




epsi

MA & TMA





Programme d'aujourd'hui

1. Rappel des deux cours précédents
 1. Intérêts et objectifs du cours
 2. Présentation du projet fil rouge
 3. Rappel du travail attendu
2. Cours : le clean code
3. Autonomie de 30 minutes
4. Passage en revue individuel, noté, des travaux
5. Bilan du travail de TMA



Dimension pédagogique

- ◆ Décalage entre le système universitaire et professionnel:
 - ◆ Beaucoup de projets réalisés dans un cadre scolaire sont des travaux faits « from scratch »,
 - ◆ Alors que vous aurez quasi systématiquement une base de code existante. Et parfois assez monstrueuse.
- ◆ Cela pose deux problèmes majeurs:
 - ◆ La difficulté pour un jeune diplômé ou avec peu d'expérience à reprendre le code édité par des tiers,
 - ◆ La tentation de reprendre un travail « from scratch » au lieu de modifier l'existant.
- ◆ C'est ce que va s'efforcer de résoudre ce cours en vous habituant :
 - ◆ A faire un code facilement éditable par un tiers,
 - ◆ À vous habituer à travailler sur des grosses bases de code difficilement éditables.

C'est donc un cours important dans votre montée en compétence.



Objectifs globaux

- ◇ Développer une application maintenable
- ◇ Améliorer progressivement son approche
 - ◇ En développant sa manière de communiquer
 - ◇ Avec l'utilisation d'outils
 - ◇ En codant proprement
- ◇ En alternant plusieurs vagues successives de:
 - ◇ Modification d'application à partir d'une base ou from scratch faite par vous-même (Maintenance applicative)
 - ◇ Reprise d'application existante faite par d'autres (Tierce Maintenance Applicative)



