Conception et Programmation objet

Initiation, Vocabulaire, Conception, UML, Programmation, IDE, Framework, C#, Java

DESCRIPTION

Cette formation vous permet de comprendre l'intérêt et toutes les possibilités offertes par le **développement objet** par rapport à une **approche fonctionnelle** (fonctions isolées). Les limites de cette dernière sont présentées de façon progressive afin que vous puissiez comprendre, grâce à de **nombreux ateliers pratiques**, qu'il existe une « autre façon » de penser le développement et de bien ancrer ainsi les principes et les spécificités de la programmation orientée objet (POO).

Au-delà de la rupture disruptive par rapport à l'approche structurée classique, il s'agit tout simplement de savoir concevoir un modèle de classes pertinent pour ses applications en respectant les bonnes pratiques de conception d'une architecture conforme aux standards professionnels. Dans un premier temps vous **manipulerez des classes simples** (commande, article, etc.) et vous prendrez progressivement la mesure de l'impact de la conception des données et des traitements sur la qualité de votre application (performances, sécurité, évolutivité, lisibilité et maintenance) à travers des petits exercices, les. La conception fera également allusion aux Design Patterns afin de vous orienter vers les bonnes pratiques dans vos futurs programmes objets en étant capable d'utiliser des modèles existants. Vous traduisez ensuite votre travail de conception sous la forme d'un diagramme de classes UML afin de disposer d'une langage visuel commun et standardisé. Au-delà de la professionnalisation de la conception cela permet aussi au formateur de limiter les ambiguïtés d'interprétation avec les participants pour la suite de la formation.

Une fois **les fondamentaux acquis (classe, propriétés, méthodes, instance, héritage, composition de classes) nous introduisons l'IA** afin de vous montrer comment elle peut vous aider dans votre processus de conception. Pour cela nous vous montrerons comment écrire des prompts efficaces **afin de vous assister dans vos réflexions**, voir être force de proposition mais tout en vous habituant à **porter un regard critique sur les réponses fournies par l'assistant IA** en « gardant le contrôle ».

Afin de ne pas rester uniquement conceptuelle, la formation vous présente des exemples d'implémentation des concepts objets dans les **principaux langages objets** (Java et Python) ainsi que les apports des environnements de développement (IDE) pour l'utilisation des objets en programmation (**Visual Studio Code, IntelliJ**). Le formateur traduit, en Java par exemple, les travaux de conception objet réalisé jusque-là et dépose cet exemple dans une branche Git. Il demande à l'IA de traduire ce programme en Python et le dépose dans une autre branche afin de

Stage pratique
Qualité du logiciel - Software Craftsmanship

Code :
CPOBJ

Durée :
3 jour(s) (21,00 heures)

Exposés : **40 %**
Cas pratiques : **40 %**
Echanges d'expérience : **20 %**

Inter-entreprises :
Prochaines sessions disponibles sur notre site web.
Tarif : 1 890,00 € HT / participant

Intra-entreprise :
Tarifs et dates sur demande.

montrer les invariants tout en faisant passer un message d'acculturation fort sur l'IA et l'importance fondamentale de la conception par rapport à la syntaxe.

Nous proposons également de mettre en œuvre un **projet simple intégrant l'IA** générative basé sur 3 classes abstraites (service, question, réponse) et qui permet de soumettre un prompt à une IA générative et de récupérer le résultat dans notre application. On fait le lien également avec le concept des classes abstraites vu auparavant car il existe plusieurs sortes de prompt (textuel, sonore, etc.) pour la classe « Question » et différents types de fonction (résumé de texte, génération d'image, traduction, etc.) pour la classe « Service ». La classe « Réponse » elle contient la « chaîne de caractères » de la réponse et d'éventuelles méta-données (version, modèle de l'IA, etc.).

