

Doxygen

Un générateur de documentation

laurent.thiry@uha.fr

¹<https://www.doxygen.nl/index.html>

Motivation: comment profiter de "stdio.h" & co. ?

Motivation: comment profiter de "stdio.h" & co. ?

```
thiry@demetrios: ~
Linux Programmer's Manual
STDIO(3)

NAME
    stdio - standard input/output library functions

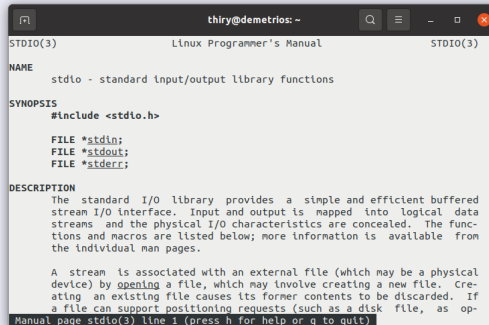
SYNOPSIS
    #include <stdio.h>

    FILE *stdin;
    FILE *stdout;
    FILE *stderr;

DESCRIPTION
    The standard I/O library provides a simple and efficient buffered
    stream I/O interface. Input and output is mapped into logical data
    streams and the physical I/O characteristics are concealed. The func-
    tions and macros are listed below; more information is available from
    the individual man pages.

    A stream is associated with an external file (which may be a physical
    device) by opening a file, which may involve creating a new file. Cre-
    ating an existing file causes its former contents to be discarded. If
    a file can support positioning requests (such as a disk file, as op-
    Manual page stdio(3) line 1 (press h for help or q to quit)
```

Motivation: comment profiter de "stdio.h" & co. ?



```
thiry@demetrios: ~  
STDIO(3) Linux Programmer's Manual STDIO(3)  
  
NAME  
    stdio - standard input/output library functions  
  
SYNOPSIS  
    #include <stdio.h>  
  
    FILE *stdin;  
    FILE *stdout;  
    FILE *stderr;  
  
DESCRIPTION  
    The standard I/O library provides a simple and efficient buffered  
    stream I/O interface. Input and output is mapped into logical data  
    streams and the physical I/O characteristics are concealed. The func-  
    tions and macros are listed below; more information is available from  
    the individual man pages.  
  
    A stream is associated with an external file (which may be a physical  
    device) by opening a file, which may involve creating a new file. Cre-  
    ating an existing file causes its former contents to be discarded. If  
    a file can support positioning requests (such as a disk file, as op-  
    Manual page stdio(3) line 1 (press h for help or q to quit)
```

⇒ Comment faire une "bonne" documentation ? Est-ce que la doc Gtk vous a été utile ?

Principe

- 1 Utiliser des `/*commentaires*/` ... pour chaque élément de code
 - fichiers
 - structures et/ou classes
 - sous-programmes et fonctions
 - ...

Principe

- ❶ Utiliser des `/*commentaires*/` ... pour chaque élément de code
 - fichiers
 - structures et/ou classes
 - sous-programmes et fonctions
 - ...
- ❷ Intégrer des \balises spécifiques
 - auteur(s)
 - champ(s)
 - paramètre(s)
 - ...

Par exemple: comment profiter du code suivant ?

```
struct Vector { int x; int y; };  
typedef struct Vector* Vect;
```

```
Vect new_Vect(int x, int y);  
Vect add(Vect v1, Vect v2);  
void print(Vect v);
```

Par exemple: comment profiter du code suivant ?

```
struct Vector { int x; int y; };  
typedef struct Vector* Vect;
```

```
Vect new_Vect(int x, int y);  
Vect add(Vect v1, Vect v2);  
void print(Vect v);
```

- Quelle sont les coordonnées (x,y) considérées ?

Par exemple: comment profiter du code suivant ?

```
struct Vector { int x; int y; };  
typedef struct Vector* Vect;
```

```
Vect new_Vect(int x, int y);  
Vect add(Vect v1, Vect v2);  
void print(Vect v);
```

- Quelle sont les coordonnées (x,y) considérées ?
- Est-ce que "add" modifie les paramètres ?

