Kalkulator wielomianów

Wygenerowano przez Doxygen 1.9.1

1 Indeks przestrzeni nazw

1.1 Lista przestrzeni nazw

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

2 Indeks hierarchiczny

2.1 Hierarchia klas

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

3 Indeks klas

3.1 Lista klas

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

4 Indeks plików

4.1 Lista plików

Tutaj znajduje się lista wszystkich plików z ich krótkimi opisami:

5 Dokumentacja przestrzeni nazw

5.1 Dokumentacja przestrzeni nazw configuration

Konfiguracja aplikacji, która musi być ustawiona, aby połączenie z GUI było możliwe.

Zmienne

- const char ∗const EVENT_NAME = "Global\\ComputeEvent"
- const char ∗const COMPLETION_EVENT_NAME = "Global\\CompletionEvent"
- const char ∗const CONFIG_FILE = "../../PythonGUI/config.ini"

5.1.1 Opis szczegółowy

Konfiguracja aplikacji, która musi być ustawiona, aby połączenie z GUI było możliwe.

5.1.2 Dokumentacja zmiennych

5.1.2.1 COMPLETION_EVENT_NAME `const char *const configuration::COMPLETION_EVENT_NAME = "Global\\CompletionEvent"`

Definicja w linii 20 pliku config.cpp.

5.1.2.2 CONFIG_FILE `const char *const configuration::CONFIG_FILE = "../../PythonGUI/config.ini"`

Definicja w linii 21 pliku config.cpp.

5.1.2.3 EVENT_NAME `const char *const configuration::EVENT_NAME = "Global\\ComputeEvent"`

Definicja w linii 19 pliku config.cpp.

5.2 Dokumentacja przestrzeni nazw dll_functions

Funkcje wielomianowe zaimplementowane w bibliotekach DLL.

Zmienne

- PolynomialFunc hornerCpp = nullptr
- PolynomialFunc hornerAsm = nullptr
- PolynomialFuncAvx hornerCppAvx = nullptr
- PolynomialFuncAvx hornerAsmAvx = nullptr

5.2.1 Opis szczegółowy

Funkcje wielomianowe zaimplementowane w bibliotekach DLL.

5.2.2 Dokumentacja zmiennych

5.2.2.1 hornerAsm `PolynomialFunc dll_functions::hornerAsm = nullptr`

Definicja w linii 27 pliku dlls.cpp.

5.2.2.2 hornerAsmAvx `PolynomialFuncAvx dll_functions::hornerAsmAvx = nullptr`

Definicja w linii 29 pliku dlls.cpp.

5.2.2.3 hornerCpp `PolynomialFunc dll_functions::hornerCpp = nullptr`

Definicja w linii 26 pliku dlls.cpp.

5.2.2.4 hornerCppAvx `PolynomialFuncAvx dll_functions::hornerCppAvx = nullptr`

Definicja w linii 28 pliku dlls.cpp.

5.3 Dokumentacja przestrzeni nazw dlls

Globalne uchwyty do bibliotek DLL.

Zmienne

- HINSTANCE h_cpp = nullptr
- HINSTANCE h_asm = nullptr

5.3.1 Opis szczegółowy

Globalne uchwyty do bibliotek DLL.

5.3.2 Dokumentacja zmiennych

5.3.2.1 h_asm `HINSTANCE dlls::h_asm = nullptr`

Definicja w linii 22 pliku dlls.cpp.

5.3.2.2 h_cpp `HINSTANCE dlls::h_cpp = nullptr`

Definicja w linii 21 pliku dlls.cpp.

5.4 Dokumentacja przestrzeni nazw gui

Przestrzenie nazw

- config
- events
- experiment
- file_utils
- main_window
- plotting

5.5 Dokumentacja przestrzeni nazw gui.config

Zmienne

- string EVENT_NAME = "Global\\ComputeEvent"
- string COMPLETION_EVENT_NAME = "Global\\CompletionEvent"
- TEMP_DIR = os.path.abspath('temp')
- CONFIG_FILE = os.path.join('config.ini')
- PROGRESS_FILE = os.path.join(TEMP_DIR, 'progress.tmp')
- COMPUTATION_TIME_FILE = os.path.join(TEMP_DIR, 'computation.time')

5.5.1 Dokumentacja zmiennych

5.5.1.1 COMPLETION_EVENT_NAME `string gui.config.COMPLETION_EVENT_NAME = "Global\\CompletionEvent"`

Definicja w linii 4 pliku config.py.

5.5.1.2 COMPUTATION_TIME_FILE `gui.config.COMPUTATION_TIME_FILE = os.path.join(TEMP_DIR, 'computation.time')`

Definicja w linii 8 pliku config.py.

5.5.1.3 CONFIG_FILE `gui.config.CONFIG_FILE = os.path.join('config.ini')`

Definicja w linii 6 pliku config.py.

5.5.1.4 EVENT_NAME `string gui.config.EVENT_NAME = "Global\\ComputeEvent"`

Definicja w linii 3 pliku config.py.

5.5.1.5 PROGRESS_FILE `gui.config.PROGRESS_FILE = os.path.join(TEMP_DIR, 'progress.tmp')`

Definicja w linii 7 pliku config.py.

5.5.1.6 TEMP_DIR `gui.config.TEMP_DIR = os.path.abspath('temp')`

Definicja w linii 5 pliku config.py.

5.6 Dokumentacja przestrzeni nazw gui.events

Funkcje

- def trigger_event (event_name)
- def monitor_progress_and_completion (app)
- def on_closing (self)

Zmienne

- is_running

5.6.1 Dokumentacja funkcji

5.6.1.1 monitor_progress_and_completion() `def gui.events.monitor_progress_and_completion ( app )`

Definicja w linii 20 pliku events.py.

5.6.1.2 on_closing() `def gui.events.on_closing ( self )`

Definicja w linii 65 pliku events.py.

5.6.1.3 trigger_event() def gui.events.trigger_event (event_name)

Definicja w linii 12 pliku events.py.

5.6.2 Dokumentacja zmiennych

5.6.2.1 is_running gui.events.is_running

Definicja w linii 66 pliku events.py.

5.7 Dokumentacja przestrzeni nazw gui.experiment

Funkcje

- def show_experiment_info ()
- def run_experiment (app)

5.7.1 Dokumentacja funkcji

5.7.1.1 run_experiment() def gui.experiment.run_experiment (app)

```
Uruchamia eksperyment iteracyjnie -
1) Generuje pliki współczynników i punktów,
2) Zapisuje config.ini,
3) trigger_event(EVENT_NAME),
4) czeka na pojawienie się computation.time,
5) odczytuje i zapisuje w pliku CSV.

Brak użycia progress.tmp. Postęp wypisujemy w GUI (status_label).
```

Definicja w linii 62 pliku experiment.py.

5.7.1.2 show_experiment_info() def gui.experiment.show_experiment_info ( )

```
Wyświetla w okienku informacyjnym opis eksperymentu:
- 3 scenariusze (n=10/100/1000, różne kroki),
- implementacja: {cpp, asm},
- AVX: {False, True},
- multithreading: zawsze True, ale threads=1..16,
- 10 uruchomień każdej konfiguracji,
- brak progress.tmp (nie śledzimy postępu w jednym uruchomieniu),
- brak zapisu wyników, brak generowania wykresu,
- odczytujemy finalny czas z computation.time.
```

Definicja w linii 21 pliku experiment.py.

5.8 Dokumentacja przestrzeni nazw gui.file_utils

Funkcje

- def load_file (title, filetypes)
- str _generate_coefficients (int n, float coeff_min, float coeff_max)
- str _generate_points (float start, float end, float step)
- def generate_coefficients_file (app, n=None, coeff_min=None, coeff_max=None)
- def generate_points_file (app, start=None, end=None, step=None)
- def write_config_file (implementation, multithreading, threads_number, avx, save_results, generate_chart, output_file, coefficients_file, points_file)
- def read_results_and_display (app)

5.8.1 Dokumentacja funkcji

5.8.1.1 _generate_coefficients()

```
str gui.file_utils._generate_coefficients (
        int n,
        float coeff_min,
        float coeff_max )  [private]
```

```
Funkcja 'niższego poziomu' generująca n współczynników
z zakresu [coeff_min, coeff_max] i zapisująca do pliku w conf.TEMP_DIR.

Zwraca ścieżkę do wygenerowanego pliku.
```

Definicja w linii 13 pliku file_utils.py.

5.8.1.2 _generate_points()

```
str gui.file_utils._generate_points (
        float start,
        float end,
        float step )  [private]
```

```
Funkcja 'niższego poziomu' generująca parametry start, end, step
i zapisująca je do pliku w conf.TEMP_DIR.

Zwraca ścieżkę do wygenerowanego pliku.
```

Definicja w linii 34 pliku file_utils.py.

5.8.1.3 generate_coefficients_file() def gui.file_utils.generate_coefficients_file (

```
app,
n = None,
coeff_min = None,
coeff_max = None )
```

```
Funkcja do generowania współczynników:
  - Jeśli n, coeff_min, coeff_max != None, to generujemy bez okna dialogowego
    i zwracamy ścieżkę (zapisujemy też w app.coefficients_file).
  - Jeśli parametry są None, to wyświetlamy okno Toplevel, by użytkownik wprowadził dane.
```

Definicja w linii 52 pliku file_utils.py.

5.8.1.4 generate_points_file() def gui.file_utils.generate_points_file (

```
app,
start = None,
end = None,
step = None )
```

```
Funkcja do generowania pliku z parametrami punktów:
  - Jeśli start, end, step != None, to generujemy bez okna dialogowego
    i zwracamy ścieżkę (zapisujemy też w app.points_file).
  - Jeśli parametry są None, to wyświetlamy okno Toplevel, by użytkownik wprowadził dane.
```

Definicja w linii 109 pliku file_utils.py.

