

SymulacjaZbiornika

0.1

Wygenerowano przez Doxygen 1.8.6

Cz, 4 cze 2015 23:55:51

Spis treści

1	Wizualizacja rozkładu ciśnienia cieczy na podstawie symulacji komputerowej	1
1.1	Opis projektu	1
1.2	Funkcjonalności aplikacji	1
2	Indeks przestrzeni nazw	3
2.1	Lista przestrzeni nazw	3
3	Indeks hierarchiczny	5
3.1	Hierarchia klas	5
4	Indeks klas	7
4.1	Lista klas	7
5	Indeks plików	9
5.1	Lista plików	9
6	Dokumentacja przestrzeni nazw	11
6.1	Dokumentacja przestrzeni nazw Ui	11
7	Dokumentacja klas	13
7.1	Dokumentacja klasy Czasteczka	13
7.1.1	Opis szczegółowy	14
7.1.2	Dokumentacja konstruktora i destruktora	14
7.1.2.1	Czasteczka	14
7.1.3	Dokumentacja funkcji składowych	14
7.1.3.1	Promien	14
7.1.3.2	Promien	15
7.1.3.3	RGB	15
7.1.3.4	RGB	15
7.1.3.5	RysujCzasteczke	15
7.1.3.6	xy	16
7.1.3.7	xy	16
7.1.4	Dokumentacja atrybutów składowych	16
7.1.4.1	_Promien	16

7.1.4.2	<code>_RGB</code>	17
7.1.4.3	<code>_xy</code>	17
7.2	Dokumentacja klasy <code>DMainWindow</code>	17
7.2.1	Opis szczegółowy	18
7.2.2	Dokumentacja konstruktora i destruktor	18
7.2.2.1	<code>DMainWindow</code>	18
7.2.2.2	<code>~DMainWindow</code>	18
7.2.3	Dokumentacja atrybutów składowych	19
7.2.3.1	<code>ui</code>	19
7.3	Dokumentacja klasy <code>Ui::DMainWindow</code>	19
7.3.1	Opis szczegółowy	19
7.4	Dokumentacja klasy <code>Kolor</code>	20
7.4.1	Opis szczegółowy	20
7.4.2	Dokumentacja konstruktora i destruktor	20
7.4.2.1	<code>Kolor</code>	20
7.4.2.2	<code>Kolor</code>	21
7.4.3	Dokumentacja funkcji składowych	21
7.4.3.1	<code>b</code>	21
7.4.3.2	<code>b</code>	22
7.4.3.3	<code>g</code>	22
7.4.3.4	<code>g</code>	23
7.4.3.5	<code>r</code>	23
7.4.3.6	<code>r</code>	23
7.4.4	Dokumentacja atrybutów składowych	23
7.4.4.1	<code>_b</code>	23
7.4.4.2	<code>_g</code>	23
7.4.4.3	<code>_r</code>	24
7.5	Dokumentacja klasy <code>OknoGlowne</code>	24
7.5.1	Opis szczegółowy	27
7.5.2	Dokumentacja konstruktora i destruktor	27
7.5.2.1	<code>OknoGlowne</code>	27
7.5.3	Dokumentacja funkcji składowych	27
7.5.3.1	<code>GdyNapis</code>	27
7.5.3.2	<code>GdyOdpowiedniCzas</code>	28
7.5.3.3	<code>on_action_Save_triggered</code>	28
7.5.3.4	<code>on_lineEdit_returnPressed</code>	28
7.5.3.5	<code>on_loadButton_clicked</code>	28
7.5.3.6	<code>on_pauseButton_clicked</code>	29
7.5.3.7	<code>on_playButton_clicked</code>	29
7.5.3.8	<code>on_sliderSzybkoscSym_valueChanged</code>	29

7.5.3.9	on_stopButton_clicked	29
7.5.3.10	paintEvent	30
7.5.3.11	WczytajSymulacjeZPliku	30
7.5.3.12	ZapiszSymulacjeDoPliku	31
7.5.3.13	ZglosNapis	31
7.5.4	Dokumentacja atrybutów składowych	31
7.5.4.1	_old_height	31
7.5.4.2	_old_width	32
7.5.4.3	_Stoper	32
7.5.4.4	action_Exit	32
7.5.4.5	action_Save	32
7.5.4.6	horizontalLayout	32
7.5.4.7	horizontalLayoutWidget	32
7.5.4.8	labelCzasSym	32
7.5.4.9	labelLiczbaCzasteczek	32
7.5.4.10	labelSzybkoscSym	32
7.5.4.11	lcdCzasSym	33
7.5.4.12	lcdLiczbaCzasteczek	33
7.5.4.13	lcdSzybkoscSym	33
7.5.4.14	licznik_plikow	33
7.5.4.15	lineEdit	33
7.5.4.16	loadButton	33
7.5.4.17	menu_Edit	33
7.5.4.18	menu_File	33
7.5.4.19	menu_Help	33
7.5.4.20	menuBar	34
7.5.4.21	pauseButton	34
7.5.4.22	playButton	34
7.5.4.23	sliderSzybkoscSym	34
7.5.4.24	statusBar	34
7.5.4.25	stopButton	34
7.5.4.26	toolBar	34
7.5.4.27	wZbiornik	34
7.6	Dokumentacja struktury params_t	35
7.6.1	Opis szczegółowy	35
7.6.2	Dokumentacja atrybutów składowych	35
7.6.2.1	dt	35
7.6.2.2	gx	35
7.6.2.3	gy	35
7.6.2.4	h	35

7.6.2.5	k	35
7.6.2.6	mass	35
7.6.2.7	mu	35
7.6.2.8	nframes	36
7.6.2.9	npframe	36
7.6.2.10	rho0	36
7.7	Dokumentacja klasy simulation	36
7.7.1	Opis szczegółowy	37
7.7.2	Dokumentacja składowych definicji typu	37
7.7.2.1	indicate_fun_t	37
7.7.3	Dokumentacja konstruktora i destruktor	37
7.7.3.1	simulation	37
7.7.4	Dokumentacja funkcji składowych	38
7.7.4.1	box_indicator	38
7.7.4.2	check_state	38
7.7.4.3	compute_accel	38
7.7.4.4	compute_density	39
7.7.4.5	damp_reflect_x	39
7.7.4.6	damp_reflect_y	40
7.7.4.7	getN	40
7.7.4.8	go	41
7.7.4.9	init	41
7.7.4.10	leapfrog_start	42
7.7.4.11	leapfrog_step	42
7.7.4.12	place_particles	43
7.7.4.13	reflect_bc	43
7.7.5	Dokumentacja przyjaciół i funkcji związanych	44
7.7.5.1	operator<<	44
7.7.6	Dokumentacja atrybutów składowych	44
7.7.6.1	a	44
7.7.6.2	n	44
7.7.6.3	p	44
7.7.6.4	params	44
7.7.6.5	rho	44
7.7.6.6	v	44
7.7.6.7	vh	44
7.8	Dokumentacja klasy Ui_DMainWindow	45
7.8.1	Opis szczegółowy	46
7.8.2	Dokumentacja funkcji składowych	46
7.8.2.1	retranslateUi	46

7.8.2.2	setupUi	46
7.8.3	Dokumentacja atrybutów składowych	46
7.8.3.1	action_Exit	46
7.8.3.2	action_Save	46
7.8.3.3	actionExit	47
7.8.3.4	actionPlay	47
7.8.3.5	centralWidget	47
7.8.3.6	horizontalLayout	47
7.8.3.7	horizontalLayoutWidget	47
7.8.3.8	label	47
7.8.3.9	lcdCzasSym	47
7.8.3.10	lcdLiczbaCzasteczek	47
7.8.3.11	lcdSzybkoscSym	47
7.8.3.12	lineCzasSym	47
7.8.3.13	lineLiczbaCzasteczek	47
7.8.3.14	lineSzybkoscSym	47
7.8.3.15	mainToolBar	48
7.8.3.16	menu_Edit	48
7.8.3.17	menu_File	48
7.8.3.18	menu_Help	48
7.8.3.19	menuBar	48
7.8.3.20	pauseButton	48
7.8.3.21	playButton	48
7.8.3.22	sliderSzybkoscSym	48
7.8.3.23	statusBar	48
7.8.3.24	stopButton	48
7.8.3.25	toolBar	48
7.8.3.26	verticalSpacer	48
7.9	Dokumentacja klasy Vector	49
7.9.1	Opis szczegółowy	50
7.9.2	Dokumentacja konstruktora i destruktora	50
7.9.2.1	Vector	50
7.9.2.2	Vector	51
7.9.3	Dokumentacja funkcji składowych	51
7.9.3.1	getX	51
7.9.3.2	getX	51
7.9.3.3	getX	52
7.9.3.4	getX	52
7.9.3.5	getY	52
7.9.3.6	getY	52

7.9.3.7	getY	52
7.9.3.8	getY	53
7.9.3.9	normSquared	53
7.9.3.10	normSquared	53
7.9.3.11	operator*	53
7.9.3.12	operator*	54
7.9.3.13	operator*==	54
7.9.3.14	operator*==	55
7.9.3.15	operator+	56
7.9.3.16	operator+	56
7.9.3.17	operator+=	57
7.9.3.18	operator+=	57
7.9.3.19	operator-	57
7.9.3.20	operator-	58
7.9.3.21	operator-=	58
7.9.3.22	operator-=	59
7.9.3.23	operator=	60
7.9.3.24	operator=	60
7.9.4	Dokumentacja przyjaciół i funkcji związanych	60
7.9.4.1	operator<<	60
7.9.4.2	operator<<	60
7.9.5	Dokumentacja atrybutów składowych	60
7.9.5.1	x	60
7.9.5.2	y	61
7.10	Dokumentacja klasy Zbiornik	61
7.10.1	Opis szczegółowy	63
7.10.2	Dokumentacja konstruktora i destruktor	63
7.10.2.1	Zbiornik	63
7.10.3	Dokumentacja funkcji składowych	64
7.10.3.1	czas_sym	64
7.10.3.2	czas_sym	64
7.10.3.3	CzyWewnatrzZbiornika	64
7.10.3.4	CzyWewnatrzZbiornika	65
7.10.3.5	CzyWewnatrzZbiornika	66
7.10.3.6	GdyOdpowiedniCzas	67
7.10.3.7	grubosc	68
7.10.3.8	grubosc	68
7.10.3.9	lewa_gora_xy	68
7.10.3.10	lewa_gora_xy	69
7.10.3.11	odpowiedni_czas	69

7.10.3.12	odpowiedni_czas	69
7.10.3.13	paintEvent	69
7.10.3.14	podstawa	70
7.10.3.15	podstawa	70
7.10.3.16	RysujZbiornik	70
7.10.3.17	RysujZbiornikZCzasteczkami	71
7.10.3.18	wysokosc	72
7.10.3.19	wysokosc	72
7.10.3.20	ZglosCzasSymulacji	72
7.10.3.21	ZglosLiczbeCzasteczek	72
7.10.3.22	ZglosNapis	73
7.10.4	Dokumentacja atrybutów składowych	73
7.10.4.1	_czas_sym	73
7.10.4.2	_grubosc	73
7.10.4.3	_lewa_gora_xy	73
7.10.4.4	_odpowiedni_czas	73
7.10.4.5	_podstawa	73
7.10.4.6	_Stoper	74
7.10.4.7	_wysokosc	74
7.10.4.8	Czasteczki	74
8	Dokumentacja plików	75
8.1	Dokumentacja pliku czasteczka.cpp	75
8.1.1	Opis szczegółowy	75
8.2	Dokumentacja pliku czasteczka.hh	75
8.2.1	Opis szczegółowy	76
8.3	Dokumentacja pliku dmainwindow.cpp	77
8.4	Dokumentacja pliku dmainwindow.h	77
8.5	Dokumentacja pliku flagi.hh	78
8.5.1	Opis szczegółowy	78
8.5.2	Dokumentacja typów wyliczanych	79
8.5.2.1	anonymous enum	79
8.5.3	Dokumentacja zmiennych	79
8.5.3.1	GRUBOSC	79
8.5.3.2	PODSTAWA	79
8.5.3.3	PROMIEN	79
8.5.3.4	STAN	79
8.5.3.5	WYSOKOSC	79
8.6	Dokumentacja pliku kolor.hh	79
8.6.1	Opis szczegółowy	80

8.7	Dokumentacja pliku main.cpp	80
8.7.1	Opis szczegółowy	81
8.7.2	Dokumentacja funkcji	81
8.7.2.1	main	81
8.8	Dokumentacja pliku moc_dmainwindow.cpp	81
8.9	Dokumentacja pliku moc_okno_glowne.cpp	82
8.10	Dokumentacja pliku moc_zbiornik.cpp	82
8.11	Dokumentacja pliku okno_glowne.cpp	82
8.11.1	Opis szczegółowy	82
8.12	Dokumentacja pliku okno_glowne.hh	82
8.12.1	Opis szczegółowy	84
8.13	Dokumentacja pliku simulation.cpp	84
8.13.1	Opis szczegółowy	85
8.13.2	Dokumentacja definicji	85
8.13.2.1	DAMP	85
8.13.2.2	LOG	85
8.13.2.3	XMAX	85
8.13.2.4	YMAX	85
8.13.3	Dokumentacja funkcji	85
8.13.3.1	funkcja_main	85
8.13.3.2	operator<<	86
8.13.3.3	operator<<	86
8.13.3.4	setup	86
8.14	Dokumentacja pliku strona.dox	86
8.15	Dokumentacja pliku ui_dmainwindow.h	86
8.16	Dokumentacja pliku vector.hh	88
8.17	Dokumentacja pliku zbiornik.cpp	88
8.17.1	Opis szczegółowy	88
8.17.2	Dokumentacja zmiennych	89
8.17.2.1	STAN	89
8.18	Dokumentacja pliku zbiornik.hh	89
8.18.1	Opis szczegółowy	90
Indeks		91

Rozdział 1

Wizualizacja rozkładu ciśnienia cieczy na podstawie symulacji komputerowej

Autor

Adam Balawender,
Krzysztof Kwieciński,
AiR, ARR, W4

Data

11.05.2015

Wersja

0.1

Aplikacja dotyczy komputerowej symulacji zachowania cieczy oraz wizualizacji jej stanu i rozkładu ciśnienia w zbiorniku z płynem.

1.1 Opis projektu

Symulacja będzie obejmowała ruch cieczy w przekroju 2D wybranego naczynia. Ciecz zostanie przedstawiona na płaszczyźnie jako zbiór oddziaływujących ze sobą cząsteczek. Postaramy się, żeby jej zachowanie było możliwie zbliżone do rzeczywistego. Ruch płynu zostanie zamodelowany metodą numeryczną SPH (particle hydrodynamics - wygładzona hydrodynamika cząstek). Pozwoli to na realistyczne odwzorowanie zachowania cieczy. Możliwe będzie badanie cieczy o różnych parametrach, dlatego też modelowane będą jej właściwości fizyczne: gęstość i lepkość. Dodatkowo mierzone będzie ciśnienie cieczy i zostanie ono zwizualizowane jako odcień koloru płynu. Im będzie on ciemniejszy, tym wyższe ciśnienie będzie odzwierciedlał.

1.2 Funkcjonalności aplikacji

Najistotniejszymi funkcjonalnościami aplikacji będą:

- symulacja zachowania cieczy w zależności od zadanych warunków początkowych,
- możliwość zdefiniowania parametrów cieczy (gęstości, lepkości),
- możliwość obserwacji wyniku symulacji (położenia cząsteczek i rozkładu ciśnień).

