

Fractale

Généré par Doxygen 1.8.9.1

Vendredi 11 Mars 2016 14 :58 :50

Table des matières

1	Index hiérarchique	1
1.1	Hiérarchie des classes	1
2	Index des classes	3
2.1	Liste des classes	3
3	Index des fichiers	5
3.1	Liste des fichiers	5
4	Documentation des classes	7
4.1	Référence de la classe Application	7
4.1.1	Documentation des constructeurs et destructeur	9
4.1.1.1	Application	9
4.1.2	Documentation des fonctions membres	9
4.1.2.1	DoForEns	9
4.1.2.2	DoForForme	9
4.1.2.3	DoForQPointF	9
4.1.2.4	getCentre	10
4.1.2.5	getK	10
4.1.2.6	getm11	10
4.1.2.7	getm12	10
4.1.2.8	getm21	10
4.1.2.9	getm22	10
4.1.2.10	getTheta	11
4.1.2.11	getv1	11
4.1.2.12	getv2	11
4.1.2.13	isHomothetie	11
4.1.2.14	isRotation	11
4.1.2.15	setCentre	11
4.1.2.16	setK	11
4.1.2.17	setm11	12
4.1.2.18	setm12	12

4.1.2.19	setm21	12
4.1.2.20	setm22	12
4.1.2.21	setv1	12
4.1.2.22	setv2	12
4.1.3	Documentation des données membres	13
4.1.3.1	Centre	13
4.1.3.2	k	13
4.1.3.3	m11	13
4.1.3.4	m12	13
4.1.3.5	m21	13
4.1.3.6	m22	13
4.1.3.7	v1	13
4.1.3.8	v2	13
4.2	Référence de la classe Forme	14
4.2.1	Documentation des constructeurs et destructeur	14
4.2.1.1	Forme	14
4.2.2	Documentation des fonctions membres	14
4.2.2.1	AddPoint	14
4.2.2.2	generateExisting	15
4.2.2.3	GetPoint	15
4.2.2.4	GetSize	15
4.2.3	Documentation des données membres	15
4.2.3.1	L	15
4.3	Référence de la classe Fractale	15
4.3.1	Documentation des constructeurs et destructeur	16
4.3.1.1	Fractale	17
4.3.2	Documentation des fonctions membres	17
4.3.2.1	AddApplication	17
4.3.2.2	AddForme	17
4.3.2.3	AddHomothetie	17
4.3.2.4	AddHomothetie	17
4.3.2.5	AddHomothetie	17
4.3.2.6	AddRotation	17
4.3.2.7	AddRotation	18
4.3.2.8	AddRotation	18
4.3.2.9	generateExisting	18
4.3.2.10	getFromEnsForme	18
4.3.2.11	getSizeEnsAppli	18
4.3.2.12	getSizeEnsForme	19
4.3.2.13	isLikeCantor	19

4.3.2.14	RunOnce	19
4.3.2.15	setLikeCantor	19
4.3.3	Documentation des données membres	19
4.3.3.1	EnsA	19
4.3.3.2	EnsF	19
4.3.3.3	isCantor	19
4.4	Référence de la classe Homothetie	19
4.4.1	Documentation des constructeurs et destructeur	21
4.4.1.1	Homothetie	21
4.4.1.2	Homothetie	21
4.4.1.3	Homothetie	21
4.4.2	Documentation des fonctions membres	21
4.4.2.1	setHomothetie	21
4.4.2.2	setHomothetie	21
4.5	Référence de la classe QMainWindow	22
4.6	Référence de la classe Rotation	22
4.6.1	Documentation des constructeurs et destructeur	23
4.6.1.1	Rotation	23
4.6.1.2	Rotation	23
4.6.1.3	Rotation	23
4.6.2	Documentation des fonctions membres	24
4.6.2.1	setRotation	24
4.6.2.2	setRotation	24
4.7	Référence de la classe SimilitudeDirecte	24
4.7.1	Documentation des constructeurs et destructeur	25
4.7.1.1	SimilitudeDirecte	25
4.7.1.2	SimilitudeDirecte	25
4.7.1.3	SimilitudeDirecte	26
4.7.2	Documentation des fonctions membres	26
4.7.2.1	setSimilitudeDirecte	26
4.7.2.2	setSimilitudeDirecte	26
4.7.2.3	setTheta	26
4.8	Référence de la classe SimilitudeIndirecte	26
4.8.1	Documentation des constructeurs et destructeur	28
4.8.1.1	SimilitudeIndirecte	28
4.8.1.2	SimilitudeIndirecte	28
4.8.1.3	SimilitudeIndirecte	28
4.8.2	Documentation des fonctions membres	28
4.8.2.1	setSimilitudeIndirecte	28
4.8.2.2	setSimilitudeIndirecte	28

4.9	Référence de la classe Window	29
4.9.1	Documentation des constructeurs et destructeur	30
4.9.1.1	Window	30
4.9.2	Documentation des fonctions membres	30
4.9.2.1	eventFilter	30
4.9.2.2	load	31
4.9.2.3	loadExistingFractal	31
4.9.2.4	refreshView	31
4.9.2.5	refreshViewColor	31
4.9.2.6	refreshViewSpecialCantor	31
4.9.2.7	Zoom	31
4.9.3	Documentation des données membres	31
4.9.3.1	B_load	31
4.9.3.2	B_next	31
4.9.3.3	fractale	31
4.9.3.4	GridLayout	31
4.9.3.5	Pen1	31
4.9.3.6	scene	31
4.9.3.7	SDI_Area	31
4.9.3.8	step	31
4.9.3.9	ToolBar	32
4.9.3.10	tweak	32
4.9.3.11	view	32
5	Documentation des fichiers	33
5.1	Référence du fichier application.cpp	33
5.1.1	Documentation des fonctions	33
5.1.1.1	operator==	33
5.2	Référence du fichier application.h	34
5.2.1	Documentation des fonctions	35
5.2.1.1	operator==	35
5.3	Référence du fichier forme.cpp	35
5.3.1	Documentation des fonctions	35
5.3.1.1	operator==	35
5.4	Référence du fichier forme.h	36
5.4.1	Documentation des fonctions	37
5.4.1.1	operator==	37
5.5	Référence du fichier fractale.cpp	37
5.6	Référence du fichier fractale.h	37
5.7	Référence du fichier homothetie.cpp	38

5.8	Référence du fichier homothetie.h	39
5.9	Référence du fichier main.cpp	41
5.9.1	Documentation des fonctions	41
5.9.1.1	main	41
5.10	Référence du fichier rotation.cpp	41
5.11	Référence du fichier rotation.h	42
5.12	Référence du fichier similitudedirecte.cpp	44
5.13	Référence du fichier similitudedirecte.h	44
5.14	Référence du fichier similitudeindirecte.cpp	46
5.15	Référence du fichier similitudeindirecte.h	46
5.16	Référence du fichier window.cpp	48
5.17	Référence du fichier window.h	48
	Index	51

Chapitre 1

Index hiérarchique

1.1 Hiérarchie des classes

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

Application	7
Homothetie	19
Rotation	22
SimilitudeDirecte	24
SimilitudeIndirecte	26
Forme	14
Fractale	15
QMainWindow	22
Window	29