5.8.1.5 load_file() def gui.file_utils.load_file (

```
title,
filetypes )
```

Definicja w linii 9 pliku file_utils.py.

5.8.1.6 read_results_and_display() def gui.file_utils.read_results_and_display (

```
app )
```

Definicja w linii 192 pliku file_utils.py.

5.8.1.7 write_config_file() def gui.file_utils.write_config_file (

```
implementation,
multithreading,
threads_number,
avx,
save_results,
generate_chart,
output_file,
coefficients_file,
points_file )
```

Definicja w linii 165 pliku file_utils.py.

5.9 Dokumentacja przestrzeni nazw gui.main_window

Komponenty

- class MainWindow

5.10 Dokumentacja przestrzeni nazw gui.plotting

Funkcje

- def plot_results (app, points, results)

5.10.1 Dokumentacja funkcji

5.10.1.1 plot_results() `def gui.plotting.plot_results ( app, points, results )`

```
Wyświetla wykres wyników wielomianu obliczonego metodą Hornera
dla zadanych punktów.

:param app: Referencja do głównej aplikacji (zawierająca m.in. zmienną generate_chart)
:param points: Lista punktów (x)
:param results: Lista wartości wielomianu w tych punktach (p(x))
```

Definicja w linii 4 pliku plotting.py.

5.11 Dokumentacja przestrzeni nazw PythonGUI

Zmienne

- root = tk.Tk()
- app = MainWindow(root)

5.11.1 Dokumentacja zmiennych

5.11.1.1 app `PythonGUI.app = MainWindow(root)`

Definicja w linii 6 pliku PythonGUI.py.

5.11.1.2 root `PythonGUI.root = tk.Tk()`

Definicja w linii 5 pliku PythonGUI.py.

6 Dokumentacja klas

6.1 Dokumentacja klasy gui.main_window.MainWindow

Diagram dziedziczenia dla gui.main_window.MainWindow

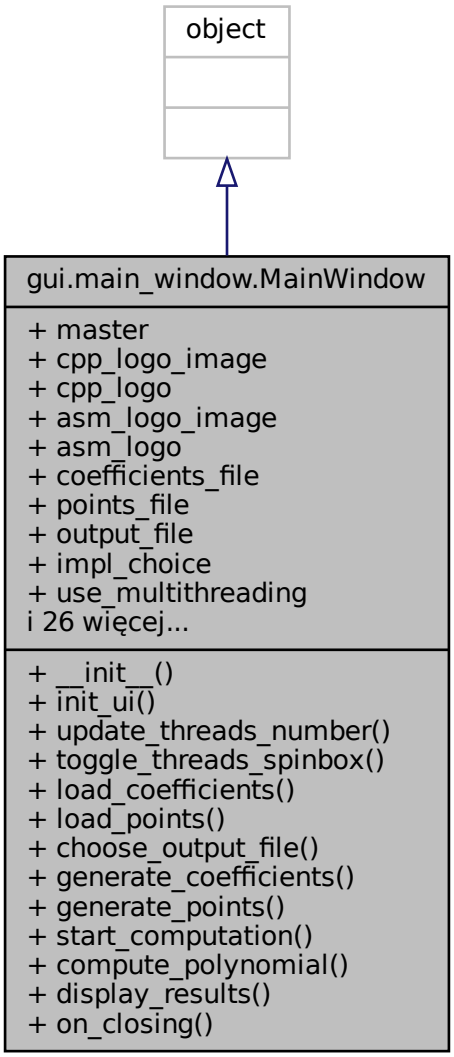
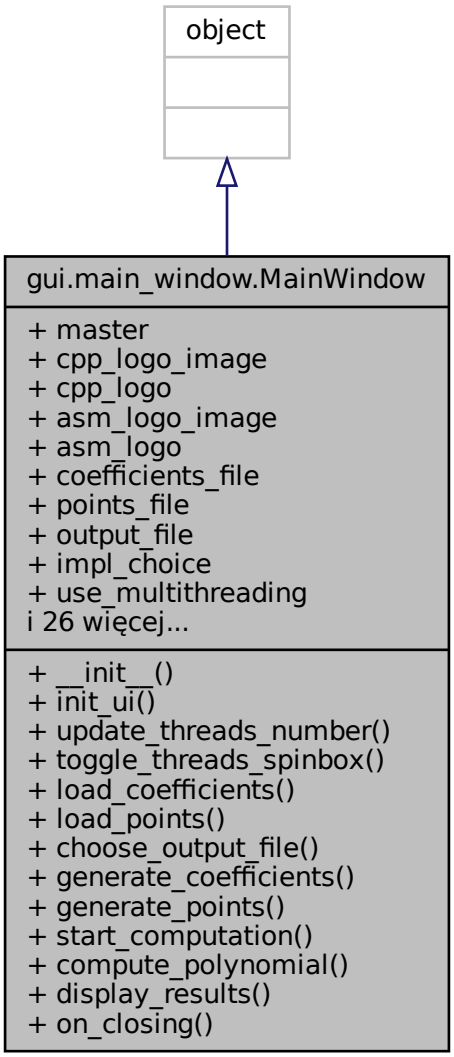


Diagram współpracy dla gui.main_window.MainWindow:



Metody publiczne

- def __init__ (self, master)
- def init_ui (self)
- def update_threads_number (self)
- def toggle_threads_spinbox (self)
- def load_coefficients (self)
- def load_points (self)
- def choose_output_file (self)
- def generate_coefficients (self)
- def generate_points (self)
- def start_computation (self)
- def compute_polynomial (self)
- def display_results (self, computation_time, points, results)
- def on_closing (self)

Atrybuty publiczne

- master
- cpp_logo_image
- cpp_logo
- asm_logo_image
- asm_logo
- coefficients_file
- points_file
- output_file
- impl_choice
- use_multithreading
- threads_number
- use_avx
- generate_chart
- save_results
- is_running
- cpp_radio
- asm_radio
- mt_checkbox
- threads_spinbox
- avx_checkbox
- save_results_checkbox
- chart_checkbox
- load_coeff_button
- gen_coeff_button
- coeff_file_label
- load_points_button
- gen_points_button
- points_file_label
- choose_output_button
- output_file_label
- compute_button
- run_experiment_button
- experiment_info_button
- status_label
- progress
- monitor_thread

6.1.1 Opis szczegółowy

```
Main window of the GUI application
```

Definicja w linii 22 pliku main_window.py.

6.1.2 Dokumentacja konstruktora i destruktora

6.1.2.1 __init__() def gui.main_window.MainWindow.__init__ (

```
self,
master )
```

Definicja w linii 24 pliku main_window.py.

6.1.3 Dokumentacja funkcji składowych

6.1.3.1 choose_output_file() def gui.main_window.MainWindow.choose_output_file (

```
self )
```

Definicja w linii 374 pliku main_window.py.

6.1.3.2 compute_polynomial() def gui.main_window.MainWindow.compute_polynomial (

```
self )
```

Definicja w linii 410 pliku main_window.py.

6.1.3.3 display_results() def gui.main_window.MainWindow.display_results (

```
self,
computation_time,
points,
results )
```

Definicja w linii 450 pliku main_window.py.

6.1.3.4 generate_coefficients() def gui.main_window.MainWindow.generate_coefficients (

```
self )
```

Definicja w linii 384 pliku main_window.py.

6.1.3.5 generate_points() def gui.main_window.MainWindow.generate_points (

```
self )
```

Definicja w linii 387 pliku main_window.py.

6.1.3.6 init_ui() def gui.main_window.MainWindow.init_ui (self)

Definicja w linii 56 pliku main_window.py.

6.1.3.7 load_coefficients() def gui.main_window.MainWindow.load_coefficients (self)

Definicja w linii 362 pliku main_window.py.

6.1.3.8 load_points() def gui.main_window.MainWindow.load_points (self)

Definicja w linii 368 pliku main_window.py.

6.1.3.9 on_closing() def gui.main_window.MainWindow.on_closing (self)

Definicja w linii 472 pliku main_window.py.

6.1.3.10 start_computation() def gui.main_window.MainWindow.start_computation (self)

Definicja w linii 390 pliku main_window.py.

6.1.3.11 toggle_threads_spinbox() def gui.main_window.MainWindow.toggle_threads_spinbox (self)

Definicja w linii 356 pliku main_window.py.

6.1.3.12 update_threads_number() def gui.main_window.MainWindow.update_threads_number (self)

Definicja w linii 350 pliku main_window.py.

6.1.4 Dokumentacja atrybutów składowych

6.1.4.1 asm_logo `gui.main_window.MainWindow.asm_logo`

Definicja w linii 39 pliku main_window.py.

6.1.4.2 asm_logo_image `gui.main_window.MainWindow.asm_logo_image`

Definicja w linii 38 pliku main_window.py.

6.1.4.3 asm_radio `gui.main_window.MainWindow.asm_radio`

Definicja w linii 105 pliku main_window.py.

6.1.4.4 avx_checkbox `gui.main_window.MainWindow.avx_checkbox`

Definicja w linii 150 pliku main_window.py.

6.1.4.5 chart_checkbox `gui.main_window.MainWindow.chart_checkbox`

Definicja w linii 166 pliku main_window.py.

6.1.4.6 choose_output_button `gui.main_window.MainWindow.choose_output_button`

Definicja w linii 265 pliku main_window.py.

6.1.4.7 coeff_file_label `gui.main_window.MainWindow.coeff_file_label`

Definicja w linii 208 pliku main_window.py.

6.1.4.8 coefficients_file `gui.main_window.MainWindow.coefficients_file`

Definicja w linii 42 pliku main_window.py.