Rozdział 2

Indeks przestrzeni nazw

2.1 Lista przestrzeni nazw

Tutaj znajdują się wszystkie przestrzenie nazw wraz z ich krótkimi opisami:

Ui	11
----	----

Rozdział 3

Indeks hierarchiczny

3.1 Hierarchia klas

Ta lista dziedziczenia posortowana jest z grubsza, choć nie całkowicie, alfabetycznie:

Czasteczka	13
Kolor	20
params_t	35
QMainWindow	
DMainWindow	17
OknoGlowne	24
QWidget	
Zbiornik	61
simulation	36
Ui_DMainWindow	45
Ui::DMainWindow	19
Vector	49

Rozdział 4

Indeks klas

4.1 Lista klas

Tutaj znajdują się klasy, struktury, unie i interfejsy wraz z ich krótkimi opisami:

Czasteczka		
	Klasa modelująca czasteczke	13
DMainWindow	17
Ui::DMainWindow	19
Kolor		
	Klasa modelująca kolor	20
OknoGlowne		
	Klasa modelujaca główne okno aplikacji	24
params_t	35
simulation	36
Ui_DMainWindow	45
Vector		
	Klasa Vector	49
Zbiornik		
	Klasa modelująca zbiornik	61

Rozdział 5

Indeks plików

5.1 Lista plików

Tutaj znajduje się lista wszystkich plików z ich krótkimi opisami:

czasteczka.cpp	
Zawiera definicje metod klasy Czasteczka	75
czasteczka.hh	
Zawiera definicje klasy Czasteczka oraz deklaracje jej metod	75
dmainwindow.cpp	77
dmainwindow.h	77
flagi.hh	
Zawiera globalne zmienne opisujace symulacje	78
kolor.hh	
Zawiera definicje klasy Kolor oraz deklaracje jej metod	79
main.cpp	
Zawiera ogolna strukture funkcji main	80
moc_dmainwindow.cpp	81
moc_okno_glowne.cpp	82
moc_zbiornik.cpp	82
okno_glowne.cpp	
Zawiera definicje metod klasy OknoGlowne	82
okno_glowne.hh	
Zawiera definicje klasy OknoGlowne i deklaracje jej metod	82
simulation.cpp	
Plik z kodem symulatora cieczy	84
ui_dmainwindow.h	86
vector.hh	88
zbiornik.cpp	
Zawiera definicje metod klasy Zbiornik	88
zbiornik.hh	
Zawiera definicje klasy Zbiornik i deklaracje jej metod	89

Rozdział 6

Dokumentacja przestrzeni nazw

6.1 Dokumentacja przestrzeni nazw Ui

Komponenty

- class [DMainWindow](#)

Rozdział 7

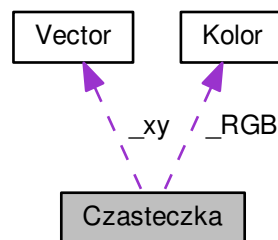
Dokumentacja klas

7.1 Dokumentacja klasy Czasteczka

Klasa modelująca czasteczke.

```
#include <czasteczka.hh>
```

Diagram współpracy dla Czasteczka:



Metody publiczne

- **Czasteczka** (**Vector** xy, int r, const **Kolor** &rgb)
Konstruktor.
- void **RysujCzasteczke** (QPainter &Rysownik, const int **Promien**, const **Kolor** RGB, const double x, const double y)
Metoda rysująca czasteczke.
- **Vector** xy () const
Interfejs pozwalający na odczyt prywatnych danych.
- **Vector** & xy ()
Interfejs pozwalający na zmianę prywatnych danych.
- int **Promien** () const
Interfejs pozwalający na odczyt prywatnych danych.
- int & **Promien** ()
Interfejs pozwalający na zmianę prywatnych danych.
- **Kolor** RGB () const

Interfejs pozwalający na odczyt prywatnych danych.

- [Kolor & RGB \(\)](#)

Interfejs pozwalający na zmianę prywatnych danych.

Atrybuty prywatne

- [Vector _xy](#)

Atrybut opisujący położenie czasteczki.

- [int _Promien](#)

Atrybut określający promień czasteczki.

- [Kolor _RGB](#)

Atrybut opisujący kolor czasteczki.

7.1.1 Opis szczegółowy

Klasa zawierająca podstawowe atrybuty czasteczki.

Definicja w linii 29 pliku czasteczka.hh.

7.1.2 Dokumentacja konstruktora i destruktora

7.1.2.1 Czasteczka::Czasteczka (Vector xy, int r, const Kolor & rgb) [inline]

Konstruktor parametryczny, inicjalizujący czasteczkę podanymi własnościami.

Parametry

in	xy	- wektor położenia czasteczki
in	r	- promień czasteczki
in	rgb	- kolor czasteczki

Definicja w linii 40 pliku czasteczka.hh.

7.1.3 Dokumentacja funkcji składowych

7.1.3.1 int Czasteczka::Promien () const [inline]

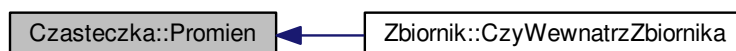
Interfejs pozwalający na odczyt prywatnych danych.

Zwraca

`_Promien` - prywatny atrybut opisujący promień czasteczki

Definicja w linii 81 pliku czasteczka.hh.

Oto graf wywołań tej funkcji:



7.1.3.2 int& Czasteczka::Promien () [inline]

Interfejs pozwalający na zmianę prywatnych danych.

Zwraca

_Promien - referencja na prywatny atrybut opisujący promień czasteczki

Definicja w linii 88 pliku czasteczka.hh.

7.1.3.3 Kolor Czasteczka::RGB () const [inline]

Interfejs pozwalający na odczyt prywatnych danych.

Zwraca

_RGB - prywatny atrybut opisujący położenie koloru czasteczki

Definicja w linii 96 pliku czasteczka.hh.

7.1.3.4 Kolor& Czasteczka::RGB () [inline]

Interfejs pozwalający na zmianę prywatnych danych.

Zwraca

_RGB - referencja na prywatny atrybut opisujący promień czasteczki

Definicja w linii 103 pliku czasteczka.hh.

7.1.3.5 void Czasteczka::RysujCzasteczke (QPainter & Rysownik, const int Promien, const Kolor RGB, const double x, const double y)

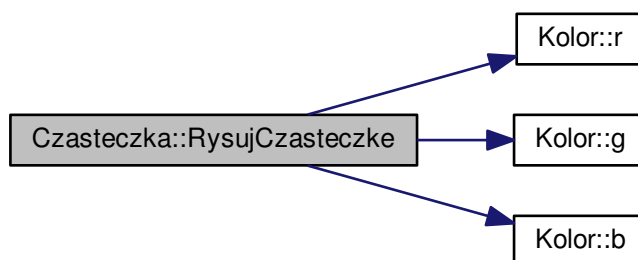
Rysuje czasteczke o zadanych parametrach

Parametry

in, out	<i>Rysownik</i>	- referencja na obiekt klasy QPainter
in	<i>Promien</i>	- promień czasteczki
in	<i>RGB</i>	- kolor czasteczki w formacie RGB
in	<i>x</i>	- położenie czasteczki na osi x
in	<i>y</i>	- położenie czasteczki na osi y

Definicja w linii 16 pliku czasteczka.cpp.

Oto graf wywołań dla tej funkcji:



7.1.3.6 Vector Czasteczka::xy () const [inline]

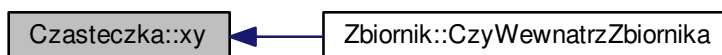
Interfejs pozwalający na odczyt prywatnych danych.

Zwraca

`_xy` - prywatny atrybut opisujący położenie czasteczki

Definicja w linii 66 pliku `czasteczka.hh`.

Oto graf wywołań tej funkcji:



7.1.3.7 Vector& Czasteczka::xy () [inline]

Interfejs pozwalający na zmianę prywatnych danych.

Zwraca

`_xy` - referencja na prywatny atrybut opisujący położenie czasteczki

Definicja w linii 73 pliku `czasteczka.hh`.

7.1.4 Dokumentacja atrybutów składowych

7.1.4.1 int Czasteczka::_Promien [private]

Atrybut określający promień czasteczki.

Definicja w linii 118 pliku `czasteczka.hh`.

7.1.4.2 Kolor Czasteczka::_RGB [private]

Atrybut opisujący kolor czasteczki.

Definicja w linii 125 pliku czasteczka.hh.

7.1.4.3 Vector Czasteczka::_xy [private]

Atrybut opisujący położenie czasteczki w kartezjanskim układzie współrzędnych.

Definicja w linii 103 pliku czasteczka.hh.

Dokumentacja dla tej klasy została wygenerowana z plików:

- [czasteczka.hh](#)
- [czasteczka.cpp](#)

7.2 Dokumentacja klasy DMainWindow

```
#include <dmainwindow.h>
```

Diagram dziedziczenia dla DMainWindow

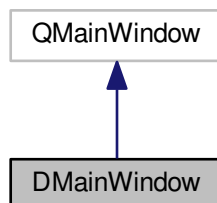
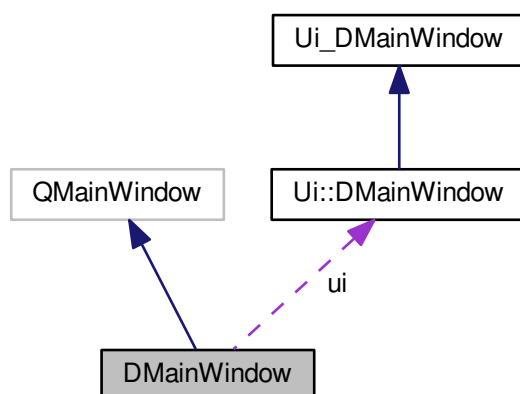


Diagram współpracy dla DMainWindow:



Metody publiczne

- `DMainWindow` (`QWidget *parent=0`)
- `~DMainWindow` ()

Atrybuty prywatne

- `Ui::DMainWindow * ui`

7.2.1 Opis szczegółowy

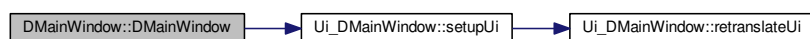
Definicja w linii 10 pliku `dmainwindow.h`.

7.2.2 Dokumentacja konstruktora i destruktor

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

Definicja w linii 4 pliku `dmainwindow.cpp`.

Oto graf wywołań dla tej funkcji:



7.2.2.2 `DMainWindow::~~DMainWindow ()`

Definicja w linii 11 pliku `dmainwindow.cpp`.

7.2.3 Dokumentacja atrybutów składowych

7.2.3.1 Ui::DMainWindow* DMainWindow::ui [private]

Definicja w linii 19 pliku dmainwindow.h.

Dokumentacja dla tej klasy została wygenerowana z plików:

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

7.3 Dokumentacja klasy Ui::DMainWindow

```
#include <ui_dmainwindow.h>
```

Diagram dziedziczenia dla Ui::DMainWindow

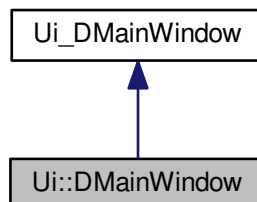
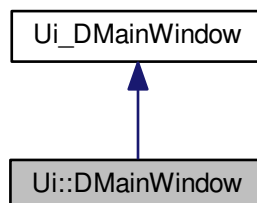


Diagram współpracy dla Ui::DMainWindow:



Dodatkowe Dziedziczone Składowe

7.3.1 Opis szczegółowy

Definicja w linii 200 pliku ui_dmainwindow.h.

Dokumentacja dla tej klasy została wygenerowana z pliku:

- [ui_dmainwindow.h](#)

7.4 Dokumentacja klasy Kolor

Klasa modelująca kolor.

```
#include <kolor.hh>
```

Metody publiczne

- [Kolor](#) (int [r](#), int [g](#), int [b](#))
Konstruktor.
- [Kolor](#) (const [Kolor](#) &rgb)
Konstruktor kopiujący.
- int [r](#) () const
Interfejs pozwalający na odczyt prywatnych danych.
- int & [r](#) ()
Interfejs pozwalający na zmianę prywatnych danych.
- int [g](#) () const
Interfejs pozwalający na odczyt prywatnych danych.
- int & [g](#) ()
Interfejs pozwalający na zmianę prywatnych danych.
- int [b](#) () const
Interfejs pozwalający na odczyt prywatnych danych.
- int & [b](#) ()
Interfejs pozwalający na zmianę prywatnych danych.

Atrybuty prywatne

- int [_r](#)
Atrybut opisujący wartość odcienia czerwonego.
- int [_g](#)
Atrybut opisujący wartość odcienia zielonego.
- int [_b](#)
Atrybut opisujący wartość odcienia niebieskiego.

7.4.1 Opis szczegółowy

Klasa opisuje kolor w formacie RGB.

Definicja w linii 24 pliku kolor.hh.

7.4.2 Dokumentacja konstruktora i destruktor

7.4.2.1 [Kolor::Kolor](#) (int [r](#), int [g](#), int [b](#)) [inline]

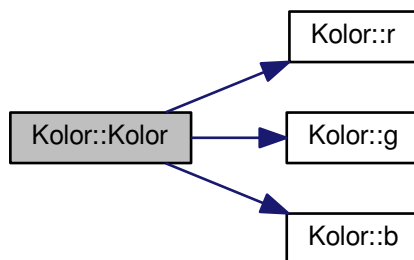
Konstruktor parametryczny, inicjalizujący kolor podanymi wartościami.

Parametry

in	<i>r</i>	- wartosc odcienia czerwonego, [0, 255]
in	<i>g</i>	- wartosc odcienia zielonego, [0, 255]
in	<i>b</i>	- wartosc odcienia niebieskiego, [0, 255]

Definicja w linii 35 pliku kolor.hh.

Oto graf wywołań dla tej funkcji:



7.4.2.2 `Kolor::Kolor (const Kolor & rgb) [inline]`

Konstruktor kopiujacy.

Parametry

in	<i>rgb</i>	- obiekt do skopiowania
----	------------	-------------------------

Definicja w linii 43 pliku kolor.hh.

7.4.3 Dokumentacja funkcji składowych

7.4.3.1 `int Kolor::b () const [inline]`

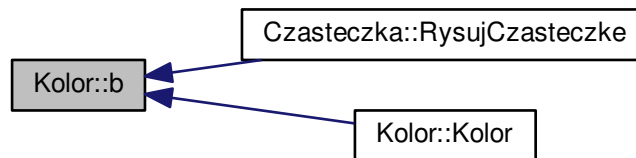
Interfejs pozwalający na odczyt prywatnych danych.

Zwraca

`_b` - prywatny atrybut opisujący niebieski odcień

Definicja w linii 81 pliku kolor.hh.

Oto graf wywoływań tej funkcji:

**7.4.3.2 `int& Kolor::b () [inline]`**

Interfejs pozwalający na zmianę prywatnych danych.

Zwraca

`_b` - referencja na prywatny atrybut opisujący niebieski odcień

Definicja w linii 88 pliku kolor.hh.

7.4.3.3 `int Kolor::g () const [inline]`

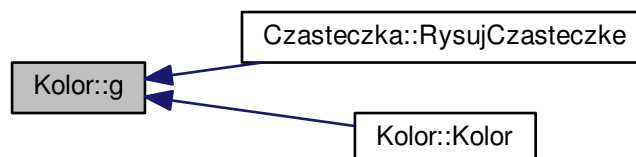
Interfejs pozwalający na odczyt prywatnych danych.

Zwraca

`_g` - prywatny atrybut opisujący zielony odcień

Definicja w linii 66 pliku kolor.hh.

Oto graf wywoływań tej funkcji:



7.4.3.4 `int& Kolor::g () [inline]`

Interfejs pozwalający na zmianę prywatnych danych.

Zwraca

`_g` - referencja na prywatny atrybut opisujący zielony odcień

Definicja w linii 73 pliku kolor.hh.

7.4.3.5 `int Kolor::r () const [inline]`

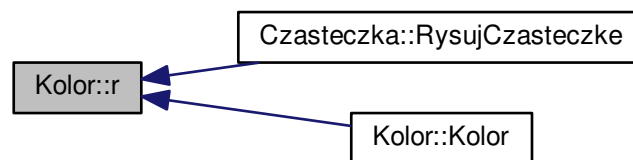
Interfejs pozwalający na odczyt prywatnych danych.

Zwraca

`_r` - prywatny atrybut opisujący czerwony odcień

Definicja w linii 51 pliku kolor.hh.

Oto graf wywołań tej funkcji:



7.4.3.6 `int& Kolor::r () [inline]`

Interfejs pozwalający na zmianę prywatnych danych.

Zwraca

`_r` - referencja na prywatny atrybut opisujący czerwony odcień

Definicja w linii 58 pliku kolor.hh.

7.4.4 Dokumentacja atrybutów składowych

7.4.4.1 `int Kolor::_b [private]`

Atrybut opisujący wartość odcienia niebieskiego.

Definicja w linii 110 pliku kolor.hh.

7.4.4.2 `int Kolor::_g [private]`

Atrybut opisujący wartość odcienia zielonego.

Definicja w linii 103 pliku kolor.hh.

Sloty publiczne

- void [GdyOdpowiedniCzas](#) ()
Slot odpowiadający za aktualizację danych. .
- void [GdyNapis](#) (const QString &)
Slot odpowiadający za wyświetlenie napisu po otrzymaniu sygnału.
- void [on_playButton_clicked](#) ()
Slot odpowiadający za obsługę stanu play.
- void [on_pauseButton_clicked](#) ()
Slot odpowiadający za obsługę stanu pauza.
- void [on_stopButton_clicked](#) ()
Slot odpowiadający za obsługę stanu stop.
- void [on_loadButton_clicked](#) ()
Slot odpowiadający za wczytanie danych z pliku.
- void [on_lineEdit_returnPressed](#) ()
Slot odpowiadający za wczytanie danych z pliku.
- void [on_sliderSzybkoscSym_valueChanged](#) (int a)
Slot odpowiadający za zmianę wartości slidera.
- void [on_action_Save_triggered](#) ()
Slot odpowiadający za przycisnięcie przycisku Save.

