2gether - Reliability Assessment Model (RAM)

Athanasios Salamanis (asal@iti.gr)

Version v1.0

Thu Jul 1 2021

Table of Contents

Namespace Index

Namespace List

Here is a list of all namespaces with brief descriptions:

Class Index

Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

File Index

File List

Here is a list of all files with brief descriptions:

Namespace Documentation

gnfnc Namespace Reference

Functions

- std::string **getExecutablePath** ()
- std::string **getExecutablePathAndMatchItWithFilename** (const std::string &fileName)

Function Documentation

std::string gnfnc::getExecutablePath ()

Returns the absolute path of the executable's directory.

Returns:

the absolute path of the executable's directory.

Definition at line 8 of file GenericFunc.cpp.

std::string gnfnc::getExecutablePathAndMatchItWithFilename (const std::string & *fileName*)

Returns the absolute path a file located in the executable's directory.

Parameters:

fileName	the name of the file (e.g., file.txt)

Returns:

the absolute path of the file

Definition at line 17 of file GenericFunc.cpp.

Class Documentation

Artist Class Reference

```
#include <Artist.h>
```

Public Member Functions

- **Artist** ()
- **Artist** (int id)
- **~Artist** ()
- void **setID** (int id)
- int **getID** () const
- void **addRAMrep** (double RAMrep)
- void **addBENrep** (double BENrep)
- void **saveRAMreps** ()
- void **saveBENreps** ()
- void **setMalicious** (bool malicious)
- bool **isMalicious** () const
- void **addRating** (**Rating** *rating)
- void **computeAverageRatingReceived** ()
- double **getAverageRatingReceived** () const
- void **computeAverageRatingGiven** ()
- double **getAverageRatingGiven** () const
- void **addArtwork** (int artworkID, **Artwork** *artwork)
- **Artwork** * **getArtwork** (int artworkID)
- std::map< int, **Artwork** * > * **getArtworks** ()
- int **getNumOfArtworks** ()
- bool **artworkExists** (int artworkID)
- void **printArtworksIDs** ()

Detailed Description

Definition at line 9 of file Artist.h.

Constructor & Destructor Documentation

Artist::Artist ()

Default constructor.

Definition at line 10 of file Artist.cpp.

Artist::Artist (int *id*)

Constructor.

Parameters:

id	the ID of the **Artist** object.

Definition at line 18 of file Artist.cpp.

Artist::~Artist ()

Destructor.

Definition at line 26 of file Artist.cpp.

Member Function Documentation

void Artist::addArtwork (int *artworkID*, Artwork * *artwork*)

Adds a new **Artwork** object in the list of **Artwork** objects of the artist.

Parameters:

artworkID	the ID of the **Artwork** object.
artwork	the **Artwork** object.

Definition at line 153 of file Artist.cpp.

void Artist::addBENrep (double *BENrep*)

Adds a new benchmark reputation value in the vector of benchmark reputation values.

Parameters:

BENrep	the benchmark reputation value to be added.

Definition at line 64 of file Artist.cpp.

void Artist::addRAMrep (double *RAMrep*)

Adds a new RAM reputation value in the vector of RAM reputation values.

Parameters:

RAMrep	the RAM reputation value to be added.

Definition at line 59 of file Artist.cpp.

void Artist::addRating (Rating * *rating*)

Adds a new **Rating** object (i.e., pointer to **Rating** object) to the vector of **Rating** objects (i.e., vector of pointers to rating objects).

Parameters:

rating	the new **Rating** object.

Definition at line 111 of file Artist.cpp.

bool Artist::artworkExists (int *artworkID*)

Checks if an **Artwork** object with specific ID exists in the list of **Artwork** objects of the **Artist**.

Parameters:

artworkID	the ID of the **Artwork** object.

Returns:

true if the **Artwork** object belongs to the **Artist**, false otherwise.

Definition at line 147 of file Artist.cpp.

void Artist::computeAverageRatingGiven ()

Computes the average of all ratings given by the **Artist** for the artworks of other artists.

Definition at line 132 of file Artist.cpp.

void Artist::computeAverageRatingReceived ()

Computes the average of the ratings received for all the artworks of the **Artist**.

Definition at line 116 of file Artist.cpp.

Artwork * Artist::getArtwork (int *artworkID*)

Returns an **Artwork** object from the list of **Artwork** objects of the **Artist**.

Parameters:

artworkID	the ID of the **Artwork** object.

Returns:

the **Artwork** object.

Definition at line 161 of file Artist.cpp.

std::map< int, Artwork * > * Artist::getArtworks ()

Returns the list of **Artwork** objects of the **Artist**.

