Documentation des données membres

4.1.3.1 QPointF Application : :Centre [protected]

Centre de l'application.

4.1.3.2 qreal Application : :k [protected]

k Rapport de l'application

4.1.3.3 qreal Application : :m11 [protected]

Décrit la matrice de rotation

$$\begin{pmatrix} \mathbf{m_{1,1}} & \mathbf{m_{1,2}} \\ \mathbf{m_{2,1}} & \mathbf{m_{2,2}} \end{pmatrix} = \begin{pmatrix} \mathbf{cos}(\theta) & -\mathbf{sin}(\theta) \\ \mathbf{sin}(\theta) & \mathbf{cos}(\theta) \end{pmatrix} \mathbf{ou} \begin{pmatrix} \mathbf{cos}(\theta) & \mathbf{sin}(\theta) \\ \mathbf{sin}(\theta) & -\mathbf{cos}(\theta) \end{pmatrix}$$

.

4.1.3.4 qreal Application : :m12 [protected]

Décrit la matrice de rotation

$$\begin{pmatrix} \mathbf{m_{1,1}} & \mathbf{m_{1,2}} \\ \mathbf{m_{2,1}} & \mathbf{m_{2,2}} \end{pmatrix} = \begin{pmatrix} \mathbf{cos}(\theta) & -\mathbf{sin}(\theta) \\ \mathbf{sin}(\theta) & \mathbf{cos}(\theta) \end{pmatrix} \mathbf{ou} \begin{pmatrix} \mathbf{cos}(\theta) & \mathbf{sin}(\theta) \\ \mathbf{sin}(\theta) & -\mathbf{cos}(\theta) \end{pmatrix}$$

.

4.1.3.5 qreal Application : :m21 [protected]

Décrit la matrice de rotation

$$\begin{pmatrix} \mathbf{m_{1,1}} & \mathbf{m_{1,2}} \\ \mathbf{m_{2,1}} & \mathbf{m_{2,2}} \end{pmatrix} = \begin{pmatrix} \mathbf{cos}(\theta) & -\mathbf{sin}(\theta) \\ \mathbf{sin}(\theta) & \mathbf{cos}(\theta) \end{pmatrix} \mathbf{ou} \begin{pmatrix} \mathbf{cos}(\theta) & \mathbf{sin}(\theta) \\ \mathbf{sin}(\theta) & -\mathbf{cos}(\theta) \end{pmatrix}$$

.

4.1.3.6 qreal Application : :m22 [protected]

Décrit la matrice de rotation

$$\begin{pmatrix} \mathbf{m_{1,1}} & \mathbf{m_{1,2}} \\ \mathbf{m_{2,1}} & \mathbf{m_{2,2}} \end{pmatrix} = \begin{pmatrix} \mathbf{cos}(\theta) & -\mathbf{sin}(\theta) \\ \mathbf{sin}(\theta) & \mathbf{cos}(\theta) \end{pmatrix} \mathbf{ou} \begin{pmatrix} \mathbf{cos}(\theta) & \mathbf{sin}(\theta) \\ \mathbf{sin}(\theta) & -\mathbf{cos}(\theta) \end{pmatrix}$$

.

4.1.3.7 qreal Application : :v1 [protected]

v1 Parametre du vecteur de translation V

$$\mathbf{V} = \begin{pmatrix} \mathbf{v_1} \\ \mathbf{v_2} \end{pmatrix}$$

4.1.3.8 qreal Application : :v2 [protected]

v2 Parametre du vecteur de translation V

$$\mathbf{V} = \begin{pmatrix} \mathbf{v_1} \\ \mathbf{v_2} \end{pmatrix}$$

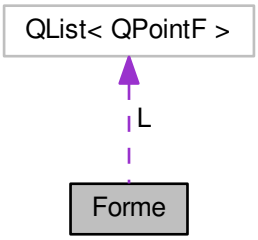
La documentation de cette classe a été générée à partir des fichiers suivants :

— application.h
— application.cpp

4.2 Référence de la classe Forme

`#include "forme.h"`

Graphe de collaboration de Forme :



Fonctions membres publiques

- Forme ()
  Forme : :Forme.
- int GetSize () const
  Forme : :GetSize.
- QPointF GetPoint (int i) const
  Forme : :GetPoint.
- void AddPoint (const QPointF &P)
  Forme : :AddPoint Ajoute le point P à la forme.
- void generateExisting (quint32 n=0)
  Forme : :generateExisting Génére une forme par défaut
  n=0 : Segment
  n=1 : Triangle.

Attributs privés

- QList< QPointF > L
  Liste des points constituant la forme dans l'ordre de tracé

4.2.1 Documentation des constructeurs et destructeur

4.2.1.1 Forme : :Forme ( )

Forme : :Forme.

4.2.2 Documentation des fonctions membres

4.2.2.1 void Forme : :AddPoint (const QPointF & *P*)

Forme : :AddPoint Ajoute le point P à la forme.

Paramètres

P	QPointF

4.2.2.2 void Forme : :generateExisting (quint32 *n* = 0)

Forme : :generateExisting Génére une forme par défaut
n=0 : Segment
n=1 : Triangle.

Paramètres

n	

4.2.2.3 QPointF Forme : :GetPoint (int *i*) const

Forme : :GetPoint.

Paramètres

i	Indice du point

Renvoie

Retourne le i-ème point(s) de la forme Nécessite que l'indice soit VALIDE.

4.2.2.4 int Forme : :GetSize ( ) const

Forme : :GetSize.

Renvoie

Nombre de points constituant la forme

4.2.3 Documentation des données membres

4.2.3.1 QList<QPointF> Forme : :L [private]

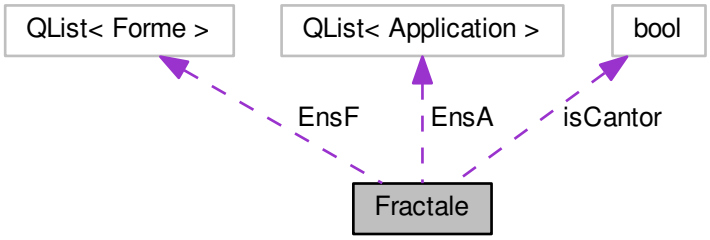
Liste des points constituant la forme dans l'ordre de tracé

La documentation de cette classe a été générée à partir des fichiers suivants :
- forme.h
- forme.cpp

4.3 Référence de la classe Fractale

```
#include "fractale.h"
```

Graphe de collaboration de Fractale :



Fonctions membres publiques

- Fractale ()
  Fractale : :Fractale.
- void AddApplication (Application A)
  Fractale : :AddApplication Ajoute une Aplication à la fractale.
- void AddHomothetie (qreal k)
  Fractale : :AddHomothetie Ajoute une Homothetie de rapport k à la fractale.
- void AddHomothetie (qreal k, QPointF Centre)
  Fractale : :AddHomothetie Ajoute une Homothetie de rapport k et de centre Centre à la fractale.
- void AddHomothetie (qreal k, qreal x, qreal y)
  Fractale : :AddHomothetie Ajoute une Homothetie de rapport k et de centre (x, y) à la fractale.
- void AddRotation (qreal theta)
  Fractale : :AddRotation Ajoute une Rotation d'angle theta à la fractale.
- void AddRotation (qreal theta, QPointF Centre)
  Fractale : :AddRotation Ajoute une Rotation d'angle theta et de centre Centre à la fractale.
- void AddRotation (qreal theta, qreal x, qreal y)
  Fractale : :AddRotation Ajoute une Rotation d'angle theta et de centre (x, y) à la fractale.
- void AddForme (Forme F)
  Fractale : :AddForme Ajoute une Forme à la fractale.
- bool isLikeCantor () const
  Fractale : :isLikeCantor.
- void setLikeCantor (bool p)
  Fractale : :setLikeCantor Attibut la valeur p à isCantor.
- void RunOnce ()
  Fractale : :RunOnce Calcul la fractale au rang suivant.
- Forme getFromEnsForme (int i) const
  Fractale : :getFromEnsForme.
- int getSizeEnsForme () const
  Fractale : :getSizeEnsForme.
- int getSizeEnsAppli () const
  Fractale : :getSizeEnsAppli.
- void generateExisting (quint32 n)
  Fractale : :generateExisting Génere une fractale selon des valeurs par defaut.