Sygnały

- void [ZglosNapis](#) (const QString &)
Sygnał zgłaszający napis.

Metody publiczne

- [OknoGlowne](#) (QWidget *wRodzic=NULL)
Konstruktor.
- virtual void [paintEvent](#) (QPaintEvent *event)
Wirtualna metoda paintEvent wyrysowująca obiekty na ekranie.
- void [ZapiszSymulacjeDoPliku](#) ()
Metoda zapisująca aktualny stan symulacji do pliku.
- void [WczytajSymulacjeZPliku](#) (const std::string nazwa_pliku)
Metoda wczytująca z pliku stan symulacji.

Atrybuty publiczne

- QTimer [_Stoper](#)
Miernik czasu.

Atrybuty prywatne

- Zbiornik * [wZbiornik](#)
Wskaźnik na zbiornik.
- QMenuBar * [menuBar](#)
Wskaźnik na pasek menu.
- QAction * [action_Save](#)
Wskaźnik na akcję przycisku menu Save.

- QAction * [action_Exit](#)
Wskaźnik na akcje przycisku menu Exit.
- QMenu * [menu_File](#)
Wskaźnik na akcje przycisku menu File.
- QMenu * [menu_Edit](#)
Wskaźnik na akcje przycisku menu Edit.
- QMenu * [menu_Help](#)
Wskaźnik na akcje przycisku menu Help.
- QStatusBar * [statusBar](#)
Wskaźnik na pasek statusowy.
- QToolBar * [toolBar](#)
Wskaźnik na pasek narzędziowy.
- QHBoxLayout * [horizontalLayout](#)
Wskaźnik na obszar do horyzontalnego rozmieszczenia przycisków.
- QWidget * [horizontalLayoutWidget](#)
Wskaźnik na widget odpowiedzialny za horyzontalne wyświetlenie przycisków.
- QPushButton * [playButton](#)
Wskaźnik na przycisk play.
- QPushButton * [pauseButton](#)
Wskaźnik na przycisk pause.
- QPushButton * [stopButton](#)
Wskaźnik na przycisk stop.
- QPushButton * [loadButton](#)
Wskaźnik na przycisk Wczytaj.
- QSlider * [sliderSzybkoscSym](#)
Wskaźnik na slider.
- QLCDNumber * [lcdSzybkoscSym](#)
Wskaźnik na LCD z szybkością symulacji.
- QLabel * [labelSzybkoscSym](#)
Wskaźnik na etykiety z szybkością symulacji.
- QLabel * [labelCzasSym](#)
Wskaźnik na etykiety z czasem symulacji.
- QLCDNumber * [lcdCzasSym](#)
Wskaźnik na LCD z czasem trwania symulacji.
- QLabel * [labelLiczbaCzasteczek](#)
Wskaźnik na etykiety z liczbą cząsteczek.
- QLCDNumber * [lcdLiczbaCzasteczek](#)
Wskaźnik na LCD z liczbą cząsteczek.
- QLineEdit * [lineEdit](#)
Wskaźnik na linijkę do wpisywania tekstu.
- double [_old_width](#)
Stara szerokość okienka.
- double [_old_height](#)
Stara wysokość okienka.

Statyczne atrybuty prywatne

- static int [licznik_plikow](#)
Służy do generowania unikatowych nazw plików wyjściowych.

7.5.1 Opis szczegółowy

Dzięki tej klasie wyświetlane jest okno główne aplikacji.

Definicja w linii 68 pliku okno_glowne.hh.

7.5.2 Dokumentacja konstruktora i destruktor

7.5.2.1 OknoGlowne::OknoGlowne (QWidget * wRodzic = NULL)

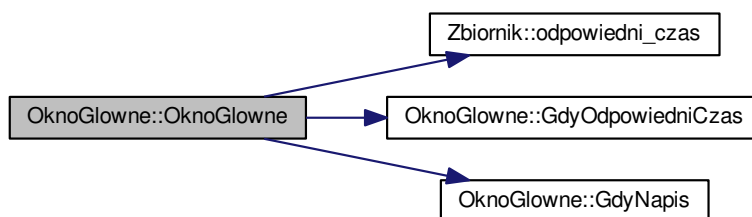
Konstruktor parametryczny.

Parametry

<i>in, out</i>	<i>wRodzic</i>	- wskaźnik na rodzica
----------------	----------------	-----------------------

Definicja w linii 16 pliku okno_glowne.cpp.

Oto graf wywołań dla tej funkcji:



7.5.3 Dokumentacja funkcji składowych

7.5.3.1 void OknoGlowne::GdyNapis (const QString & Napis) [slot]

Odpowiada za wyświetlenie napisu na belce statusowej.

Parametry

<i>in</i>	<i>Napis</i>	- napis do wyświetlenia
-----------	--------------	-------------------------

Definicja w linii 223 pliku okno_glowne.cpp.

Oto graf wywołań tej funkcji:



7.5.3.2 void OknoGlowne::GdyOdpowiedniCzas () [slot]

Odpowiada za uaktualnienie okienka w odpowiednich momentach.

Definicja w linii 215 pliku okno_glowne.cpp.

Oto graf wywołań tej funkcji:

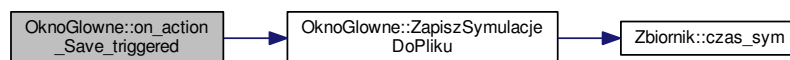


7.5.3.3 void OknoGlowne::on_action_Save_triggered () [slot]

Odpowiada za wykonanie odpowiednich czynności po przycisnięciu Save.

Definicja w linii 204 pliku okno_glowne.cpp.

Oto graf wywołań dla tej funkcji:

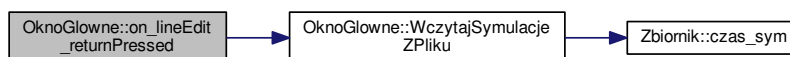


7.5.3.4 void OknoGlowne::on_lineEdit_returnPressed () [slot]

Odpowiada za wczytanie danych z pliku.

Definicja w linii 191 pliku okno_glowne.cpp.

Oto graf wywołań dla tej funkcji:

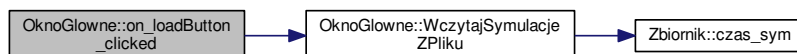


7.5.3.5 void OknoGlowne::on_loadButton_clicked () [slot]

Odpowiada za wczytanie danych z pliku.

Definicja w linii 185 pliku okno_glowne.cpp.

Oto graf wywołań dla tej funkcji:



7.5.3.6 void OknoGlowne::on_pauseButton_clicked () [slot]

Odpowiada za wykonanie odpowiednich czynności w trakcie stanu pauza.

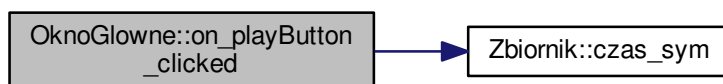
Definicja w linii 177 pliku `okno_glowne.cpp`.

7.5.3.7 void OknoGlowne::on_playButton_clicked () [slot]

Odpowiada za wykonanie odpowiednich czynności w trakcie stanu play.

Definicja w linii 167 pliku `okno_glowne.cpp`.

Oto graf wywołań dla tej funkcji:

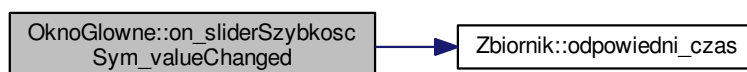


7.5.3.8 void OknoGlowne::on_sliderSzybkoscSym_valueChanged (int a) [slot]

Odpowiada za wykonanie odpowiednich czynności po zmianie wartości slidera.

Definicja w linii 197 pliku `okno_glowne.cpp`.

Oto graf wywołań dla tej funkcji:



7.5.3.9 void OknoGlowne::on_stopButton_clicked () [slot]

Odpowiada za wykonanie odpowiednich czynności w trakcie stanu stop.

Definicja w linii 181 pliku okno_glowne.cpp.

7.5.3.10 `void OknoGlowne::paintEvent (QPaintEvent * event) [virtual]`

Odziedziczona wirtualna metoda paintEvent. Rysuje zbiornik i przyciski w nowym miejscu.

Parametry

<i>in, out</i>	<i>event</i>	- wskaźnik obiekt klasy QPaintEvent
----------------	--------------	-------------------------------------

Definicja w linii 227 pliku okno_glowne.cpp.

Oto graf wywołań dla tej funkcji:

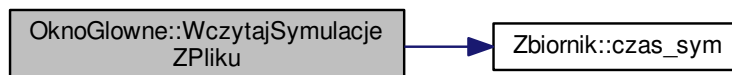


7.5.3.11 `void OknoGlowne::WczytajSymulacjeZPliku (const std::string nazwa_pliku)`

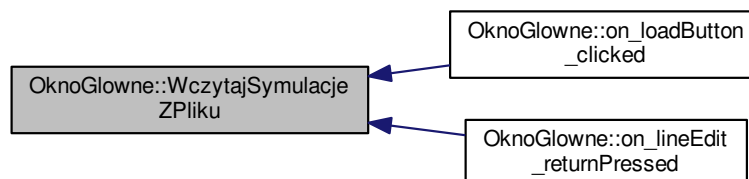
Wczytuje stan symulacji (czas, liczba czasteczek, dane czasteczek).

Definicja w linii 266 pliku okno_glowne.cpp.

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:

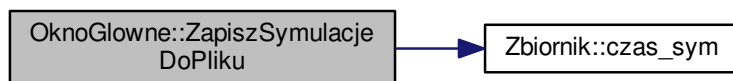


7.5.3.12 void OknoGlowne::ZapiszSymulacjeDoPliku ()

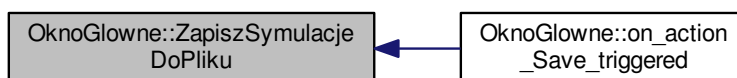
Zapisuje aktualny stan symulacji (czas, liczba czasteczek, dane czasteczek) do pliku.

Definicja w linii 243 pliku okno_glowne.cpp.

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



7.5.3.13 void OknoGlowne::ZglosNapis (const QString & _t1) [signal]

Sygnal zgłaszający napis do odpowiedniego slotu.

Parametry

in	<code>_t1</code>	- napis do zgłoszenia
----	------------------	-----------------------

Definicja w linii 121 pliku moc_okno_glowne.cpp.

Oto graf wywoływań tej funkcji:



7.5.4 Dokumentacja atrybutów składowych

7.5.4.1 double OknoGlowne::_old_height [private]

Stara wysokosc okienka.

Definicja w linii 367 pliku okno_glowne.hh.

7.5.4.2 `double OknoGlowne::_old_width` [private]

Stara szerokosc okienka.

Definicja w linii 361 pliku `okno_glowne.hh`.

7.5.4.3 `QTimer OknoGlowne::_Stoper`

Miernik czasu.

Definicja w linii 353 pliku `okno_glowne.hh`.

7.5.4.4 `QAction* OknoGlowne::action_Exit` [private]

Wskaznik na akcje przycisku menu Exit.

Definicja w linii 212 pliku `okno_glowne.hh`.

7.5.4.5 `QAction* OknoGlowne::action_Save` [private]

Wskaznik na akcje przycisku menu Save.

Definicja w linii 205 pliku `okno_glowne.hh`.

7.5.4.6 `QHBoxLayout* OknoGlowne::horizontalLayout` [private]

Wskaznik na obszar do horyzontalnego rozmieszczenia przyciskow.

Definicja w linii 254 pliku `okno_glowne.hh`.

7.5.4.7 `QWidget* OknoGlowne::horizontalLayoutWidget` [private]

Wskaznik na widget odpowiedzialny za horyzontalne wyswietlenie przyciskow.

Definicja w linii 261 pliku `okno_glowne.hh`.

7.5.4.8 `QLabel* OknoGlowne::labelCzasSym` [private]

Etykieta dla czasu symulacji.

Definicja w linii 317 pliku `okno_glowne.hh`.

7.5.4.9 `QLabel* OknoGlowne::labelLiczbaCzasteczek` [private]

Etykieta dla liczby symulowanych czasteczek.

Definicja w linii 331 pliku `okno_glowne.hh`.

7.5.4.10 `QLabel* OknoGlowne::labelSzybkoscSym` [private]

Etykieta dla szybkosci symulacji.

Definicja w linii 310 pliku `okno_glowne.hh`.

7.5.4.11 `QLCDNumber* OknoGlowne::lcdCzasSym` `[private]`

Wskaźnik na LCD z szybkością symulacji. Wyświetla jej czas trwania.

Definicja w linii 324 pliku `okno_glowne.hh`.

7.5.4.12 `QLCDNumber* OknoGlowne::lcdLiczbaCzasteczek` `[private]`

Wyświetla liczbę symulowanych cząsteczek.

Definicja w linii 338 pliku `okno_glowne.hh`.

7.5.4.13 `QLCDNumber* OknoGlowne::lcdSzyboscSym` `[private]`

Wskaźnik na LCD z szybkością symulacji. Wyświetla jej szybkość.

Definicja w linii 303 pliku `okno_glowne.hh`.

7.5.4.14 `int OknoGlowne::licznik_plikow` `[static], [private]`

Służy do generowania unikatowych nazw plików wyjściowych.

Definicja w linii 184 pliku `okno_glowne.hh`.

7.5.4.15 `QLineEdit* OknoGlowne::lineEdit` `[private]`

Wskazuje linijkę do wpisywania tekstu.

Definicja w linii 345 pliku `okno_glowne.hh`.

7.5.4.16 `QPushButton* OknoGlowne::loadButton` `[private]`

Wskaźnik na przycisk Wczytaj. Wczytuje symulację.

Definicja w linii 289 pliku `okno_glowne.hh`.

7.5.4.17 `QMenu* OknoGlowne::menu_Edit` `[private]`

Wskaźnik na akcję przycisku menu Edit.

Definicja w linii 226 pliku `okno_glowne.hh`.

7.5.4.18 `QMenu* OknoGlowne::menu_File` `[private]`

Wskaźnik na akcję przycisku menu File.

Definicja w linii 219 pliku `okno_glowne.hh`.

7.5.4.19 `QMenu* OknoGlowne::menu_Help` `[private]`

Wskaźnik na akcję przycisku menu Help.

Definicja w linii 233 pliku `okno_glowne.hh`.

7.5.4.20 QMenuBar* OknoGlowne::menuBar [private]

Wskaźnik na pasek menu.

Definicja w linii 198 pliku okno_glowne.hh.

7.5.4.21 QPushButton* OknoGlowne::pauseButton [private]

Wskaźnik na przycisk pause. Wstrzymuje symulację.

Definicja w linii 275 pliku okno_glowne.hh.

7.5.4.22 QPushButton* OknoGlowne::playButton [private]

Wskaźnik na przycisk play. Uruchamia symulację.

Definicja w linii 268 pliku okno_glowne.hh.

7.5.4.23 QSlider* OknoGlowne::sliderSzybkoscSym [private]

Wskaźnik na slider. Steruje szybkością symulacji.

Definicja w linii 296 pliku okno_glowne.hh.

7.5.4.24 QStatusBar* OknoGlowne::statusBar [private]

Wskaźnik na pasek statusowy.

Definicja w linii 240 pliku okno_glowne.hh.

7.5.4.25 QPushButton* OknoGlowne::stopButton [private]

Wskaźnik na przycisk stop. Zatrzymuje symulację.

Definicja w linii 282 pliku okno_glowne.hh.

7.5.4.26 QToolBar* OknoGlowne::toolBar [private]

Wskaźnik na pasek narzędziowy.

Definicja w linii 247 pliku okno_glowne.hh.

7.5.4.27 Zbiornik* OknoGlowne::wZbiornik [private]

Wskaźnik na zbiornik.

Definicja w linii 191 pliku okno_glowne.hh.

Dokumentacja dla tej klasy została wygenerowana z plików:

- [okno_glowne.hh](#)
- [moc_okno_glowne.cpp](#)
- [okno_glowne.cpp](#)

7.6 Dokumentacja struktury params_t

Atrybuty publiczne

- unsigned [nframes](#)
- unsigned [npframe](#)
- float [h](#)
- float [dt](#)
- float [rho0](#)
- float [k](#)
- float [mu](#)
- float [gx](#)
- float [gy](#)
- float [mass](#)

7.6.1 Opis szczegółowy

Definicja w linii 137 pliku simulation.cpp.

7.6.2 Dokumentacja atrybutów składowych

7.6.2.1 float params_t::dt

Definicja w linii 141 pliku simulation.cpp.