6.1.4.9 compute_button `gui.main_window.MainWindow.compute_button`

Definicja w linii 283 pliku main_window.py.

6.1.4.10 cpp_logo `gui.main_window.MainWindow.cpp_logo`

Definicja w linii 36 pliku main_window.py.

6.1.4.11 cpp_logo_image `gui.main_window.MainWindow.cpp_logo_image`

Definicja w linii 35 pliku main_window.py.

6.1.4.12 cpp_radio `gui.main_window.MainWindow.cpp_radio`

Definicja w linii 94 pliku main_window.py.

6.1.4.13 experiment_info_button `gui.main_window.MainWindow.experiment_info_button`

Definicja w linii 314 pliku main_window.py.

6.1.4.14 gen_coeff_button `gui.main_window.MainWindow.gen_coeff_button`

Definicja w linii 198 pliku main_window.py.

6.1.4.15 gen_points_button `gui.main_window.MainWindow.gen_points_button`

Definicja w linii 235 pliku main_window.py.

6.1.4.16 generate_chart `gui.main_window.MainWindow.generate_chart`

Definicja w linii 49 pliku main_window.py.

6.1.4.17 impl_choice gui.main_window.MainWindow.impl_choice

Definicja w linii 45 pliku main_window.py.

6.1.4.18 is_running gui.main_window.MainWindow.is_running

Definicja w linii 51 pliku main_window.py.

6.1.4.19 load_coeff_button gui.main_window.MainWindow.load_coeff_button

Definicja w linii 188 pliku main_window.py.

6.1.4.20 load_points_button gui.main_window.MainWindow.load_points_button

Definicja w linii 225 pliku main_window.py.

6.1.4.21 master gui.main_window.MainWindow.master

Definicja w linii 26 pliku main_window.py.

6.1.4.22 monitor_thread gui.main_window.MainWindow.monitor_thread

Definicja w linii 438 pliku main_window.py.

6.1.4.23 mt_checkbox gui.main_window.MainWindow.mt_checkbox

Definicja w linii 125 pliku main_window.py.

6.1.4.24 output_file gui.main_window.MainWindow.output_file

Definicja w linii 44 pliku main_window.py.

6.1.4.25 output_file_label gui.main_window.MainWindow.output_file_label

Definicja w linii 275 pliku main_window.py.

6.1.4.26 points_file gui.main_window.MainWindow.points_file

Definicja w linii 43 pliku main_window.py.

6.1.4.27 points_file_label gui.main_window.MainWindow.points_file_label

Definicja w linii 245 pliku main_window.py.

6.1.4.28 progress gui.main_window.MainWindow.progress

Definicja w linii 334 pliku main_window.py.

6.1.4.29 run_experiment_button gui.main_window.MainWindow.run_experiment_button

Definicja w linii 301 pliku main_window.py.

6.1.4.30 save_results gui.main_window.MainWindow.save_results

Definicja w linii 50 pliku main_window.py.

6.1.4.31 save_results_checkbox gui.main_window.MainWindow.save_results_checkbox

Definicja w linii 158 pliku main_window.py.

6.1.4.32 status_label gui.main_window.MainWindow.status_label

Definicja w linii 327 pliku main_window.py.

6.1.4.33 threads_number `gui.main_window.MainWindow.threads_number`

Definicja w linii 47 pliku main_window.py.

6.1.4.34 threads_spinbox `gui.main_window.MainWindow.threads_spinbox`

Definicja w linii 138 pliku main_window.py.

6.1.4.35 use_avx `gui.main_window.MainWindow.use_avx`

Definicja w linii 48 pliku main_window.py.

6.1.4.36 use_multithreading `gui.main_window.MainWindow.use_multithreading`

Definicja w linii 46 pliku main_window.py.

Dokumentacja dla tej klasy została wygenerowana z pliku:

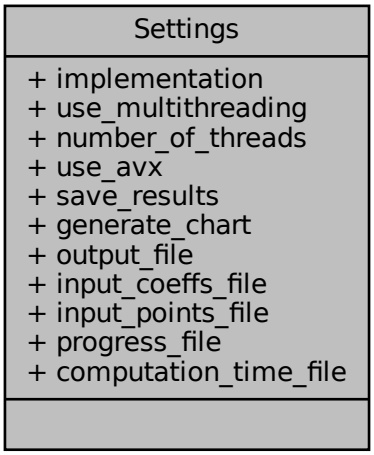
- PythonGUI/gui/main_window.py

6.2 Dokumentacja struktury Settings

Struktura przechowująca ustawienia aplikacji wczytywane przy każdym wywołaniem z GUI.

`#include <config.h>`

Diagram współpracy dla Settings:



Atrybuty publiczne

- std::string implementation
- bool use_multithreading
- int number_of_threads
- bool use_avx
- bool save_results
- bool generate_chart
- std::string output_file
- std::string input_coeffs_file
- std::string input_points_file
- std::string progress_file
- std::string computation_time_file

6.2.1 Opis szczegółowy

Struktura przechowująca ustawienia aplikacji wczytywane przy każdym wywołaniem z GUI.

Definicja w linii 31 pliku config.h.

6.2.2 Dokumentacja atrybutów składowych

6.2.2.1 computation_time_file `std::string Settings::computation_time_file`

Definicja w linii 42 pliku config.h.

6.2.2.2 generate_chart `bool Settings::generate_chart`

Definicja w linii 37 pliku config.h.

6.2.2.3 implementation `std::string Settings::implementation`

Definicja w linii 32 pliku config.h.

6.2.2.4 input_coeffs_file `std::string Settings::input_coeffs_file`

Definicja w linii 39 pliku config.h.

6.2.2.5 input_points_file `std::string Settings::input_points_file`

Definicja w linii 40 pliku config.h.

6.2.2.6 number_of_threads `int Settings::number_of_threads`

Definicja w linii 34 pliku config.h.

6.2.2.7 output_file `std::string Settings::output_file`

Definicja w linii 38 pliku config.h.

6.2.2.8 progress_file `std::string Settings::progress_file`

Definicja w linii 41 pliku config.h.

6.2.2.9 save_results `bool Settings::save_results`

Definicja w linii 36 pliku config.h.

6.2.2.10 use_avx `bool Settings::use_avx`

Definicja w linii 35 pliku config.h.

6.2.2.11 use_multithreading `bool Settings::use_multithreading`

Definicja w linii 33 pliku config.h.

Dokumentacja dla tej struktury została wygenerowana z pliku:

- App/config.h

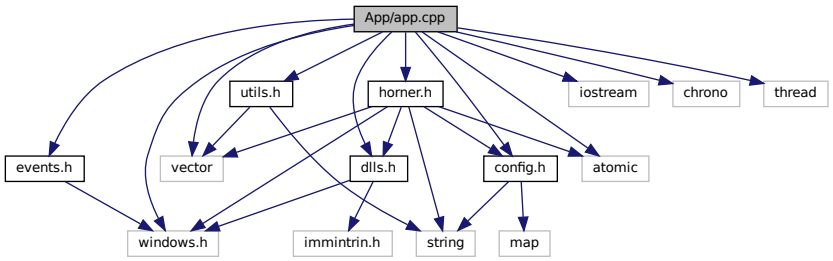
7 Dokumentacja plików

7.1 Dokumentacja pliku App/app.cpp

Głowny plik aplikacji serwera do obliczania wartości wielomianów.

```
#include <windows.h>
#include <iostream>
#include <vector>
#include <chrono>
#include <thread>
#include <atomic>
#include "config.h"
#include "events.h"
#include "horner.h"
#include "utils.h"
#include "dlls.h"
```

Wykres zależności załączania dla app.cpp:



Funkcje

- int main (int argc, char ∗∗argv)

  Funckcja main ładuje biblioteki DLL, tworzy eventy i w nieskończonej pętli oczekuje na zdarzenie.

7.1.1 Opis szczegółowy

Głowny plik aplikacji serwera do obliczania wartości wielomianów.

Autor

krzsztfwtk

Wersja

2.1

Data

2024-12-20

Copyright

Copyright (c) 2025 krzsztfwtk

7.1.2 Dokumentacja funkcji

7.1.2.1 main() `int main ( int argc, char ** argv )`

Funckcja main ładuje biblioteki DLL, tworzy eventy i w nieskończonej pętli oczekuje na zdarzenie.

Parametry

argc	Liczba argumentów.
argv	Opcjonalny wybór scenariusza testowego.

Zwraca

int Sukces: 0, Błąd: 1

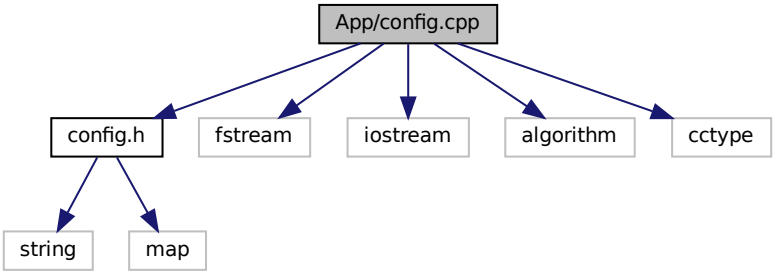
Definicja w linii 52 pliku app.cpp.

7.2 Dokumentacja pliku App/config.cpp

Obługa konfiguracji globalnej lub tymczasowej.

```
#include "config.h"
#include <fstream>
#include <iostream>
#include <algorithm>
#include <cctype>
```

Wykres zależności załączania dla config.cpp:



Przestrzenie nazw

- configuration

  Konfiguracja aplikacji, która musi być ustawiona, aby połączenie z GUI było możliwe.

Funkcje

- std::string trim (const std::string &str)
- int stringToInt (const std::string &str)
- bool readConfigINI (const std::string &filename, Settings &settings)

  Wczytuje ustawienia aplikacji z pliku konfiguracyjnego.