Attributs privés

- QList< Application > EnsA
  EnsA Stocke l'ensemble des Applications définissant la fractale.
- QList< Forme > EnsF
  EnsF Stocke L'ensemble des points constituant la fractale au cours du temps.
- bool isCantor

4.3.1 Documentation des constructeurs et destructeur

4.3.1.1 Fractale : :Fractale ( )

Fractale : :Fractale.

4.3.2 Documentation des fonctions membres

4.3.2.1 void Fractale : :AddApplication (Application *A*)

Fractale : :AddApplication Ajoute une Aplication à la fractale.

Paramètres

A	Application

4.3.2.2 void Fractale : :AddForme (Forme *F*)

Fractale : :AddForme Ajoute une Forme à la fractale.

Paramètres

F	Forme

4.3.2.3 void Fractale : :AddHomothetie (qreal *k*)

Fractale : :AddHomothetie Ajoute une Homothetie de rapport k à la fractale.

Paramètres

k	Rapport de l'Homothetie

4.3.2.4 void Fractale : :AddHomothetie (qreal *k*, QPointF *Centre*)

Fractale : :AddHomothetie Ajoute une Homothetie de rapport k et de centre Centre à la fractale.

Paramètres

k	Rapport de l'Homothetie
Centre	Centre de l'Homothetie

4.3.2.5 void Fractale : :AddHomothetie (qreal *k*, qreal *x*, qreal *y*)

Fractale : :AddHomothetie Ajoute une Homothetie de rapport k et de centre (x, y) à la fractale.

Paramètres

k	Rapport de l'Homothetie
x	Cx
y	Cy

4.3.2.6 void Fractale : :AddRotation (qreal *theta*)

Fractale : :AddRotation Ajoute une Rotation d'angle theta à la fractale.

Paramètres

theta	Angle de la rotation

4.3.2.7 void Fractale : :AddRotation (qreal *theta,* QPointF *Centre*)

Fractale : :AddRotation Ajoute une Rotation d'angle theta et de centre Centre à la fractale.

Paramètres

theta	Angle de la rotation
Centre	Centre de la rotation

4.3.2.8 void Fractale : :AddRotation (qreal *theta,* qreal *x,* qreal *y*)

Fractale : :AddRotation Ajoute une Rotation d'angle theta et de centre (x, y) à la fractale.

Paramètres

theta	angle de la rotation
x	abcisse du centre de la rotation
y	ordonnée du centre de la rotation

4.3.2.9 void Fractale : :generateExisting (quint32 *n*)

Fractale : :generateExisting Génere une fractale selon des valeurs par defaut.

Paramètres

n	Type de fractal par defaut n=0 : Cantor n=2 : Triangle de Sierpinski n=3 : Courbe de Koch n=4 : Flocon de Koch n=5 : Hata's tree-like set n=6 : Lévy Curve n=7 : PentaKun n=8 : Sierpinski carpet

4.3.2.10 Forme Fractale : :getFromEnsForme (int *i*) const

Fractale : :getFromEnsForme.

Paramètres

i	indice de la forme

Renvoie

Retourne la Forme d'indice i

4.3.2.11 int Fractale : :getSizeEnsAppli ( ) const

Fractale : :getSizeEnsAppli.

Renvoie

Retourne le nombre d'application définissant la fractale

4.3.2.12 int Fractale : :getSizeEnsForme ( ) const

Fractale : :getSizeEnsForme.

Renvoie

Retourne le nombre de Forme

4.3.2.13 bool Fractale : :isLikeCantor ( ) const

Fractale : :isLikeCantor.

Renvoie

Retourne si la fractale est de type Cantor

4.3.2.14 void Fractale : :RunOnce ( )

Fractale : :RunOnce Calcul la fractale au rang suivant.

4.3.2.15 void Fractale : :setLikeCantor (bool *p*)

Fractale : :setLikeCantor Attibut la valeur p à isCantor.

Paramètres

p	Nouvelle Valeur pour isCantor

4.3.3 Documentation des données membres

4.3.3.1 QList<Application> Fractale : :EnsA [private]

EnsA Stocke l'ensemble des Applications définissant la fractale.

4.3.3.2 QList<Forme> Fractale : :EnsF [private]

EnsF Stocke L'ensemble des points constituant la fractale au cours du temps.

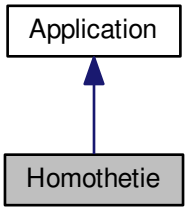
4.3.3.3 bool Fractale : :isCantor [private]

La documentation de cette classe a été générée à partir des fichiers suivants :
- fractale.h
- fractale.cpp

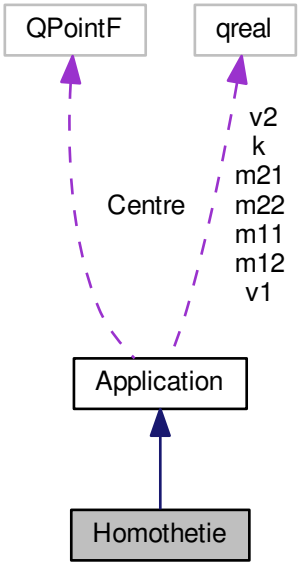
4.4 Référence de la classe Homothetie

```
#include "homothetie.h"
```

Graphe d'héritage de Homothetie :



Graphe de collaboration de Homothetie :



Fonctions membres publiques

- Homothetie ()
  Homothetie : :Homothetie.
- Homothetie (qreal K)
  Homothetie : :Homothetie Construit une Homothetie de rapport k.
- Homothetie (qreal K, QPointF C)
  Homothetie : :Homothetie Construit une Homothetie de rapport k et de centre C.
- void setHomothetie (qreal K)
  Homothetie : :setHomothetie Définie une Homothetie de rapport k.
- void setHomothetie (qreal K, QPointF C)
  Homothetie : :setHomothetie Définie une Homothetie de rapport k et de centre C.

Membres hérités additionnels

4.4.1 Documentation des constructeurs et destructeur

4.4.1.1 Homothetie : :Homothetie ( )

Homothetie : :Homothetie.

4.4.1.2 Homothetie : :Homothetie (qreal *K*)

Homothetie : :Homothetie Construit une Homothetie de rapport k.

Paramètres

K	Rapport de l'Homothetie

4.4.1.3 Homothetie : :Homothetie (qreal *K*, QPointF *C*)

Homothetie : :Homothetie Construit une Homothetie de rapport k et de centre C.

Paramètres

K	Rapport de l'Homothetie
C	Centre de l'Homothetie

4.4.2 Documentation des fonctions membres

4.4.2.1 void Homothetie : :setHomothetie (qreal *K*)

Homothetie : :setHomothetie Définie une Homothetie de rapport k.

Paramètres

K	Rapport de l'Homothetie

4.4.2.2 void Homothetie : :setHomothetie (qreal *K*, QPointF *C*)

Homothetie : :setHomothetie Définie une Homothetie de rapport k et de centre C.