En fil rouge, la formation aborde la conception des structures de données et des traitements dans une approche objet d'une application web proposant un **catalogue d'articles et des commandes associées.** Nous cherchons à l'enrichir tout en intégrant progressivement les possibilités offertes par l'IA sans perdre de vue notre objectif pédagogique fondamental qui est de savoir concevoir une application évolutive grâce aux concepts objets. Nous faisons donc en sorte que notre application puisse facilement s'enrichir de nouveaux services (gestion de stocks, service bancaire paiement, exposition d'API, etc.) en se basant sur des classes représentatives de celles qu'on trouve dans les projets professionnels (par exemple une classe dédiée aux traitements des entrées/sorties pour les bases de données, une classe abstraite pour utiliser différents moteurs d'IA et rendre « intelligente » notre application, etc.) avec un impact minimum sur notre code grâce à nos choix de conception.

OBJECTIFS PEDAGOGIQUES

Objectif opérationnel :

Savoir utiliser ses connaissances en programmation orientée objet pour développer et comprendre comment s'appuyer sur l'IA dans le cadre de ses développements.

Objectifs pédagogiques :

À l'issue de cette **formation Conception et Programmation objet**, vous aurez acquis les connaissances et compétences nécessaires pour :

- Comprendre les principes et les spécificités de la conception Objet
- Passer d'une approche fonctionnelle à une approche Objet
- Savoir concevoir un modèle de classes pour ses applications

- Comprendre l'apport des Frameworks dans une approche Objet
- Découvrir l'impact de l'IA dans la programmation orientée objet
- Mettre en œuvre les concepts objets à travers un programme simple intégrant la programmation orientée objet et l'IA

PUBLIC CIBLE

Ce cours s'adresse aux développeurs ou aux chefs de projet pratiquant la programmation structurée traditionnelle. Le public visé cherche à comprendre l'approche objet et comment mettre en œuvre ses modalités concrètes lors de futurs développements.

Cette formation intéressera également tout développeur (objet ou non) désirant être plus à l'aise dans l'utilisation ou la conception des classes dans un projet applicatif.

PRE-REQUIS

Les participants à cette formation possèdent une première expérience en conception d'applications et en développement logiciel.

Quel que soit le langage utilisé (C, Cobol, Shell, Python, ...), il est important d'avoir déjà développé et livré une application, même de taille modeste, seul ou en équipe.

J'évalue mes connaissances pour vérifier que je dispose des prérequis nécessaires pour profiter pleinement de cette formation en faisant ce test.

METHODE PEDAGOGIQUE
Formation avec apports théoriques, échanges sur les contextes des participants et retours d'expérience pratique des formateurs, complétés de travaux pratiques et de mises en situation.

PROFIL DES INTERVENANTS

Cette formation est dispensée par un·e ou plusieurs consultant·es d'OCTO Technology ou de son réseau de partenaires, expert·es reconnus des sujets traités.

Le processus de sélection de nos formateurs et formatrices est exigeant et repose sur une évaluation rigoureuse leurs capacités techniques, de leur expérience professionnelle et de leurs compétences pédagogiques.

MODALITÉS D'ÉVALUATION ET FORMALISATION À L'ISSUE DE LA

FORMATION
L'évaluation des acquis se fait tout au long de la session au travers des ateliers et des mises en pratique. Afin de valider les compétences acquises lors de la formation, un formulaire d'auto-positionnement est envoyé en amont et en aval de celle-ci. Une évaluation à chaud est également effectuée en fin de session pour mesurer la satisfaction des stagiaires et un certificat de réalisation leur est adressé individuellement.

PROGRAMME PEDAGOGIQUE DETAILLE

Jour 1

DE LA PROGRAMMATION FONCTIONNELLE VERS LA PROGRAMMATION OBJET

- Bien comprendre les limites de la programmation dite « structurée »
- Les objectifs du monde Objet
- En quoi l'approche objet facilite-t-elle les ateliers de développement logiciels ?
- Que retrouve-t-on de la programmation structurée dans la programmation objet ?