Chapitre 2

Index des classes

2.1 Liste des classes

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

Application	7
Forme	14
Fractale	15
Homothetie	19
QMainWindow	22
Rotation	22
SimilitudeDirecte	24
SimilitudeIndirecte	26
Window	29

Chapitre 3

Index des fichiers

3.1 Liste des fichiers

Liste de tous les fichiers avec une brève description :

application.cpp	33
application.h	34
forme.cpp	35
forme.h	36
fractale.cpp	37
fractale.h	37
homothetie.cpp	38
homothetie.h	39
main.cpp	41
rotation.cpp	41
rotation.h	42
similitudedirecte.cpp	44
similitudedirecte.h	44
similitudeindirecte.cpp	46
similitudeindirecte.h	46
window.cpp	48
window.h	48

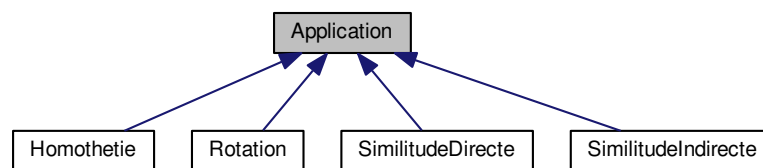
Chapitre 4

Documentation des classes

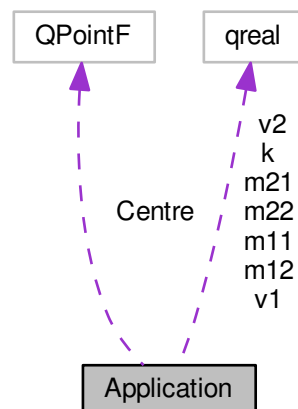
4.1 Référence de la classe Application

```
#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

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

- qreal [m12](#)
Décrit la matrice de rotation

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

- qreal [m21](#)
Décrit la matrice de rotation

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

— qreal [m22](#)

Décrit la matrice de rotation

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

— qreal [v1](#)

v1 Parametre du vecteur de translation V

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

— qreal [v2](#)

v2 Parametre du vecteur de translation V

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

— 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

<i>EnsForme</i>	
-----------------	--

Renvoie

Image de l'application sur chaque des forme

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

[Application : :DoForForme](#).

Paramètres

<i>F</i>	Forme
----------	-----------------------

Renvoie

Image de l'application sur la forme

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

[Application : :DoForQPointF](#).

Paramètres

<i>P</i>	
----------	--

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

<i>P</i>	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

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

.

4.1.3.4 `qreal Application::m12` [protected]

Décrit la matrice de rotation

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

.

4.1.3.5 `qreal Application::m21` [protected]

Décrit la matrice de rotation

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

.

4.1.3.6 `qreal Application::m22` [protected]

Décrit la matrice de rotation

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

.

4.1.3.7 `qreal Application::v1` [protected]

v1 Parametre du vecteur de translation V

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

4.1.3.8 `qreal Application::v2` [protected]

v2 Parametre du vecteur de translation V

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

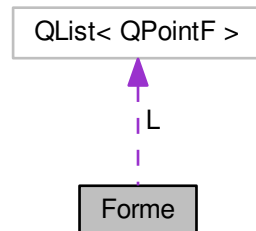
La documentation de cette classe a été générée à partir des fichiers suivants :

- [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 Génère 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

<i>P</i>	QPointF
----------	---------

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

Forme : :generateExisting Génère une forme par défaut

n=0 : Segment

n=1 : Triangle.

Paramètres

<i>n</i>	
----------	--

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

Forme : :GetPoint.

Paramètres

<i>i</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

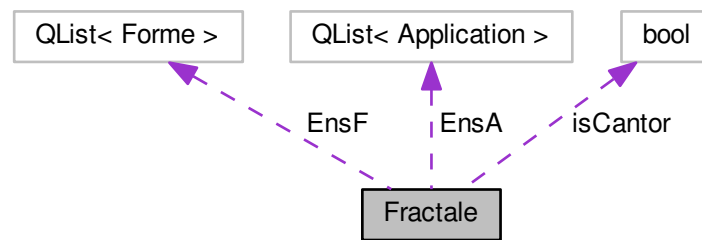
La documentation de cette classe a été générée à partir des fichiers suivants :

- [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 Attribut 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 défaut.

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 Application 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

<i>theta</i>	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

<i>theta</i>	Angle de la rotation
<i>Centre</i>	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

<i>theta</i>	angle de la rotation
<i>x</i>	abscisse du centre de la rotation
<i>y</i>	ordonnée du centre de la rotation

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

[Fractale : :generateExisting](#) Génère une fractale selon des valeurs par défaut.

Paramètres

<i>n</i>	Type de fractal par défaut 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>i</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](#) Attribut la valeur p à isCantor.

Paramètres

<i>p</i>	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]`

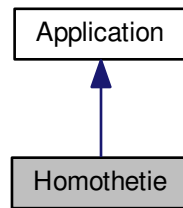
La documentation de cette classe a été générée à partir des fichiers suivants :

- [fractale.h](#)
- [fractale.cpp](#)

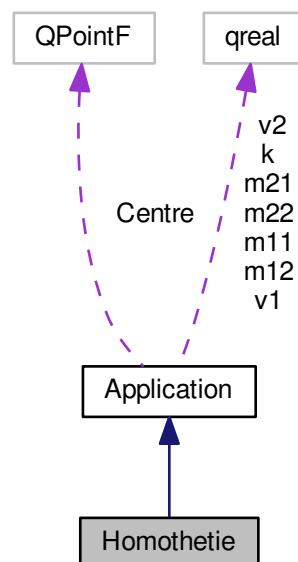
4.4 Référence de la classe Homothetie

```
#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 Définit une *Homothetie* de rapport *k*.
- `void setHomothetie (qreal K, QPointF C)`
Homothetie : :setHomothetie Définit 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](#) Définit une [Homothetie](#) de rapport k.

Paramètres

K	Rapport de l' Homothetie
-----	--

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

[Homothetie](#) : [setHomothetie](#) Définit une [Homothetie](#) de rapport k et de centre C.

Paramètres

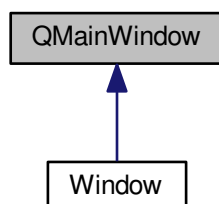
K	Rapport de l' Homothetie
C	Centre de l' Homothetie

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 :



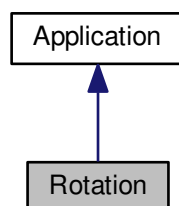
La documentation de cette classe a été générée à partir du fichier suivant :

— [window.h](#)