Zmienne

- const char ∗const configuration::EVENT_NAME = "Global\\ComputeEvent"
- const char ∗const configuration::COMPLETION_EVENT_NAME = "Global\\CompletionEvent"
- const char ∗const configuration::CONFIG_FILE = "../../PythonGUI/config.ini"

7.2.1 Opis szczegółowy

Obługa konfiguracji globalnej lub tymczasowej.

Autor

krzsztfwtk

Wersja

2.1

Data

2024-12-20

Copyright

7.2.2 Dokumentacja funkcji

7.2.2.1 readConfigINI() `bool readConfigINI ( const std::string & filename, Settings & settings )`

Wczytuje ustawienia aplikacji z pliku konfiguracyjnego.

Parametry

filename	Nazwa pliku konfiguracyjnego.
settings	Struktura przechowująca wczytane ustawienia.

Zwraca

true Jeśli udało się wczytać ustawienia.

false Jeśli nie udało się wczytać ustawień.

Definicja w linii 45 pliku config.cpp.

7.2.2.2 stringToInt() `int stringToInt ( const std::string & str )`

Definicja w linii 31 pliku config.cpp.

7.2.2.3 trim() `std::string trim ( const std::string & str )`

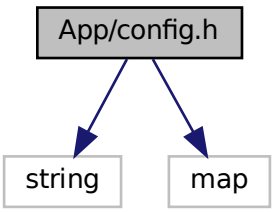
Definicja w linii 24 pliku config.cpp.

7.3 Dokumentacja pliku App/config.h

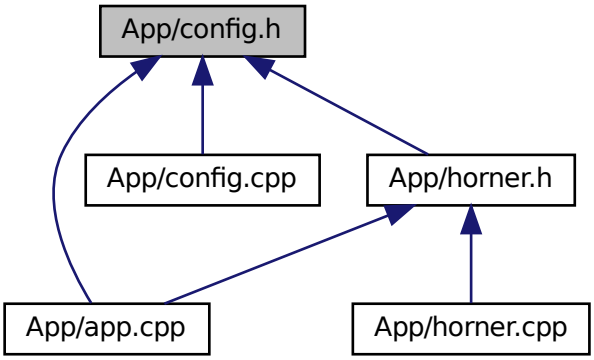
Obługa konfiguracji globalnej lub tymczasowej.

```
#include <string>
#include <map>
```

Wykres zależności załączania dla config.h:



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



Komponenty

- struct Settings

  Struktura przechowująca ustawienia aplikacji wczytywane przy każdym wywołaniem z GUI.

Przestrzenie nazw

- configuration

  Konfiguracja aplikacji, która musi być ustawiona, aby połączenie z GUI było możliwe.

Funkcje

- bool readConfigINI (const std::string &filename, Settings &settings)

  Wczytuje ustawienia aplikacji z pliku konfiguracyjnego.

7.3.1 Opis szczegółowy

Obługa konfiguracji globalnej lub tymczasowej.

Autor

krzsztfwtk

Wersja

2.1

Data

2024-12-20

Copyright

Copyright (c) 2025 krzsztfwtk

7.3.2 Dokumentacja funkcji

7.3.2.1 readConfigINI() `bool readConfigINI ( const std::string & filename, Settings & settings )`

Wczytuje ustawienia aplikacji z pliku konfiguracyjnego.

Parametry

filename	Nazwa pliku konfiguracyjnego.
settings	Struktura przechowująca wczytane ustawienia.

Zwraca

true Jeśli udało się wczytać ustawienia.

false Jeśli nie udało się wczytać ustawień.

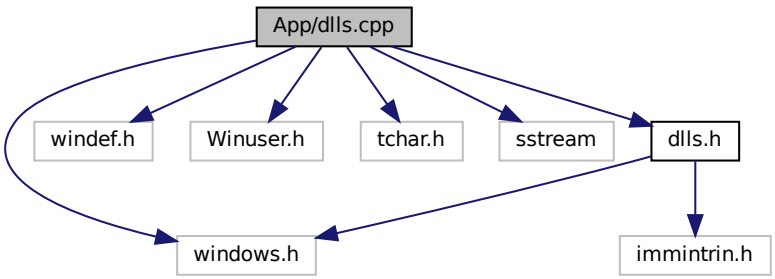
Definicja w linii 45 pliku config.cpp.

7.4 Dokumentacja pliku App/dlls.cpp

Obługa dynamicznego ładowania bibliotek.

```
#include <windows.h>
#include <windef.h>
#include <Winuser.h>
#include <tchar.h>
#include <sstream>
#include "dlls.h"
```

Wykres zależności załączania dla dlls.cpp:



Przestrzenie nazw

- dlls

  Globalne uchwyty do bibliotek DLL.
- dll_functions

  Funkcje wielomianowe zaimplementowane w bibliotekach DLL.

Funkcje

- static void ResultMessageBox (float x, float result)
- static std::wstring ChooseDll (const wchar_t *title)
- void loadDLLFunctions ()
- void LoadDlls ()

  Funkcja do ładowania bibliotek DLL.

Zmienne

- HINSTANCE dlls::h_cpp = nullptr
- HINSTANCE dlls::h_asm = nullptr
- PolynomialFunc dll_functions::hornerCpp = nullptr
- PolynomialFunc dll_functions::hornerAsm = nullptr
- PolynomialFuncAvx dll_functions::hornerCppAvx = nullptr
- PolynomialFuncAvx dll_functions::hornerAsmAvx = nullptr

7.4.1 Opis szczegółowy

Obługa dynamicznego ładowania bibliotek.

Autor

krzsztfwtk

Wersja

2.1

Data

2024-12-20

Copyright

7.4.2 Dokumentacja funkcji

7.4.2.1 ChooseDll() `static std::wstring ChooseDll ( const wchar_t * title ) [static]`

Definicja w linii 39 pliku dlls.cpp.

7.4.2.2 loadDLLFunctions() `void loadDLLFunctions ( )`

Definicja w linii 59 pliku dlls.cpp.

7.4.2.3 LoadDlls() `void LoadDlls ( )`

Funkcja do ładowania bibliotek DLL.

Definicja w linii 66 pliku dlls.cpp.

7.4.2.4 ResultMessageBox() `static void ResultMessageBox ( float x, float result ) [static]`

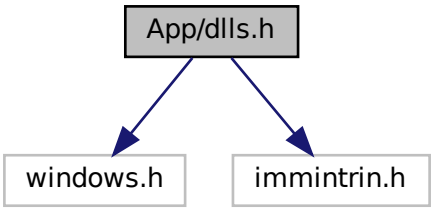
Definicja w linii 32 pliku dlls.cpp.

7.5 Dokumentacja pliku App/dlls.h

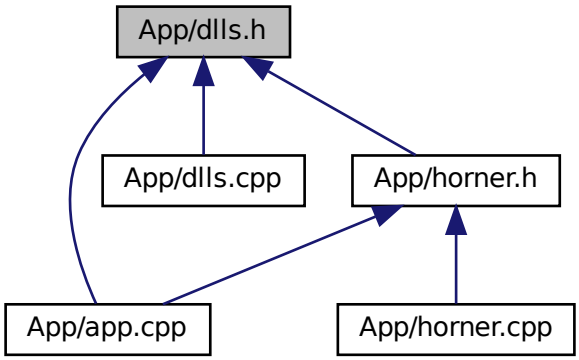
Obługa dynamicznego ładowania bibliotek.

```
#include <windows.h>
#include <immintrin.h>
```

Wykres zależności załączania dla dlls.h:



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



Przestrzenie nazw

- dlls

  Globalne uchwyty do bibliotek DLL.
- dll_functions

  Funkcje wielomianowe zaimplementowane w bibliotekach DLL.

Definicje typów

- typedef float(__stdcall ∗ PolynomialFunc) (float ∗x, float ∗a, int n)

  Typedef dla funkcji wielomianowej, która może być zaimplementowana w ASM DLL lub CPP DLL.
- typedef __m256(__stdcall ∗ PolynomialFuncAvx) (float ∗x, float ∗a, int n)

Funkcje

- void LoadDlls ()

  Funkcja do ładowania bibliotek DLL.

7.5.1 Opis szczegółowy

Obługa dynamicznego ładowania bibliotek.

Autor

krzsztfwtk

Wersja

2.1

Data

2024-12-20

Copyright

7.5.2 Dokumentacja definicji typów

7.5.2.1 PolynomialFunc `typedef float(__stdcall* PolynomialFunc) (float *x, float *a, int n)`

Typedef dla funkcji wielomianowej, która może być zaimplementowana w ASM DLL lub CPP DLL.

Definicja w linii 21 pliku dlls.h.

7.5.2.2 PolynomialFuncAvx `typedef __m256(__stdcall* PolynomialFuncAvx) (float *x, float *a, int n)`

Definicja w linii 22 pliku dlls.h.

7.5.3 Dokumentacja funkcji

7.5.3.1 LoadDlls() `void LoadDlls ( )`

Funkcja do ładowania bibliotek DLL.

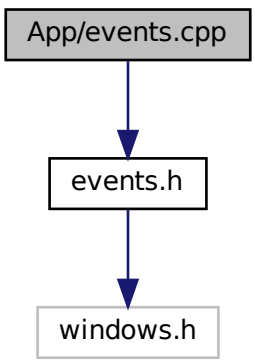
Definicja w linii 66 pliku dlls.cpp.

7.6 Dokumentacja pliku App/events.cpp

Obługa zdarzeń systemowych Windows.

```
#include "events.h"
```

Wykres zależności załączania dla events.cpp:



Funkcje

- HANDLE createEvent (const char ∗eventName, BOOL manualReset)

  Utowrzenie eventu WinAPI.