Returns:

the list of **Artwork** objects of the **Artist**.

Definition at line 182 of file Artist.cpp.

double Artist::getAverageRatingGiven () const

Returns the average of all ratings given by the **Artist** for the artworks of other artists.

Returns:

the average of all ratings given by the **Artist** for the artworks of other artists.

Definition at line 142 of file Artist.cpp.

double Artist::getAverageRatingReceived () const

Returns the average of the ratings received for all the artworks of the **Artist**.

Returns:

the average of the ratings received for all the artworks of the **Artist**.

Definition at line 127 of file Artist.cpp.

int Artist::getID () const

Returns the ID of the **Artist** object.

Returns:

the ID of the **Artist** object.

Definition at line 54 of file Artist.cpp.

int Artist::getNumOfArtworks ()

Returns the number of **Artwork** objects of the **Artist**.

Returns:

the number of **Artwork** objects of the **Artist**.

Definition at line 187 of file Artist.cpp.

bool Artist::isMalicious () const

Returns the status of **Artist** (i.e., malicious, non-malicious).

Returns:

a flag indicating if the **Artist** is malicious or not.

Definition at line 106 of file Artist.cpp.

void Artist::printArtworksIDs ()

Prints the IDs of the **Artwork** objects of the **Artist**.

Definition at line 173 of file Artist.cpp.

void Artist::saveBENreps ()

Saves the vector of benchmark reputation values into a file.

Definition at line 85 of file Artist.cpp.

void Artist::saveRAMreps ()

Saves the vector of RAM reputation values into a file.

Definition at line 69 of file Artist.cpp.

void Artist::setID (int *id*)

Sets the ID of the **Artist** object.

Parameters:

id	the ID of the **Artist** object.

Definition at line 49 of file Artist.cpp.

void Artist::setMalicious (bool *malicious*)

Sets the **Artist** as malicious.

Parameters:

malicious	flag (true) to set **Artist** as malicious.

Definition at line 101 of file Artist.cpp.

The documentation for this class was generated from the following files:

- **Artist.h**
- **Artist.cpp**

Artwork Class Reference

`#include <Artwork.h>`

Public Member Functions

- **Artwork** ()
- **Artwork** (int id, int ownerId)
- **~Artwork** ()
- void **setID** (int id)
- int **getID** () const
- void **setOwnerID** (int ownerId)
- int **getOwnerID** () const
- void **addRating** (**Rating** *rating)
- void **computeAverageRating** ()
- double **getAverageRating** () const

Detailed Description

Definition at line 8 of file Artwork.h.

Constructor & Destructor Documentation

Artwork::Artwork ()

Default constructor.

Definition at line 5 of file Artwork.cpp.

Artwork::Artwork (int *id*, int *ownerId*)

Constructor.

Parameters:

id	the ID of the **Artwork** object.
ownerId	the ID of the **Artist** who owns the **Artwork** object.

Definition at line 12 of file Artwork.cpp.

Artwork::~Artwork ()

Destructor.

Definition at line 19 of file Artwork.cpp.

Member Function Documentation

void Artwork::addRating (Rating * *rating*)

Adds a new **Rating** object (i.e., pointer to **Rating** object) to the vector of **Rating** objects (i.e., vector of pointers to rating objects).

Parameters:

rating	the new **Rating** object.

Definition at line 51 of file Artwork.cpp.

void Artwork::computeAverageRating ()

Computes the average of all ratings received for the **Artwork** object from other artists of the platform.

Definition at line 56 of file Artwork.cpp.

double Artwork::getAverageRating () const

Returns the average of all ratings received for the **Artwork** object from other artists of the platform.

Returns:

the average of all ratings received for the **Artwork** object from other artists of the platform.

Definition at line 66 of file Artwork.cpp.

int Artwork::getID () const

Returns the ID of the **Artwork** object.

Returns:

the ID of the **Artwork** object.

Definition at line 36 of file Artwork.cpp.

int Artwork::getOwnerID () const

Returns the ID of the **Artist** who owns the **Artwork** object.

Returns:

the ID of the **Artist** who owns the **Artwork** object.

Definition at line 46 of file Artwork.cpp.

void Artwork::setID (int *id*)

Sets the ID of the **Artwork** object.

Parameters:

id	the ID of the **Artwork** object.

Definition at line 31 of file Artwork.cpp.

void Artwork::setOwnerID (int *ownerId*)

Sets the ID of the **Artist** who owns the **Artwork** object.

Parameters:

id	the ID of the **Artist** who owns the **Artwork** object.