Paramètres

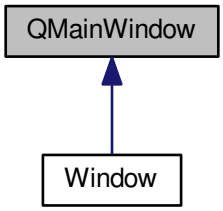
K	Rapport de l'Homothetie
C	Centre de l'Homothetie

La documentation de cette classe a été générée à partir des fichiers suivants :

— homothetie.h
— homothetie.cpp

4.5 Référence de la classe QMainWindow

Graphe d'héritage de QMainWindow :

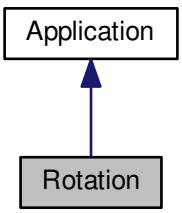


La documentation de cette classe a été générée à partir du fichier suivant :
— window.h

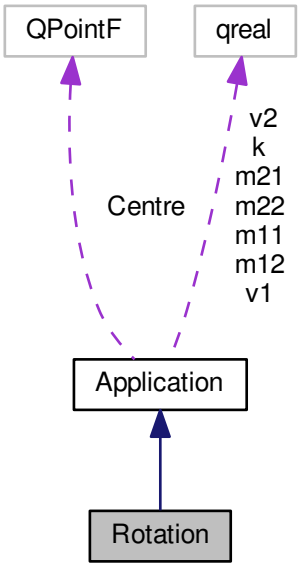
4.6 Référence de la classe Rotation

```
#include "rotation.h"
```

Graphe d'héritage de Rotation :



Graphe de collaboration de Rotation :



Fonctions membres publiques

- Rotation ()
  Rotation : :Rotation.
- Rotation (qreal theta)
- Rotation (qreal theta, QPointF C)
  Rotation : :Rotation.
- void setRotation (qreal theta)
  Rotation : :setRotation.
- void setRotation (qreal theta, QPointF C)
  Rotation : :setRotation.

Membres hérités additionnels

4.6.1 Documentation des constructeurs et destructeur

4.6.1.1 Rotation : :Rotation ( )

Rotation : :Rotation.

4.6.1.2 Rotation : :Rotation (qreal *theta*)

4.6.1.3 Rotation : :Rotation (qreal *theta,* QPointF *C*)

Rotation : :Rotation.

Paramètres

theta	
C	

4.6.2 Documentation des fonctions membres

4.6.2.1 void Rotation : :setRotation (qreal *theta*)

Rotation : :setRotation.

Paramètres

theta	

4.6.2.2 void Rotation : :setRotation (qreal *theta,* QPointF *C*)

Rotation : :setRotation.

Paramètres

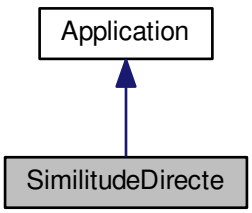
theta	
C	

La documentation de cette classe a été générée à partir des fichiers suivants :
- rotation.h
- rotation.cpp

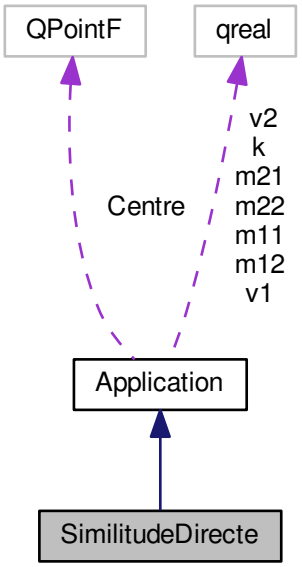
4.7 Référence de la classe SimilitudeDirecte

```
#include "similitudedirecte.h"
```

Graphe d'héritage de SimilitudeDirecte :



Graphe de collaboration de SimilitudeDirecte :



Fonctions membres publiques

- SimilitudeDirecte ()
  SimilitudeDirecte : :SimilitudeDirecte.
- SimilitudeDirecte (qreal K, qreal theta, QPointF C)
  SimilitudeDirecte : :SimilitudeDirecte.
- SimilitudeDirecte (qreal K, qreal theta, QPointF C, QPointF P)
  SimilitudeDirecte : :SimilitudeDirecte.
- void setSimilitudeDirecte (qreal K, qreal theta, QPointF C)
  SimilitudeDirecte : :setSimilitudeDirecte.
- void setSimilitudeDirecte (qreal K, qreal theta, QPointF C, QPointF P)
  SimilitudeDirecte : :setSimilitudeDirecte.
- void setTheta (qreal theta)
  SimilitudeDirecte : :setTheta.

Membres hérités additionnels

4.7.1 Documentation des constructeurs et destructeur

4.7.1.1 SimilitudeDirecte : :SimilitudeDirecte ( )

SimilitudeDirecte : :SimilitudeDirecte.

4.7.1.2 SimilitudeDirecte : :SimilitudeDirecte (qreal *K,* qreal *theta,* QPointF *C*)

SimilitudeDirecte : :SimilitudeDirecte.

Paramètres

K	
theta	
C	

4.7.1.3 SimilitudeDirecte : :SimilitudeDirecte (qreal *K,* qreal *theta,* QPointF *C,* QPointF *P*)

SimilitudeDirecte : :SimilitudeDirecte.

Paramètres

K	
theta	
C	
P	

4.7.2 Documentation des fonctions membres

4.7.2.1 void SimilitudeDirecte : :setSimilitudeDirecte (qreal *K,* qreal *theta,* QPointF *C*)

SimilitudeDirecte : :setSimilitudeDirecte.

Paramètres

K	
theta	
C	

4.7.2.2 void SimilitudeDirecte : :setSimilitudeDirecte (qreal *K,* qreal *theta,* QPointF *C,* QPointF *P*)

SimilitudeDirecte : :setSimilitudeDirecte.

Paramètres

K	
theta	
C	
P	

4.7.2.3 void SimilitudeDirecte : :setTheta (qreal *theta*)

SimilitudeDirecte : :setTheta.

Paramètres

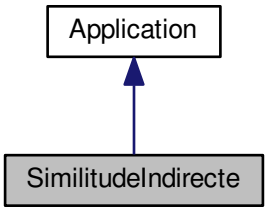
theta	

La documentation de cette classe a été générée à partir des fichiers suivants :
- similitudedirecte.h
- similitudedirecte.cpp

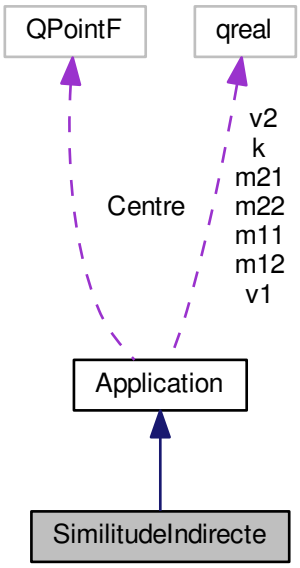
4.8 Référence de la classe SimilitudeIndirecte

```
#include "similitudeindirecte.h"
```

Graphe d'héritage de SimilitudeIndirecte :



Graphe de collaboration de SimilitudeIndirecte :



Fonctions membres publiques

- SimilitudeIndirecte ()
  SimilitudeIndirecte : :SimilitudeIndirecte.
- SimilitudeIndirecte (qreal K, qreal theta, QPointF P)
  SimilitudeIndirecte : :SimilitudeIndirecte.
- SimilitudeIndirecte (qreal K, qreal theta, QPointF P, QPointF C)
  SimilitudeIndirecte : :SimilitudeIndirecte.
- void setSimilitudeIndirecte (qreal K, qreal theta, QPointF P)
  SimilitudeIndirecte : :setSimilitudeIndirecte.
- void setSimilitudeIndirecte (qreal K, qreal theta, QPointF P, QPointF C)
  SimilitudeIndirecte : :setSimilitudeIndirecte.

Membres hérités additionnels

4.8.1 Documentation des constructeurs et destructeur

4.8.1.1 SimilitudeIndirecte : :SimilitudeIndirecte ( )

SimilitudeIndirecte : :SimilitudeIndirecte.

4.8.1.2 SimilitudeIndirecte : :SimilitudeIndirecte (qreal *K,* qreal *theta,* QPointF *P*)

SimilitudeIndirecte : :SimilitudeIndirecte.

Paramètres

K	
theta	
P	

4.8.1.3 SimilitudeIndirecte : :SimilitudeIndirecte (qreal *K,* qreal *theta,* QPointF *P,* QPointF *C*)

SimilitudeIndirecte : :SimilitudeIndirecte.

Paramètres

K	
theta	
P	
C	

4.8.2 Documentation des fonctions membres

4.8.2.1 void SimilitudeIndirecte : :setSimilitudeIndirecte (qreal *K,* qreal *theta,* QPointF *P*)

SimilitudeIndirecte : :setSimilitudeIndirecte.