7.6.1 Opis szczegółowy

Obługa zdarzeń systemowych Windows.

Autor

krzsztfwtk

Wersja

2.1

Data

2024-12-20

Copyright

Copyright (c) 2025 krzsztfwtk

7.6.2 Dokumentacja funkcji

7.6.2.1 createEvent() `HANDLE createEvent ( const char * eventName, BOOL manualReset )`

Utowrzenie eventu WinAPI.

Parametry

eventName	Nazwa eventu.
manualReset	TRUE dla manual-reset, FALSE dla auto-reset

Zwraca

HANDLE Uchwyt do zdarzenia systemowego.

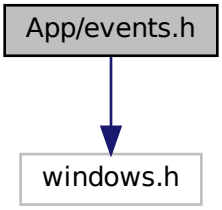
Definicja w linii 14 pliku events.cpp.

7.7 Dokumentacja pliku App/events.h

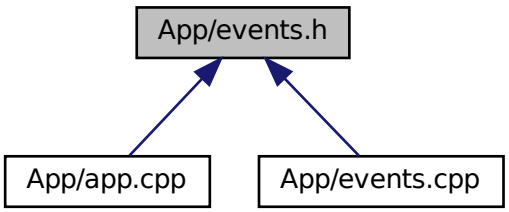
Obługa zdarzeń systemowych Windows.

```
#include <windows.h>
```

Wykres zależności załączania dla events.h:



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



Funkcje

- HANDLE createEvent (const char ∗eventName, BOOL manualReset)

  Utowrzenie eventu WinAPI.

7.7.1 Opis szczegółowy

Obługa zdarzeń systemowych Windows.

Autor

krzsztfwtk

Wersja

2.1

Data

2024-12-20

Copyright

7.7.2 Dokumentacja funkcji

7.7.2.1 createEvent() `HANDLE createEvent ( const char * eventName, BOOL manualReset )`

Utowrzenie eventu WinAPI.

Parametry

eventName	Nazwa eventu.
manualReset	TRUE dla manual-reset, FALSE dla auto-reset

Zwraca

HANDLE Uchwyt do zdarzenia systemowego.

Definicja w linii 14 pliku events.cpp.

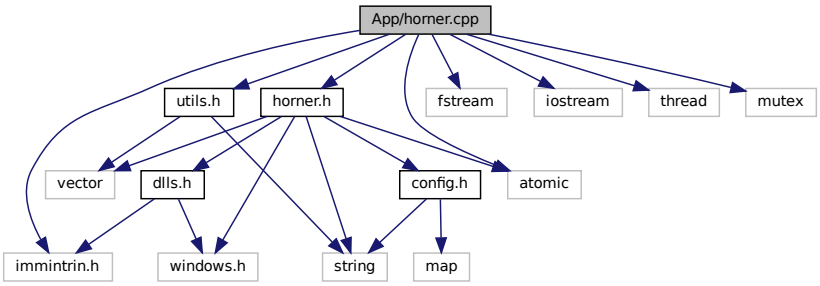
7.8 Dokumentacja pliku App/horner.cpp

Plik z implementacją funkcji obliczających wartości wielomianu.

```
#include "horner.h"
#include "utils.h"
#include <fstream>
#include <iostream>
#include <thread>
#include <immintrin.h>
#include <atomic>
```

```
#include <mutex>
```

Wykres zależności załączania dla horner.cpp:



Funkcje

- void computePolynomial (const Settings &settings, const std::vector< float > &coefficients, const std::vector< float > &points)

  Funkcja do obsługi różnych implementacji i konfiguracji obliczania wielomianu.
- void hornerScalar (float coeffs[ ], int n, float points[ ], int numPoints, float results[ ], int start, int end, std::atomic< int > &progressCounter, int progressUpdateInterval, const std::string &progressFile, PolynomialFunc polynomialFunc)

  Oblicza wartości wielomianu stopnia n w punktach z tablicy points wykorzystując skalnarną implementację.
- void hornerAvx (float coeffs[ ], int n, float points[ ], int numPoints, float results[ ], int start, int end, std::atomic< int > &progressCounter, int progressUpdateInterval, const std::string &progressFile, PolynomialFuncAvx polynomialFuncAvx, PolynomialFunc polynomialFunc)

  Oblicza wartości wielomianu stopnia n w punktach z tablicy points za pomocą implementacji AVX.
- void hornerScalarMultithreaded (float coeffs[ ], int n, float points[ ], int numPoints, float results[ ], int numThreads, int progressUpdateInterval, const std::string &progressFile, PolynomialFunc polynomialFunc)

  Oblicza wartości wielomianu stopnia n w punktach z tablicy points, korzystając ze skalarnych obliczeń wielowątkowych.
- void hornerAvxMultithreaded (float coeffs[ ], int n, float points[ ], int numPoints, float results[ ], int numThreads, int progressUpdateInterval, const std::string &progressFile, PolynomialFuncAvx polynomialFuncAvx, PolynomialFunc polynomialFunc)

  Oblicza wartości wielomianu stopnia n w punktach z tablicy points, wykorzystując implementację AVX oraz wielowątkowość.

7.8.1 Opis szczegółowy

Plik z implementacją funkcji obliczających wartości wielomianu.

Autor

krzsztfwtk

Wersja

2.1

Data

2024-12-20

Copyright

7.8.2 Dokumentacja funkcji

7.8.2.1 computePolynomial()

```
void computePolynomial (
        const Settings & settings,
        const std::vector< float > & coefficients,
        const std::vector< float > & points )
```

Funkcja do obsługi różnych implementacji i konfiguracji obliczania wielomianu.

Parametry

settings	Konfiguracja wywołania funkcji obliczającej wielomian.
coefficients	Lista współczynników wielomianu.
points	Zbiór punktów, dla których obliczane są wartości wielomianu.

Definicja w linii 22 pliku horner.cpp.

7.8.2.2 hornerAvx()

```
void hornerAvx (
        float coeffs[],
        int n,
        float points[],
        int numPoints,
        float results[],
        int start,
        int end,
        std::atomic< int > & progressCounter,
        int progressUpdateInterval,
        const std::string & progressFile,
        PolynomialFuncAvx polynomialFuncAvx,
        PolynomialFunc polynomialFunc )
```

Oblicza wartości wielomianu stopnia n w punktach z tablicy points za pomocą implementacji AVX.

Parametry

coeffs	Lista współczynników wielomianu.
n	Liczba współczynników.
points	Zbiór punktów, dla których obliczane są wartości wielomianu.
numPoints	Liczba punktów, dla których obliczane są wartości wielomianu.
results	Wartości wielomianu w punktach.
start	Początkowy indeks punktów, dla których obliczane są wartości wielomianu.
end	Końcowy indeks punktów, dla których obliczane są wartości wielomianu.
progressCounter	Poziom postępu obliczeń.
progressUpdateInterval	Interwał pomiędzy aktualizacjami postępu.
progressFile	Plik, do którego zapisywany jest poziom postępu obliczeń.
polynomialFuncAvx	Funkcja AVX obliczająca wartość wielomianu.
polynomialFunc	Funkcja skalarna obliczająca wartość wielomianu.

Definicja w linii 147 pliku horner.cpp.

7.8.2.3 hornerAvxMultithreaded()

```
void hornerAvxMultithreaded (
        float coeffs[],
        int n,
        float points[],
        int numPoints,
        float results[],
        int numThreads,
        int progressUpdateInterval,
        const std::string & progressFile,
        PolynomialFuncAvx polynomialFuncAvx,
        PolynomialFunc polynomialFunc )
```

Oblicza wartości wielomianu stopnia n w punktach z tablicy points, wykorzystując implementację AVX oraz wielowątkowość.

Parametry

coeffs	Lista współczynników wielomianu.
n	Liczba współczynników wielomianu.
points	Zbiór punktów, dla których obliczane są wartości wielomianu.
numPoints	Liczba punktów, dla których obliczane są wartości wielomianu.
results	Tablica, do której zapisywane są wyniki obliczeń.
numThreads	Liczba wątków używanych w obliczeniach.
progressUpdateInterval	Interwał pomiędzy aktualizacjami poziomu postępu.
progressFile	Ścieżka do pliku, w którym zapisywany jest poziom postępu obliczeń.
polynomialFuncAvx	Funkcja AVX obliczająca wartość wielomianu.
polynomialFunc	Funkcja skalarnie obliczająca wartość wielomianu.

Definicja w linii 220 pliku horner.cpp.

7.8.2.4 hornerScalar()

```
void hornerScalar (
        float coeffs[],
        int n,
        float points[],
        int numPoints,
        float results[],
        int start,
        int end,
        std::atomic< int > & progressCounter,
        int progressUpdateInterval,
        const std::string & progressFile,
        PolynomialFunc polynomialFunc )
```

Oblicza wartości wielomianu stopnia n w punktach z tablicy points wykorzystując skalnarną implementację.

Parametry

coeffs	Lista współczynników wielomianu.
n	Liczba współczynników.
points	Zbiór punktów, dla których obliczane są wartości wielomianu.
numPoints	Liczba punktów dla których obliczane są wartości wielomianu.
results	Wartości wielomianu w punktach.
start	Indeks początkowy punktów dla których obliczane są wartości wielomianu.
end	Indeks końcowy punktów dla których obliczane są wartości wielomianu.
progressCounter	Poziom postępu obliczeń.
progressUpdateInterval	Przerwa między aktualizacjami postępu obliczeń.
progressFile	Plik, do którego zapisywany jest poziom postępu obliczeń.
polynomialFunc	Funckja skalarna obliczająca wartość wielomianu.

Definicja w linii 125 pliku horner.cpp.