Definition at line 41 of file Artwork.cpp.

The documentation for this class was generated from the following files:

- **Artwork.h**
- **Artwork.cpp**

Rating Class Reference

```
#include <Rating.h>
```

Public Member Functions

- **Rating** ()
- **Rating** (int fromArtist, int forArtwork, int stars)
- **~Rating** ()
- void **setFromArtist** (int fromArtist)
- int **getFromArtist** () const
- void **setForArtwork** (int forArtwork)
- int **getForArtwork** () const
- void **setStars** (int stars)
- int **getStars** () const

Detailed Description

Definition at line 4 of file Rating.h.

Constructor & Destructor Documentation

Rating::Rating ()

Default constructor.

Definition at line 3 of file Rating.cpp.

Rating::Rating (int *fromArtist*, int *forArtwork*, int *stars*)

Constructor.

Parameters:

fromArtist	the id of the **Artist** object that made the **Rating**.
forArtwork	the id of the **Artwork** object that received the **Rating**.
stars	the actual rating value (integer in [1, 5]).

Definition at line 10 of file Rating.cpp.

Rating::~Rating ()

Destructor.

Definition at line 17 of file Rating.cpp.

Member Function Documentation

int Rating::getForArtwork () const

Returns the ID of the **Artwork** object that received the **Rating**.

Returns:

the ID of the **Artwork** object that received the **Rating**.

Definition at line 36 of file Rating.cpp.

int Rating::getFromArtist () const

Returns the ID of the **Artist** object that made the **Rating**.

Returns:

the ID of the **Artist** object that made the **Rating**.

Definition at line 26 of file Rating.cpp.

int Rating::getStars () const

Returns the actual rating value (integer in [1, 5]).

Returns:

the actual rating value (integer in [1, 5]).

Definition at line 46 of file Rating.cpp.

void Rating::setForArtwork (int *forArtwork*)

Sets the ID of the **Artwork** object that received the **Rating**.

Parameters:

id	the ID of the **Artwork** object that received the **Rating**.

Definition at line 31 of file Rating.cpp.

void Rating::setFromArtist (int *fromArtist*)

Sets the ID of the **Artist** object that made the **Rating**.

Parameters:

id	the ID of the **Artist** object that made the **Rating**.

Definition at line 21 of file Rating.cpp.

void Rating::setStars (int *stars*)

Sets the actual rating value (integer in [1, 5]).

Parameters:

stars	the actual rating value (integer in [1, 5]).

Definition at line 41 of file Rating.cpp.

The documentation for this class was generated from the following files:

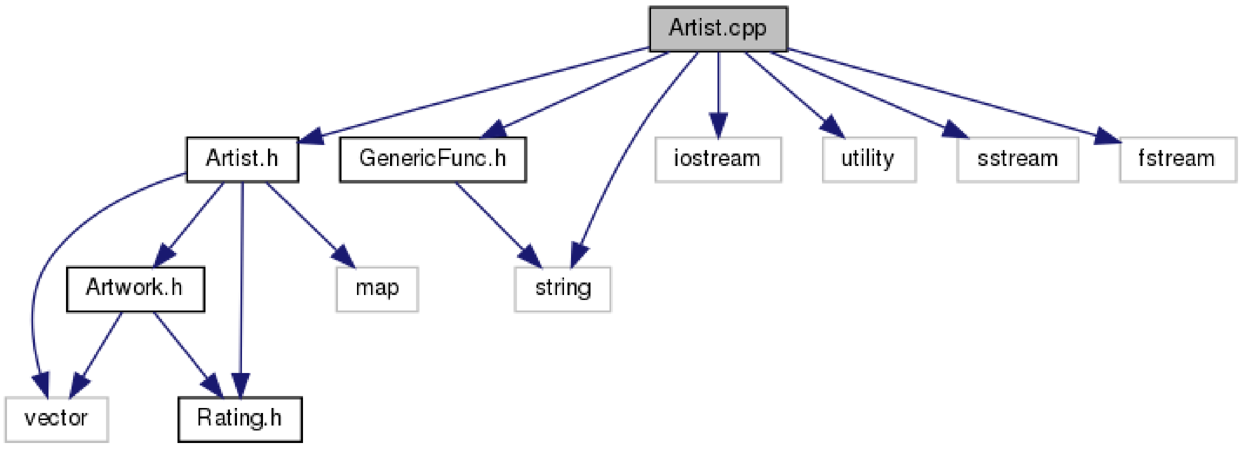
- **Rating.h**
- **Rating.cpp**

File Documentation

Artist.cpp File Reference

```
#include "Artist.h"
#include "GenericFunc.h"
#include <iostream>
#include <utility>
#include <sstream>
#include <string>
#include <fstream>
```