7.6.2.2 float params_t::gx

Definicja w linii 145 pliku simulation.cpp.

7.6.2.3 float params_t::gy

Definicja w linii 146 pliku simulation.cpp.

7.6.2.4 float params_t::h

Definicja w linii 140 pliku simulation.cpp.

7.6.2.5 float params_t::k

Definicja w linii 143 pliku simulation.cpp.

7.6.2.6 float params_t::mass

Definicja w linii 147 pliku simulation.cpp.

7.6.2.7 float params_t::mu

Definicja w linii 144 pliku simulation.cpp.

7.6.2.8 unsigned params_t::nframes

Definicja w linii 138 pliku simulation.cpp.

7.6.2.9 unsigned params_t::npframe

Definicja w linii 139 pliku simulation.cpp.

7.6.2.10 float params_t::rho0

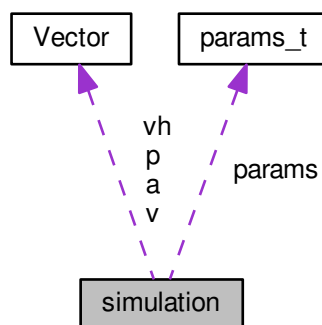
Definicja w linii 142 pliku simulation.cpp.

Dokumentacja dla tej struktury została wygenerowana z pliku:

- [simulation.cpp](#)

7.7 Dokumentacja klasy simulation

Diagram współpracy dla simulation:



Typy publiczne

- typedef bool(simulation::* [indicate_fun_t](#))(float, float)

Metody publiczne

- [simulation](#) (const unsigned &_n, [params_t](#) *_param)
konstruktor klasy
- void [compute_density](#) ([params_t](#) *_p)
funkcja
- void [compute_accel](#) ([params_t](#) *_params)
- void [damp_reflect_x](#) (float _barrier, unsigned i)
- void [damp_reflect_y](#) (float _barrier, unsigned i)
- void [reflect_bc](#) ()

- void `leapfrog_start` (float dt)
- void `leapfrog_step` (float dt)
- void `check_state` ()
- bool `box_indicator` (float x, float y)
- `simulation` & `place_particles` (`params_t` *`params`, `indicate_fun_t` `indicate_fun`)
- void `init` ()
- void `go` ()
- const unsigned & `getN` () const

Atrybuty prywatne

- unsigned `n`
Liczba cząstek.
- float * `rho`
Gęstości.
- `Vector` * `p`
Pozycje.
- `Vector` * `vh`
Prędkości (half step)
- `Vector` * `v`
Prędkości (full step)
- `params_t` * `params`
Przyspieszenia.

Przyjaciele

- std::ostream & `operator<<` (std::ostream &`_os`, const `simulation` &`_s`)

7.7.1 Opis szczegółowy

Definicja w linii 162 pliku simulation.cpp.

7.7.2 Dokumentacja składowych definicji typu

7.7.2.1 `typedef bool(simulation::* simulation::indicate_fun_t)(float, float)`

Definicja w linii 314 pliku simulation.cpp.

7.7.3 Dokumentacja konstruktora i destruktor

7.7.3.1 `simulation::simulation (const unsigned &_n, params_t *_param) [inline]`

Alokuję pamięć na potrzeby przechowania stanu symulacji

Parametry

<code>in</code>	<code>rozmiar</code>
-----------------	----------------------

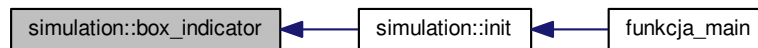
Definicja w linii 181 pliku simulation.cpp.

7.7.4 Dokumentacja funkcji składowych

7.7.4.1 `bool simulation::box_indicator (float x, float y) [inline]`

Definicja w linii 316 pliku simulation.cpp.

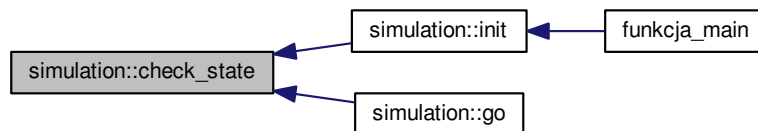
Oto graf wywołań tej funkcji:



7.7.4.2 `void simulation::check_state () [inline]`

Definicja w linii 306 pliku simulation.cpp.

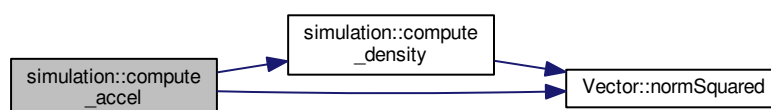
Oto graf wywołań tej funkcji:



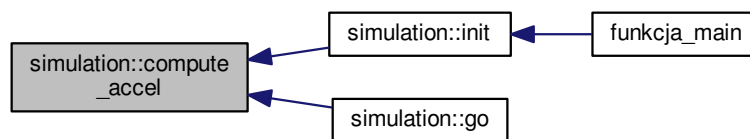
7.7.4.3 `void simulation::compute_accel (params_t * params) [inline]`

Definicja w linii 213 pliku simulation.cpp.

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



7.7.4.4 `void simulation::compute_density (params_t* _p) [inline]`

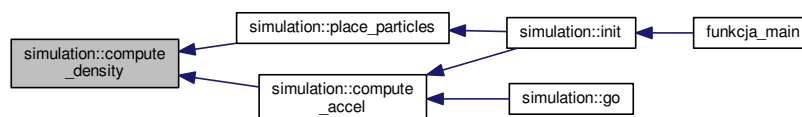
$$|I| = |\alpha|$$

Definicja w linii 191 pliku `simulation.cpp`.

Oto graf wywołań dla tej funkcji:



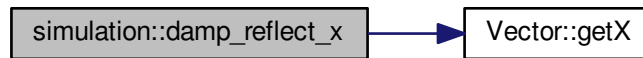
Oto graf wywoływań tej funkcji:



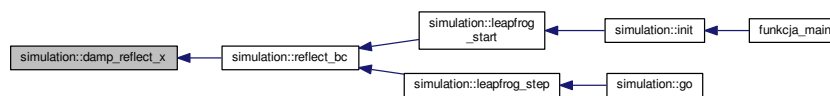
7.7.4.5 `void simulation::damp_reflect_x (float _barrier, unsigned i) [inline]`

Definicja w linii 252 pliku `simulation.cpp`.

Oto graf wywołań dla tej funkcji:



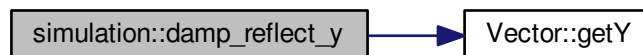
Oto graf wywoływań tej funkcji:



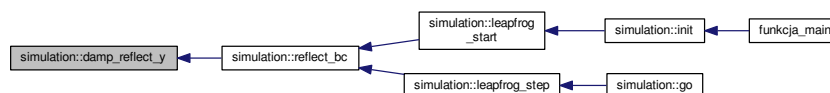
7.7.4.6 void simulation::damp_reflect_y (float *barrier*, unsigned *i*) [inline]

Definicja w linii 265 pliku `simulation.cpp`.

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



7.7.4.7 const unsigned& simulation::getN () const [inline]

Definicja w linii 356 pliku `simulation.cpp`.

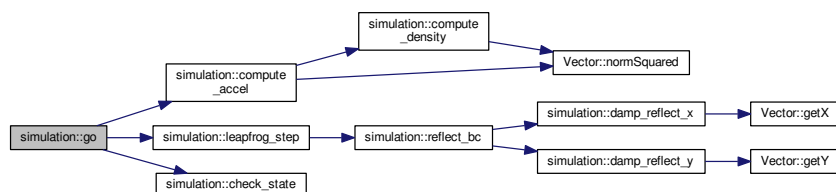
Oto graf wywołań tej funkcji:



7.7.4.8 void simulation::go () [inline]

Definicja w linii 349 pliku simulation.cpp.

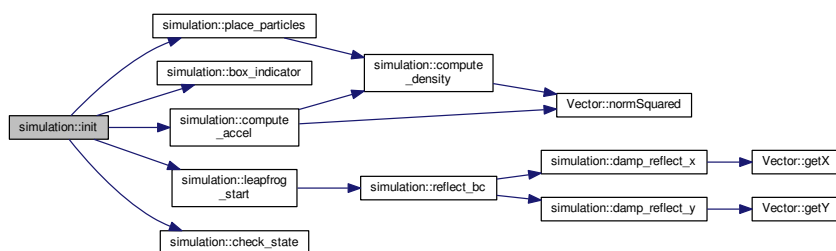
Oto graf wywołań dla tej funkcji:



7.7.4.9 void simulation::init () [inline]

Definicja w linii 343 pliku simulation.cpp.

Oto graf wywołań dla tej funkcji:



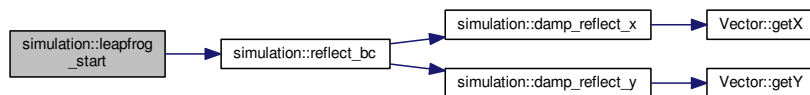
Oto graf wywołań tej funkcji:



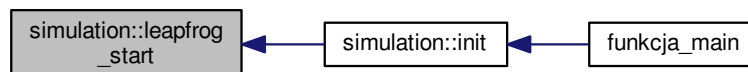
7.7.4.10 void simulation::leapfrog_start (float dt) [inline]

Definicja w linii 289 pliku simulation.cpp.

Oto graf wywołań dla tej funkcji:



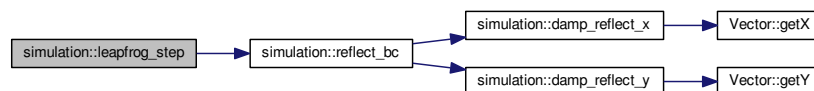
Oto graf wywołań tej funkcji:



7.7.4.11 void simulation::leapfrog_step (float dt) [inline]

Definicja w linii 297 pliku simulation.cpp.

Oto graf wywołań dla tej funkcji:



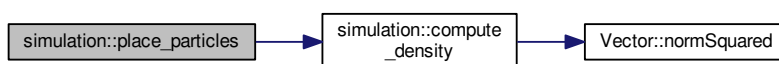
Oto graf wywoływań tej funkcji:



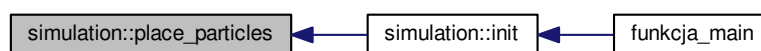
7.7.4.12 `simulation& simulation::place_particles (params_t * params, indicate_fun_t indicate_fun) [inline]`

Definicja w linii 320 pliku `simulation.cpp`.

Oto graf wywołań dla tej funkcji:



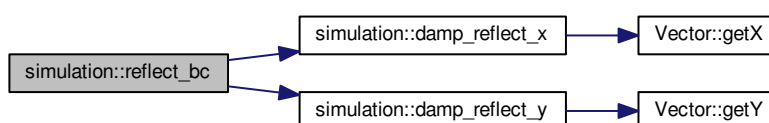
Oto graf wywoływań tej funkcji:



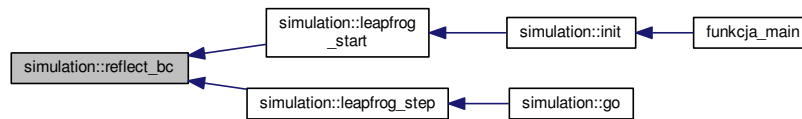
7.7.4.13 `void simulation::reflect_bc () [inline]`

Definicja w linii 279 pliku `simulation.cpp`.

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



7.7.5 Dokumentacja przyjaciół i funkcji związanych

7.7.5.1 `std::ostream& operator<< (std::ostream & _os, const simulation & _s)` `[friend]`

Definicja w linii 360 pliku simulation.cpp.

7.7.6 Dokumentacja atrybutów składowych

7.7.6.1 `Vector* simulation::a` `[private]`

Definicja w linii 170 pliku simulation.cpp.

7.7.6.2 `unsigned simulation::n` `[private]`

Definicja w linii 165 pliku simulation.cpp.

7.7.6.3 `Vector* simulation::p` `[private]`

Definicja w linii 167 pliku simulation.cpp.

7.7.6.4 `params_t* simulation::params` `[private]`

Definicja w linii 172 pliku simulation.cpp.

7.7.6.5 `float* simulation::rho` `[private]`

Definicja w linii 166 pliku simulation.cpp.

7.7.6.6 `Vector* simulation::v` `[private]`

Definicja w linii 169 pliku simulation.cpp.

7.7.6.7 `Vector* simulation::vh` `[private]`

Definicja w linii 168 pliku simulation.cpp.

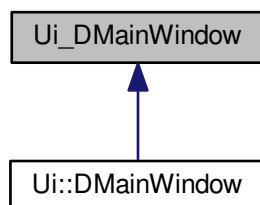
Dokumentacja dla tej klasy została wygenerowana z pliku:

- [simulation.cpp](#)

7.8 Dokumentacja klasy Ui_DMainWindow

```
#include <ui_dmainwindow.h>
```

Diagram dziedziczenia dla Ui_DMainWindow



Metody publiczne

- void [setupUi](#) (QMainWindow *[DMainWindow](#))
- void [retranslateUi](#) (QMainWindow *[DMainWindow](#))

Atrybuty publiczne

- QAction * [action_Save](#)
- QAction * [action_Exit](#)
- QAction * [actionExit](#)
- QAction * [actionPlay](#)
- QWidget * [centralWidget](#)
- QWidget * [horizontalLayoutWidget](#)
- QHBoxLayout * [horizontalLayout](#)
- QPushButton * [playButton](#)
- QSpacerItem * [verticalSpacer](#)
- QPushButton * [pauseButton](#)
- QPushButton * [stopButton](#)
- QSlider * [sliderSzybkoscSym](#)
- QLCDNumber * [lcdSzybkoscSym](#)
- QLineEdit * [lineSzybkoscSym](#)
- QLineEdit * [lineCzasSym](#)
- QLCDNumber * [lcdCzasSym](#)
- QLineEdit * [lineLiczbaCzasteczek](#)
- QLCDNumber * [lcdLiczbaCzasteczek](#)
- QLabel * [label](#)
- QMenuBar * [menuBar](#)
- QMenu * [menu_File](#)
- QMenu * [menu_Edit](#)
- QMenu * [menu_Help](#)
- QToolBar * [mainToolBar](#)
- QStatusBar * [statusBar](#)
- QToolBar * [toolBar](#)

7.8.1 Opis szczegółowy

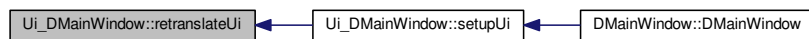
Definicja w linii 33 pliku ui_dmainwindow.h.

7.8.2 Dokumentacja funkcji składowych

7.8.2.1 `void Ui_DMainWindow::retranslateUi (QMainWindow * DMainWindow) [inline]`

Definicja w linii 177 pliku ui_dmainwindow.h.

Oto graf wywołań tej funkcji:



7.8.2.2 `void Ui_DMainWindow::setupUi (QMainWindow * DMainWindow) [inline]`

Definicja w linii 63 pliku ui_dmainwindow.h.

Oto graf wywołań dla tej funkcji:



Oto graf wywołań tej funkcji:



7.8.3 Dokumentacja atrybutów składowych

7.8.3.1 `QAction* Ui_DMainWindow::action_Exit`

Definicja w linii 37 pliku ui_dmainwindow.h.

7.8.3.2 `QAction* Ui_DMainWindow::action_Save`

Definicja w linii 36 pliku ui_dmainwindow.h.

7.8.3.3 QAction* Ui_DMainWindow::actionExit

Definicja w linii 38 pliku ui_dmainwindow.h.

7.8.3.4 QAction* Ui_DMainWindow::actionPlay

Definicja w linii 39 pliku ui_dmainwindow.h.

7.8.3.5 QWidget* Ui_DMainWindow::centralWidget

Definicja w linii 40 pliku ui_dmainwindow.h.

7.8.3.6 QHBoxLayout* Ui_DMainWindow::horizontalLayout

Definicja w linii 42 pliku ui_dmainwindow.h.

7.8.3.7 QWidget* Ui_DMainWindow::horizontalLayoutWidget

Definicja w linii 41 pliku ui_dmainwindow.h.

7.8.3.8 QLabel* Ui_DMainWindow::label

Definicja w linii 54 pliku ui_dmainwindow.h.

7.8.3.9 QLCDNumber* Ui_DMainWindow::lcdCzasSym

Definicja w linii 51 pliku ui_dmainwindow.h.

7.8.3.10 QLCDNumber* Ui_DMainWindow::lcdLiczbaCzasteczek

Definicja w linii 53 pliku ui_dmainwindow.h.

7.8.3.11 QLCDNumber* Ui_DMainWindow::lcdSzybkoscSym

Definicja w linii 48 pliku ui_dmainwindow.h.

7.8.3.12 QLineEdit* Ui_DMainWindow::lineCzasSym

Definicja w linii 50 pliku ui_dmainwindow.h.

