

Info2 - Compilation Séparée

17.5.2021

- Modèle: ↗ fichier source C (main + fonctions)
↳ gestion des arguments
↳ gestion des erreurs
↳ gestion des fichiers

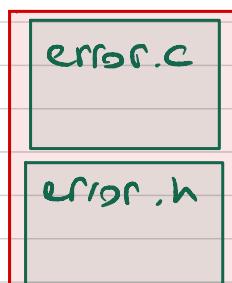
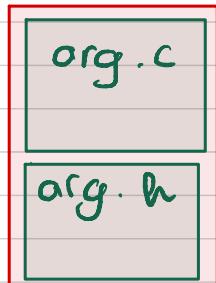
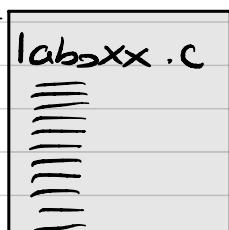
Maintenabilité: facile

Portage des fonctions: compliqué

Décomposition du source C en plusieurs

MODULES officielles

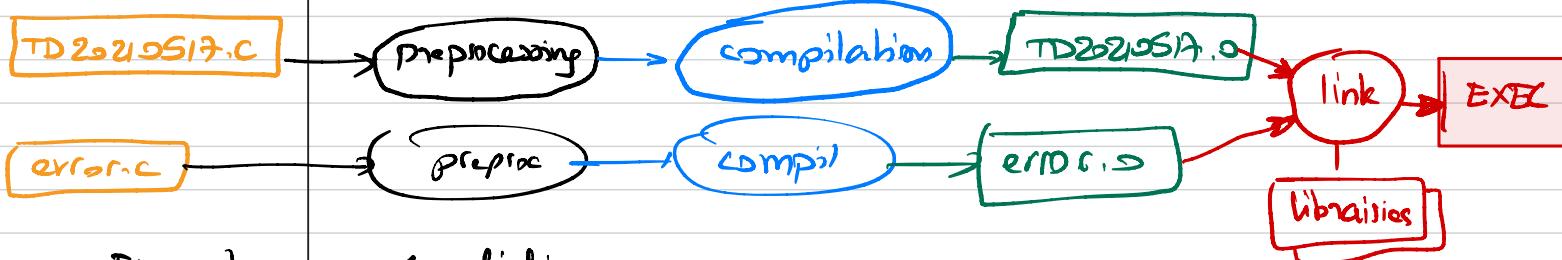
- ↳ .c
↳ .h



module "arg"

module "error"

Compilation "make"



Exemple avec 1 fichier

```
→ TD20210517 git:(main) ✘ make
→ gcc -std=c99 -Wall -g -Iinclude -c TD20210517.c -o obj/TD20210517.o -MMD -MF obj/TD20210517.d
→ gcc -o app obj/TD20210517.o -lm
```

Exemple avec 2 fichiers

```
→ TD20210517 git:(main) ✘ make
→ gcc -std=c99 -Wall -g -Iinclude -c error.c -o obj/error.o -MMD -MF obj/error.d
→ gcc -std=c99 -Wall -g -Iinclude -c TD20210517.c -o obj/TD20210517.o -MMD -MF obj/TD20210517.d
→ gcc -o app obj/error.o obj/TD20210517.o -lm
```

Tous les fichiers .c sont compilés SÉPARÉMENT

TD 20210517

↳ TD 20210517.c

└ main

└ void displayErrors (eErrorCode) ;

Créer le type enum eErrorCode avec :

E_NO_ERROR

E_ERROR_1

E_ERROR_2

Main

Répondre à l'entrée d'affichage : n

Si n = 2 → displayError(E_NO_ERROR)

Si n < 2

Erreur

E_ERROR_1

E_ERROR_2

TD 20210517

TD 20210517.c

```
#include < stdio.h> ..  
#include "error.h"
```

main ()
=

ERROR

error.c "implémentation du module"

#include "error.h"

code de la fonction displayError .

error.h "présentation du module"

#include < stdio.h> // printf

typedef enum {

...

