



PROGRAMME FORMATION AGILE SOFTWARE CRAFTMANSHIP

DATES 2011

► 20 et 21 avril

Tél : +33 (0)1 53 89 99 99
Fax : +33 (0)1 53 89 99 97
Email: info@xebia-training.fr

Animée par Robert Cecil Martin

Avec près de 35 ans d'expérience, Robert Martin est une des figures emblématiques du monde du développement logiciel agile. Il est l'auteur de plusieurs ouvrages et articles sur le « craftsmanship » logiciel ou comment développer un logiciel dans les règles de l'Art, l'artisanat logiciel. Il est notamment l'auteur de : « Clean Code: A Handbook of Agile Software Craftmanship »

PRÉSENTATION

Même s'il est vrai qu'un logiciel peut fonctionner malgré un code de mauvaise qualité, ce même code peut également être la source de pertes de productivité graves pour une entreprise.

Cette formation pratique de 2 jours est inspirée des principes et bonnes pratiques énumérées dans l'ouvrage de Robert Martin : Clean code : A Handbook of Agile Software Craftmanship.

A LA FIN DE CETTE FORMATION, VOUS :

- Prendrez pleinement conscience des règles de l'art du « Software Craftmanship »
- Serez en mesure de faire la différence entre du bon et du mauvais code
- Saurez comment rédiger du bon code et corriger du mauvais code
- Saurez créer des bonnes conventions de nommage, des bonnes fonctionnalités, de bons objets et des bonnes classes
- Saurez formater du code pour une lisibilité maximale
- Saurez mettre en pratique la gestion des erreurs sans obscurcir la logique du code
- Maîtriserez les techniques de tests unitaires et de refactoring

INFORMATIONS

FORMATIONS	TYPE	LIEU	DATES	DURÉE	PRIX € HT
Formations agile software craftsmanship par Robert Cecil Martin	Inter-entreprise	Paris / Selon les demandes du client	20 et 21 avril	2 jours	2000

[S'INSCRIRE A LA FORMATION](#)



Première journée :

1- *Le code propre*

La première partie de journée traitera du mauvais code et le coût que celui-ci engendre (« The grand redesign in the sky »). Vous aborderez également les règles de l'art indispensables pour écrire du code propre, les opinions des experts et les différentes écoles

2- *Des conventions de nommages pertinentes*

Cette seconde partie traitera les sujets suivants:

- L'importance d'utiliser des noms cohérents avec les intentions du code,
- Comment éviter les informations erronées,
- Comment faire des distinctions et des nuances pertinentes
- L'importance d'utiliser des noms prononçables,
- Comment utiliser des noms que l'on peut retrouver,
- Comment éviter le code crypté et le « Mind Mapping »,
- Les noms de classe,
- Les noms de méthodes,
- Les raisons pour lesquelles il faut éviter les jeux de mots,
- Comment identifier des noms de domaines pertinents et éviter les noms de domaine problématiques.

3- *Les méthodes*

Cette partie traitera de l'importance de créer des méthodes simples et faciles à utiliser, de se concentrer sur une tâche à la fois et d'éviter trop d'abstractions. Robert Martin abordera également les arguments et paramètres à utiliser lorsque l'on nomme une méthode et les effets secondaires que ces choix peuvent engendrer.

4- *Les commentaires*

Dans cette partie, vous étudierez quand et pour quelles raisons il faut rédiger des commentaires, les méthodes pour s'exprimer à travers le code et la différence entre des bons et mauvais commentaires.

5- *L'indentation*

Ici, vous abordez l'utilité de l'indentation et ce que sont des indentations verticales et horizontales.

6- *Résumé*

Seconde journée

1- *Les objets et structures de données*

Cette première partie traitera en profondeur des abstractions de données, de l'anti- symétrie des données et des objets, de la loi de Demeter ainsi que les objets de transfert de données (DTO).

2- *La gestion des erreurs*

La seconde partie abordera les meilleures pratiques de gestion des erreurs (l'utilisation des exceptions, apporter un contexte à ces exceptions, définir des classes d'exceptions, la définition d'un flux normal, etc).

3- *Les frontières*

La troisième partie abordera les thèmes tel que le « Third Party code », l'exploration des frontières, Log4J, utiliser du code qui n'existe pas encore, etc...

4- *Les tests unitaires*

Vous aborderez dans la 4eme partie les 3 lois du Test Driven Development, comment garder des tests propres, les tests de langages pour chacun des différents domaines, OneAssert et F.I.R.S.T.

5- *Les classes*

La cinquième partie traitera de l'organisation des classes, des petites classes, le « single responsibility principle » (terme inventé par Robert), comment maintenir la cohésion et l'organisation du changement.

6- *Les Heuristiques*

7- *Conclusion*