7.8.3.13 QLineEdit* Ui_DMainWindow::lineLiczbaCzasteczek

Definicja w linii 52 pliku ui_dmainwindow.h.

7.8.3.14 QLineEdit* Ui_DMainWindow::lineSzybkoscSym

Definicja w linii 49 pliku ui_dmainwindow.h.

7.8.3.15 QToolBar* Ui_MainWindow::mainToolBar

Definicja w linii 59 pliku ui_dmainwindow.h.

7.8.3.16 QMenu* Ui_MainWindow::menu_Edit

Definicja w linii 57 pliku ui_dmainwindow.h.

7.8.3.17 QMenu* Ui_MainWindow::menu_File

Definicja w linii 56 pliku ui_dmainwindow.h.

7.8.3.18 QMenu* Ui_MainWindow::menu_Help

Definicja w linii 58 pliku ui_dmainwindow.h.

7.8.3.19 QMenuBar* Ui_MainWindow::menuBar

Definicja w linii 55 pliku ui_dmainwindow.h.

7.8.3.20 QPushButton* Ui_MainWindow::pauseButton

Definicja w linii 45 pliku ui_dmainwindow.h.

7.8.3.21 QPushButton* Ui_MainWindow::playButton

Definicja w linii 43 pliku ui_dmainwindow.h.

7.8.3.22 QSlider* Ui_MainWindow::sliderSzybkoscSym

Definicja w linii 47 pliku ui_dmainwindow.h.

7.8.3.23 QStatusBar* Ui_MainWindow::statusBar

Definicja w linii 60 pliku ui_dmainwindow.h.

7.8.3.24 QPushButton* Ui_MainWindow::stopButton

Definicja w linii 46 pliku ui_dmainwindow.h.

7.8.3.25 QToolBar* Ui_MainWindow::toolBar

Definicja w linii 61 pliku ui_dmainwindow.h.

7.8.3.26 QSpacerItem* Ui_MainWindow::verticalSpacer

Definicja w linii 44 pliku ui_dmainwindow.h.

Dokumentacja dla tej klasy została wygenerowana z pliku:

- [ui_dmainwindow.h](#)

7.9 Dokumentacja klasy Vector

klasa `Vector`

```
#include <vector.hh>
```

Metody publiczne

- `Vector` (float `_x`=0, float `_y`=0)
Współrzędne wektora.
- const float `getX` () const
zwraga pierwszą współrzędną wektora
- const float `getY` () const
pobiera drugą współrzędną wektora
- float & `getX` ()
pobiera pierwszą współrzędną wektora
- float & `getY` ()
pobiera drugą współrzędną wektora
- `Vector` & `operator+=` (const `Vector` & `_v`)
operator dodawania
- `Vector` & `operator-=` (const `Vector` & `_v`)
operator odejmowania
- `Vector` & `operator*=` (const float & `_c`)
operator skalowania
- `Vector operator+` (const `Vector` & `_v`) const
operator dodawania
- `Vector operator-` (const `Vector` & `_v`) const
operator odejmowania
- `Vector operator*` (const float & `_c`) const
operator skalowania
- `Vector` & `operator=` (const `Vector` & `_v`)
operator przypisania
- float `normSquared` ()
kwadrat normy
- `Vector` (float `_x`=0, float `_y`=0)
Współrzędne wektora.
- const float `getX` () const
zwraga pierwszą współrzędną wektora
- const float `getY` () const
pobiera drugą współrzędną wektora
- float & `getX` ()
pobiera pierwszą współrzędną wektora
- float & `getY` ()
pobiera drugą współrzędną wektora
- `Vector` & `operator+=` (const `Vector` & `_v`)
operator dodawania
- `Vector` & `operator-=` (const `Vector` & `_v`)
operator odejmowania
- `Vector` & `operator*=` (const float & `_c`)
operator skalowania
- `Vector operator+` (const `Vector` & `_v`) const

operator dodawania

- `Vector operator-` (const `Vector` &_v) const

operator odejmowania

- `Vector operator*` (const float &_c) const

operator skalowania

- `Vector & operator=` (const `Vector` &_v)

operator przypisania

- float `normSquared` ()

kwadrat normy

Atrybuty prywatne

- float `x`
- float `y`

Przyjaciele

- `std::ostream & operator<<` (std::ostream &_os, const `Vector` &_s)
- `std::ostream & operator<<` (std::ostream &_os, const `Vector` &_s)

7.9.1 Opis szczegółowy

Posiada metody do obsługi dwuelementowych wektorów o współrzędnych typu float

Definicja w linii 9 pliku vector.hh.

7.9.2 Dokumentacja konstruktora i destruktor

7.9.2.1 `Vector::Vector (float _x = 0, float _y = 0) [inline]`

konstruktor wektora

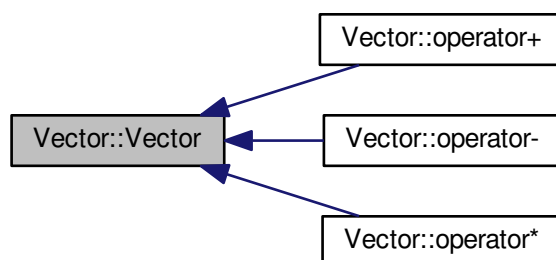
Ustawia współrzędne wektora

Parametry

in	<code>_x</code>	- pierwsza współrzędna (domyślnie: 0)
in	<code>_y</code>	- druga współrzędna (domyślnie: 0)

Definicja w linii 21 pliku vector.hh.

Oto graf wywoływań tej funkcji:



7.9.2.2 Vector::Vector (float _x = 0, float _y = 0) [inline]

konstruktor wektora

Ustawia współrzędne wektora

Parametry

in	_x	- pierwsza współrzędna (domyślnie: 0)
in	_y	- druga współrzędna (domyślnie: 0)

Definicja w linii 47 pliku simulation.cpp.

7.9.3 Dokumentacja funkcji składowych

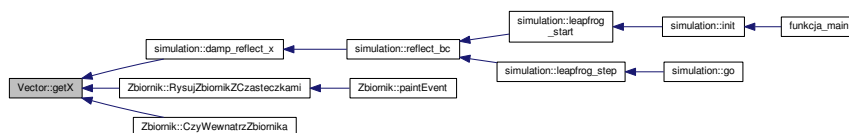
7.9.3.1 const float Vector::getX () const [inline]

Zwraca

pierwsza współrzędna wektora jako stała

Definicja w linii 26 pliku vector.hh.

Oto graf wywoływań tej funkcji:



7.9.3.2 float& Vector::getX () [inline]

Zwraca

pierwsza współrzędna wektora jako stała

Definicja w linii 36 pliku vector.hh.

7.9.3.3 const float Vector::getX () const [inline]**Zwraca**

pierwsza współrzędna wektora jako stała

Definicja w linii 52 pliku simulation.cpp.

7.9.3.4 float& Vector::getX () [inline]**Zwraca**

pierwsza współrzędna wektora jako stała

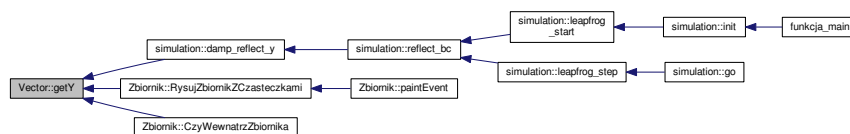
Definicja w linii 62 pliku simulation.cpp.

7.9.3.5 const float Vector::getY () const [inline]**Zwraca**

druga współrzędna wektora jako stała

Definicja w linii 31 pliku vector.hh.

Oto graf wywoływań tej funkcji:

**7.9.3.6 float& Vector::getY () [inline]****Zwraca**

druga współrzędna wektora jako stała

Definicja w linii 41 pliku vector.hh.

7.9.3.7 const float Vector::getY () const [inline]**Zwraca**

druga współrzędna wektora jako stała

Definicja w linii 57 pliku simulation.cpp.

7.9.3.8 float& Vector::getY () [inline]

Zwraca

druga współrzędna wektora jako stała

Definicja w linii 67 pliku simulation.cpp.

7.9.3.9 float Vector::normSquared () [inline]

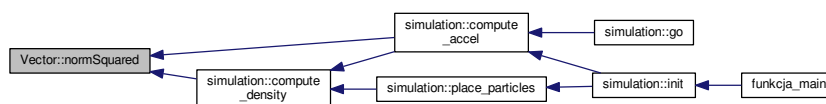
Zwraca sumę kwadratów współrzędnych wektora

Zwraca

wartość kwadratu normy

Definicja w linii 104 pliku vector.hh.

Oto graf wywoływań tej funkcji:



7.9.3.10 float Vector::normSquared () [inline]

Zwraca sumę kwadratów współrzędnych wektora

Zwraca

wartość kwadratu normy

Definicja w linii 130 pliku simulation.cpp.

7.9.3.11 Vector Vector::operator* (const float &_c) const [inline]

Skaluje wektor

Parametry

in	_c	- współczynnik skalowania
----	----	---------------------------

Zwraca

przeskalowany wektor

Definicja w linii 89 pliku vector.hh.

Oto graf wywołań dla tej funkcji:

**7.9.3.12 Vector Vector::operator*(const float &_c) const [inline]**

Skaluje wektor

Parametry

in	_c	- współczynnik skalowania
----	----	---------------------------

Zwraca

przeskalowany wektor

Definicja w linii 115 pliku simulation.cpp.

Oto graf wywołań dla tej funkcji:

**7.9.3.13 Vector& Vector::operator*=(const float &_c) [inline]**

Skaluje wektor

Parametry

in	_c	- współczynnik
----	----	----------------

Zwraca

referencja na pierwotny obiekt

Definicja w linii 65 pliku vector.hh.

7.9.3.14 `Vector& Vector::operator*=(const float &_c) [inline]`

Skaluje wektor

Parametry

<code>in</code>	<code>_c</code>	- współczynnik
-----------------	-----------------	----------------

Zwraca

referencja na pierwotny obiekt

Definicja w linii 91 pliku `simulation.cpp`.

7.9.3.15 `Vector Vector::operator+ (const Vector &_v) const` `[inline]`

Dodaje wartość dwóch wektorów

Parametry

<code>in</code>	<code>_v</code>	- inny wektor
-----------------	-----------------	---------------

Zwraca

wektor będący sumą

Definicja w linii 73 pliku `vector.hh`.

Oto graf wywołań dla tej funkcji:

**7.9.3.16** `Vector Vector::operator+ (const Vector &_v) const` `[inline]`

Dodaje wartość dwóch wektorów

Parametry

<code>in</code>	<code>_v</code>	- inny wektor
-----------------	-----------------	---------------

Zwraca

wektor będący sumą

Definicja w linii 99 pliku simulation.cpp.

Oto graf wywołań dla tej funkcji:

**7.9.3.17 Vector& Vector::operator+=(const Vector &_v) [inline]**

Dodaje wartość innego wektora do klasy

Parametry

in	_v	- inny wektor
----	----	---------------

Zwraca

referencja na pierwotny obiekt

Definicja w linii 49 pliku vector.hh.

7.9.3.18 Vector& Vector::operator+=(const Vector &_v) [inline]

Dodaje wartość innego wektora do klasy

Parametry

in	_v	- inny wektor
----	----	---------------

Zwraca

referencja na pierwotny obiekt

Definicja w linii 75 pliku simulation.cpp.

7.9.3.19 Vector Vector::operator- (const Vector &_v) const [inline]

Odejmuje wartość innego wektora od klasy

Parametry

in	_v	- inny wektor
----	----	---------------

Zwraca

wektor będący różnicą

Definicja w linii 81 pliku vector.hh.

Oto graf wywołań dla tej funkcji:

**7.9.3.20 Vector Vector::operator- (const Vector &_v) const [inline]**

Odejmuje wartość innego wektora od klasy

Parametry

in	_v	- inny wektor
----	----	---------------

Zwraca

wektor będący różnicą

Definicja w linii 107 pliku simulation.cpp.

Oto graf wywołań dla tej funkcji:

**7.9.3.21 Vector& Vector::operator-= (const Vector &_v) [inline]**

Odejmuje wartość innego wektora od klasy

Parametry

in	_v	- inny wektor
----	----	---------------

Zwraca

referencja na pierwotny obiekt

Definicja w linii 57 pliku vector.hh.

7.9.3.22 **Vector& Vector::operator= (const **Vector** &_v)** [inline]

Odejmuje wartość innego wektora od klasy

Parametry

<code>in</code>	<code>_v</code>	- inny wektor
-----------------	-----------------	---------------

Zwraca

referencja na pierwotny obiekt

Definicja w linii 83 pliku `simulation.cpp`.

7.9.3.23 `Vector& Vector::operator=(const Vector &_v) [inline]`

Przypisuje współrzędne innego wektora do klasy

Parametry

<code>in</code>	<code>_v</code>	- inny wektor
-----------------	-----------------	---------------

Zwraca

referencja na pierwotny obiekt

Definicja w linii 97 pliku `vector.hh`.

7.9.3.24 `Vector& Vector::operator=(const Vector &_v) [inline]`

Przypisuje współrzędne innego wektora do klasy

Parametry

<code>in</code>	<code>_v</code>	- inny wektor
-----------------	-----------------	---------------

Zwraca

referencja na pierwotny obiekt

Definicja w linii 123 pliku `simulation.cpp`.

7.9.4 Dokumentacja przyjaciół i funkcji związanych**7.9.4.1 `std::ostream& operator<< (std::ostream &_os, const Vector &_s) [friend]`**

Definicja w linii 133 pliku `simulation.cpp`.

7.9.4.2 `std::ostream& operator<< (std::ostream &_os, const Vector &_s) [friend]`

Definicja w linii 133 pliku `simulation.cpp`.

7.9.5 Dokumentacja atrybutów składowych**7.9.5.1 `float Vector::x [private]`**

Definicja w linii 11 pliku `vector.hh`.

7.9.5.2 float Vector::y [private]

Definicja w linii 11 pliku vector.hh.

Dokumentacja dla tej klasy została wygenerowana z plików:

- [vector.hh](#)
- [simulation.cpp](#)

7.10 Dokumentacja klasy Zbiornik

Klasa modelująca zbiornik.

```
#include <zbiornik.hh>
```

Diagram dziedziczenia dla Zbiornik

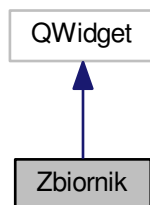
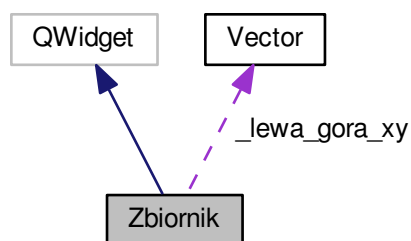


Diagram współpracy dla Zbiornik:



Sloty publiczne

- void [GdyOdpowiedniCzas](#) ()

Slot odpowiadający za aktualizację danych. .

Sygnaly

- void [ZglosNapis](#) (const QString &
Sygnal zgłaszający napis.
- void [ZglosLiczbeCzasteczek](#) (const int)
Sygnal zgłaszający liczbę czasteczek.
- void [ZglosCzasSymulacji](#) (const double)
Sygnal zgłaszający czas trwania symulacji.

Metody publiczne

- [Zbiornik](#) (QWidget *wRodzic, const [Vector](#) &[lewa_gora_xy](#), const double [podstawa](#), const double [wysokosc](#), const double [grubosc](#))
Konstruktor parametryczny.
- virtual void [paintEvent](#) (QPaintEvent *event)
Wirtualna metoda paintEvent wyrysowująca obiekt na ekranie.
- void [RysujZbiornik](#) (QPainter &Rysownik, const int Podstawa, const int Wysokosc, const int Grubosc, const int x, const int y)
Metoda rysująca zbiornik.
- void [RysujZbiornikZCzasteczkami](#) (QPainter &Rysownik)
Metoda rysująca zbiornik wraz z czasteczkami.
- bool [CzyWewnatrzZbiornika](#) (const double x, const double y) const
Metoda sprawdzająca czy punkt znajduje się wewnątrz zbiornika.
- bool [CzyWewnatrzZbiornika](#) (const [Vector](#) &xy) const
Metoda sprawdzająca czy punkt znajduje się wewnątrz zbiornika.
- bool [CzyWewnatrzZbiornika](#) (const [Czasteczka](#) &cz) const
Metoda sprawdzająca czy czasteczka znajduje się wewnątrz zbiornika.
- [Vector](#) [lewa_gora_xy](#) () const
Interfejs pozwalający na odczyt prywatnych danych.
- [Vector](#) & [lewa_gora_xy](#) ()
Interfejs pozwalający na zmianę prywatnych danych.
- double [podstawa](#) () const
Interfejs pozwalający na odczyt prywatnych danych.
- double & [podstawa](#) ()
Interfejs pozwalający na zmianę prywatnych danych.
- double [wysokosc](#) () const
Interfejs pozwalający na odczyt prywatnych danych.
- double & [wysokosc](#) ()
Interfejs pozwalający na zmianę prywatnych danych.
- double [grubosc](#) () const
Interfejs pozwalający na odczyt prywatnych danych.
- double & [grubosc](#) ()
Interfejs pozwalający na zmianę prywatnych danych.
- double [czas_sym](#) () const
Interfejs pozwalający na odczyt prywatnych danych.
- double & [czas_sym](#) ()
Interfejs pozwalający na zmianę prywatnych danych.
- int [odpowiedni_czas](#) () const
Interfejs pozwalający na odczyt prywatnych danych.
- int & [odpowiedni_czas](#) ()
Interfejs pozwalający na zmianę prywatnych danych.