Paramètres

K	
theta	
P	

4.8.2.2 void SimilitudeIndirecte : :setSimilitudeIndirecte (qreal *K,* qreal *theta,* QPointF *P,* QPointF *C*)

SimilitudeIndirecte : :setSimilitudeIndirecte.

Paramètres

K	
theta	
P	
C	

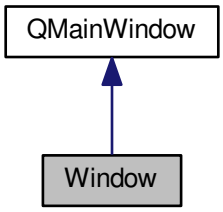
La documentation de cette classe a été générée à partir des fichiers suivants :

— similitudeindirecte.h
— similitudeindirecte.cpp

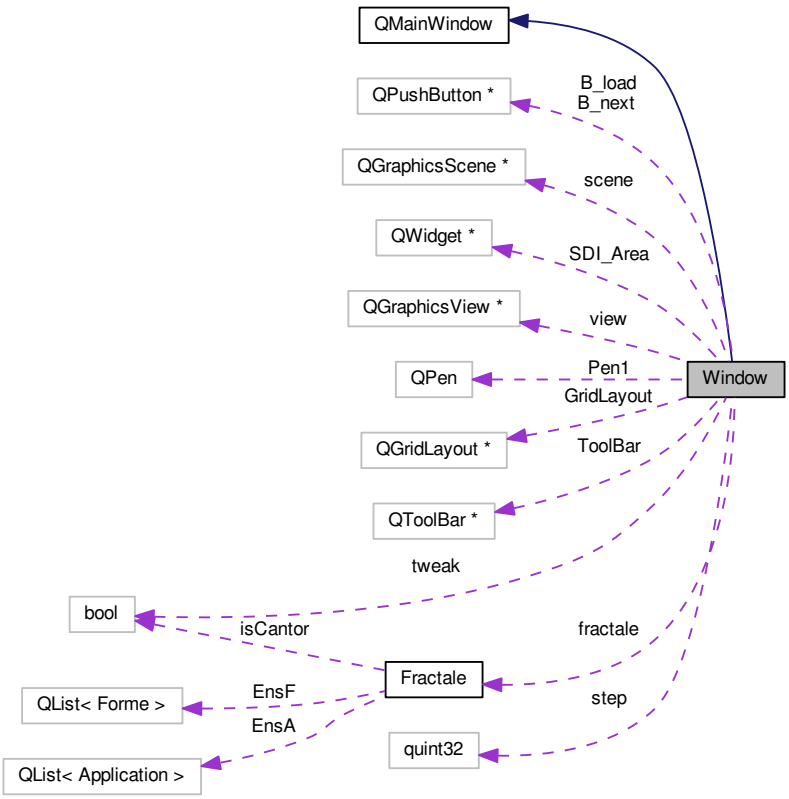
4.9 Référence de la classe Window

`#include "window.h"`

Graphe d'héritage de Window :



Graphe de collaboration de Window :



Connecteurs publics

— void load ()

Window : :load *Affiche une fenetre permettant de choisir une fractale par defaut.*
— void refreshView ()
Window : :refreshView *Dessine la fractale.*

Fonctions membres publiques

— Window ()
Window : :Window *Constructeur par defaut de la classe Window*
Crée les differents objets nécessaire.

Attributs publics

— bool tweak
— Fractale ∗ fractale

Fonctions membres protégées

— virtual bool eventFilter (QObject ∗obj, QEvent ∗event)
Window : :eventFilter *Filtre d'évenement pour capturer les evenemts spécifique.*

Fonctions membres privées

— void Zoom (QGraphicsSceneWheelEvent ∗event)
Window : :Zoom *Est Appelé quand l'évenement QEvent : :GraphicsSceneWheel est capturé*
— void loadExistingFractal (quint64)
— void refreshViewSpecialCantor ()
Window : :refreshViewSpecialCantor *Dessine de la fractale de typeCantor.*
— void refreshViewColor ()
Window : :refreshViewColor *Dessine la fractale en colorant l'intérieur de la figure.*

Attributs privés

— QWidget ∗ SDI_Area
— QGridLayout ∗ GridLayout
— QToolBar ∗ ToolBar
— QGraphicsView ∗ view
— QGraphicsScene ∗ scene
— QPen Pen1
— QPushButton ∗ B_next
— QPushButton ∗ B_load
— quint32 step

4.9.1 Documentation des constructeurs et destructeur

4.9.1.1 Window : :Window ( )

Window : :Window Constructeur par defaut de la classe Window
Crée les differents objets nécessaire.

4.9.2 Documentation des fonctions membres

4.9.2.1 bool Window : :eventFilter (QObject ∗ *obj,* QEvent ∗ *event*) [protected], [virtual]

Window : :eventFilter Filtre d'évenement pour capturer les evenemts spécifique.

Paramètres

obj	
event	

Renvoie

4.9.2.2 void Window : :load ( ) [slot]

Window : :load Affiche une fenetre permettant de choisir une fractale par defaut.

4.9.2.3 void Window : :loadExistingFractal (quint64) [private]

4.9.2.4 void Window : :refreshView ( ) [slot]

Window : :refreshView Dessine la fractale.

4.9.2.5 void Window : :refreshViewColor ( ) [private]

Window : :refreshViewColor Dessine la fractale en colorant l'intérieur de la figure.

4.9.2.6 void Window : :refreshViewSpecialCantor ( ) [private]

Window : :refreshViewSpecialCantor Dessine de la fractale de typeCantor.

4.9.2.7 void Window : :Zoom (QGraphicsSceneWheelEvent * *event*) [private]

Window : :Zoom Est Appelé quand l'évenement QEvent : :GraphicsSceneWheel est capturé

Paramètres

event	

4.9.3 Documentation des données membres

4.9.3.1 QPushButton* Window : :B_load [private]

4.9.3.2 QPushButton* Window : :B_next [private]

4.9.3.3 Fractale* Window : :fractale

4.9.3.4 QGridLayout* Window : :GridLayout [private]

4.9.3.5 QPen Window : :Pen1 [private]

4.9.3.6 QGraphicsScene* Window : :scene [private]

4.9.3.7 QWidget* Window : :SDl_Area [private]

4.9.3.8 quint32 Window : :step [private]

4.9.3.9 QToolBar* Window : :ToolBar [private]

4.9.3.10 bool Window : :tweak

4.9.3.11 QGraphicsView* Window : :view [private]

La documentation de cette classe a été générée à partir des fichiers suivants :
— window.h
— window.cpp

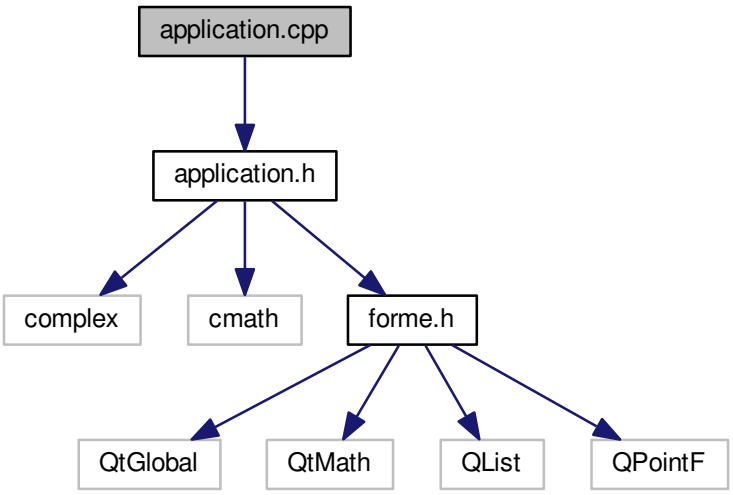
Chapitre 5

Documentation des fichiers

5.1 Référence du fichier application.cpp

```
#include "application.h"
```

Graphe des dépendances par inclusion de application.cpp :