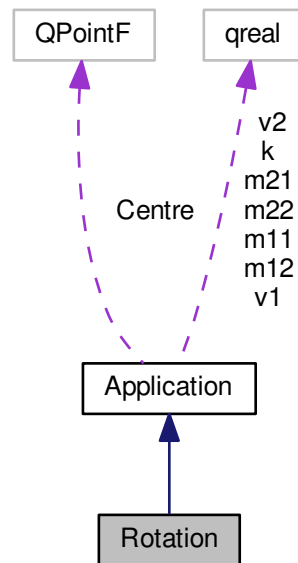
4.6 Référence de la classe Rotation

```
#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

<i>theta</i>	
<i>C</i>	

4.6.2 Documentation des fonctions membres

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

[Rotation : :setRotation.](#)

Paramètres

<i>theta</i>	
--------------	--

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

[Rotation : :setRotation.](#)

Paramètres

<i>theta</i>	
<i>C</i>	

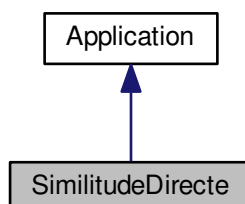
La documentation de cette classe a été générée à partir des fichiers suivants :

- [rotation.h](#)
- [rotation.cpp](#)

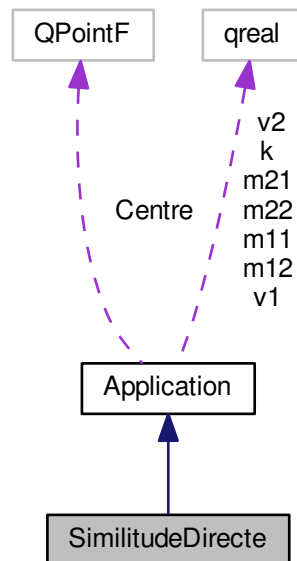
4.7 Référence de la classe SimilitudeDirecte

```
#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	
θ	
C	

4.7.1.3 `SimilitudeDirecte : :SimilitudeDirecte (qreal K , qreal θ , QPointF C , QPointF P)`

[SimilitudeDirecte : :SimilitudeDirecte.](#)

Paramètres

K	
θ	
C	
P	

4.7.2 Documentation des fonctions membres

4.7.2.1 `void SimilitudeDirecte : :setSimilitudeDirecte (qreal K , qreal θ , QPointF C)`

[SimilitudeDirecte : :setSimilitudeDirecte.](#)

Paramètres

K	
θ	
C	

4.7.2.2 `void SimilitudeDirecte : :setSimilitudeDirecte (qreal K , qreal θ , QPointF C , QPointF P)`

[SimilitudeDirecte : :setSimilitudeDirecte.](#)

Paramètres

K	
θ	
C	
P	

4.7.2.3 `void SimilitudeDirecte : :setTheta (qreal θ)`

[SimilitudeDirecte : :setTheta.](#)

Paramètres

θ	
----------	--

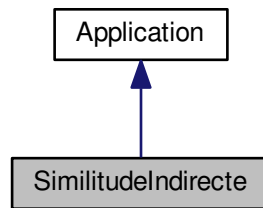
La documentation de cette classe a été générée à partir des fichiers suivants :

- [similitudedirecte.h](#)
- [similitudedirecte.cpp](#)

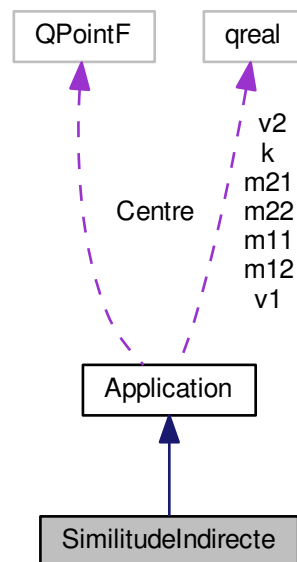
4.8 Référence de la classe SimilitudeIndirecte

```
#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

<i>K</i>	
<i>theta</i>	
<i>P</i>	

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

[SimilitudeIndirecte : :SimilitudeIndirecte.](#)

Paramètres

<i>K</i>	
<i>theta</i>	
<i>P</i>	
<i>C</i>	

4.8.2 Documentation des fonctions membres

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

[SimilitudeIndirecte : :setSimilitudeIndirecte.](#)

Paramètres

<i>K</i>	
<i>theta</i>	
<i>P</i>	

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

[SimilitudeIndirecte : :setSimilitudeIndirecte.](#)

Paramètres

<i>K</i>	
<i>theta</i>	
<i>P</i>	
<i>C</i>	

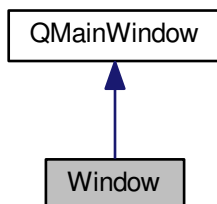
La documentation de cette classe a été générée à partir des fichiers suivants :

- [similitudeindirecte.h](#)
- [similitudeindirecte.cpp](#)

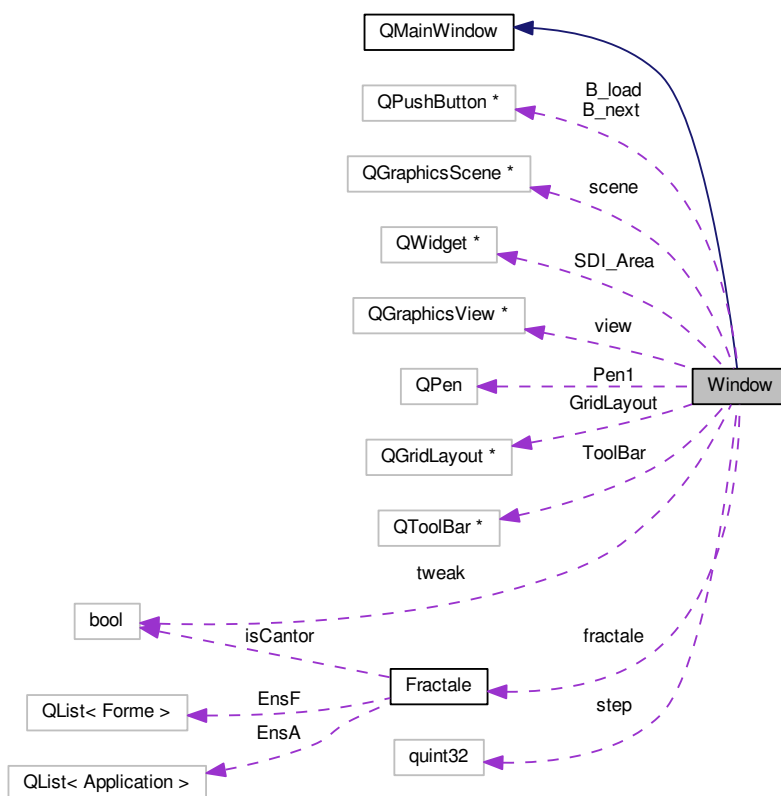
4.9 Référence de la classe Window

```
#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 default.
- void [refreshView](#) ()
- [Window : :refreshView](#) Dessine la fractale.

Fonctions membres publiques

- [Window](#) ()
- [Window : :Window](#) Constructeur par default 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 default 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

<i>obj</i>	
<i>event</i>	

Renvoie

4.9.2.2 void Window : :load () [slot]

[Window : :load](#) Affiche une fenetre permettant de choisir une fractale par default.