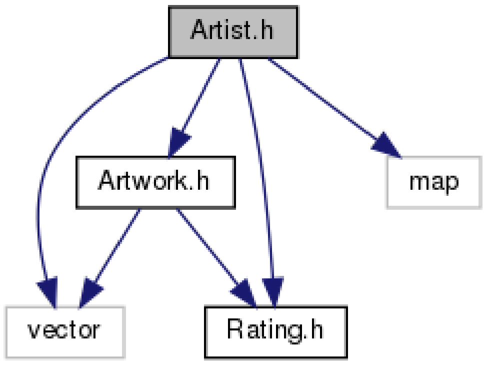
Include dependency graph for Artist.cpp:



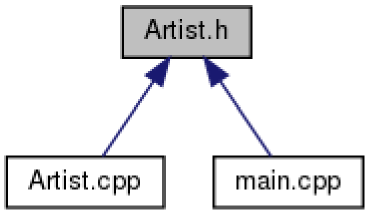
Artist.h File Reference

```
#include "Artwork.h"
#include "Rating.h"
#include <map>
#include <vector>
```

Include dependency graph for Artist.h:



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



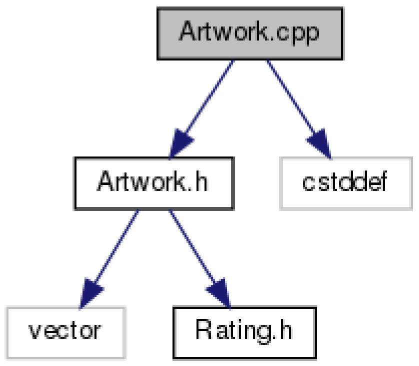
Classes

- class **Artist**

Artwork.cpp File Reference

```
#include "Artwork.h"
#include <cstddef>
```

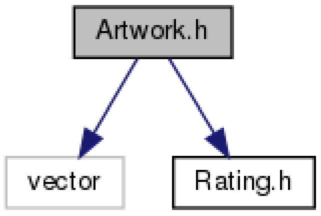
Include dependency graph for Artwork.cpp:



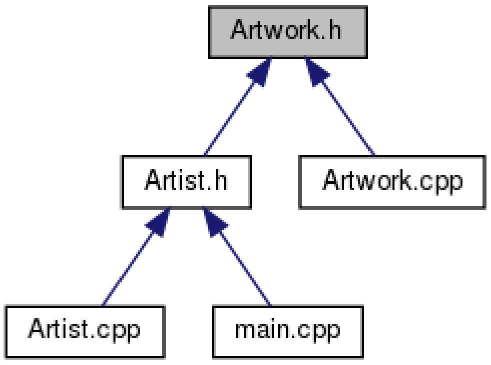
Artwork.h File Reference

```
#include <vector>
#include "Rating.h"
```

Include dependency graph for Artwork.h:



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

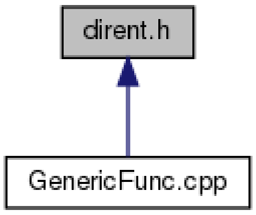


Classes

- class **Artwork**

dirent.h File Reference

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



Macros

- #define **DIRENT_H_INCLUDED**

Macro Definition Documentation

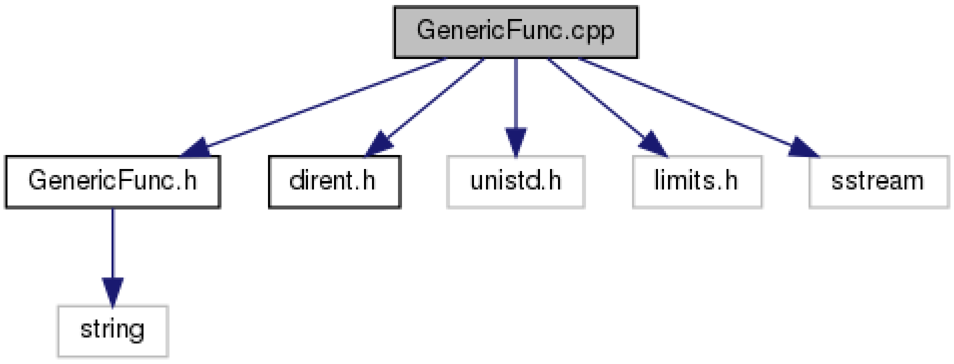
#define DIRENT_H_INCLUDED

Definition at line 83 of file dirent.h.