Fonctions

— bool operator== (Application const &A, Application const &B)
operator ==

5.1.1 Documentation des fonctions

5.1.1.1 bool operator== (Application const & *A*, Application const & *B*)

operator ==

Paramètres

A	
B	

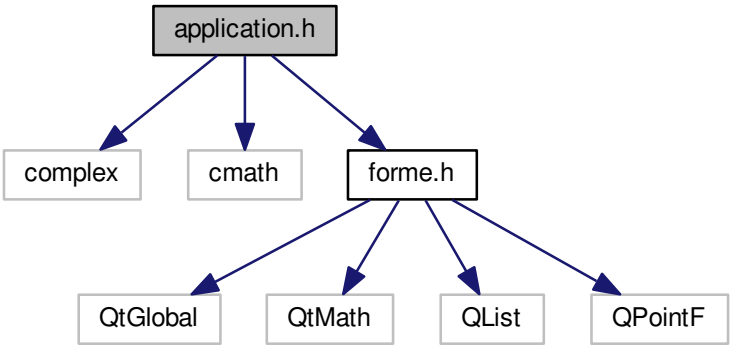
Renvoie

true si A==B, false si A !=B

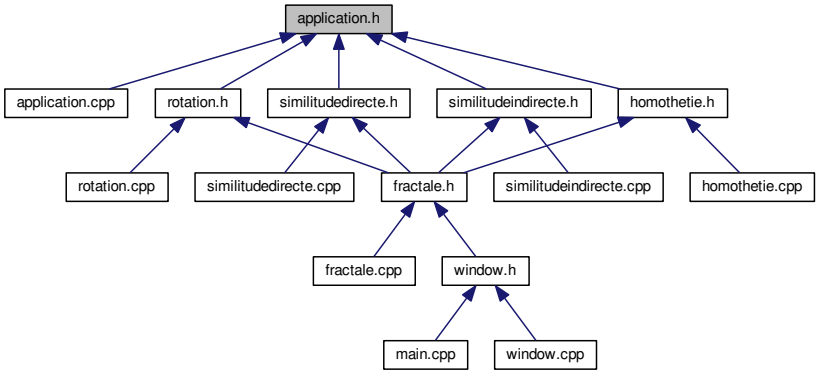
5.2 Référence du fichier application.h

```
#include <complex>
#include <cmath>
#include "forme.h"
```

Graphe des dépendances par inclusion de application.h :



Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :



Classes

— class Application

Fonctions

— bool operator== (Application const &A, Application const &B)
operator ==

5.2.1 Documentation des fonctions

5.2.1.1 bool operator== (Application const & *A*, Application const & *B*)

operator ==

Paramètres

A	
B	

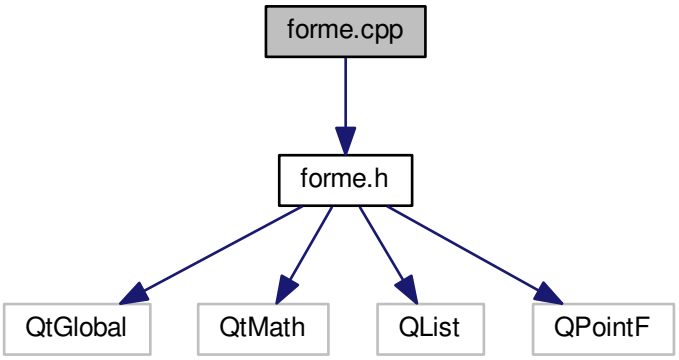
Renvoie

true si A==B, false si A !=B

5.3 Référence du fichier forme.cpp

```
#include "forme.h"
```

Graphe des dépendances par inclusion de forme.cpp :



Fonctions

— bool operator== (Forme const &A, Forme const &B)
operator == Teste si les deux formes sont egales

5.3.1 Documentation des fonctions

5.3.1.1 bool operator== (Forme const & *A*, Forme const & *B*)

operator == Teste si les deux formes sont egales

Paramètres

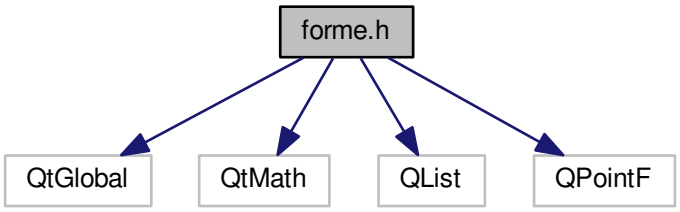
A	Forme 1
B	Forme 2

Renvoie

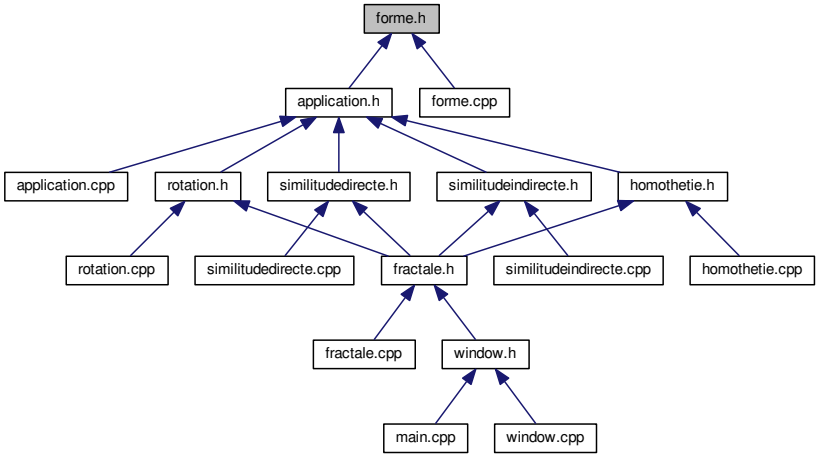
5.4 Référence du fichier forme.h

```
#include <QtGlobal>
#include <QtMath>
#include <QList>
#include <QPointF>
```

Graphe des dépendances par inclusion de forme.h :



Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :



Classes

— class Forme

Fonctions

— bool operator== (Forme const &A, Forme const &B)
operator == Teste si les deux formes sont egales

5.4.1 Documentation des fonctions

5.4.1.1 bool operator== (Forme const & *A*, Forme const & *B*)

operator == Teste si les deux formes sont egales

Paramètres

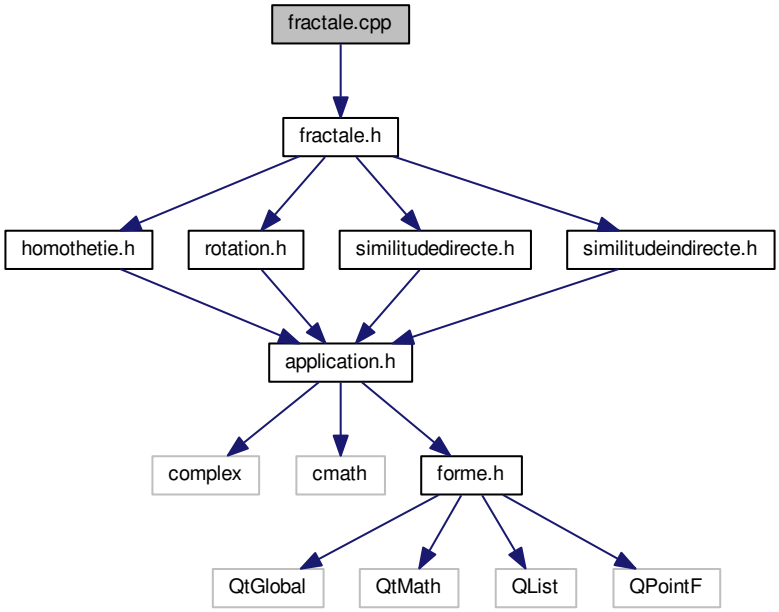
A	Forme 1
B	Forme 2

Renvoie

5.5 Référence du fichier fractale.cpp

```
#include "fractale.h"
```

Graphe des dépendances par inclusion de fractale.cpp :