Travaux pratiques

Objectif : Comprendre le fonctionnement de la programmation orientée objet

Descriptif : Sur la base d'une application classique (commandes, articles), le formateur présente les fonctionnalités de l'application conçue dans une approche fonctionnelle (non objet). Les participants sont amenés à répondre à certaines questions concernant la lisibilité des traitements, des variables, et sur l'évolution du logiciel. Les limites de la programmation classique sont volontairement amplifiées afin d'être clairement identifiées.

APPRENDRE À CONCEVOIR DES CLASSES D'OBJET

- Définir un comportement commun au sein d'une classe
- Comment « reconnaître » une classe ?
- Savoir faire un effort d'abstraction et rester conceptuel
- Liens entre classe et objets (instance)
- Exemples de classes « mal conçues » qui complexifient le développement
- Savoir définir un dictionnaire de données et les attributs d'une classe

- Qu'entend-on par « encapsuler » les données ?
- Pourquoi utiliser des accesseurs ?
- L'origine et la fin d'un objet (constructeurs, destructeurs)
- Savoir définir les traitements (méthodes) d'une classe
- Liens entre états d'un objet et méthodes
- Ne pas mélanger la logique métier et la logique technique
- Exemple d'une classe dédiée aux I/O (Data Access Object)
- Bonnes pratiques de nommage
- Utilisation de classes existantes

Travaux pratiques

Objectif : Être capable de créer des classes d'objet métiers pour une application et comprendre l'intérêt de la séparation de la logique métier avec la technique (sauvegarde des articles et des commandes par exemple)

Descriptif : « Reconception » objet de l'application précédente (commandes, articles) conçue sur un modèle fonctionnel. Élaboration du dictionnaire de données conception des classes, attributs (dictionnaire de données ventilé dans les attributs), traitements (extraction des verbes d'action, des états d'une commande, etc.). Création d'une classe ArticleDAO implémentant les méthodes de sauvegarde et de chargement des articles.

Jour 2

LIENS ENTRE LES CLASSES ET LES OBJETS (HÉRITAGE, VISIBILITÉ, COMPOSITION, ASSOCIATION)

- Visibilité des attributs entre les objets (publique, protégé, privé)
- Visibilité des méthodes entre les objets
- La puissance de l'héritage
- Réflexions sur l'héritage multiple
- Ne pas confondre composition et héritage
- Intérêt du polymorphisme
- La surcharge des opérateurs
- Appels de méthode d'une autre classe (message)

Travaux pratiques

Objectif : Distinguer héritage, composition, association

Descriptif : Création d'une classe de commande en ligne (spécialisation de commande pour montrer l'héritage), de composition d'articles au sein

d'une classe « Ligne de commande », et d'association sous la forme d'articles qui ne font pas partie de la commande (au catalogue ou en stock par exemple).

INTERFACES ET ABSTRACTIONS

- Notions de classe virtuelle ou abstraite (factorisation de comportements)
- Utilisation par héritage
- Imposer un contrat à des classes (interface)
- Différences entre classe abstraite et interface

Travaux pratiques

Objectif : Comprendre l'intérêt d'un niveau d'abstraction supplémentaire pour l'évolutivité de notre application fil rouge (commandes, articles)
Descriptif : Amélioration du modèle précédent avec l'utilisation de classes abstraites. On enrichit l'application fil rouge avec le paiement en ligne (classe abstraite pour intégrer différents mécanismes de paiement bancaire « classique », Paypal, bitcoin, etc. sans déstructurer notre conception précédente.

INTRODUCTION À LA MODÉLISATION OBJET AVEC UML

- UML (Unified Modeling Language) ne représente pas une démarche mais un formalisme
- Les deux types de vues (statiques, dynamiques)
- Tour d'horizon (rapide) des différents diagrammes UML
- L'assistance proposée par l'IA pour valider la modélisation (classes prédictives)

Travaux pratiques

Objectif : Comprendre l'apport de l'UML pour l'analyse et la documentation

Descriptif : Modélisation de notre application fil rouge avec UML, Soumission de notre modèle de classe à une IA pour comparaisons des résultats.