7.8.2.5 hornerScalarMultithreaded()

```
void hornerScalarMultithreaded (
        float coeffs[],
        int n,
        float points[],
        int numPoints,
        float results[],
        int numThreads,
        int progressUpdateInterval,
        const std::string & progressFile,
        PolynomialFunc polynomialFunc )
```

Oblicza wartości wielomianu stopnia n w punktach z tablicy points, korzystając ze skalarnych obliczeń wielowątkowych.

Parametry

coeffs	Lista współczynników wielomianu.
n	Liczba współczynników wielomianu.
points	Zbiór punktów, dla których obliczane są wartości wielomianu.
numPoints	Liczba punktów, dla których obliczane są wartości wielomianu.
results	Tablica, do której zapisywane są wyniki obliczeń.
numThreads	Liczba wątków używanych w obliczeniach.
progressUpdateInterval	Interwał pomiędzy aktualizacjami poziomu postępu.
progressFile	Ścieżka do pliku, w którym zapisywany jest poziom postępu obliczeń.
polynomialFunc	Funkcja skalarnie obliczająca wartość wielomianu.

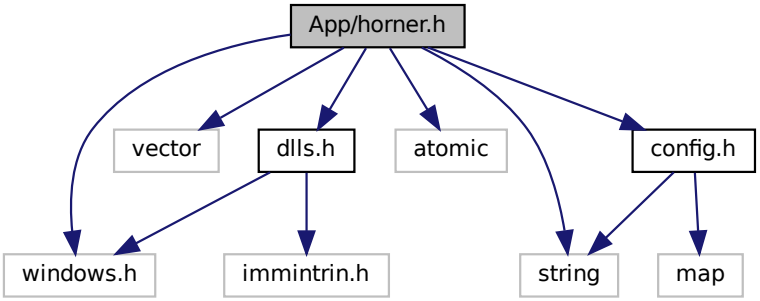
Definicja w linii 192 pliku horner.cpp.

7.9 Dokumentacja pliku App/horner.h

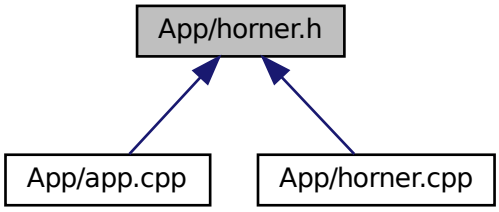
Plik nagłówkowy z funkcjami obliczające wartości wielomianu.

```
#include <windows.h>
#include <vector>
#include <string>
#include <atomic>
#include "config.h"
#include "dlls.h"
```

Wykres zależności załączania dla horner.h:



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



Definicje

- #define NOMINMAX

Funkcje

- void computePolynomial (const Settings &settings, const std::vector< float > &coefficients, const std::vector< float > &points)

  Funkcja do obsługi różnych implementacji i konfiguracji obliczania wielomianu.
- bool writeProgress (const std::string &filename, int progress)

  Zapisuje postęp obliczeń do pliku.

- void hornerScalar (float coeffs[ ], int n, float points[ ], int numPoints, float results[ ], int start, int end, std::atomic< int > &progressCounter, int progressUpdateInterval, const std::string &progressFile, PolynomialFunc polynomialFunc)

  Oblicza wartości wielomianu stopnia n w punktach z tablicy points wykorzystując skalnarną implementację.
- void hornerAvx (float coeffs[ ], int n, float points[ ], int numPoints, float results[ ], int start, int end, std::atomic< int > &progressCounter, int progressUpdateInterval, const std::string &progressFile, PolynomialFuncAvx polynomialFuncAvx, PolynomialFunc polynomialFunc)

  Oblicza wartości wielomianu stopnia n w punktach z tablicy points za pomocą implementacji AVX.
- void hornerScalarMultithreaded (float coeffs[ ], int n, float points[ ], int numPoints, float results[ ], int numThreads, int progressUpdateInterval, const std::string &progressFile, PolynomialFunc polynomialFunc)

  Oblicza wartości wielomianu stopnia n w punktach z tablicy points, korzystając ze skalarnych obliczeń wielowątkowych.
- void hornerAvxMultithreaded (float coeffs[ ], int n, float points[ ], int numPoints, float results[ ], int numThreads, int progressUpdateInterval, const std::string &progressFile, PolynomialFuncAvx polynomialFuncAvx, PolynomialFunc polynomialFunc)

  Oblicza wartości wielomianu stopnia n w punktach z tablicy points, wykorzystując implementację AVX oraz wielowątkowość.

7.9.1 Opis szczegółowy

Plik nagłówkowy z funkcjami obliczające wartości wielomianu.

Autor

krzsztfwtk

Wersja

2.1

Data

2024-12-20

Copyright

Copyright (c) 2025 krzsztfwtk

7.9.2 Dokumentacja definicji

7.9.2.1 NOMINMAX `#define NOMINMAX`

Definicja w linii 14 pliku horner.h.

7.9.3 Dokumentacja funkcji

7.9.3.1 computePolynomial() `void computePolynomial (`

```
const Settings & settings,
const std::vector< float > & coefficients,
const std::vector< float > & points )
```

Funkcja do obsługi różnych implementacji i konfiguracji obliczania wielomianu.

Parametry

settings	Konfiguracja wywołania funkcji obliczającej wielomian.
coefficients	Lista współczynników wielomianu.
points	Zbiór punktów, dla których obliczane są wartości wielomianu.

Definicja w linii 22 pliku horner.cpp.

7.9.3.2 hornerAvx()

```
void hornerAvx (
        float coeffs[],
        int n,
        float points[],
        int numPoints,
        float results[],
        int start,
        int end,
        std::atomic< int > & progressCounter,
        int progressUpdateInterval,
        const std::string & progressFile,
        PolynomialFuncAvx polynomialFuncAvx,
        PolynomialFunc polynomialFunc )
```

Oblicza wartości wielomianu stopnia n w punktach z tablicy points za pomocą implementacji AVX.

Parametry

coeffs	Lista współczynników wielomianu.
n	Liczba współczynników.
points	Zbiór punktów, dla których obliczane są wartości wielomianu.
numPoints	Liczba punktów, dla których obliczane są wartości wielomianu.
results	Wartości wielomianu w punktach.
start	Początkowy indeks punktów, dla których obliczane są wartości wielomianu.
end	Końcowy indeks punktów, dla których obliczane są wartości wielomianu.
progressCounter	Poziom postępu obliczeń.
progressUpdateInterval	Interwał pomiędzy aktualizacjami postępu.
progressFile	Plik, do którego zapisywany jest poziom postępu obliczeń.
polynomialFuncAvx	Funkcja AVX obliczająca wartość wielomianu.
polynomialFunc	Funkcja skalarna obliczająca wartość wielomianu.

Definicja w linii 147 pliku horner.cpp.

7.9.3.3 hornerAvxMultithreaded()

```
void hornerAvxMultithreaded (
        float coeffs[],
        int n,
        float points[],
        int numPoints,
```

```
float results[],
int numThreads,
int progressUpdateInterval,
const std::string & progressFile,
PolynomialFuncAvx polynomialFuncAvx,
PolynomialFunc polynomialFunc )
```

Oblicza wartości wielomianu stopnia n w punktach z tablicy points, wykorzystując implementację AVX oraz wielowątkowość.

Parametry

coeffs	Lista współczynników wielomianu.
n	Liczba współczynników wielomianu.
points	Zbiór punktów, dla których obliczane są wartości wielomianu.
numPoints	Liczba punktów, dla których obliczane są wartości wielomianu.
results	Tablica, do której zapisywane są wyniki obliczeń.
numThreads	Liczba wątków używanych w obliczeniach.
progressUpdateInterval	Interwał pomiędzy aktualizacjami poziomu postępu.
progressFile	Ścieżka do pliku, w którym zapisywany jest poziom postępu obliczeń.
polynomialFuncAvx	Funkcja AVX obliczająca wartość wielomianu.
polynomialFunc	Funkcja skalarnie obliczająca wartość wielomianu.

Definicja w linii 220 pliku horner.cpp.

7.9.3.4 hornerScalar() `void hornerScalar (`

```
float coeffs[],
int n,
float points[],
int numPoints,
float results[],
int start,
int end,
std::atomic< int > & progressCounter,
int progressUpdateInterval,
const std::string & progressFile,
PolynomialFunc polynomialFunc )
```

Oblicza wartości wielomianu stopnia n w punktach z tablicy points wykorzystując skalnarną implementację.

Parametry

coeffs	Lista współczynników wielomianu.
n	Liczba współczynników.
points	Zbiór punktów, dla których obliczane są wartości wielomianu.
numPoints	Liczba punktów dla których obliczane są wartości wielomianu.
results	Wartości wielomianu w punktach.
start	Indeks początkowy punktów dla których obliczane są wartości wielomianu.
end	Indeks końcowy punktów dla których obliczane są wartości wielomianu.
progressCounter	Poziom postępu obliczeń.

Parametry

progressUpdateInterval	Przerwa między aktualizacjami postępu obliczeń.
progressFile	Plik, do którego zapisywany jest poziom postępu obliczeń.
polynomialFunc	Funckja skalarna obliczająca wartość wielomianu.

Definicja w linii 125 pliku horner.cpp.

7.9.3.5 hornerScalarMultithreaded()

```
void hornerScalarMultithreaded (
        float coeffs[],
        int n,
        float points[],
        int numPoints,
        float results[],
        int numThreads,
        int progressUpdateInterval,
        const std::string & progressFile,
        PolynomialFunc polynomialFunc )
```

Oblicza wartości wielomianu stopnia n w punktach z tablicy points, korzystając ze skalarnych obliczeń wielowątkowych.