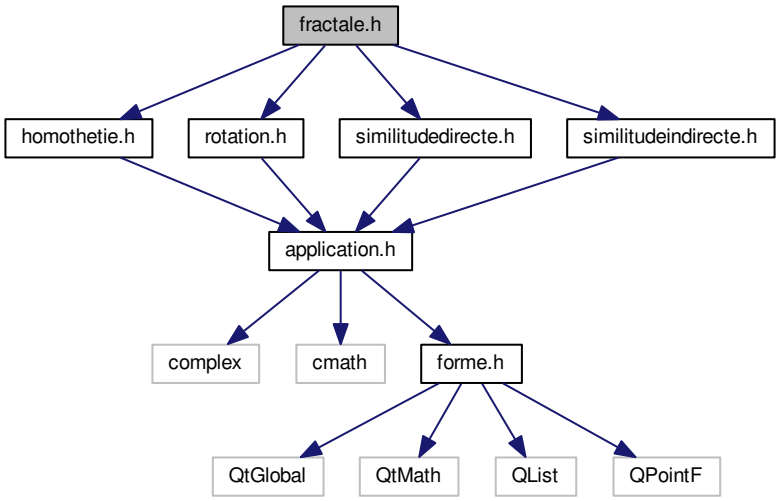


5.6 Référence du fichier fractale.h

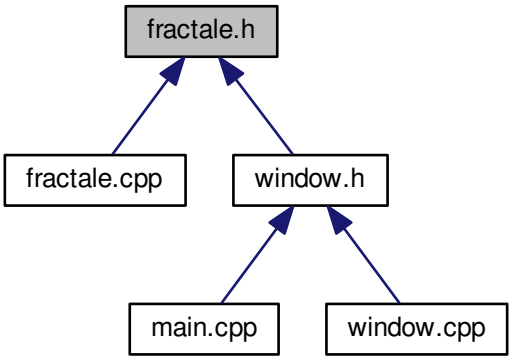
```
#include "homothetie.h"
```

```
#include "rotation.h"
#include "similitudedirecte.h"
#include "similitudeindirecte.h"
```

Graphe des dépendances par inclusion de fractale.h :



Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :



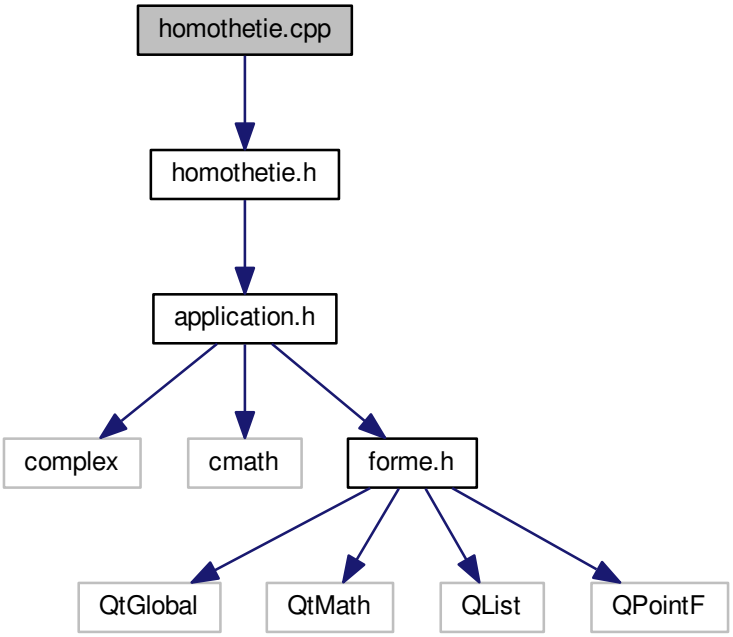
Classes

— class Fractale

5.7 Référence du fichier homothetie.cpp

```
#include "homothetie.h"
```

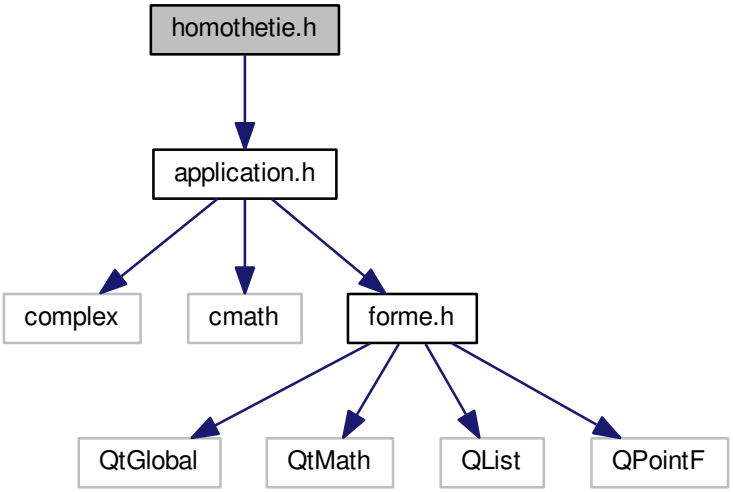
Graphe des dépendances par inclusion de homothetie.cpp :



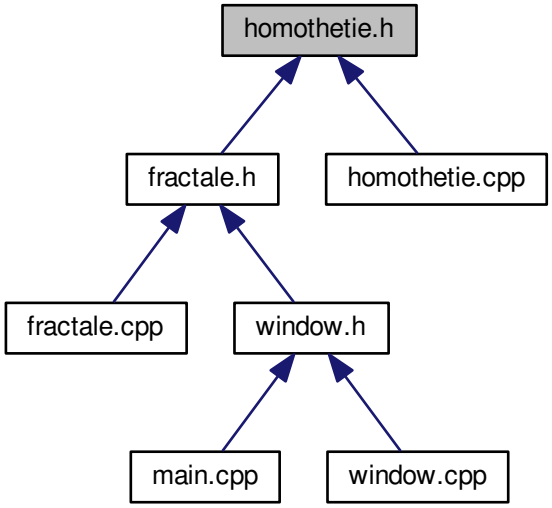
5.8 Référence du fichier homothetie.h

```
#include "application.h"
```

Graphe des dépendances par inclusion de homothetie.h :



Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :



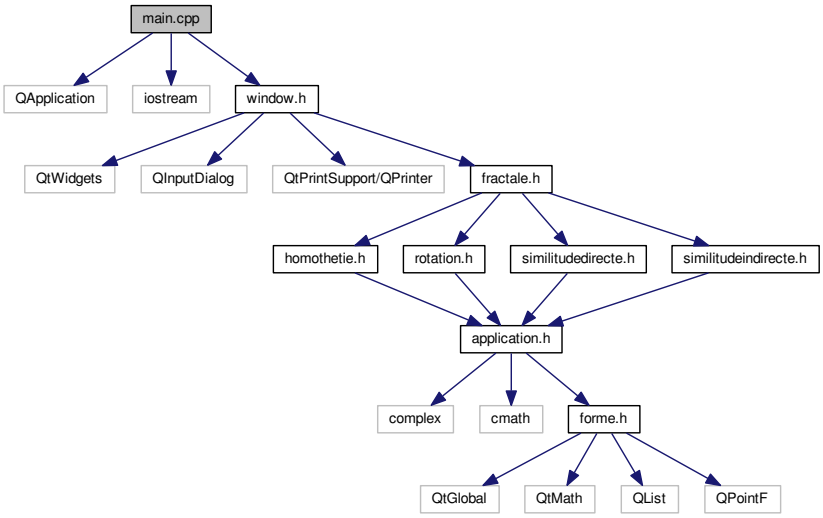
Classes

— class Homothetie

5.9 Référence du fichier main.cpp

```
#include <QApplication>
#include <iostream>
#include "window.h"
```

Graphe des dépendances par inclusion de main.cpp :



Fonctions

— int main (int argc, char ∗argv[ ])