GenericFunc.cpp File Reference

```
#include "GenericFunc.h"
#include "dirent.h"
#include <unistd.h>
#include <limits.h>
#include <sstream>
```

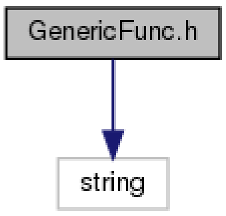
Include dependency graph for GenericFunc.cpp:



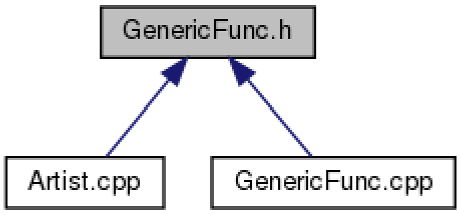
GenericFunc.h File Reference

```
#include <string>
```

Include dependency graph for GenericFunc.h:



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



Namespaces

- **gnfnc**

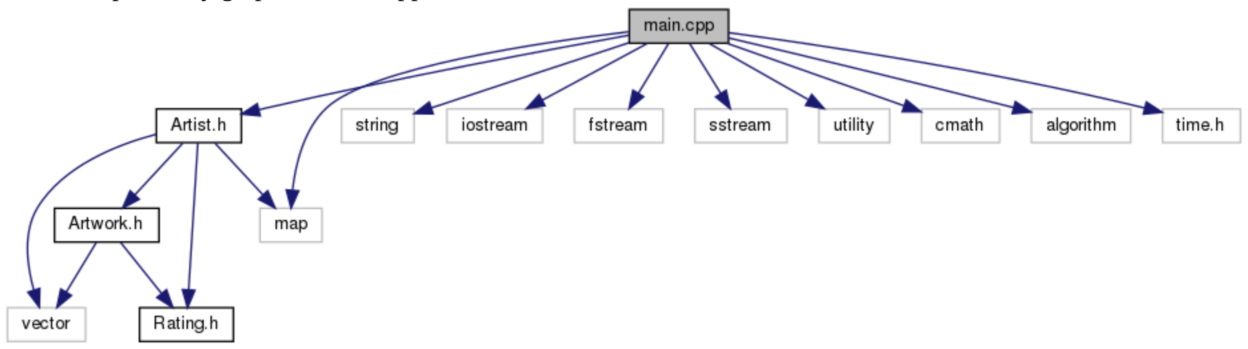
Functions

- std::string **gnfnc::getExecutablePath** ()
- std::string **gnfnc::getExecutablePathAndMatchItWithFilename** (const std::string &fileName)

main.cpp File Reference

```
#include "Artist.h"
#include <string>
#include <iostream>
#include <fstream>
#include <sstream>
#include <map>
#include <utility>
#include <cmath>
#include <algorithm>
#include <time.h>
```

Include dependency graph for main.cpp:



Functions

- int **main** (int argc, char **argv)

Function Documentation

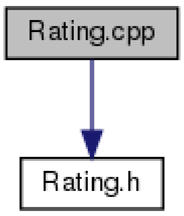
int main (int *argc*, char ** *argv*)

Definition at line 13 of file main.cpp.

Rating.cpp File Reference

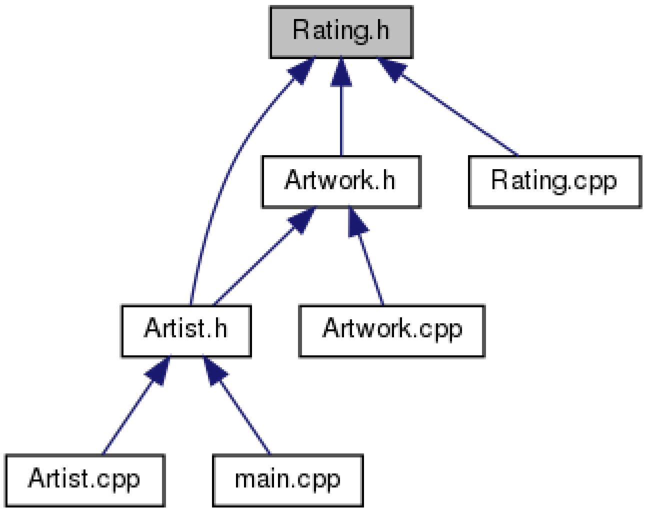
```
#include "Rating.h"
```

Include dependency graph for Rating.cpp:



Rating.h File Reference

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



Classes

- class **Rating**

Index

INDEX