Par exemple: comment profiter du code suivant ?

```
struct Vector { int x; int y; };  
typedef struct Vector* Vect;
```

```
Vect new_Vect(int x, int y);  
Vect add(Vect v1, Vect v2);  
void print(Vect v);
```

- Quelle sont les coordonnées (x,y) considérées ?
- Est-ce que "add" modifie les paramètres ?
- Est-ce que "print" fait appel l'imprimante ?
- ...

Avec des commentaires ... c'est plus facilement (ré)utilisable

```
/**  
 * "Vect.h"  
 * v0.1  
 */  
  
/**  
 * Vecteur 2D en coordonnées cartésiennes entières.  
 */  
struct Vector { int x; int y; };  
typedef struct Vector* Vect;  
  
/**  
 * Constructeur  
 */  
Vect new_Vect(int x, int y);
```

En ajoutant des \balises (éléments sémantiques) pour les fichiers:

```
/**  
 * \file Vect.h  
 * \brief Vecteurs 2D avec opérations usuelles (add,print)  
 * \author laurent.thiry@uha.fr  
 * \version 0.1  
 *  
 * Exemple de code source documenté avec Doxygen.  
 */
```

... pour les structures (ou des classes C++, etc.):

```
/**  
 * \struct Vector Vect.h Vect.c  
 * \brief vecteur 2D avec coordonnées entières (4 octets)  
 */  
struct Vector {  
    /*! abscisse */  
    int x;  
    /*! ordonnée */  
    int y;  
};  
typedef struct Vector* Vect;
```

... pour les programmes:

```
/**  
 * \fn new_Vect(int x, int y)  
 * \brief constructeur  
 * \param x abscisse entière  
 * \return référence sur un nouveau vecteur  
 */  
Vect new_Vect(int x, int y);
```

... pour les programmes:

```
/**  
 * \fn new_Vect(int x, int y)  
 * \brief constructeur  
 * \param x abscisse entière  
 * \return référence sur un nouveau vecteur  
 */  
Vect new_Vect(int x, int y);
```

⇒ *Doxygen* permet alors de générer une documentation Html à partir de ces nouvelles informations ! Et plus encore...

Installation

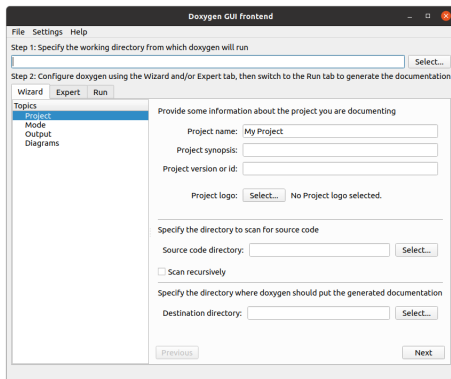
```
sudo apt install graphviz doxygen doxygen-gui
```


Installation

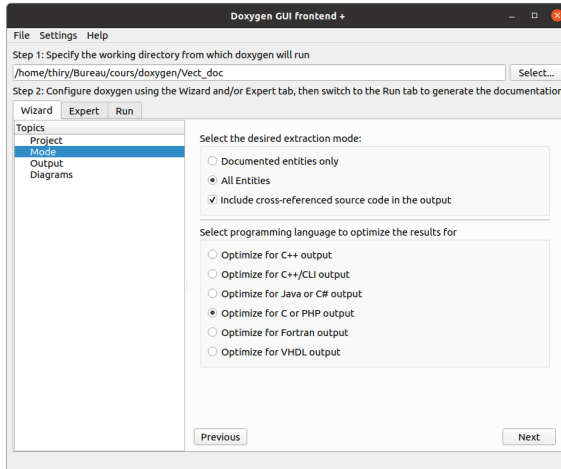
```
sudo apt install graphviz doxygen doxygen-gui
```