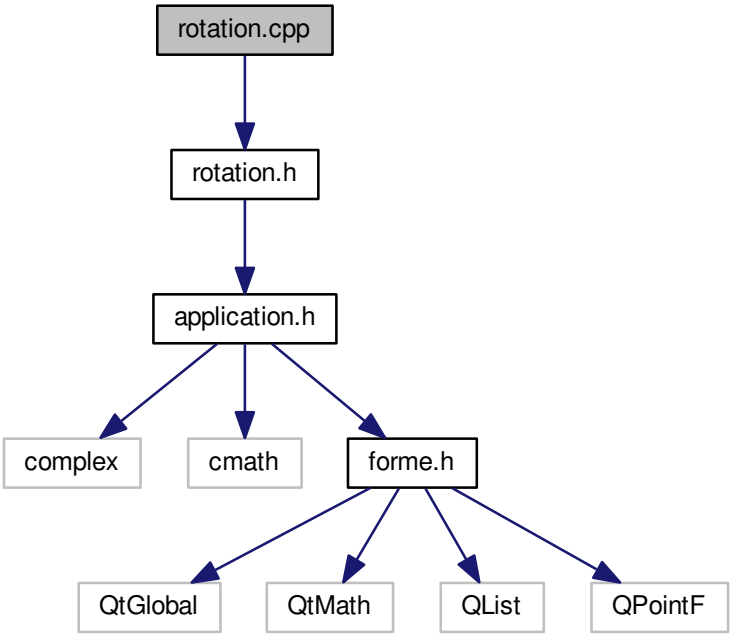
5.9.1 Documentation des fonctions

5.9.1.1 int main (int *argc,* char ∗ *argv[ ]*)

5.10 Référence du fichier rotation.cpp

```
#include "rotation.h"
```

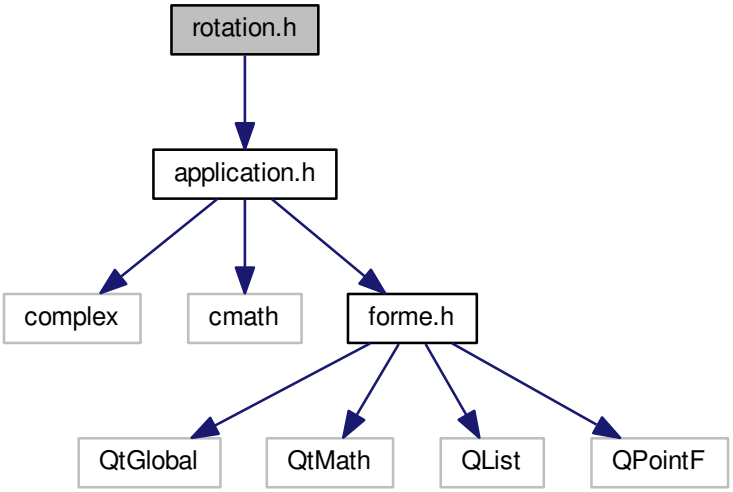
Graphe des dépendances par inclusion de rotation.cpp :



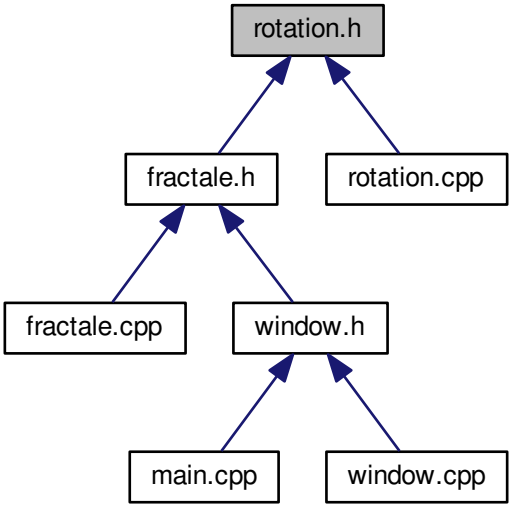
5.11 Référence du fichier rotation.h

```
#include "application.h"
```

Graphe des dépendances par inclusion de rotation.h :



Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :



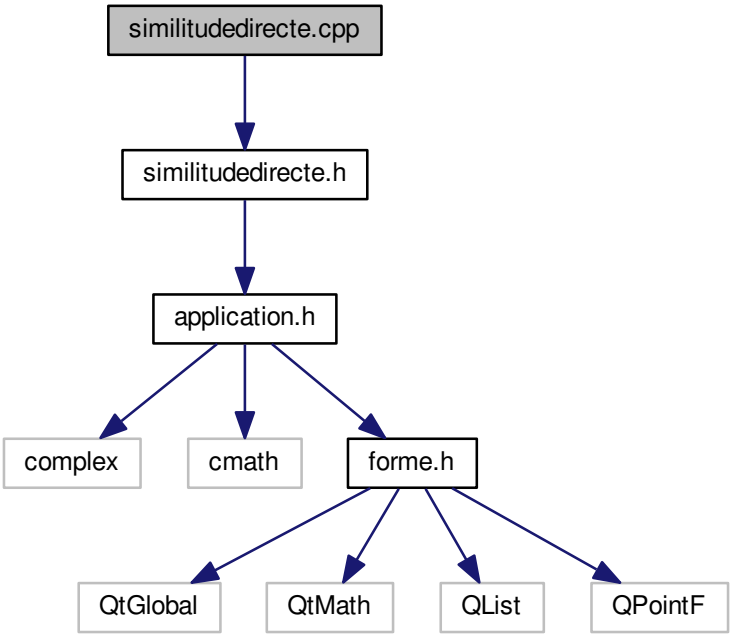
Classes

— class Rotation

5.12 Référence du fichier similitudedirecte.cpp

```
#include "similitudedirecte.h"
```

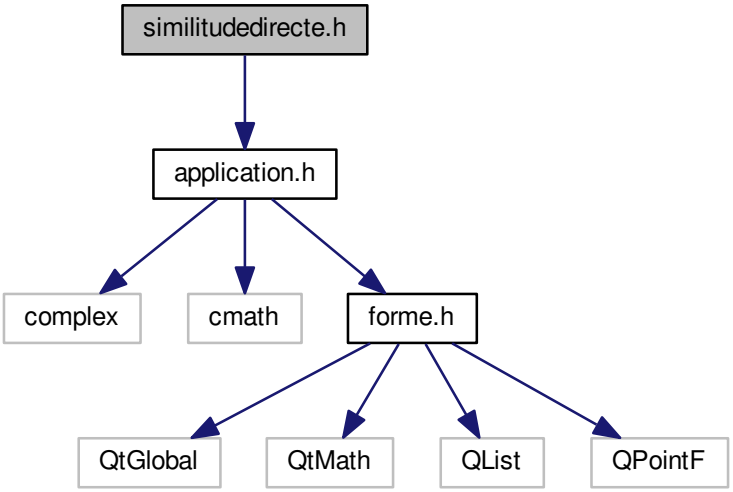
Graphe des dépendances par inclusion de similitudedirecte.cpp :



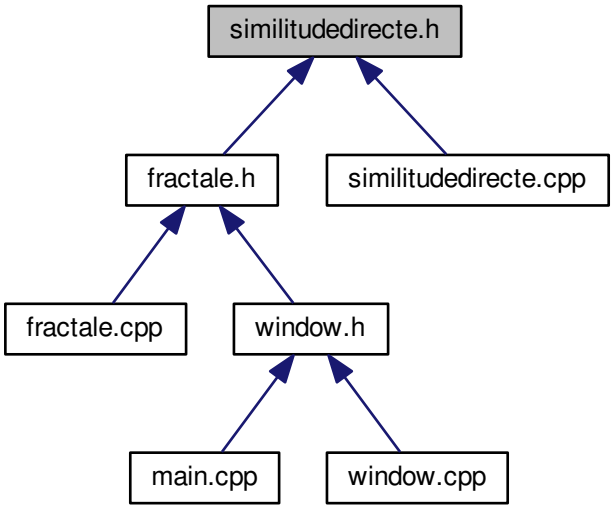
5.13 Référence du fichier similitudedirecte.h

```
#include "application.h"
```