Parametry

coeffs	Lista współczynników wielomianu.
n	Liczba współczynników wielomianu.
points	Zbiór punktów, dla których obliczane są wartości wielomianu.
numPoints	Liczba punktów, dla których obliczane są wartości wielomianu.
results	Tablica, do której zapisywane są wyniki obliczeń.
numThreads	Liczba wątków używanych w obliczeniach.
progressUpdateInterval	Interwał pomiędzy aktualizacjami poziomu postępu.
progressFile	Ścieżka do pliku, w którym zapisywany jest poziom postępu obliczeń.
polynomialFunc	Funkcja skalarnie obliczająca wartość wielomianu.

Definicja w linii 192 pliku horner.cpp.

7.9.3.6 writeProgress()

```
bool writeProgress (
        const std::string & filename,
        int progress )
```

Zapisuje postęp obliczeń do pliku.

Parametry

filename	Plik, do którego zapisywany jest postęp obliczeń.
progress	Procet postępu obliczeń.

Zwraca

true Poprawnie zapisano postęp obliczeń.

false Nieuadło się zapisać postępu obliczeń.

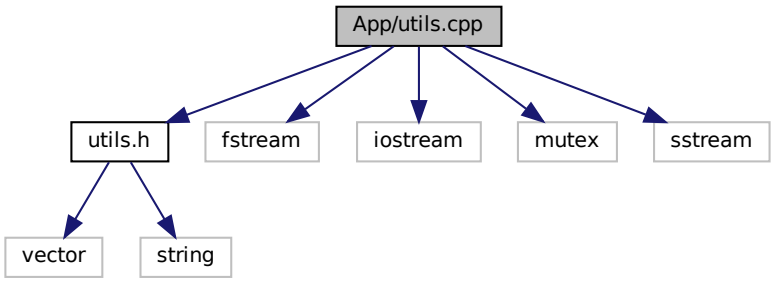
Definicja w linii 110 pliku utils.cpp.

7.10 Dokumentacja pliku App/utils.cpp

Funkcje pomocnicze wykorzystywane w różnych modułach.

```
#include "utils.h"
#include <fstream>
#include <iostream>
#include <mutex>
#include <sstream>
```

Wykres zależności załączania dla utils.cpp:



Funkcje

- bool readCoefficients (const std::string &filename, std::vector< float > &coefficients)

  *Wczytuje wsp*czynniki wielomianu z pliku.*
- bool readPoints (const std::string &filename, std::vector< float > &points)

  *Wczytuje punkty, dla kt*rych b*dzie obliczany wielomian, z pliku.*
- bool writeResults (const std::string &filename, const std::vector< float > &results)

  Zapisuje wyniki oblicze do pliku.*
- bool writeComputationTime (const std::string &filename, double computationTime)

  Zapisuje czas oblicze do pliku.*
- bool writeProgress (const std::string &filename, int progress)

  Zapisuje postęp obliczeń do pliku.

7.10.1 Opis szczegółowy

Funkcje pomocnicze wykorzystywane w różnych modułach.

Autor

krzsztfwtk

Wersja

2.1

Data

2024-12-20

Copyright

Copyright (c) 2025 krzsztfwtk

7.10.2 Dokumentacja funkcji

7.10.2.1 readCoefficients()

```
bool readCoefficients (
        const std::string & filename,
        std::vector< float > & coefficients )
```

Wczytuje wsp�czynniki wielomianu z pliku.

Parametry

filename	Nazwa pliku zawieraj�cego wsp�czynniki.
coefficients	Wektor, do kt�rego zostan� zapisane wsp�czynniki.

Zwraca

true Je�li odczyt zako�czy si sukcesem.

false Je�li odczyt nie powi�d si (np. plik nie istnieje lub ma z�y format).

Definicja w linii 20 pliku utils.cpp.

7.10.2.2 readPoints()

```
bool readPoints (
        const std::string & filename,
        std::vector< float > & points )
```

Wczytuje punkty, dla kt�rych b�dzie obliczany wielomian, z pliku.

Parametry

filename	Nazwa pliku zawieraj\u00f8cego punkty.
points	Wektor, do kt\u00f8rego zostan\u00f8 zapisane punkty.

Zwraca

true Je\u00f8li operacja zako\u00f8czy\u00f8a si\u00f8 powodzeniem.

false Je\u00f8li operacja nie powiod\u00f8a si\u00f8.

Definicja w linii 51 pliku utils.cpp.

7.10.2.3 writeComputationTime() `bool writeComputationTime (`

```
const std::string & filename,
double computationTime )
```

Zapisuje czas oblicze\u00f8 do pliku.

Parametry

filename	Nazwa pliku, do kt\u00f8rego zostanie zapisany czas oblicze\u00f8.
computationTime	Czas oblicze\u00f8 w sekundach.

Zwraca

true Je\u00f8li operacja zako\u00f8czy\u00f8a si\u00f8 powodzeniem.

false Je\u00f8li operacja nie powiod\u00f8a si\u00f8.

Definicja w linii 98 pliku utils.cpp.

7.10.2.4 writeProgress() `bool writeProgress (`

```
const std::string & filename,
int progress )
```

Zapisuje postęp obliczeń do pliku.

Zapisuje post\u00f8p oblicze\u00f8 do pliku.

Parametry

filename	Plik, do którego zapisywany jest postęp obliczeń.
progress	Procet postępu obliczeń.

Zwraca

true Poprawnie zapisano postęp obliczeń.

false Nieuadło się zapisać postępu obliczeń.

Definicja w linii 110 pliku utils.cpp.

7.10.2.5 writeResults()

```
bool writeResults (
        const std::string & filename,
        const std::vector< float > & results )
```

Zapisuje wyniki oblicze do pliku.

Parametry

filename	Nazwa pliku, do kt rego zostan zapisane wyniki.
results	Wektor zawieraj cy wyniki oblicze .

Zwraca

true Je li operacja zako czy a si powodzeniem.

false Je li operacja nie powiod a si .

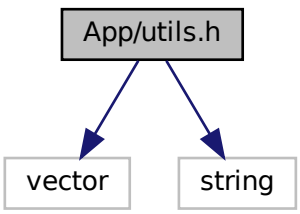
Definicja w linii 84 pliku utils.cpp.

7.11 Dokumentacja pliku App/utils.h

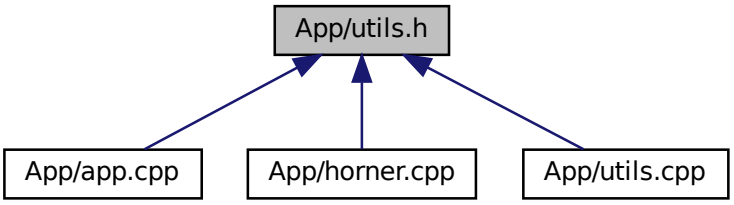
Funkcje pomocnicze wykorzystywane w r nych modu ach.

```
#include <vector>
#include <string>
```

Wykres zależności załączania dla utils.h:



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



Funkcje

- bool readCoefficients (const std::string &filename, std::vector< float > &coefficients)
  Wczytuje wsp�czynniki wielomianu z pliku.
- bool readPoints (const std::string &filename, std::vector< float > &points)
  Wczytuje punkty, dla kt�rych b�dzie obliczany wielomian, z pliku.
- bool writeResults (const std::string &filename, const std::vector< float > &results)
  Zapisuje wyniki oblicze� do pliku.
- bool writeComputationTime (const std::string &filename, double computationTime)
  Zapisuje czas oblicze� do pliku.
- bool writeProgress (const std::string &filename, int progress)
  Zapisuje post�p oblicze� do pliku.

7.11.1 Opis szczegółowy

Funkcje pomocnicze wykorzystywane w r�nych modu�ach.

Autor

krzsztfwtk

Wersja

2.1

Data

2024-12-20

Copyright

7.11.2 Dokumentacja funkcji

7.11.2.1 readCoefficients() `bool readCoefficients ( const std::string & filename, std::vector< float > & coefficients )`

Wczytuje wsp�czynniki wielomianu z pliku.

Parametry

filename	Nazwa pliku zawieraj�cego wsp�czynniki.
coefficients	Wektor, do kt�rego zostan� zapisane wsp�czynniki.

Zwraca

true Je�li odczyt zako�czy� si� sukcesem.

false Je�li odczyt nie powi�d� si� (np. plik nie istnieje lub ma z�y format).

Definicja w linii 20 pliku utils.cpp.

7.11.2.2 readPoints() `bool readPoints (`

```
const std::string & filename,
std::vector< float > & points )
```

Wczytuje punkty, dla kt�rych b�dzie obliczany wielomian, z pliku.

Parametry

filename	Nazwa pliku zawieraj�cego punkty.
points	Wektor, do kt�rego zostan� zapisane punkty.

Zwraca

true Je�li operacja zako�czy�a si� powodzeniem.

false Je�li operacja nie powiod�a si�.

Definicja w linii 51 pliku utils.cpp.

7.11.2.3 writeComputationTime() `bool writeComputationTime (`

```
const std::string & filename,
double computationTime )
```

Zapisuje czas oblicze� do pliku.

Parametry

filename	Nazwa pliku, do kt�rego zostanie zapisany czas oblicze�.
computationTime	Czas oblicze� w sekundach.

Zwraca

true Je[illegible]li operacja zako[illegible]czy[illegible]a si[illegible] powodzeniem.

false Je[illegible]li operacja nie powiod[illegible]a si[illegible].

Definicja w linii 98 pliku utils.cpp.

7.11.2.4 writeProgress() `bool writeProgress ( const std::string & filename, int progress )`

Zapisuje post[illegible]p oblicze[illegible] do pliku.

Parametry

filename	Nazwa pliku, do kt[illegible]rego zostanie zapisany post[illegible]p oblicze[illegible].
progress	Aktualny post[illegible]p w procentach.