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

<i>event</i>	
--------------	--

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 : :SDI_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](#)

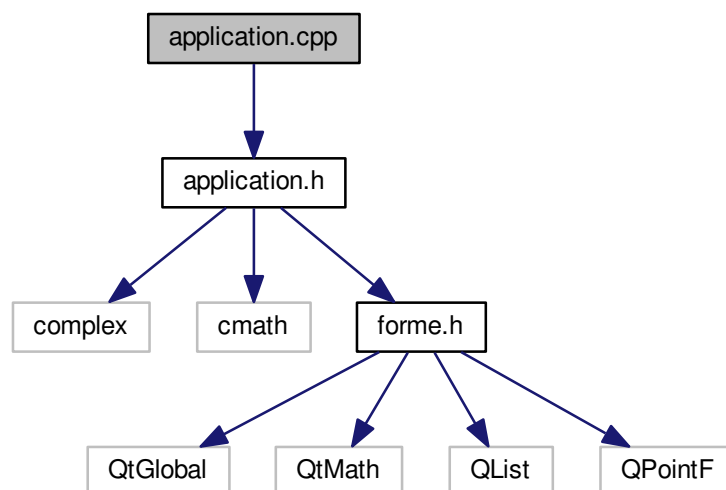
Chapitre 5

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	

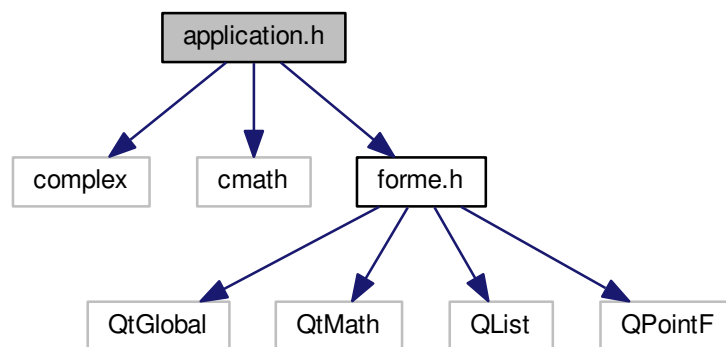
Renvoie

true si $A=B$, false si $A \neq B$

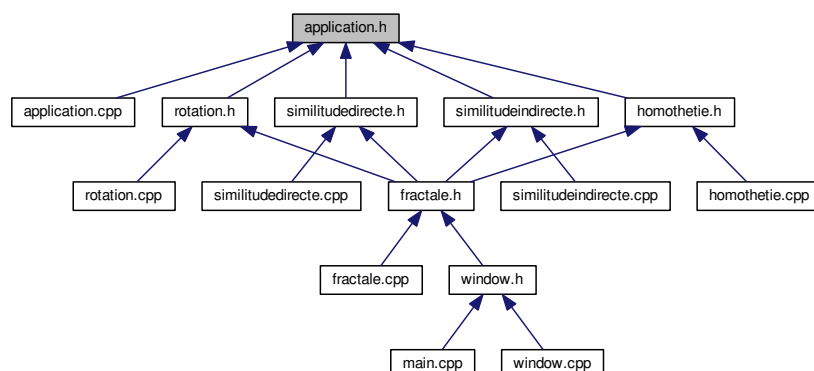
5.2 Référence du fichier application.h

```
#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

<i>A</i>	
<i>B</i>	

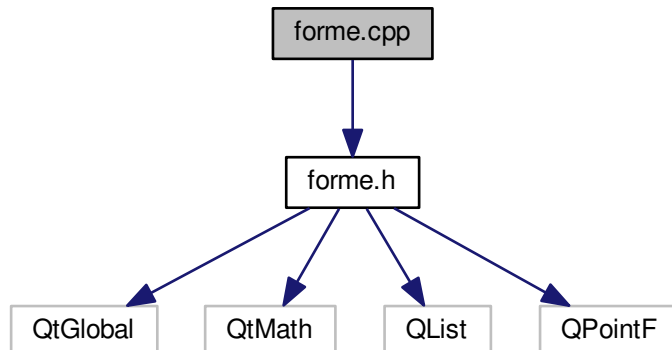
Renvoie

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

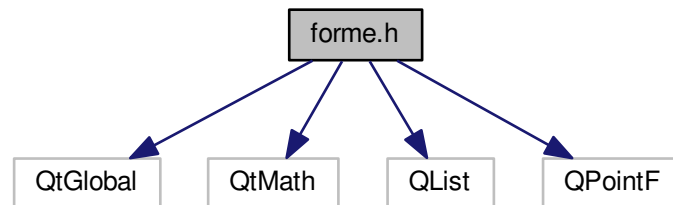
<i>A</i>	Forme 1
<i>B</i>	Forme 2

Renvoi

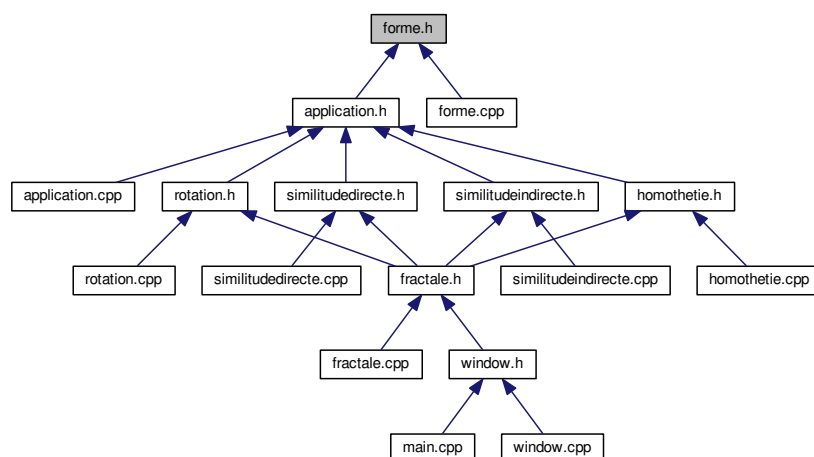
5.4 Référence du fichier forme.h

```
#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

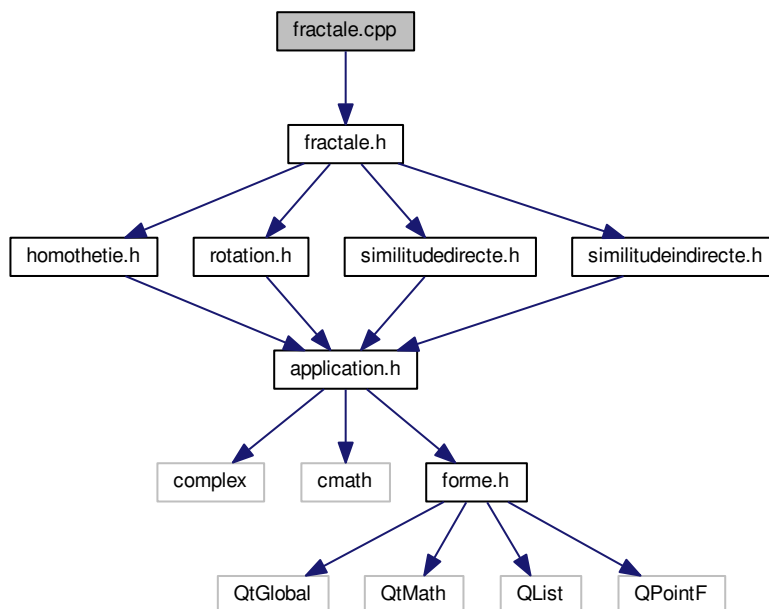
<i>A</i>	Forme 1
<i>B</i>	Forme 2

Renvoie

5.5 Référence du fichier fractale.cpp

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

Graphe des dépendances par inclusion de fractale.cpp :