Jour 3

PRÉSENTATION DES DESIGN PATTERNS

- Principes des solutions de conception cataloguées
- Méthodologie : définition des besoins techniques, des classes "types" du pattern, des collaborations entre classes
- Présentation des patrons de conception : origine, les 3 familles (création, structuration et comportement), autres patrons
- Présentation des principaux patrons de conception de chaque catégorie
- Documentation d'un patron de conception et présentation des différents diagrammes UML utilisés

Quiz : On présente une problématique et il faut choisir le bon modèle (patron ou Pattern en anglais)

INTRODUCTION À LA MISE EN OEUVRE DES CONCEPTS OBJETS (JAVA OU PYTHON)

- Présentation rapide des IDE Visual Studio et IntelliJ
- Tour d'horizon des plugins standard disponibles
- Les apports des assistants IA comme GitHub Copilot (autocompletion intelligente, génération de fonction, suggestion de code, etc.)

Travaux pratiques

Objectif : Mise en œuvre des concepts au sein du projet fil rouge (commande, article).

Descriptif : Nous implémentons les classes de notre application précédente en Java ou Python selon l'environnement préféré par les participants. Il s'agit d'une simple prise en main et le code est volontairement très simple car on ne prévoit pas de détailler la syntaxe de ces langages.

On commente un exemple de code accédant à une base de données (connexion, chargement de données) pour comprendre comment les classes que nous avons conçues « fonctionnent » dans une « vraie » application simplifiée.

IMPACT DE L'IA DANS LA PROGRAMMATION OBJET

- Briques techniques de l'IA (machine learning, deep learning, modèle, entraînement, etc.)
- Écrire des prompts (IA générative) efficaces pour le développeur
- Comment la complexité des bibliothèques d'IA (LLM, réseau de

neurones, etc.) est encapsulée dans des bibliothèques Java, Python, C++, etc. ?
- Intégration d'un composant IA dans un projet objet métier (classes intelligentes qui prennent des décisions)
- Familles de classes autour de l'IA (ModeleIA, ClassifieurIA, RegresseurIA, ClusteringIA)

Travaux pratiques

Objectif : Comprendre qu'une IA peut être vue comme un objet (ou plusieurs) « facile » à utiliser pour le développeur sans compétence technique en IA (Machine Learning, Data Science, etc.). Ce TP constitue également une excellente synthèse de tous les concepts objets vus dans la formation notamment au niveau de l'impact minimum dans le code si l'on désire intégrer de nouveaux services à notre application.

Descriptif : L'atelier consiste à développer « from scratch » une application proposant un prompt capable d'interagir avec différents modèles d''IA et récupérer les résultats (traduction, image, réusmé de texte, etc.) dans notre application.Pour cela nous prévoyons d'utiliser 3 classes abstraites « QUESTION » (attributs prompt, instructions, fichiers,etc. permettant de définir la question et son contexte, « REPONSE »proposant les attributs de la réponse (texte ou autre format selon la question, méta données éventuelles comme le nom du modèle d'IA, sa verison, etc. et « SERVICE » qui est la classe qui spécifie le service demandé (traduction, résumé, génération d'image, etc.). On voit bien ici que notre programme est extensible et capable d'intégrer de nouveaux services sans modification de code

Objectif : Consolider l'intégration de l'IA dans sa conception

Descriptif : L'atelier consiste à faire évoluer notre projet fil rouge en intégrant de l'IA avec l'utilisation d'une classe intelligente qui prend des décisions comme recommander un article, prédire une rupture de stock, détecter une anomalie, etc. On met à jour le diagramme UML de notre projet avec des éléments comme IAProcessor ou DecisionEngine

Accessibilité
L'inclusion est sujet important pour OCTO Academy.
Nos référent·es sont à votre disposition pour faciliter l'adaptation de votre formation à vos besoins spécifiques.
Pour les contacter : academy.accessibilite@octo.com