Atrybuty publiczne

- `std::list< Czasteczka > Czasteczki`
Lista wszystkich czasteczek.
- `QTimer _Stoper`
Miernik czasu dla zbiornika.

Atrybuty prywatne

- `Vector _lewa_gora_xy`
Wektor polozenia lewego gornego punktu zbiornika.
- `double _podstawa`
Dlugosc podstawy zbiornika.
- `double _wysokosc`
Wysokosc zbiornika.
- `double _grubosc`
Grubosc sciany zbiornika.
- `double _czas_sym`
Miernik czasu dla zbiornika.
- `int _odpowiedni_czas`
Interwal dla timeout'ow z timera [ms].

7.10.1 Opis szczegółowy

Dzięki tej klasie możliwe jest wyrysowywanie na ekranie zbiornika.

Definicja w linii 45 pliku `zbiornik.hh`.

7.10.2 Dokumentacja konstruktora i destruktoru

7.10.2.1 `Zbiornik::Zbiornik (QWidget * wRodzic, const Vector & lewa_gora_xy, const double podstawa, const double wysokosc, const double grubosc)`

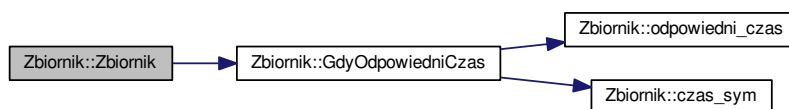
Konstruktor parametryczny.

Parametry

<code>in, out</code>	<code>wRodzic</code>	- wskaźnik na rodzica
<code>in</code>	<code>lewa_gora_xy</code>	- położenie lewego gornego rogu zbiornika
<code>in</code>	<code>podstawa</code>	- długość podstawy zbiornika
<code>in</code>	<code>wysokosc</code>	- wysokość zbiornika
<code>in</code>	<code>grubosc</code>	- grubość ściany zbiornika

Definicja w linii 23 pliku `zbiornik.cpp`.

Oto graf wywołań dla tej funkcji:



7.10.3 Dokumentacja funkcji składowych

7.10.3.1 `double Zbiornik::czas_sym () const [inline]`

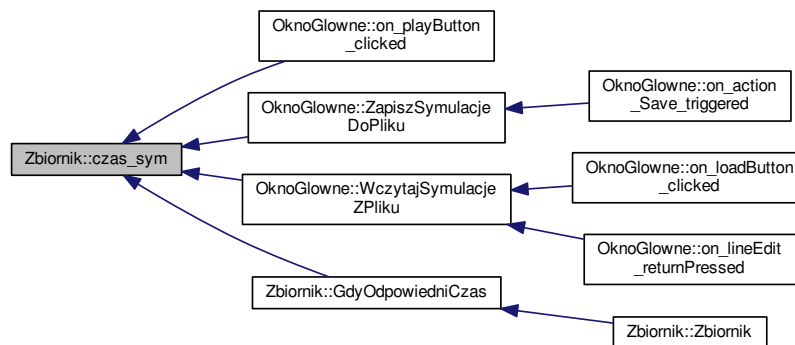
Interfejs pozwalający na odczyt prywatnych danych.

Zwraca

`_czas_sym` - prywatny atrybut opisujący pomiar czasu

Definicja w linii 191 pliku `zbiornik.hh`.

Oto graf wywołań tej funkcji:



7.10.3.2 `double& Zbiornik::czas_sym () [inline]`

Interfejs pozwalający na zmianę prywatnych danych.

Zwraca

`_czas_sym` - referencja na prywatny atrybut opisujący pomiar czasu

Definicja w linii 198 pliku `zbiornik.hh`.

7.10.3.3 `bool Zbiornik::CzyWewnatrzZbiornika (const double x, const double y) const`

Sprawdza, czy punkt znajduje się wewnątrz zbiornika.

Parametry

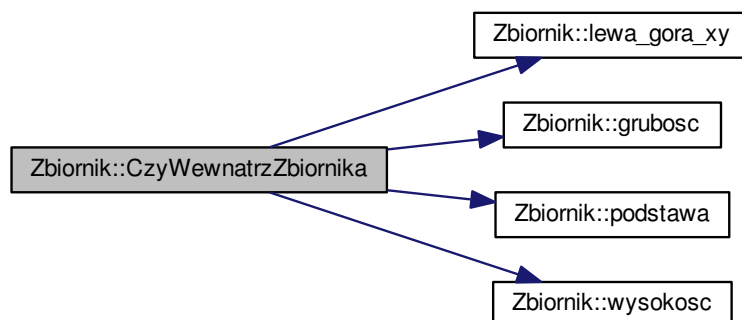
<code>in</code>	<code>x</code>	- położenie punktu na osi X,
<code>in</code>	<code>y</code>	- położenie punktu na osi Y,

Zwraca

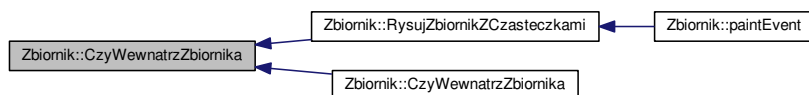
`true` - jeśli znajduje się wewnątrz zbiornika,
`false` - jeśli nie znajduje się wewnątrz zbiornika.

Definicja w linii 72 pliku `zbiornik.cpp`.

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



7.10.3.4 `bool Zbiornik::CzyWewnatrzZbiornika (const Vector & xy) const`

Sprawdza, czy punkt znajduje sie wewnatrz zbiornika.

Parametry

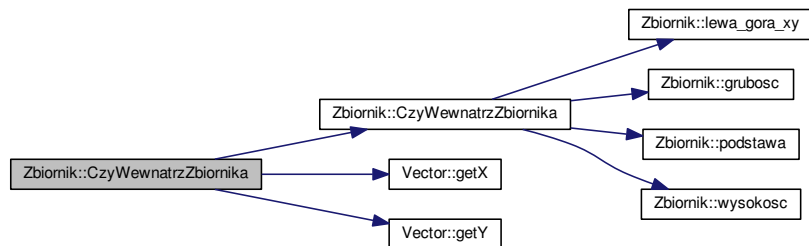
<code>in</code>	<code>xy</code>	- położenie punktu
-----------------	-----------------	--------------------

Zwraca

true - jeśli znajduje się wewnątrz zbiornika,
false - jeśli nie znajduje się wewnątrz zbiornika.

Definicja w linii 81 pliku zbiornik.cpp.

Oto graf wywołań dla tej funkcji:



7.10.3.5 bool Zbiornik::CzyWewnatrzZbiornika (const Czasteczka & cz) const

Sprawdza, czy czasteczka znajduje się wewnątrz zbiornika.

Parametry

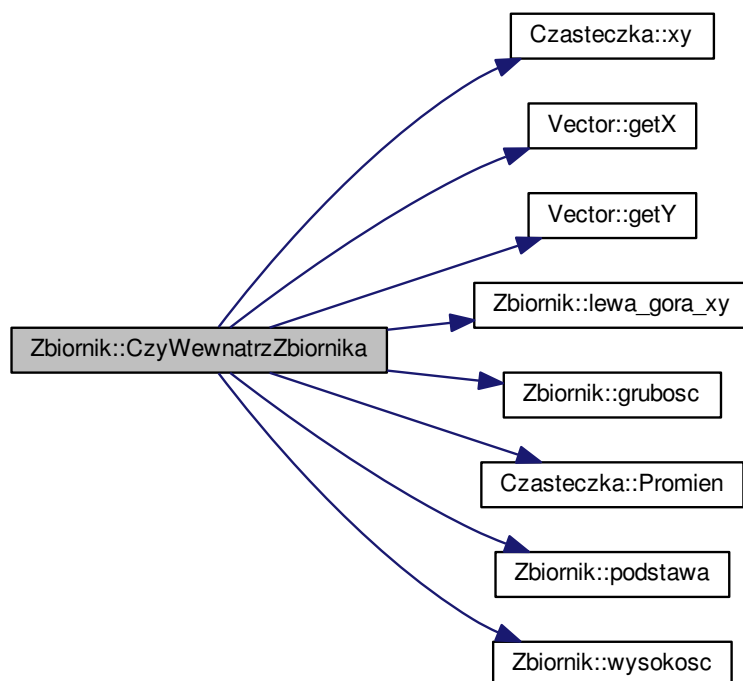
in	cz	- czasteczka do sprawdzenia
----	----	-----------------------------

Zwraca

true - jeśli znajduje się wewnątrz zbiornika,
false - jeśli nie znajduje się wewnątrz zbiornika.

Definicja w linii 89 pliku zbiornik.cpp.

Oto graf wywołań dla tej funkcji:

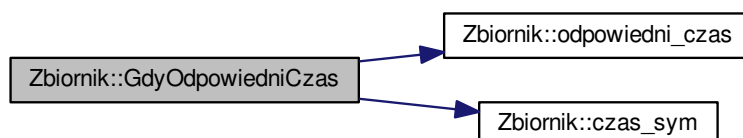


7.10.3.6 void Zbiornik::GdyOdpowiedniCzas () [slot]

Odpowiada za uaktualnienie zbiornika w odpowiednich momentach.

Definicja w linii 110 pliku `zbiornik.cpp`.

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



7.10.3.7 `double Zbiornik::grubosc () const [inline]`

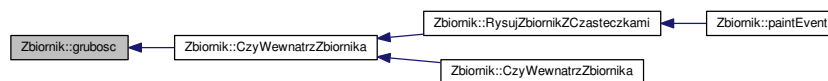
Interfejs pozwalający na odczyt prywatnych danych.

Zwraca

`_grubosc` - prywatny atrybut opisujący grubosc sciany

Definicja w linii 176 pliku `zbiornik.hh`.

Oto graf wywoływań tej funkcji:



7.10.3.8 `double& Zbiornik::grubosc () [inline]`

Interfejs pozwalający na zmianę prywatnych danych.

Zwraca

`_grubosc` - referencja na prywatny atrybut opisujący grubosc sciany

Definicja w linii 183 pliku `zbiornik.hh`.

7.10.3.9 `Vector Zbiornik::lewa_gora_xy () const [inline]`

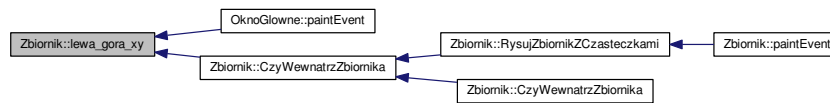
Interfejs pozwalający na odczyt prywatnych danych.

Zwraca

`_lewa_gora_xy` - prywatny atrybut opisujący wektor położenia lewego górnego punktu zbiornika

Definicja w linii 134 pliku `zbiornik.hh`.

Oto graf wywoływań tej funkcji:



7.10.3.10 Vector& Zbiornik::lewa_gora_xy () [inline]

Interfejs pozwalający na zmianę prywatnych danych.

Zwraca

_podstawa - referencja na prywatny atrybut opisujący wektor położenia lewego górnego punktu zbiornika

Definicja w linii 141 pliku zbiornik.hh.

7.10.3.11 int Zbiornik::odpowiedni_czas () const [inline]

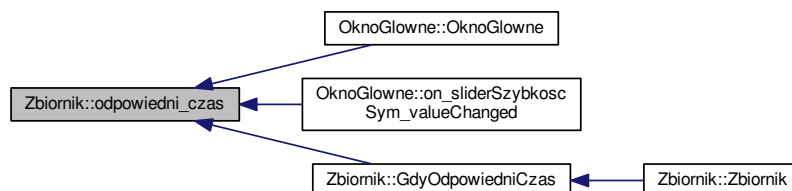
Interfejs pozwalający na odczyt prywatnych danych.

Zwraca

odpowiedni_czas - prywatny atrybut opisujący czas trwania symulacji

Definicja w linii 206 pliku zbiornik.hh.

Oto graf wywoływań tej funkcji:



7.10.3.12 int& Zbiornik::odpowiedni_czas () [inline]

Interfejs pozwalający na zmianę prywatnych danych.

Zwraca

odpowiedni_czas - referencja na prywatny atrybut opisujący czas trwania symulacji

Definicja w linii 213 pliku zbiornik.hh.

7.10.3.13 void Zbiornik::paintEvent (QPaintEvent * event) [virtual]

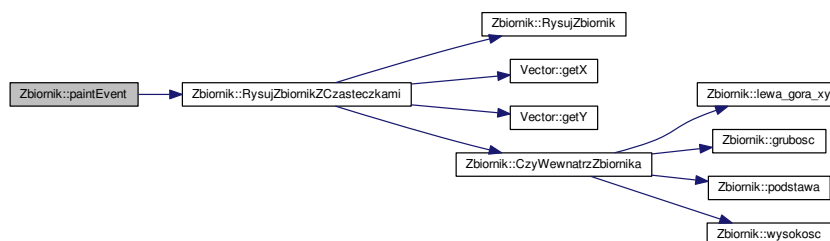
Odziedziczona wirtualna metoda paintEvent.

Parametry

<code>in, out</code>	<code>event</code>	- wskaźnik obiekt klasy <code>QPaintEvent</code>
----------------------	--------------------	--------------------------------------------------

Definicja w linii 104 pliku `zbiornik.cpp`.

Oto graf wywołań dla tej funkcji:



7.10.3.14 `double Zbiornik::podstawa () const [inline]`

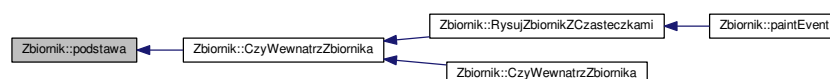
Interfejs pozwalający na odczyt prywatnych danych.

Zwraca

`_podstawa` - prywatny atrybut opisujący długość podstaw

Definicja w linii 148 pliku `zbiornik.hh`.

Oto graf wywoływań tej funkcji:



7.10.3.15 `double& Zbiornik::podstawa () [inline]`

Interfejs pozwalający na zmianę prywatnych danych.

Zwraca

`_podstawa` - referencja na prywatny atrybut opisujący długość podstawy

Definicja w linii 155 pliku `zbiornik.hh`.

7.10.3.16 `void Zbiornik::RysujZbiornik (QPainter & Rysownik, const int Podstawa, const int Wysokosc, const int Grubosc, const int x, const int y)`

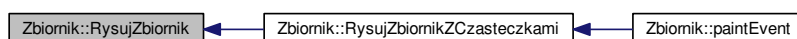
Rysuje zbiornik o zadanych parametrach.

Parametry

in, out	<i>Rysownik</i>	- referencja na obiekt klasy QPainter
in	<i>Podstawa</i>	- dlugosc podstawy zbiornika
in	<i>Wysokosc</i>	- wysokosc zbiornika
in	<i>Grubosc</i>	- grubosc wyrysowywanego odcinka
in	<i>x</i>	- polozenie lewego gornego punktu zbiornika na osi x
in	<i>y</i>	- polozenie lewego gornego punktu zbiornika na osi y

Definicja w linii 40 pliku zbiornik.cpp.

Oto graf wywoływań tej funkcji:



7.10.3.17 void Zbiornik::RysujZbiornikZCzasteczkami (QPainter & Rysownik)

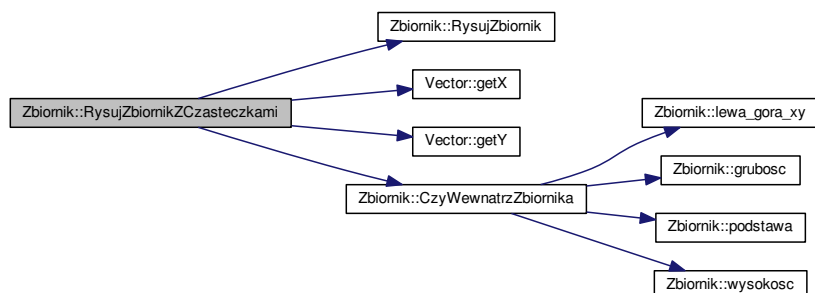
Rysuje zbiornik oraz czasteczki.