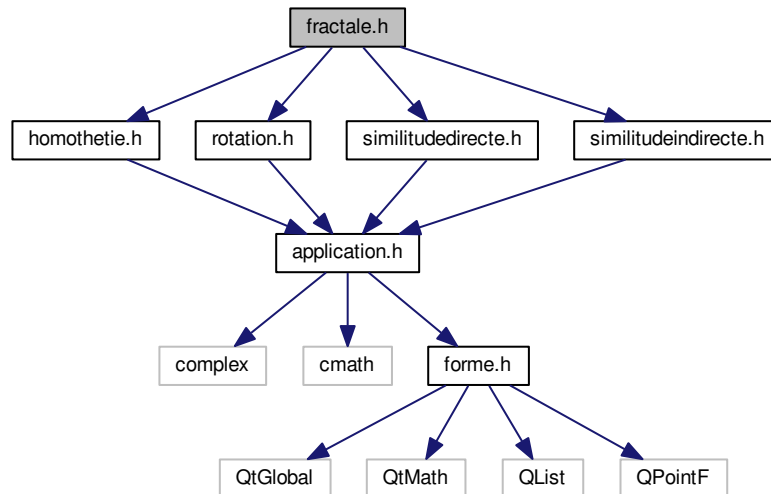


5.6 Référence du fichier fractale.h

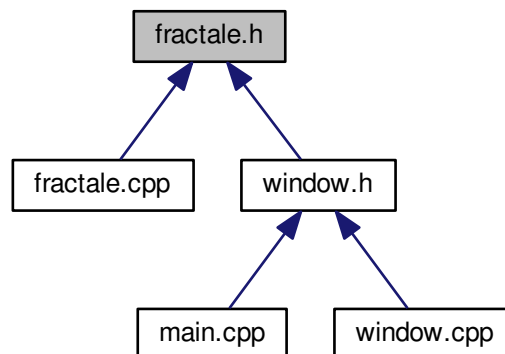
```
#include "homothetie.h"
```

```
#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 :



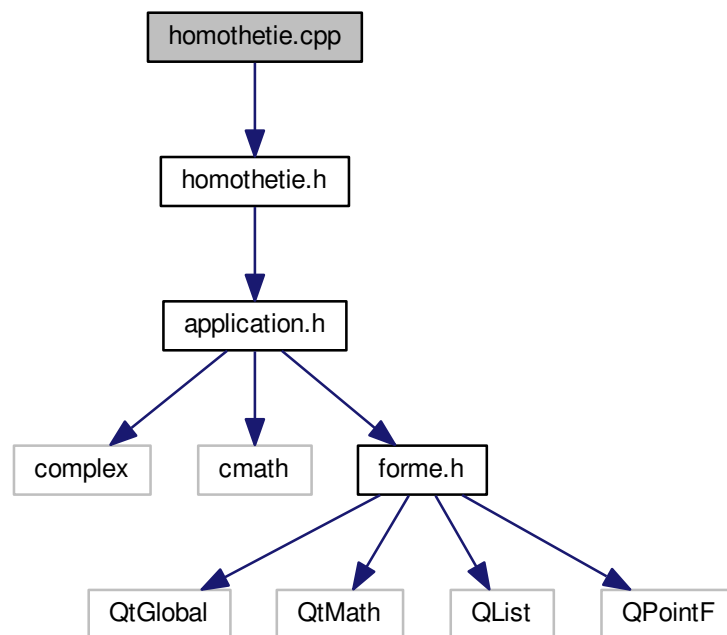
Classes

— class [Fractale](#)

5.7 Référence du fichier homothetie.cpp

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

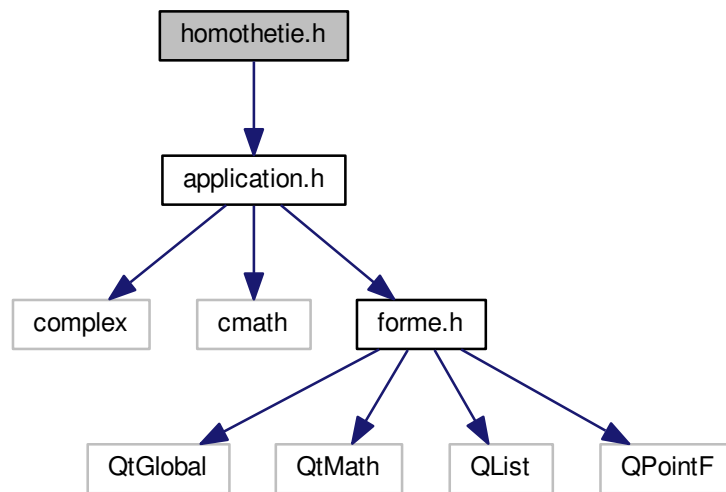
Graphe des dépendances par inclusion de homothetie.cpp :



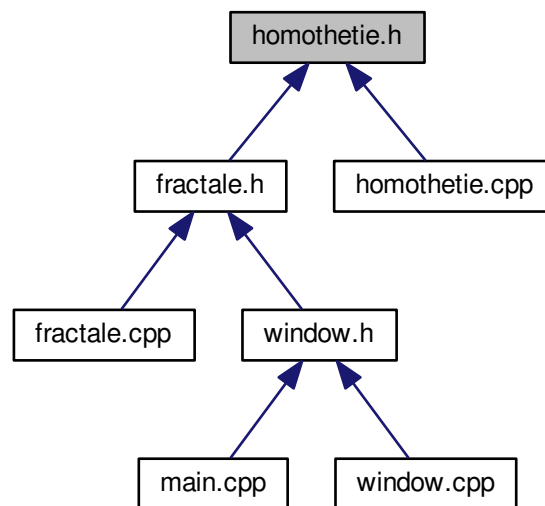
5.8 Référence du fichier homothetie.h

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

Graphe des dépendances par inclusion de homothetie.h :



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



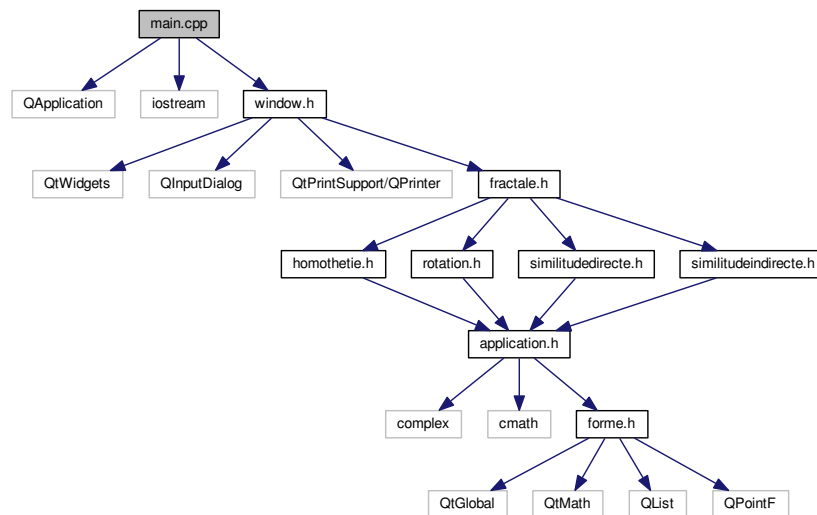
Classes

— 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[])

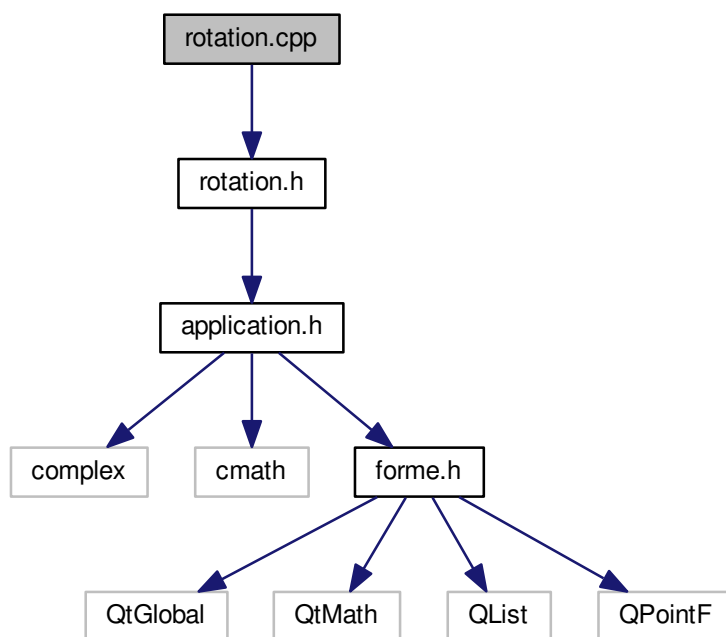
5.9.1 Documentation des fonctions

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

