Deepeyes : Implémentation de Vision   
  
Plongez au cœur du projet "Deepeyes" et explorez la fusion captivante entre l'art visuel et la technologie de pointe. Découvrez comment l'implémentation réfléchie de l'API "Vision" d'Azure a donné vie à une expérience d'analyse d'images exceptionnelle. De la détection d'objets à la transcription de texte, en passant par l'identification des couleurs principales, cet article vous guide à travers les choix stratégiques qui sous-tendent chaque aspect de "Deepeyes". Rejoignez-nous pour un voyage au sein de la transformation d'images en récits informatifs, enrichissants et esthétiques grâce à l'intelligence artificielle.

## Vision, c’est quoi ?

Ce service est proposé par Microsoft Azure et permet d'intégrer des fonctionnalités d'analyse d'images et de vision par ordinateur dans des applications, des sites web et d'autres solutions logicielles. L'API "Vision" d'Azure offre un large éventail de fonctionnalités, telles que la détection d'objets, la reconnaissance de texte, la classification d'images, la génération de miniatures et bien plus encore. Elle permet aux développeurs de créer des applications intelligentes qui peuvent interpréter et extraire des informations utiles à partir d'images et de vidéos, ouvrant ainsi la voie à des applications innovantes dans divers domaines tels que la sécurité, l'e-commerce, la santé, l'automatisation industrielle, et bien d'autres.  
  
La raison d’être de Deepeyes est de démontrer les capacités de ce service afin de permettre aux utilisateurs mieux comprendre son utilisation et ce qu’il peut accomplir.

### Un projet divisé par les versions.

A l’origine, Deepeyes utilisait la version 3.2 de Vision. Il s’agissait d’une version très complète, avec déjà beaucoup de fonctionnalité.   
Pendant le développement, Microsoft a sorti une preview de la version 4.0, qui avait ses avantages (de nouvelles fonctionnalité intéressantes) et ses inconvénient (la suppression d’autre fonctionnalité). Afin de garder le travail accomplis et toujours dans un esprit de démonstration, nous avons fait le choix d’utiliser les 2 versions.

Cette implémentation est possible grâce au fait que les packages utiliser pour les 2 versions sont des packages différents. On évite ainsi tout conflit entre les 2 versions au sein d’un même projet !

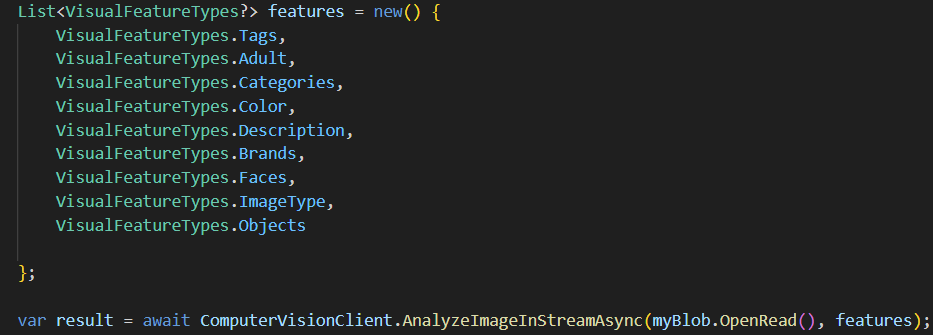
## L'implémentation

### Azure Ai Vision 3.2

Pour cette version, les fonctionnalités ayant été exploité sont les suivantes :

* Tags : ce qui permet de récuperer des tags d’une image. Ils décrivent brièvement ce que contient l’image.
* Adult : permet de savoir si une image contient du contenue pour adulte.
* Categories : catégorise l’image.
* Color : renvoie la couleur dominante d’une image
* Description : créer une phrase complète qui décris l’image dans son ensemble.
* Brands : détecte la présence de marque dans une image.
* Faces : Si des visages sont détectés sur l’image, renvoie une description de ces derniers
* ImageType : renvoie le type de l’image (dessin, clipart,..)
* Objects : détecte la présence d’objets dans l’image.

L’implémentations est relativement simple. Après avoir créé une instance de la classe “ComputerVisionClient”, on lui passe simplement la liste des fonctionnalités souhaité ainsi que l’image à analyser, et on récupère la totalité des informations en format JSON !



### Azure AI Vision 4.0 (preview)

Pour cette version, nous utilisons qu’une seule fonctionnalité : DenseCaptions.   
Cette fonctionnalité ressemble à la détection d’objet de la version précédente mais permet, en plus, de récupérer les coordonnées de ces derniers sur l’image. On peut donc voir ce que l’intelligence artificielle interprète !

L'implémentation est très proche de la version précédente mais ici, une chose est importante : Le client est maintenant un “VisionServiceOptions”, ce n’est donc plus lui qui contient les méthodes a appelé. On va plutôt créer un “ImageAnalyzer” qui en plus de recevoir l’image est les fonctionnalités, aura également besoin des options permettant la connexion au service.