Parametry

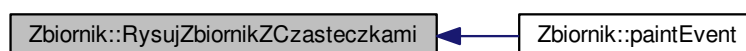
in, out	<i>Rysownik</i>	- referencja na obiekt klasy QPainter
---------	-----------------	---------------------------------------

Definicja w linii 56 pliku zbiornik.cpp.

Oto graf wywołań dla tej funkcji:



Oto graf wywoływań tej funkcji:



7.10.3.18 `double Zbiornik::wysokosc () const [inline]`

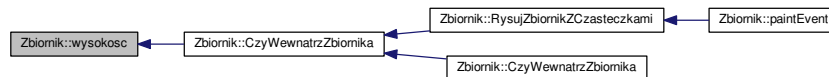
Interfejs pozwalający na odczyt prywatnych danych.

Zwraca

`_podstawa` - prywatny atrybut opisujący wysokosc

Definicja w linii 162 pliku `zbiornik.hh`.

Oto graf wywołań tej funkcji:



7.10.3.19 `double& Zbiornik::wysokosc () [inline]`

Interfejs pozwalający na zmianę prywatnych danych.

Zwraca

`_podstawa` - referencja na prywatny atrybut opisujący wysokosc

Definicja w linii 169 pliku `zbiornik.hh`.

7.10.3.20 `void Zbiornik::ZglosCzasSymulacji (const double _t1) [signal]`

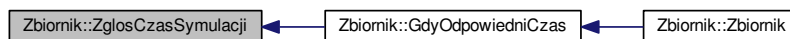
Sygnal zgłaszający czas trwania symulacji do odpowiedniego slotu.

Parametry

<code>in</code>	<code>_t1</code>	- czas do zgłoszenia
-----------------	------------------	----------------------

Definicja w linii 119 pliku `moc_zbiornik.cpp`.

Oto graf wywołań tej funkcji:



7.10.3.21 `void Zbiornik::ZglosLiczbeCzasteczek (const int _t1) [signal]`

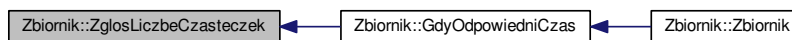
Sygnal zgłaszający liczbę czasteczek do odpowiedniego slotu.

Parametry

in	_t1	- liczba czasteczek do zgloszenia
----	-----	-----------------------------------

Definicja w linii 112 pliku moc_zbiornik.cpp.

Oto graf wywoływań tej funkcji:



7.10.3.22 void Zbiornik::ZglosNapis (const QString & _t1) [signal]

Sygnal zgłaszający napis do odpowiedniego slotu.

Parametry

in	_t1	- napis do zgloszenia
----	-----	-----------------------

Definicja w linii 105 pliku moc_zbiornik.cpp.

7.10.4 Dokumentacja atrybutów składowych

7.10.4.1 double Zbiornik::_czas_sym [private]

Miernik czasu zbiornika. Służy do mierzenia i wyświetlania czasu.

Definicja w linii 294 pliku zbiornik.hh.

7.10.4.2 double Zbiornik::_grubosc [private]

Grubość ściany zbiornika.

Definicja w linii 287 pliku zbiornik.hh.

7.10.4.3 Vector Zbiornik::_lewa_gora_xy [private]

Wektor położenia lewego górnego punktu zbiornika.

Definicja w linii 269 pliku zbiornik.hh.

7.10.4.4 int Zbiornik::_odpowiedni_czas [private]

Interwał dla timeout'ów z timera [ms].

Definicja w linii 300 pliku zbiornik.hh.

7.10.4.5 double Zbiornik::_podstawa [private]

Długość podstawy zbiornika.

Definicja w linii 275 pliku zbiornik.hh.

7.10.4.6 QTimer Zbiornik::_Stoper

Miernik czasu zbiornika. Służy do odswieżania ekranu.

Definicja w linii 261 pliku zbiornik.hh.

7.10.4.7 double Zbiornik::_wysokosc [private]

Wysokosc zbiornika.

Definicja w linii 281 pliku zbiornik.hh.

7.10.4.8 std::list<Czasteczka> Zbiornik::Czasteczki

Lista wszystkich czasteczek.

Definicja w linii 254 pliku zbiornik.hh.

Dokumentacja dla tej klasy została wygenerowana z plików:

- [zbiornik.hh](#)
- [moc_zbiornik.cpp](#)
- [zbiornik.cpp](#)

Rozdział 8

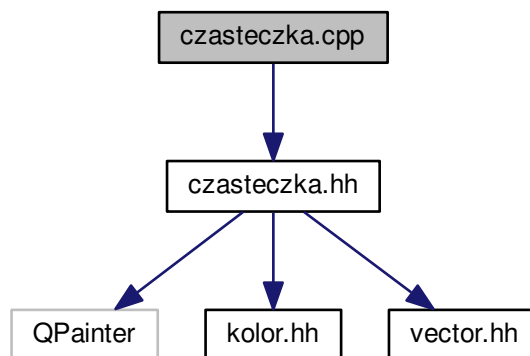
Dokumentacja plików

8.1 Dokumentacja pliku czasteczka.cpp

Zawiera definicje metod klasy [Czasteczka](#).

```
#include "czasteczka.hh"
```

Wykres zależności załączania dla czasteczka.cpp:



8.1.1 Opis szczegółowy

W pliku znajdują się:

- definicje konstruktorów, metod i przeciążeń klasy [Czasteczka](#).

Definicja w pliku [czasteczka.cpp](#).

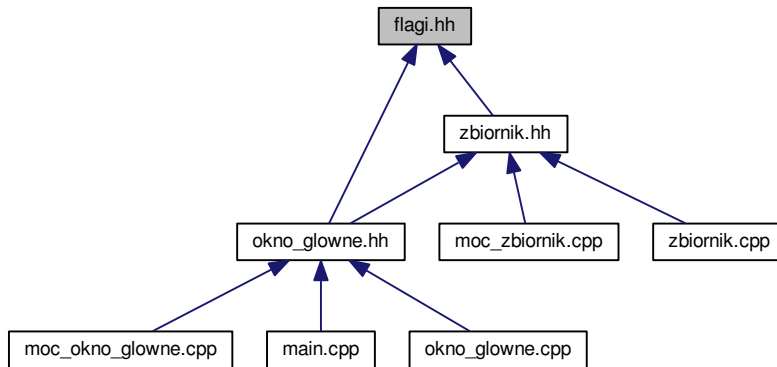
8.2 Dokumentacja pliku czasteczka.hh

Zawiera definicje klasy [Czasteczka](#) oraz deklaracje jej metod.

8.5 Dokumentacja pliku flagi.hh

Zawiera globalne zmienne opisujące symulacje.

Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



Wyliczenia

- enum { **eSTOP**, **ePAUSE**, **ePLAY** }

Typ wyliczeniowy określający stany symulacji.

Zmienne

- int **STAN**
Stan symulacji.
- const int **PROMIEN** = 5
Promień czasteczki.
- const int **PODSTAWA** = 200
Długość podstawy zbiornika.
- const int **WYSOKOSC** = 200
Wysokość zbiornika.
- const int **GRUBOSC** = 10
Grubość krawędzi zbiornika.

8.5.1 Opis szczegółowy

W pliku znajdują się:

- zmienne opisujące stan symulacji,
- początkowy promień czasteczek,
- założone wymiary zbiornika,

Definicja w pliku **flagi.hh**.

8.5.2 Dokumentacja typów wyliczanych

8.5.2.1 anonymous enum

Stany symulacji to Play/Pause/Stop.

Wartości wyliczeń

eSTOP

ePAUSE

ePLAY

Definicja w linii 27 pliku flagi.hh.

8.5.3 Dokumentacja zmiennych

8.5.3.1 const int GRUBOSC = 10

Grubosc krawedzi zbiornika. Grubosc jest symetryczna wzgledem osi.

Definicja w linii 59 pliku flagi.hh.

8.5.3.2 const int PODSTAWA = 200

Dlugosc podstawy zbiornika.

Definicja w linii 44 pliku flagi.hh.

8.5.3.3 const int PROMIEN = 5

Promien czasteczki.

Definicja w linii 37 pliku flagi.hh.

8.5.3.4 int STAN

Stan symulacji to Play/Pause/Stop. Inicjalizacja w pliku [zbiornik.cpp](#). Wybór początkowego stanu w konstruktorze okienka.

Stan symulacji.

Początkowy stan symulacji to Play/Pauza/Stop.

Definicja w linii 21 pliku zbiornik.cpp.

8.5.3.5 const int WYSOKOSC = 200

Wysokosc zbiornika.

Definicja w linii 51 pliku flagi.hh.

8.6 Dokumentacja pliku kolor.hh

Zawiera definicje klasy [Kolor](#) oraz deklaracje jej metod.

Funkcje

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

8.7.1 Opis szczegółowy

W funkcji `main` wyróżnia się:

- utworzenie obiektu klasy `QApplication` i wstępne przetworzenie argumentów z linii wywołania,
- utworzenie okna aplikacji,
- wymuszenie ukazania się okna,
- uruchomienie obsługi petli zdarzeń dla całej aplikacji.

Definicja w pliku `main.cpp`.

8.7.2 Dokumentacja funkcji

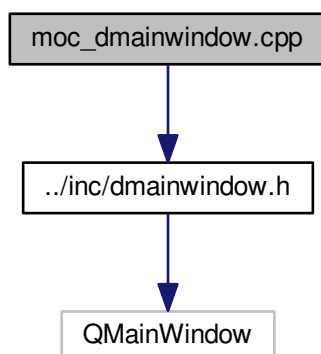
8.7.2.1 int main (int argc, char * argv[])

Definicja w linii 19 pliku `main.cpp`.

8.8 Dokumentacja pliku moc_dmainwindow.cpp

```
#include "../inc/dmainwindow.h"
```

Wykres zależności załączania dla `moc_dmainwindow.cpp`:

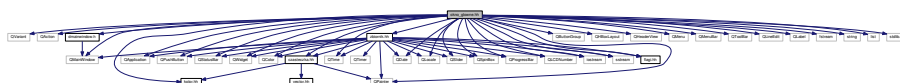



```

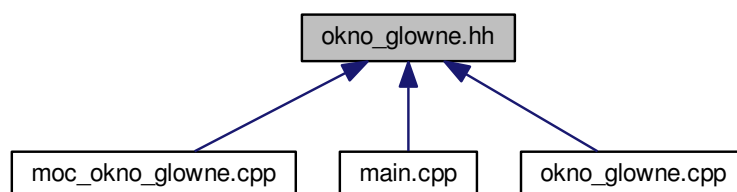
#include <QVariant>
#include <QAction>
#include <QApplication>
#include <QButtonGroup>
#include <QHBoxLayout>
#include <QHeaderView>
#include <QMainWindow>
#include <QMenu>
#include <QMenuBar>
#include <QPushButton>
#include <QStatusBar>
#include <QToolBar>
#include <QWidget>
#include <QColor>
#include <QPainter>
#include <QTime>
#include <QDate>
#include <QLocale>
#include <QSlider>
#include <QSpinBox>
#include <QProgressBar>
#include <QLCDNumber>
#include <QLineEdit>
#include <QLabel>
#include <iostream>
#include <sstream>
#include <fstream>
#include <string>
#include <list>
#include <stdlib.h>
#include "flagi.hh"
#include "kolor.hh"
#include "czasteczka.hh"
#include "zbiornik.hh"
#include "dmainwindow.h"

```

Wykres zależności załączania dla okno_glowne.hh:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



Komponenty

- class [OknoGlowne](#)

Klasa modelująca główne okno aplikacji.

8.12.1 Opis szczegółowy

W pliku znajdują się:

- definicja klasy [OknoGlowne](#) (modeluje główne okno aplikacji),
- deklaracje konstruktorów, metod i przeciążeń ww. klasy.

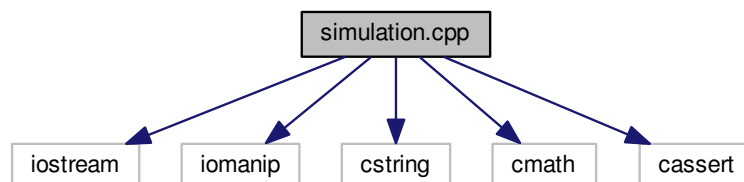
Definicja w pliku [okno_glowne.hh](#).

8.13 Dokumentacja pliku simulation.cpp

Plik z kodem symulatora cieczy.

```
#include <iostream>
#include <iomanip>
#include <cstring>
#include <cmath>
#include <cassert>
```

Wykres zależności załączania dla simulation.cpp:



Komponenty

- class [Vector](#)
klasa [Vector](#)
- struct [params_t](#)
- class [simulation](#)

Definicje

- `#define` [DAMP](#) 0.75
- `#define` [XMAX](#) 2.0
- `#define` [YMAX](#) 2.0
- `#define` [LOG](#)(MSG)
makro do logowania

Funkcje

- `std::ostream & operator<< (std::ostream &_os, const Vector &_s)`
- `void setup (params_t *params)`
- `std::ostream & operator<< (std::ostream &_os, const simulation &_s)`
- `int funkcja_main ()`

8.13.1 Opis szczegółowy

Zawiera definicje klas i funkcji pozwalających na przeprowadzenie symulacji zachowania sparymetryzowanego modelu cieczy w zadanym środowisku od określonego warunku początkowego.

Definicja w pliku `simulation.cpp`.

8.13.2 Dokumentacja definicji

8.13.2.1 #define DAMP 0.75

Definicja w linii 6 pliku `simulation.cpp`.

8.13.2.2 #define LOG(MSG)

Wartość:

```
do { std::cerr << "\e[32m" << __FILE__ << "\e[31m:\e[33m" << __LINE__ \
    << "\e[35m:\e[34m" << std::setw(16) << __FUNCTION__ << "\e[36m : \e[37m" << MSG << "\e[0m\n"; } while(
    false )
```

Łatwe w użyciu makro poprawiające czytelność komunikatów. Pętla do...while zastosowana aby wymusić poprawne użycie średnika po poleceniu.

Parametry

<code>in</code>	<code>MSG</code>	- wiadomość do wyświetlenia
-----------------	------------------	-----------------------------

Definicja w linii 18 pliku `simulation.cpp`.

8.13.2.3 #define XMAX 2.0

Definicja w linii 7 pliku `simulation.cpp`.

8.13.2.4 #define YMAX 2.0

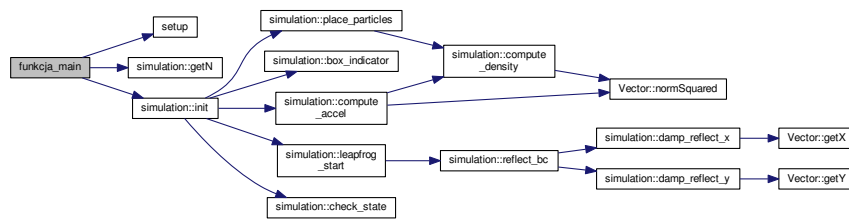
Definicja w linii 8 pliku `simulation.cpp`.

8.13.3 Dokumentacja funkcji

8.13.3.1 int funkcja_main ()

Definicja w linii 369 pliku `simulation.cpp`.

Oto graf wywołań dla tej funkcji:



8.13.3.2 `std::ostream& operator<< (std::ostream & _os, const Vector & _s)`

Definicja w linii 133 pliku simulation.cpp.

8.13.3.3 `std::ostream& operator<< (std::ostream & _os, const simulation & _s)`

Definicja w linii 360 pliku simulation.cpp.

8.13.3.4 `void setup (params_t * params)`

Definicja w linii 150 pliku simulation.cpp.

Oto graf wywoływań tej funkcji:



8.14 Dokumentacja pliku strona.dox

8.15 Dokumentacja pliku ui_dmainwindow.h

```
#include <QtCore/QVariant>
```



```

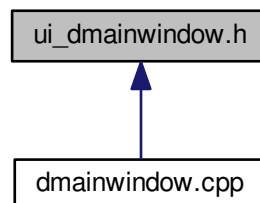
#include <QtGui/QAction>
#include <QtGui/QApplication>
#include <QtGui/QButtonGroup>
#include <QtGui/QHBoxLayout>
#include <QtGui/QHeaderView>
#include <QtGui/QLCDNumber>
#include <QtGui/QLabel>
#include <QtGui/QLineEdit>
#include <QtGui/QMainWindow>
#include <QtGui/QMenu>
#include <QtGui/QMenuBar>
#include <QtGui/QPushButton>
#include <QtGui/QSlider>
#include <QtGui/QSpacerItem>
#include <QtGui/QStatusBar>
#include <QtGui/QToolBar>
#include <QtGui/QWidget>

```

Wykres zależności załączania dla ui_dmainwindow.h:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



Komponenty

- class [Ui_DMainWindow](#)
- class [Ui::DMainWindow](#)

Przestrzenie nazw

- [Ui](#)

- definicje konstruktorow, metod i przeciazen klasy [Zbiornik](#).