Présentation du projet fil rouge

Cf navigateur



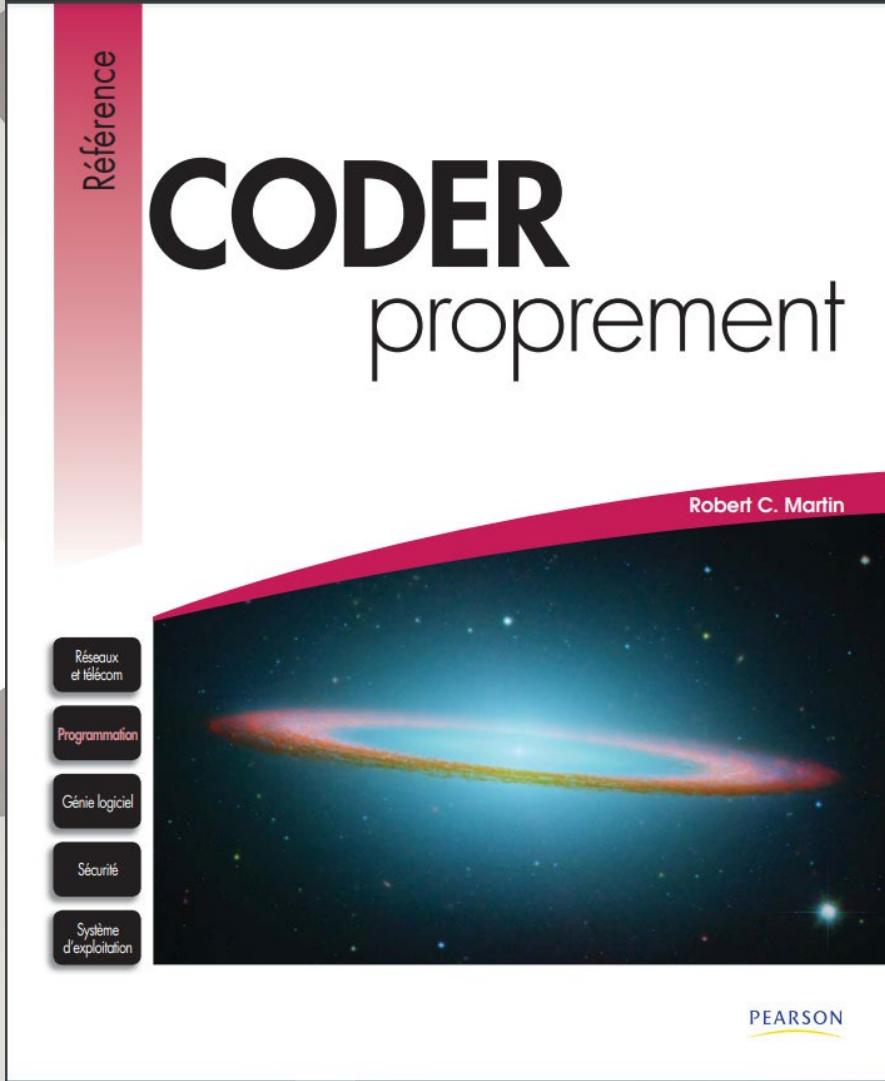
Présentation détaillée Epsilon PHP

- ◇ Cahier Des Charges
- ◇ Spécifications détaillées
- ◇ Modèle Logique de Donnée
- ◇ Revue de code
 - ◇ Front
 - ◇ Back
- ◇ <https://github.com/tvinchent-epsi/epsilonPhp>



Modifications attendues

- ◇ (très facile, 10 points) Branchement front : mêler le design dans le dossier « front » au backoffice dans le dossier « back ».
- ◇ (facile, 5 points) MultiUpload : donner la possibilité à l'utilisateur d'uploader plusieurs fichiers en même temps.
- ◇ (moyen, 10 points) Listing fichiers : faire une page qui présente, pour un utilisateur donné, les fichiers uploadés, si possible organisés par modules.
- ◇ (difficile, 3 points) isJoinable. Améliorer l'application pour qu'elle contienne des règles d'accès au module
- ◇ Attention :
 - ◇ Certaines modifications ne nécessitent pas une compréhension générale du code, d'autres demandent une revue détaillée du fonctionnement de l'application..
 - ◇ Les points correspondent à l'importance fonctionnelle de la modification aux yeux du client, pas à la difficulté.



Écrire du code propre

- ◇ Utiliser des noms significatifs pour les variables, fonctions etc. :
 - ◇ Révélateur des intentions
 - ◇ Se placer à la place de qqn qui effectue une recherche
- ◇ Fonctions
 - ◇ Faire court
 - ◇ Faire 1 seule chose
 - ◇ Ne pas se répéter
- ◇ Commentaires :
 - ◇ Expliquer le pourquoi (le code explique de lui-même le comment)
- ◇ Gestion des erreurs
 - ◇ try catch

Référence : Clean code, Robert C. Martin (oncle Bob)



Autonomie

- ◆ Finaliser votre travail avant revue notée
- ◆ Rappel
 - ◆ (très facile, 10 points) Branchement front : mêler le design dans le dossier « front » au backoffice dans le dossier « back ».
 - ◆ (facile, 5 points) MultiUpload : donner la possibilité à l'utilisateur d'uploader plusieurs fichiers en même temps.
 - ◆ (moyen, 10 points) Listing fichiers : faire une page qui présente, pour un utilisateur donné, les fichiers uploadés, si possible organisés par modules.
 - ◆ (difficile, 3 points) isJoinable. Améliorer l'application pour qu'elle contienne des règles d'accès au module
 - ◆ Attention :
 - ◆ Certaines modifications ne nécessitent pas une compréhension générale du code, d'autres demandent une revue détaillée du fonctionnement de l'application..
 - ◆ Les points correspondent à l'importance fonctionnelle de la modification aux yeux du client, pas à la difficulté.



Revue notée

- ◊ Je note en fonction de la présence ou non des fonctionnalités précédentes
- ◊ Pendant ce temps, les autres peuvent préparer le prochain travail en déposant un projet le plus clair possible (cf clean code) sur leur espace GIT



Bilan de la maintenance applicative

- ◆ Difficultés rencontrées
- ◆ Arbitrage des fonctionnalités à implémenter
- ◆ Ouverture : idées pour la suite ? (en rapport avec le BTS SIO ?)