} eErrorCode;

prototyp de la fonction displayError

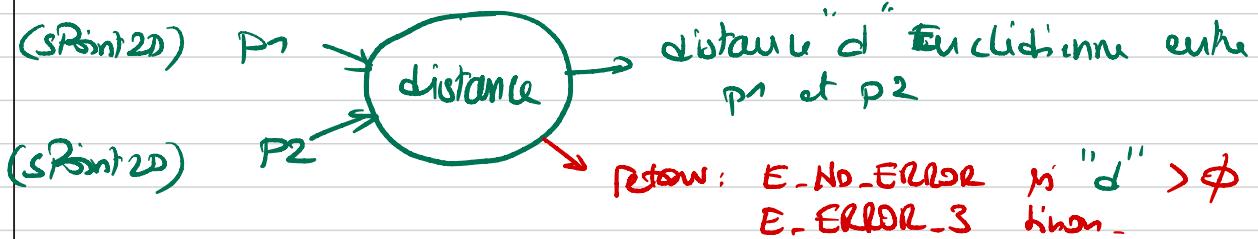
15'50

TD 2020S17

nonlinear module

computation . h

- type structure "SPoint2D" 2 champs reels x et y
- fonction "distance"



Depuis le main:

$$P_1 = (10, 40)$$

$$P_2 = (-15, 37)$$

- calcul + affichage distance (P_1, P_2)
- (P_2, P_2)

computation.c

computation.h

16h20

```

→ TD20210517 git:(main) ✘ make
gcc -std=c99 -Wall -g -Iinclude -c error.c -o obj/error.o -MMD -MF obj/error.d
gcc -std=c99 -Wall -g -Iinclude -c TD20210517.c -o obj/TD20210517.o -MMD -MF obj/TD20210517.d
In file included from computation.h:4:0,
                 from TD20210517.c:16:
error.h:7:5: error: redeclaration of enumerator 'E_NO_ERROR'
    E_NO_ERROR = 0,
    ^~~~~~

```

```

#include <stdio.h> // standard library for
#include <stdlib.h>
#include <stdint.h>

#include "error.h"
#include "computation.h"

int main(int argc, char const *argv[])
{
    sPoint2D bidon = {5, 4}; // x=5 y=4
    sPoint2D p1 = {.x = 10, .y = 40};
    sPoint2D p2 = {.x = -15, .y = 37};
    eErrorCode e = E_NO_ERROR;
    double d = 0.;

    ...
}

```

Preprocessing

fichier intermédiaire qui sera compilé après

```

#include <stdio.h>

typedef enum
{
    E_NO_ERROR = 0,
    E_ERROR_1,
    E_ERROR_2,
    E_ERROR_3,
} eErrorCode;

void displayError(const eErrorCode e);

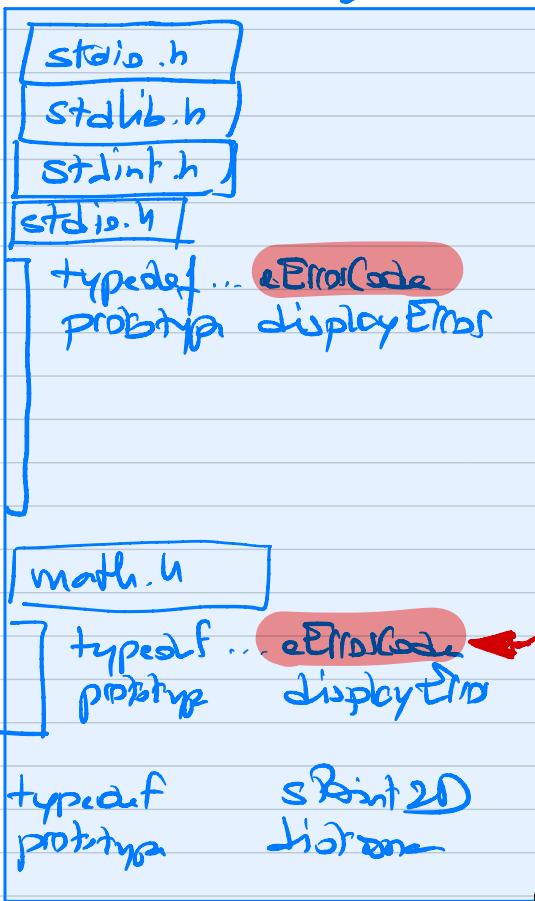
// computation.h

#include <math.h>
#include "error.h" ← 2x

typedef struct {
    double x;
    double y;
} sPoint2D;

eErrorCode distance(const sPo...

```



Protection contre les multiples inclusions:

→ mettre dans TOUS vos fichiers .h

En début de fichier.

```

// error.h
#pragma once

#include <stdio.h>

typedef enum
{
    E_NO_ERROR = 0,
    E_ERROR_1,
    E_ERROR_2,
    E_ERROR_3,
} eErrorCode;

void displayError(const eErrorCode e);

```