Lancement

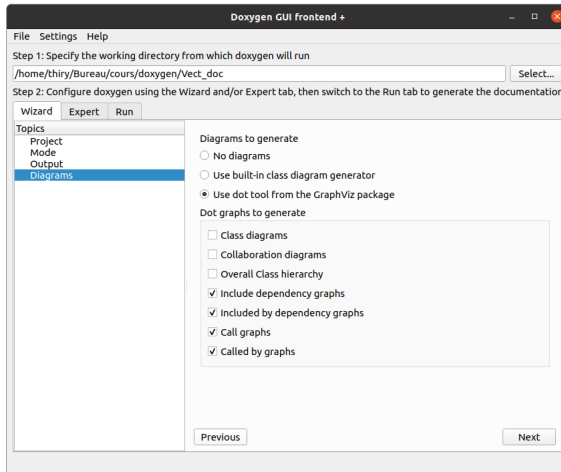
doxywizard



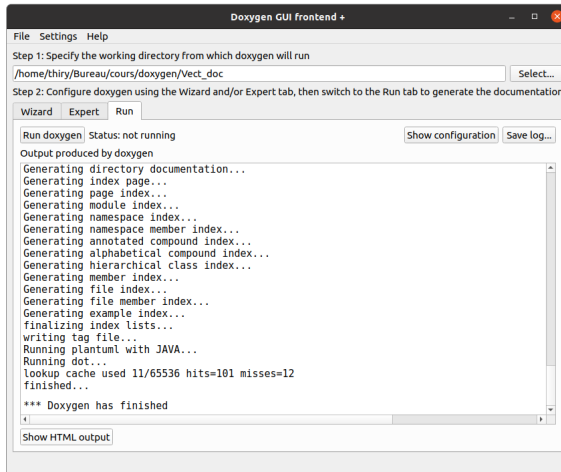
⇒ Prendre du temps pour bien configurer les paramètres pour avoir un meilleur résultat ...



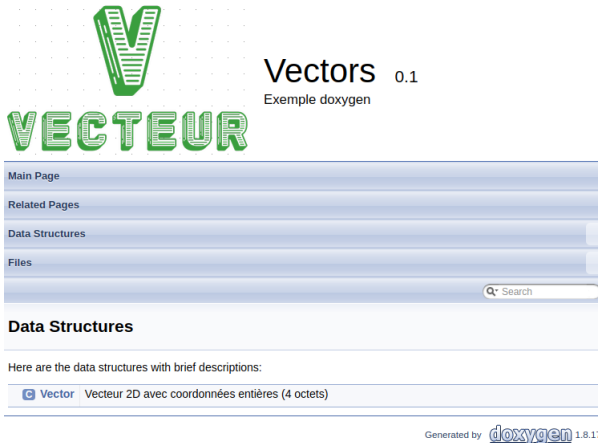
⇒ Doxygen permet de générer le graphe de dépendances !



Enfin, on génère la documentation...



Et voilà !



The screenshot shows a web page generated by Doxygen. At the top left is a logo consisting of a large green 'V' above the word 'VECTEUR' in green, all on a grid background. To the right of the logo, the text 'Vectors 0.1' is displayed, followed by 'Exemple doxygen' in a smaller font. Below this is a navigation bar with links: 'Main Page', 'Related Pages', 'Data Structures', and 'Files'. A search box is located on the right side of this bar. The main content area is titled 'Data Structures' and contains the text 'Here are the data structures with brief descriptions:'. Below this text is a table with one entry: a blue icon, the word 'Vector', and the description 'Vecteur 2D avec coordonnées entières (4 octets)'. At the bottom right of the page, it says 'Generated by doxygen 1.8.17'.


Vectors 0.1
Exemple doxygen

Main Page
Related Pages
Data Structures
Files

Search

Data Structures

Here are the data structures with brief descriptions:

 Vector	Vecteur 2D avec coordonnées entières (4 octets)
--	---

Generated by doxygen 1.8.17

¹see./doc/html/index.html

Et il y a un peu plus que les commentaires :-)

◆ new_Vect()

```
Vect new_Vect ( int x,  
               int y  
             )
```

constructeur

Parameters

x abscisse entière

Returns

référence sur un nouveau vecteur

Definition at line 5 of file [Vect.c](#).

Here is the caller graph for this function:



Merci de votre attention

Avez-vous des questions ?