5.10 Référence du fichier rotation.cpp

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

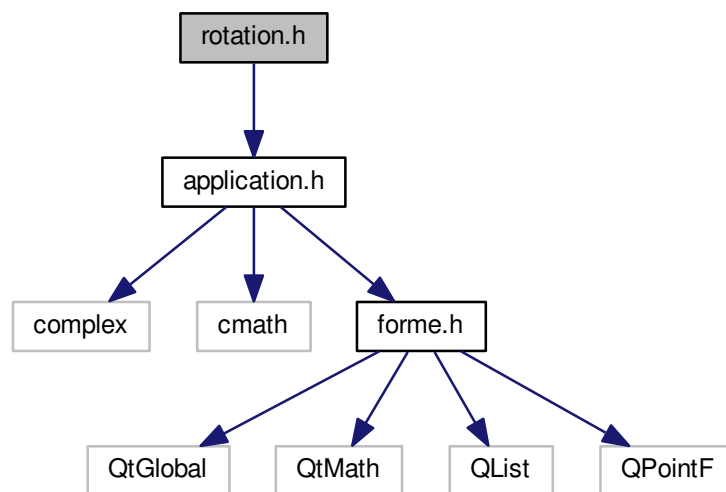
Graphe des dépendances par inclusion de rotation.cpp :



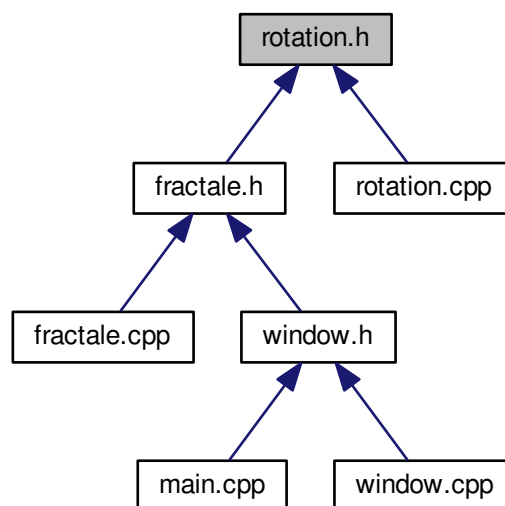
5.11 Référence du fichier rotation.h

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

Graphe des dépendances par inclusion de rotation.h :



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



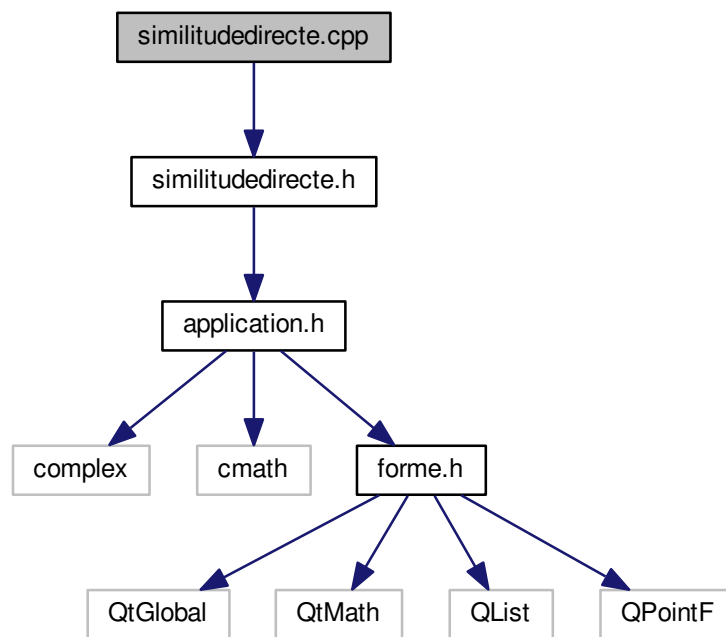
Classes

— class [Rotation](#)

5.12 Référence du fichier similitudedirecte.cpp

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

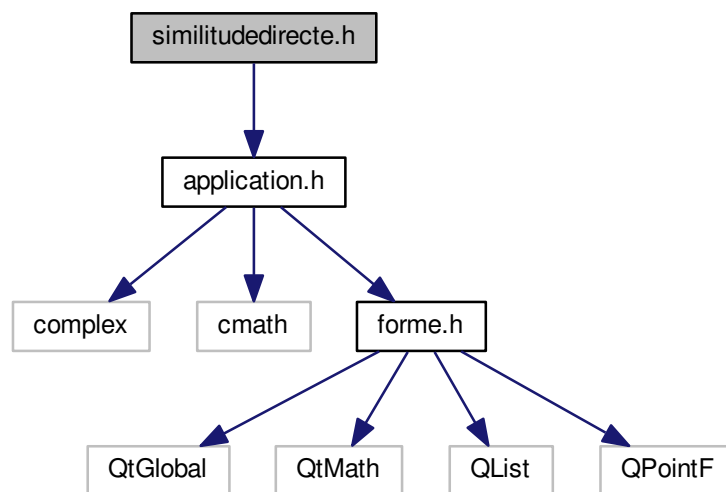
Graphe des dépendances par inclusion de similitudedirecte.cpp :



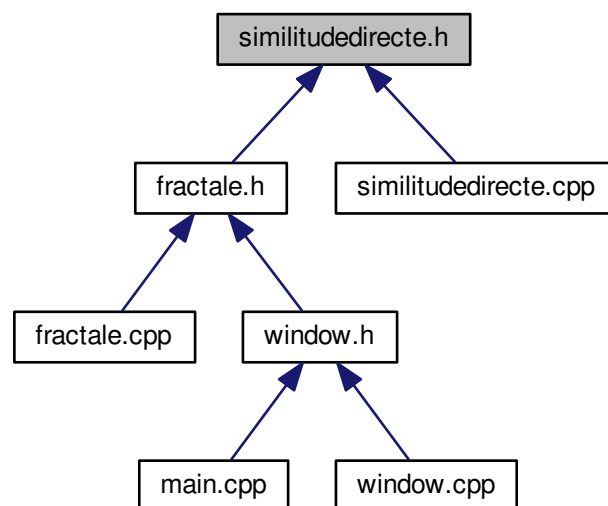
5.13 Référence du fichier similitudedirecte.h

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

Graphe des dépendances par inclusion de similitudedirecte.h :