Zwraca

true Je[illegible]li operacja zako[illegible]czy[illegible]a si[illegible] powodzeniem.

false Je[illegible]li operacja nie powiod[illegible]a si[illegible].

Zapisuje post[illegible]p oblicze[illegible] do pliku.

Parametry

filename	Plik, do którego zapisywany jest postęp obliczeń.
progress	Procet postępu obliczeń.

Zwraca

true Poprawnie zapisano postęp obliczeń.

false Nieuadło się zapisać postępu obliczeń.

Definicja w linii 110 pliku utils.cpp.

7.11.2.5 writeResults() `bool writeResults ( const std::string & filename, const std::vector< float > & results )`

Zapisuje wyniki oblicze[illegible] do pliku.

Parametry

filename	Nazwa pliku, do kt[illegible]rego zostan[illegible] zapisane wyniki.
results	Wektor zawieraj[illegible]cy wyniki oblicze[illegible].

Zwraca

true Je♦li operacja zako♦czy♦a si♦ powodzeniem.

false Je♦li operacja nie powiod♦a si♦.

Definicja w linii 84 pliku utils.cpp.

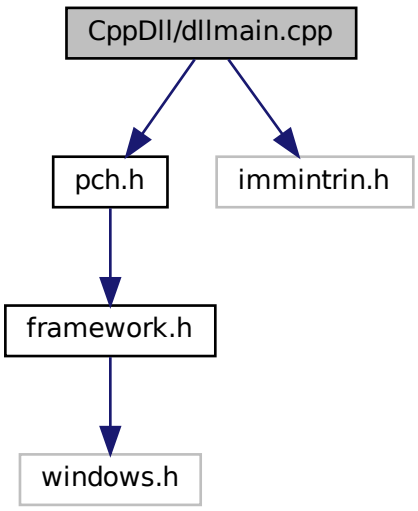
7.12 Dokumentacja pliku AsmDll/AsmDll.asm

7.13 Dokumentacja pliku CppDll/dllmain.cpp

Określa punkt wejścia dla aplikacji DLL.

```
#include "pch.h"
#include <immintrin.h>
```

Wykres zależności załączania dla dllmain.cpp:



Funkcje

- __declspec (dllexport) float HornerPolynomial(float ∗x
- for (int i=n - 1;i >=0;i--)
- BOOL APIENTRY DllMain (HMODULE hModule, DWORD ul_reason_for_call, LPVOID lpReserved)

Zmienne

- float a [ ]
- float int n
- float value = ∗x
- return result = _mm256_setzero_ps()

7.13.1 Opis szczegółowy

Określa punkt wejścia dla aplikacji DLL.

Autor

krzsztfwtk

Wersja

2.1

Data

2024-12-20

Copyright

Copyright (c) 2025 krzsztfwtk

7.13.2 Dokumentacja funkcji

7.13.2.1 __declspec() `__declspec ( dllexport )`

7.13.2.2 DllMain() `BOOL APIENTRY DllMain ( HMODULE hModule, DWORD ul_reason_for_call, LPVOID lpReserved )`

Definicja w linii 37 pliku dllmain.cpp.

7.13.2.3 for() `for ( int i = n - 1; i >= 0; i-- )`

Definicja w linii 20 pliku dllmain.cpp.

7.13.3 Dokumentacja zmiennych

7.13.3.1 a `float a`

Definicja w linii 15 pliku dllmain.cpp.

7.13.3.2 n `float int n`

Wartość początkowa:
```
{
    float result = 0.0f
```

Definicja w linii 15 pliku dllmain.cpp.

7.13.3.3 result `return result = _mm256_setzero_ps()`

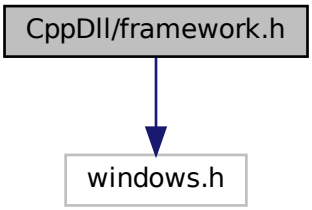
Definicja w linii 24 pliku dllmain.cpp.

7.13.3.4 value `float value = *x`

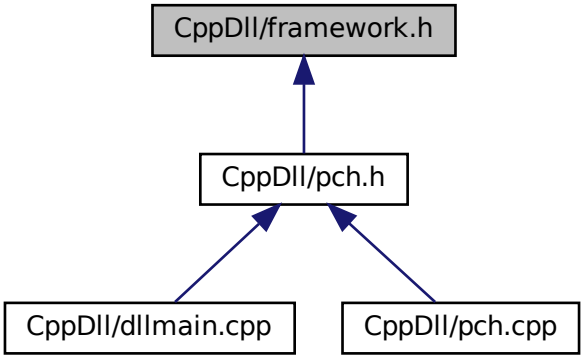
Definicja w linii 18 pliku dllmain.cpp.

7.14 Dokumentacja pliku CppDll/framework.h

```
#include <windows.h>
```
Wykres zależności załączania dla framework.h:



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



Definicje

- #define WIN32_LEAN_AND_MEAN

7.14.1 Dokumentacja definicji

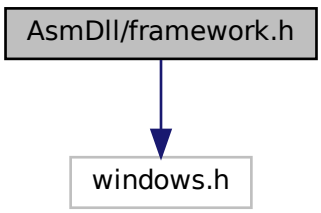
7.14.1.1 WIN32_LEAN_AND_MEAN `#define WIN32_LEAN_AND_MEAN`

Definicja w linii 3 pliku framework.h.

7.15 Dokumentacja pliku AsmDll/framework.h

```
#include <windows.h>
```

Wykres zależności załączania dla framework.h:



Definicje

- #define WIN32_LEAN_AND_MEAN

7.15.1 Dokumentacja definicji

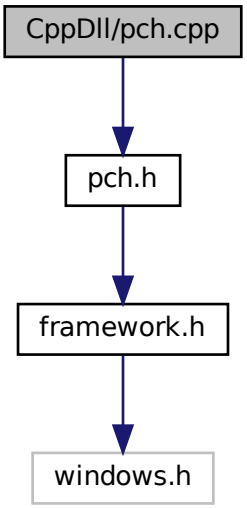
7.15.1.1 WIN32_LEAN_AND_MEAN `#define WIN32_LEAN_AND_MEAN`

Definicja w linii 3 pliku framework.h.

7.16 Dokumentacja pliku CppDll/pch.cpp

```
#include "pch.h"
```

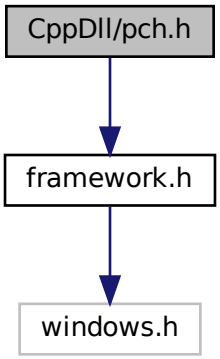
Wykres zależności załączania dla pch.cpp:



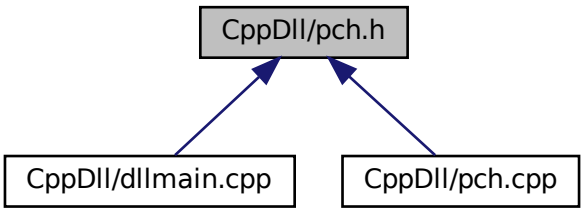
7.17 Dokumentacja pliku CppDll/pch.h

```
#include "framework.h"
```

Wykres zależności załączania dla pch.h:



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



7.18 Dokumentacja pliku PythonGUI/gui/__init__.py

Przestrzenie nazw

- gui

7.19 Dokumentacja pliku PythonGUI/gui/config.py

Przestrzenie nazw

- gui.config

Zmienne

- string gui.config.EVENT_NAME = "Global\\ComputeEvent"
- string gui.config.COMPLETION_EVENT_NAME = "Global\\CompletionEvent"
- gui.config.TEMP_DIR = os.path.abspath('temp')
- gui.config.CONFIG_FILE = os.path.join('config.ini')
- gui.config.PROGRESS_FILE = os.path.join(TEMP_DIR, 'progress.tmp')
- gui.config.COMPUTATION_TIME_FILE = os.path.join(TEMP_DIR, 'computation.time')

7.20 Dokumentacja pliku PythonGUI/gui/events.py

Przestrzenie nazw

- gui.events

Funkcje

- def gui.events.trigger_event (event_name)
- def gui.events.monitor_progress_and_completion (app)
- def gui.events.on_closing (self)

Zmienne

- gui.events.is_running

7.21 Dokumentacja pliku PythonGUI/gui/experiment.py

Przestrzenie nazw

- gui.experiment

Funkcje

- def gui.experiment.show_experiment_info ()
- def gui.experiment.run_experiment (app)

7.22 Dokumentacja pliku PythonGUI/gui/file_utils.py

Przestrzenie nazw

- gui.file_utils

Funkcje

- def gui.file_utils.load_file (title, filetypes)
- str gui.file_utils._generate_coefficients (int n, float coeff_min, float coeff_max)
- str gui.file_utils._generate_points (float start, float end, float step)
- def gui.file_utils.generate_coefficients_file (app, n=None, coeff_min=None, coeff_max=None)
- def gui.file_utils.generate_points_file (app, start=None, end=None, step=None)
- def gui.file_utils.write_config_file (implementation, multithreading, threads_number, avx, save_results, generate_chart, output_file, coefficients_file, points_file)
- def gui.file_utils.read_results_and_display (app)

7.23 Dokumentacja pliku PythonGUI/gui/main_window.py

Komponenty

- class gui.main_window.MainWindow

Przestrzenie nazw

- gui.main_window

7.24 Dokumentacja pliku PythonGUI/gui/plotting.py

Przestrzenie nazw

- gui.plotting

Funkcje

- def gui.plotting.plot_results (app, points, results)

7.25 Dokumentacja pliku PythonGUI/PythonGUI.py

Przestrzenie nazw

- PythonGUI

Zmienne

- PythonGUI.root = tk.Tk()
- PythonGUI.app = MainWindow(root)

Indeks