Graphe des dépendances par inclusion de similitudedirecte.h :



Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :



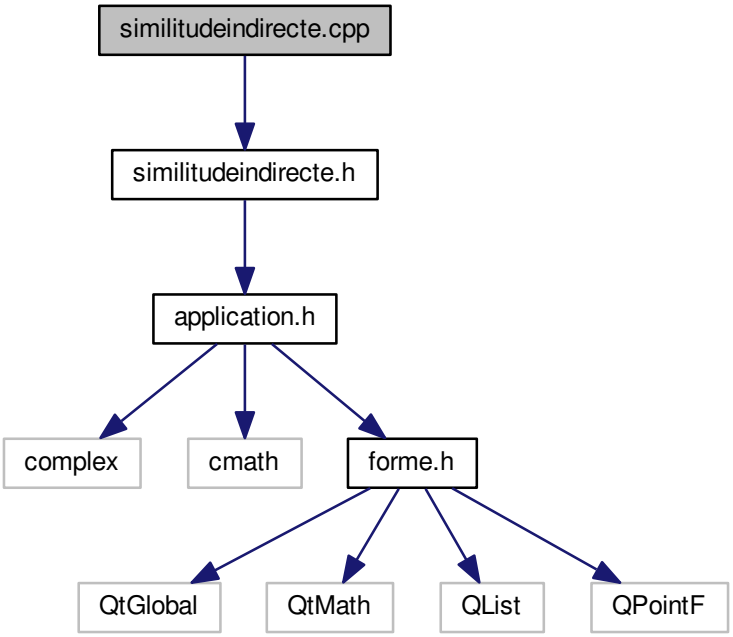
Classes

— class SimilitudeDirecte

5.14 Référence du fichier similitudeindirecte.cpp

```
#include "similitudeindirecte.h"
```

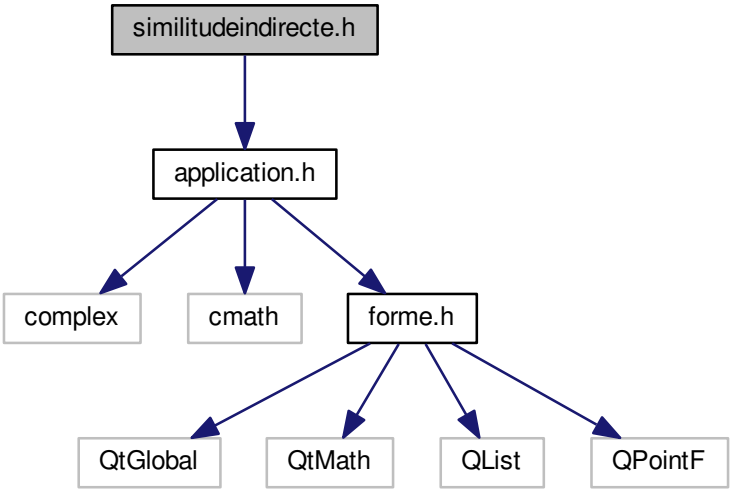
Graphe des dépendances par inclusion de similitudeindirecte.cpp :



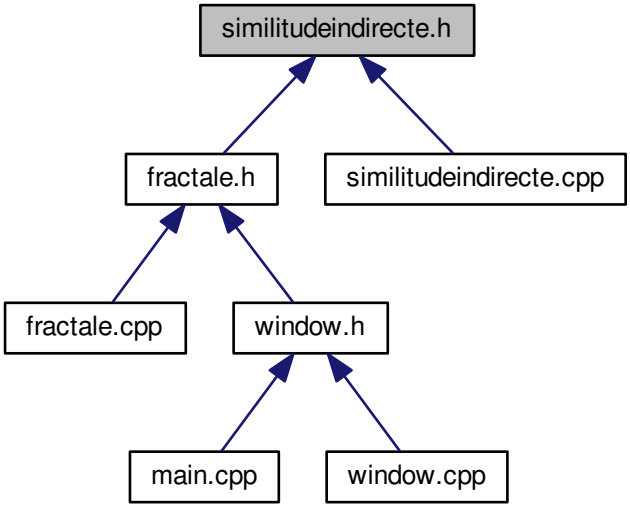
5.15 Référence du fichier similitudeindirecte.h

```
#include "application.h"
```

Graphe des dépendances par inclusion de similitudeindirecte.h :



Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :



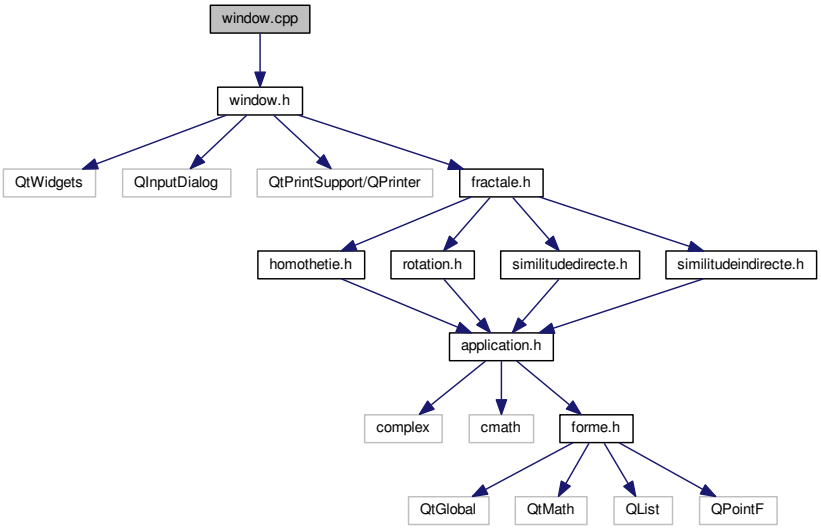
Classes

— class SimilitudeIndirecte

5.16 Référence du fichier window.cpp

```
#include "window.h"
```

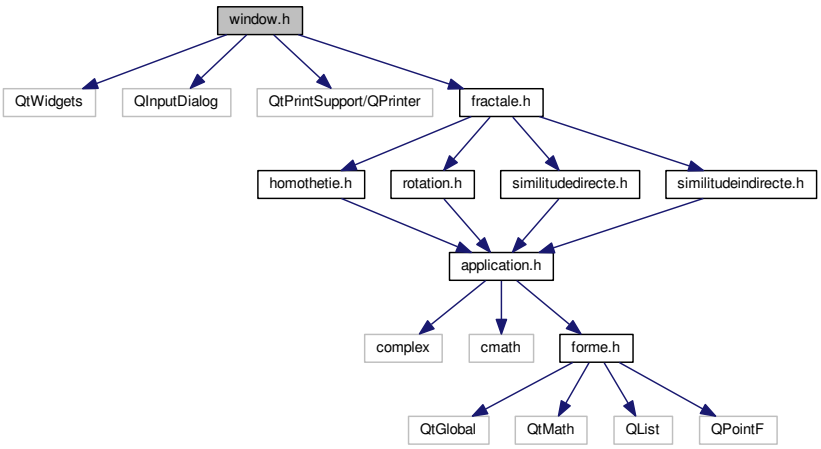
Graphe des dépendances par inclusion de window.cpp :



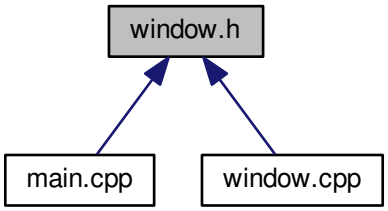
5.17 Référence du fichier window.h

```
#include <QtWidgets>
#include <QInputDialog>
#include <QtPrintSupport/QPrinter>
#include "fractale.h"
```

Graphe des dépendances par inclusion de window.h :



Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :



Classes

— class Window

Index