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



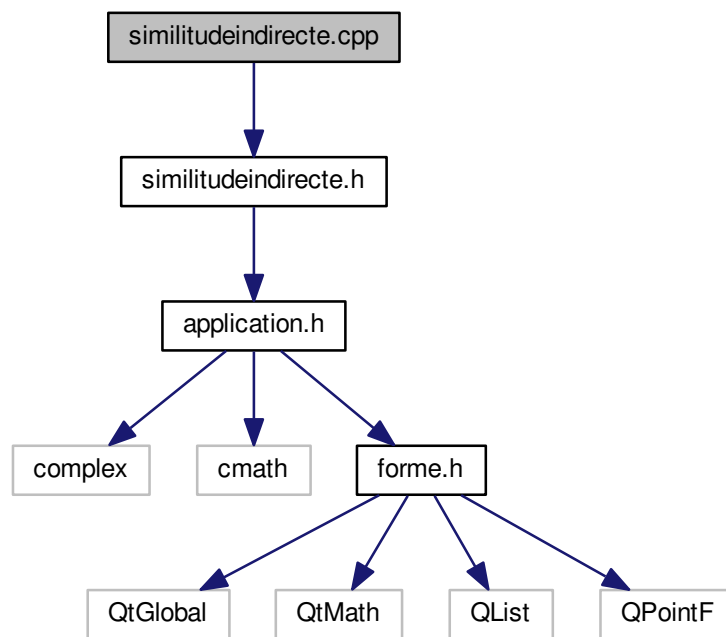
Classes

— class [SimilitudeDirecte](#)

5.14 Référence du fichier similitudeindirecte.cpp

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

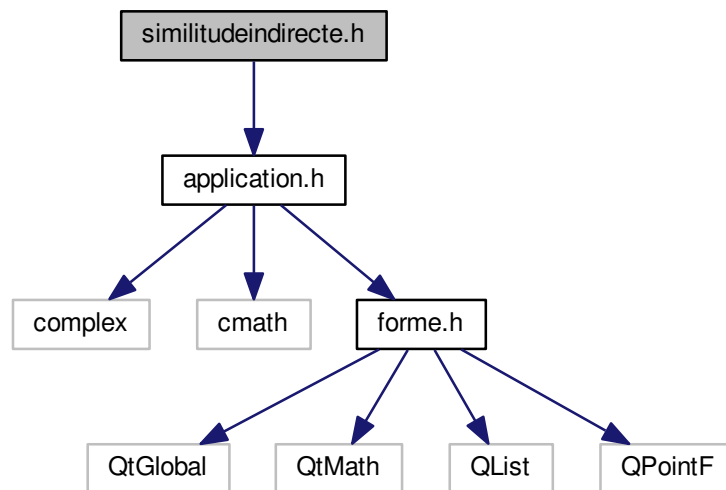
Graphe des dépendances par inclusion de similitudeindirecte.cpp :



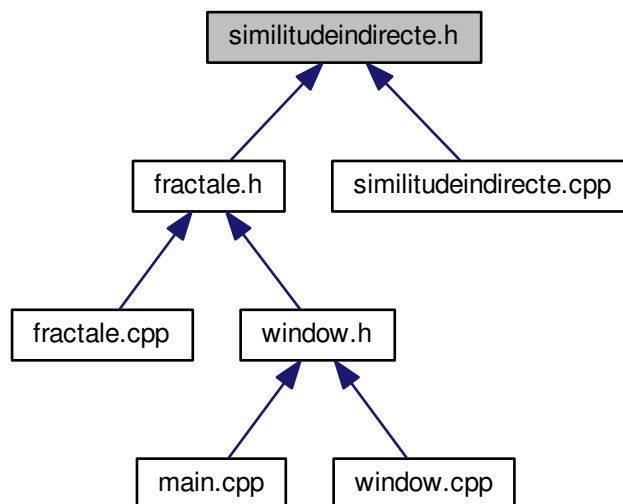
5.15 Référence du fichier similitudeindirecte.h

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

Graphe des dépendances par inclusion de similitudeindirecte.h :



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



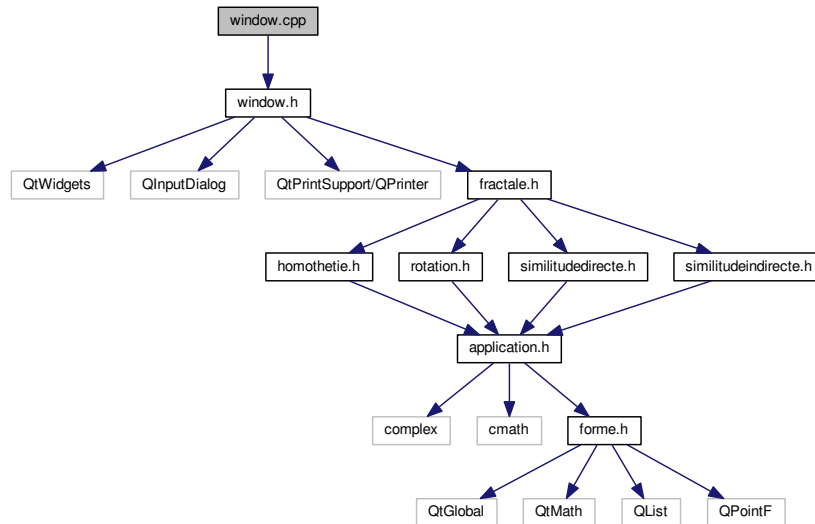
Classes

— class [SimilitudeIndirecte](#)

5.16 Référence du fichier window.cpp

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

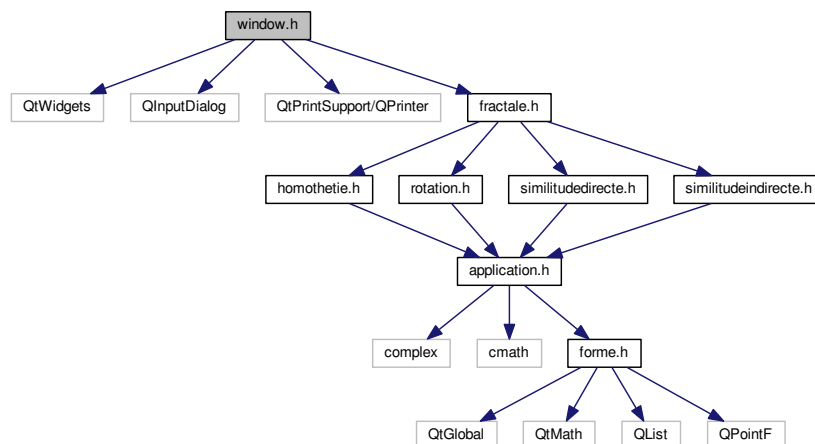
Graphe des dépendances par inclusion de window.cpp :



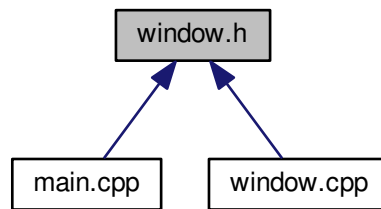
5.17 Référence du fichier window.h

