Je vais, par l’intermédiaire de cette vidéo, essayer de vous expliquer ce qu’est la programmation orientée objet

**Définition :** dans le langage informatique, la Programmation Orientée Objet (POO) consiste en la mise en relation d’objets, c’est-à-dire d’éléments de programmation, avec un langage spécifique qui leur permet de communiquer entre eux.

Les langages de programmation orientée objet apportent 4 aspects primordiaux par rapport aux autres types de langage.

ENCAPSULATION – ABSTRACTION – HERITAGE – POLYMORPHISME

Cités de la sorte, ces termes peuvent paraître plus ou moins récalcitrants.

Essayons de les expliquer en prenant l’exemple d’un objet de la vie de tous les jours, que tout le monde connaît : une maison !

L’encapsulation permet de réunir des variables et des fonctions au sein d’une même entité appelée une classe. Une classe c’est quoi ? Dans le cas de la maison, on pourrait la comparer au plan qui sert à la construire. AFFICHER CLASSE

Un objet est une version de la classe, plus spécifique. AFFICHER OBJET

L’accès aux attributs, qui sont, au final, les variables qui composent l’objet, et aux méthodes, qui sont les fonctions, peut être réglementé : publique, privé ou encore protégé. Par exemple, les attributs ne peuvent pas être changés par un autre objet si ce n’est par l’intermédiaire de méthodes publiques de la classe maison comme la méthode changerCouleur() de notre classe. On pourrait avoir un objet “Peintre” qui pourrait alors utiliser la méthode changerCouleur pour repeindre la maison, et passer la maisonClientDupont du rouge au vert par exemple.

L’abstraction est le fait de pouvoir utiliser des méthodes fournies par une classe, sans vraiment savoir comment tout fonctionne derrière. Dans notre maison, si j’appuie sur un interrupteur, la lumière s’allume sans que j’aie besoin de savoir comment tout est construit derrière le mur.

L’héritage, qu’est-ce que c’est ? Eh bien, comme son nom l’indique, une classe « enfant » de notre classe maison va naturellement hériter des attributs et méthodes de sa classe « parent » (maison, donc !). AFFICHER HERITAGE

Comme on peut le voir sur l’exemple, on peut créer une nouvelle classe, enfant de la classe maison, à laquelle on ajoute une spécificité : un garage. On peut définir ce garage en fonction du nombre de véhicule que l’on souhaite pouvoir garer à l’intérieur par exemple. La classe maisonGarage hérite donc de tous les attributs (couleur, toiture, …) et des méthodes (changeCouleur, appuiInterrupteur) de la classe maison, et est donc plus spécifique.

Le polymorphisme est un peu plus compliqué à expliquer. Ce mot vient du grec « qui peut avoir plusieurs formes ». En fait, si on reprend la méthode appuiInterrupteur de notre classe maison, le polymorphisme consiste en le fait que cette fonction puisse être utilisée dans n’importe quel cas. Par exemple, un interrupteur peut allumer une lumière, ouvrir un volet, déployer un store, ou même, dans notre classe MaisonGarage, ouvrir la porte de garage. Cette seule méthode peut donc être appliquée à plusieurs usages différents.