Definicja w pliku [zbiornik.cpp](#).

8.17.2 Dokumentacja zmiennych

8.17.2.1 int STAN = ePAUSE

Stan symulacji.

Początkowy stan symulacji to Play/Pauza/Stop.

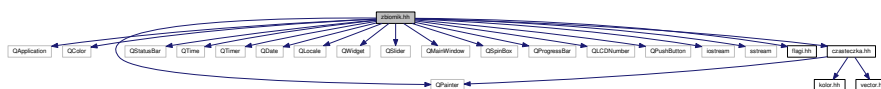
Definicja w linii 21 pliku [zbiornik.cpp](#).

8.18 Dokumentacja pliku zbiornik.hh

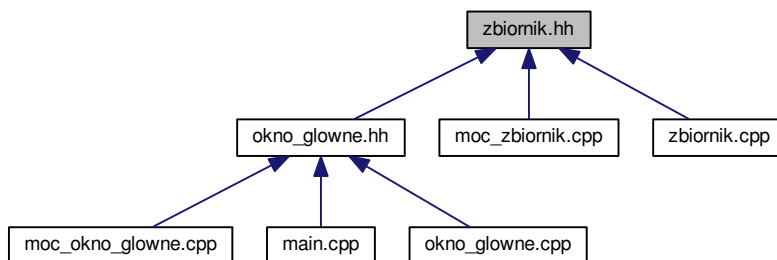
Zawiera definicje klasy [Zbiornik](#) i deklaracje jej metod.

```
#include <QApplication>
#include <QColor>
#include <QPainter>
#include <QStatusBar>
#include <QTime>
#include <QTimer>
#include <QDate>
#include <QLocale>
#include <QWidget>
#include <QSlider>
#include <QMainWindow>
#include <QSpinBox>
#include <QProgressBar>
#include <QLCDNumber>
#include <QPushButton>
#include <iostream>
#include <sstream>
#include "flagi.hh"
#include "czasteczka.hh"
```

Wykres zależności załączania dla [zbiornik.hh](#):



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



Komponenty

- class [Zbiornik](#)
Klasa modelująca zbiornik.

8.18.1 Opis szczegółowy

W pliku znajdują się:

- definicja klasy [Zbiornik](#) (modeluje pojęcie Zbiornika),
- deklaracje konstruktorów, metod i przeciążeń ww. klasy.

Definicja w pliku [zbiornik.hh](#).

Skorowidz

- ~DMainWindow
 - DMainWindow, [18](#)
- _Promien
 - Czasteczka, [16](#)
- _RGB
 - Czasteczka, [16](#)
- _Stoper
 - OknoGlowne, [32](#)
 - Zbiornik, [73](#)
- _b
 - Kolor, [23](#)
- _czas_sym
 - Zbiornik, [73](#)
- _g
 - Kolor, [23](#)
- _grubosc
 - Zbiornik, [73](#)
- _lewa_gora_xy
 - Zbiornik, [73](#)
- _odpowiedni_czas
 - Zbiornik, [73](#)
- _old_height
 - OknoGlowne, [31](#)
- _old_width
 - OknoGlowne, [31](#)
- _podstawa
 - Zbiornik, [73](#)
- _r
 - Kolor, [23](#)
- _wysokosc
 - Zbiornik, [74](#)
- _xy
 - Czasteczka, [17](#)
- a
 - simulation, [44](#)
- action_Exit
 - OknoGlowne, [32](#)
 - Ui_DMainWindow, [46](#)
- action_Save
 - OknoGlowne, [32](#)
 - Ui_DMainWindow, [46](#)
- actionExit
 - Ui_DMainWindow, [46](#)
- actionPlay
 - Ui_DMainWindow, [47](#)
- b
 - Kolor, [21](#), [22](#)
- box_indicator
 - simulation, [38](#)
- centralWidget
 - Ui_DMainWindow, [47](#)
- check_state
 - simulation, [38](#)
- compute_accel
 - simulation, [38](#)
- compute_density
 - simulation, [39](#)
- czas_sym
 - Zbiornik, [64](#)
- Czasteczka, [13](#)
 - _Promien, [16](#)
 - _RGB, [16](#)
 - _xy, [17](#)
 - Czasteczka, [14](#)
 - Promien, [14](#)
 - RGB, [15](#)
 - RysujCzasteczke, [15](#)
 - xy, [16](#)
- czasteczka.cpp, [75](#)
- czasteczka.hh, [75](#)
- Czasteczki
 - Zbiornik, [74](#)
- CzyWewnatrzZbiornika
 - Zbiornik, [64–66](#)
- DAMP
 - simulation.cpp, [85](#)
- DMainWindow, [17](#)
 - ~DMainWindow, [18](#)
 - DMainWindow, [18](#)
 - DMainWindow, [18](#)
 - ui, [19](#)
- damp_reflect_x
 - simulation, [39](#)
- damp_reflect_y
 - simulation, [40](#)
- dmainwindow.cpp, [77](#)
- dmainwindow.h, [77](#)
- dt
 - params_t, [35](#)
- ePAUSE
 - flagi.hh, [79](#)
- ePLAY
 - flagi.hh, [79](#)
- eSTOP
 - flagi.hh, [79](#)

- flagi.hh
 - ePAUSE, 79
 - ePLAY, 79
 - eSTOP, 79
- flagi.hh, 78
 - GRUBOSC, 79
 - PODSTAWA, 79
 - PROMIEN, 79
 - STAN, 79
 - WYSOKOSC, 79
- funkcja_main
 - simulation.cpp, 85
- g
 - Kolor, 22
- GRUBOSC
 - flagi.hh, 79
- GdyNapis
 - OknoGlowne, 27
- GdyOdpowiedniCzas
 - OknoGlowne, 27
 - Zbiornik, 67
- getN
 - simulation, 40
- getX
 - Vector, 51, 52
- getY
 - Vector, 52
- go
 - simulation, 41
- grubosc
 - Zbiornik, 68
- gx
 - params_t, 35
- gy
 - params_t, 35
- h
 - params_t, 35
- horizontalLayout
 - OknoGlowne, 32
 - Ui_DMainWindow, 47
- horizontalLayoutWidget
 - OknoGlowne, 32
 - Ui_DMainWindow, 47
- indicate_fun_t
 - simulation, 37
- init
 - simulation, 41
- k
 - params_t, 35
- Kolor, 20
 - _b, 23
 - _g, 23
 - _r, 23
 - b, 21, 22
 - g, 22
 - Kolor, 20, 21
 - r, 23
- kolor.hh, 79
- LOG
 - simulation.cpp, 85
- label
 - Ui_DMainWindow, 47
- labelCzasSym
 - OknoGlowne, 32
- labelLiczbaCzasteczek
 - OknoGlowne, 32
- labelSzybkoscSym
 - OknoGlowne, 32
- lcdCzasSym
 - OknoGlowne, 32
 - Ui_DMainWindow, 47
- lcdLiczbaCzasteczek
 - OknoGlowne, 33
 - Ui_DMainWindow, 47
- lcdSzybkoscSym
 - OknoGlowne, 33
 - Ui_DMainWindow, 47
- leapfrog_start
 - simulation, 42
- leapfrog_step
 - simulation, 42
- lewa_gora_xy
 - Zbiornik, 68, 69
- licznik_plikow
 - OknoGlowne, 33
- lineCzasSym
 - Ui_DMainWindow, 47
- lineEdit
 - OknoGlowne, 33
- lineLiczbaCzasteczek
 - Ui_DMainWindow, 47
- lineSzybkoscSym
 - Ui_DMainWindow, 47
- loadButton
 - OknoGlowne, 33
- main
 - main.cpp, 81
- main.cpp, 80
 - main, 81
- mainToolBar
 - Ui_DMainWindow, 47
- mass
 - params_t, 35
- menu_Edit
 - OknoGlowne, 33
 - Ui_DMainWindow, 48
- menu_File
 - OknoGlowne, 33
 - Ui_DMainWindow, 48
- menu_Help
 - OknoGlowne, 33
 - Ui_DMainWindow, 48

- menuBar
 - OknoGlowne, 33
 - Ui_DMainWindow, 48
- moc_dmainwindow.cpp, 81
- moc_okno_glowne.cpp, 82
- moc_zbiornik.cpp, 82
- mu
 - params_t, 35
- n
 - simulation, 44
- nframes
 - params_t, 35
- normSquared
 - Vector, 53
- npframe
 - params_t, 36
- odpowiedni_czas
 - Zbiornik, 69
- okno_glowne.cpp, 82
- okno_glowne.hh, 82
- OknoGlowne, 24
 - _Stoper, 32
 - _old_height, 31
 - _old_width, 31
 - action_Exit, 32
 - action_Save, 32
 - GdyNapis, 27
 - GdyOdpowiedniCzas, 27
 - horizontalLayout, 32
 - horizontalLayoutWidget, 32
 - labelCzasSym, 32
 - labelLiczbaCzasteczek, 32
 - labelSzybkoscSym, 32
 - lcdCzasSym, 32
 - lcdLiczbaCzasteczek, 33
 - lcdSzybkoscSym, 33
 - licznik_plikow, 33
 - lineEdit, 33
 - loadButton, 33
 - menu_Edit, 33
 - menu_File, 33
 - menu_Help, 33
 - menuBar, 33
 - OknoGlowne, 27
 - OknoGlowne, 27
 - on_action_Save_triggered, 28
 - on_lineEdit_returnPressed, 28
 - on_loadButton_clicked, 28
 - on_pauseButton_clicked, 29
 - on_playButton_clicked, 29
 - on_sliderSzybkoscSym_valueChanged, 29
 - on_stopButton_clicked, 29
 - paintEvent, 30
 - pauseButton, 34
 - playButton, 34
 - sliderSzybkoscSym, 34
 - statusBar, 34
 - stopButton, 34
 - toolBar, 34
 - wZbiornik, 34
 - WczytajSymulacjeZPliku, 30
 - ZapiszSymulacjeDoPliku, 30
 - ZglosNapis, 31
- on_action_Save_triggered
 - OknoGlowne, 28
- on_lineEdit_returnPressed
 - OknoGlowne, 28
- on_loadButton_clicked
 - OknoGlowne, 28
- on_pauseButton_clicked
 - OknoGlowne, 29
- on_playButton_clicked
 - OknoGlowne, 29
- on_sliderSzybkoscSym_valueChanged
 - OknoGlowne, 29
- on_stopButton_clicked
 - OknoGlowne, 29
- operator<<
 - simulation, 44
 - simulation.cpp, 86
 - Vector, 60
- operator*
 - Vector, 53, 54
- operator*=
 - Vector, 54
- operator+
 - Vector, 56
- operator+=
 - Vector, 57
- operator-
 - Vector, 57, 58
- operator-=
 - Vector, 58
- operator=
 - Vector, 60
- p
 - simulation, 44
- PODSTAWA
 - flagi.hh, 79
- PROMIEN
 - flagi.hh, 79
- paintEvent
 - OknoGlowne, 30
 - Zbiornik, 69
- params
 - simulation, 44
- params_t, 35
 - dt, 35
 - gx, 35
 - gy, 35
 - h, 35
 - k, 35
 - mass, 35
 - mu, 35
 - nframes, 35

- npframe, 36
- rho0, 36
- pauseButton
 - OknoGlowne, 34
 - Ui_DMainWindow, 48
- place_particles
 - simulation, 43
- playButton
 - OknoGlowne, 34
 - Ui_DMainWindow, 48
- podstawa
 - Zbiornik, 70
- Promien
 - Czasteczka, 14
- r
 - Kolor, 23
- RGB
 - Czasteczka, 15
- reflect_bc
 - simulation, 43
- retranslateUi
 - Ui_DMainWindow, 46
- rho
 - simulation, 44
- rho0
 - params_t, 36
- RysujCzasteczke
 - Czasteczka, 15
- RysujZbiornik
 - Zbiornik, 70
- RysujZbiornikZCzasteczkami
 - Zbiornik, 71
- STAN
 - flagi.hh, 79
 - zbiornik.cpp, 89
- setup
 - simulation.cpp, 86
- setupUi
 - Ui_DMainWindow, 46
- simulation, 36
 - a, 44
 - box_indicator, 38
 - check_state, 38
 - compute_accel, 38
 - compute_density, 39
 - damp_reflect_x, 39
 - damp_reflect_y, 40
 - getN, 40
 - go, 41
 - indicate_fun_t, 37
 - init, 41
 - leapfrog_start, 42
 - leapfrog_step, 42
 - n, 44
 - operator<<, 44
 - p, 44
 - params, 44
 - place_particles, 43
 - reflect_bc, 43
 - rho, 44
 - simulation, 37
 - v, 44
 - vh, 44
- simulation.cpp, 84
 - DAMP, 85
 - funkcja_main, 85
 - LOG, 85
 - operator<<, 86
 - setup, 86
 - XMAX, 85
 - YMAX, 85
- sliderSzybkoscSym
 - OknoGlowne, 34
 - Ui_DMainWindow, 48
- statusBar
 - OknoGlowne, 34
 - Ui_DMainWindow, 48
- stopButton
 - OknoGlowne, 34
 - Ui_DMainWindow, 48
- strona.dox, 86
- toolBar
 - OknoGlowne, 34
 - Ui_DMainWindow, 48
- Ui, 11
- ui
 - DMainWindow, 19
- Ui::DMainWindow, 19
- Ui_DMainWindow, 45
 - action_Exit, 46
 - action_Save, 46
 - actionExit, 46
 - actionPlay, 47
 - centralWidget, 47
 - horizontalLayout, 47
 - horizontalLayoutWidget, 47
 - label, 47
 - lcdCzasSym, 47
 - lcdLiczbaCzasteczek, 47
 - lcdSzybkoscSym, 47
 - lineCzasSym, 47
 - lineLiczbaCzasteczek, 47
 - lineSzybkoscSym, 47
 - mainToolBar, 47
 - menu_Edit, 48
 - menu_File, 48
 - menu_Help, 48
 - menuBar, 48
 - pauseButton, 48
 - playButton, 48
 - retranslateUi, 46
 - setupUi, 46
 - sliderSzybkoscSym, 48
 - statusBar, 48

- stopButton, 48
- toolBar, 48
- verticalSpacer, 48
- ui_dmainwindow.h, 86
- v
 - simulation, 44
- Vector, 49
 - getX, 51, 52
 - getY, 52
 - normSquared, 53
 - operator<<, 60
 - operator*, 53, 54
 - operator*=: 54
 - operator+, 56
 - operator+=, 57
 - operator-, 57, 58
 - operator-=, 58
 - operator=, 60
 - Vector, 50, 51
 - x, 60
 - y, 60
- vector.hh, 88
- verticalSpacer
 - Ui_DMainWindow, 48
- vh
 - simulation, 44
- WYSOKOSC
 - flagi.hh, 79
- wZbiornik
 - OknoGlowne, 34
- WczytajSymulacjeZPliku
 - OknoGlowne, 30
- wysokosc
 - Zbiornik, 71, 72
- x
 - Vector, 60
- XMAX
 - simulation.cpp, 85
- xy
 - Czasteczka, 16
- y
 - Vector, 60
- YMAX
 - simulation.cpp, 85
- ZapiszSymulacjeDoPliku
 - OknoGlowne, 30
- Zbiornik, 61
 - _Stoper, 73
 - _czas_sym, 73
 - _grubosc, 73
 - _lewa_gora_xy, 73
 - _odpowiedni_czas, 73
 - _podstawa, 73
 - _wysokosc, 74
 - czas_sym, 64
 - Czasteczki, 74
 - CzyWewnatrzZbiornika, 64–66
 - GdyOdpowiedniCzas, 67
 - grubosc, 68
 - lewa_gora_xy, 68, 69
 - odpowiedni_czas, 69
 - paintEvent, 69
 - podstawa, 70
 - RysujZbiornik, 70
 - RysujZbiornikZCzasteczkami, 71
 - wysokosc, 71, 72
 - Zbiornik, 63
 - ZglosCzasSymulacji, 72
 - ZglosLiczbeCzasteczek, 72
 - ZglosNapis, 73
- zbiornik.cpp, 88
 - STAN, 89
- zbiornik.hh, 89
 - ZglosCzasSymulacji
 - Zbiornik, 72
 - ZglosLiczbeCzasteczek
 - Zbiornik, 72
 - ZglosNapis
 - OknoGlowne, 31
 - Zbiornik, 73