```
#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

- AddApplication
 - Fractale, [17](#)
- AddForme
 - Fractale, [17](#)
- AddHomothetie
 - Fractale, [17](#)
- AddPoint
 - Forme, [14](#)
- AddRotation
 - Fractale, [17](#), [18](#)
- Application, [7](#)
 - Application, [9](#)
 - Centre, [13](#)
 - DoForEns, [9](#)
 - DoForForme, [9](#)
 - DoForQPointF, [9](#)
 - getCentre, [10](#)
 - getK, [10](#)
 - getTheta, [10](#)
 - getm11, [10](#)
 - getm12, [10](#)
 - getm21, [10](#)
 - getm22, [10](#)
 - getv1, [11](#)
 - getv2, [11](#)
 - isHomothetie, [11](#)
 - isRotation, [11](#)
 - k, [13](#)
 - m11, [13](#)
 - m12, [13](#)
 - m21, [13](#)
 - m22, [13](#)
 - setCentre, [11](#)
 - setK, [11](#)
 - setm11, [12](#)
 - setm12, [12](#)
 - setm21, [12](#)
 - setm22, [12](#)
 - setv1, [12](#)
 - setv2, [12](#)
 - v1, [13](#)
 - v2, [13](#)
- application.cpp, [33](#)
 - operator==, [33](#)
- application.h, [34](#)
 - operator==, [35](#)
- B_load
 - Window, [31](#)
- B_next
 - Window, [31](#)
- Centre
 - Application, [13](#)
- DoForEns
 - Application, [9](#)
- DoForForme
 - Application, [9](#)
- DoForQPointF
 - Application, [9](#)
- EnsA
 - Fractale, [19](#)
- EnsF
 - Fractale, [19](#)
- eventFilter
 - Window, [30](#)
- Forme, [14](#)
 - AddPoint, [14](#)
 - Forme, [14](#)
 - generateExisting, [15](#)
 - GetPoint, [15](#)
 - GetSize, [15](#)
 - L, [15](#)
- forme.cpp, [35](#)
 - operator==, [35](#)
- forme.h, [36](#)
 - operator==, [37](#)
- Fractale, [15](#)
 - AddApplication, [17](#)
 - AddForme, [17](#)
 - AddHomothetie, [17](#)
 - AddRotation, [17](#), [18](#)
 - EnsA, [19](#)
 - EnsF, [19](#)
 - Fractale, [16](#)
 - generateExisting, [18](#)
 - getFromEnsForme, [18](#)
 - getSizeEnsAppli, [18](#)
 - getSizeEnsForme, [19](#)
 - isCantor, [19](#)
 - isLikeCantor, [19](#)
 - RunOnce, [19](#)
 - setLikeCantor, [19](#)
- fractale
 - Window, [31](#)
- fractale.cpp, [37](#)
- fractale.h, [37](#)

- generateExisting
 - Forme, [15](#)
 - Fractale, [18](#)
- getCentre
 - Application, [10](#)
- getFromEnsForme
 - Fractale, [18](#)
- getK
 - Application, [10](#)
- GetPoint
 - Forme, [15](#)
- GetSize
 - Forme, [15](#)
- getSizeEnsAppli
 - Fractale, [18](#)
- getSizeEnsForme
 - Fractale, [19](#)
- getTheta
 - Application, [10](#)
- getm11
 - Application, [10](#)
- getm12
 - Application, [10](#)
- getm21
 - Application, [10](#)
- getm22
 - Application, [10](#)
- getv1
 - Application, [11](#)
- getv2
 - Application, [11](#)
- GridLayout
 - Window, [31](#)
- Homothetie, [19](#)
 - Homothetie, [21](#)
 - setHomothetie, [21](#)
- homothetie.cpp, [38](#)
- homothetie.h, [39](#)
- isCantor
 - Fractale, [19](#)
- isHomothetie
 - Application, [11](#)
- isLikeCantor
 - Fractale, [19](#)
- isRotation
 - Application, [11](#)
- k
 - Application, [13](#)
- L
 - Forme, [15](#)
- load
 - Window, [31](#)
- loadExistingFractal
 - Window, [31](#)
- m11
 - Application, [13](#)
- m12
 - Application, [13](#)
- m21
 - Application, [13](#)
- m22
 - Application, [13](#)
- main
 - main.cpp, [41](#)
- main.cpp, [41](#)
 - main, [41](#)
- operator==
 - application.cpp, [33](#)
 - application.h, [35](#)
 - forme.cpp, [35](#)
 - forme.h, [37](#)
- Pen1
 - Window, [31](#)
- QMainWindow, [22](#)
- refreshView
 - Window, [31](#)
- refreshViewColor
 - Window, [31](#)
- refreshViewSpecialCantor
 - Window, [31](#)
- Rotation, [22](#)
 - Rotation, [23](#)
 - setRotation, [24](#)
- rotation.cpp, [41](#)
- rotation.h, [42](#)
- RunOnce
 - Fractale, [19](#)
- SDI_Area
 - Window, [31](#)
- scene
 - Window, [31](#)
- setCentre
 - Application, [11](#)
- setHomothetie
 - Homothetie, [21](#)
- setK
 - Application, [11](#)
- setLikeCantor
 - Fractale, [19](#)
- setRotation
 - Rotation, [24](#)
- setSimilitudeDirecte
 - SimilitudeDirecte, [26](#)
- setSimilitudeIndirecte
 - SimilitudeIndirecte, [28](#)
- setTheta
 - SimilitudeDirecte, [26](#)
- setm11
 - Application, [12](#)

- setm12
 - Application, [12](#)
- setm21
 - Application, [12](#)
- setm22
 - Application, [12](#)
- setv1
 - Application, [12](#)
- setv2
 - Application, [12](#)
- SimilitudeDirecte, [24](#)
 - setSimilitudeDirecte, [26](#)
 - setTheta, [26](#)
 - SimilitudeDirecte, [25](#), [26](#)
- SimilitudeIndirecte, [26](#)
 - setSimilitudeIndirecte, [28](#)
 - SimilitudeIndirecte, [28](#)
- similitudedirecte.cpp, [44](#)
- similitudedirecte.h, [44](#)
- similitudeindirecte.cpp, [46](#)
- similitudeindirecte.h, [46](#)
- step
 - Window, [31](#)
- ToolBar
 - Window, [31](#)
- tweak
 - Window, [32](#)
- v1
 - Application, [13](#)
- v2
 - Application, [13](#)
- view
 - Window, [32](#)
- Window, [29](#)
 - B_load, [31](#)
 - B_next, [31](#)
 - eventFilter, [30](#)
 - fractale, [31](#)
 - GridLayout, [31](#)
 - load, [31](#)
 - loadExistingFractal, [31](#)
 - Pen1, [31](#)
 - refreshView, [31](#)
 - refreshViewColor, [31](#)
 - refreshViewSpecialCantor, [31](#)
 - SDI_Area, [31](#)
 - scene, [31](#)
 - step, [31](#)
 - ToolBar, [31](#)
 - tweak, [32](#)
 - view, [32](#)
 - Window, [30](#)
 - Zoom, [31](#)
- window.cpp, [48](#)
- window.h, [48](#)
- Zoom
 - Window, [31](#)