

Compte rendu de l'application IOS

Daltonipette

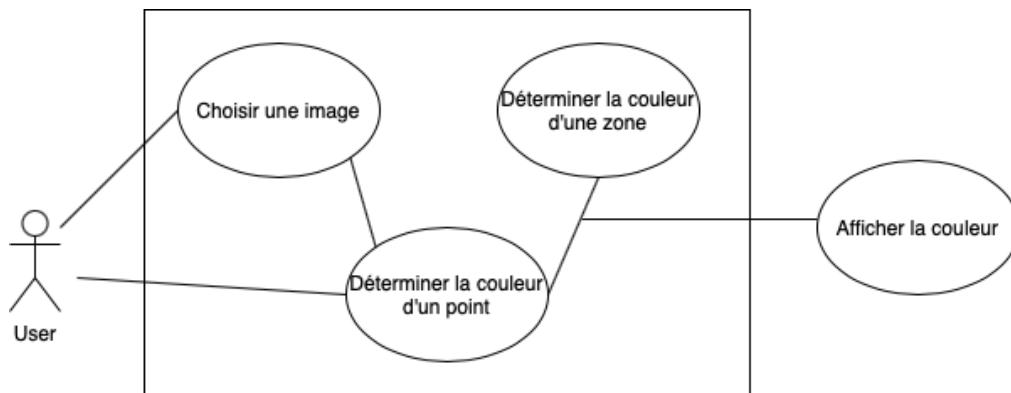
Sommaire

I.	Principe de l'application et fonctionnement	2
II.	Utilisation des fonctionnalités de l'appareil	4
III.	Respect du cahier des charges	4
IV.	Les difficultés rencontrées	6
V.	Ce que nous avons appris	6
VI.	Conclusion	6

I. Principe de l'application et fonctionnement

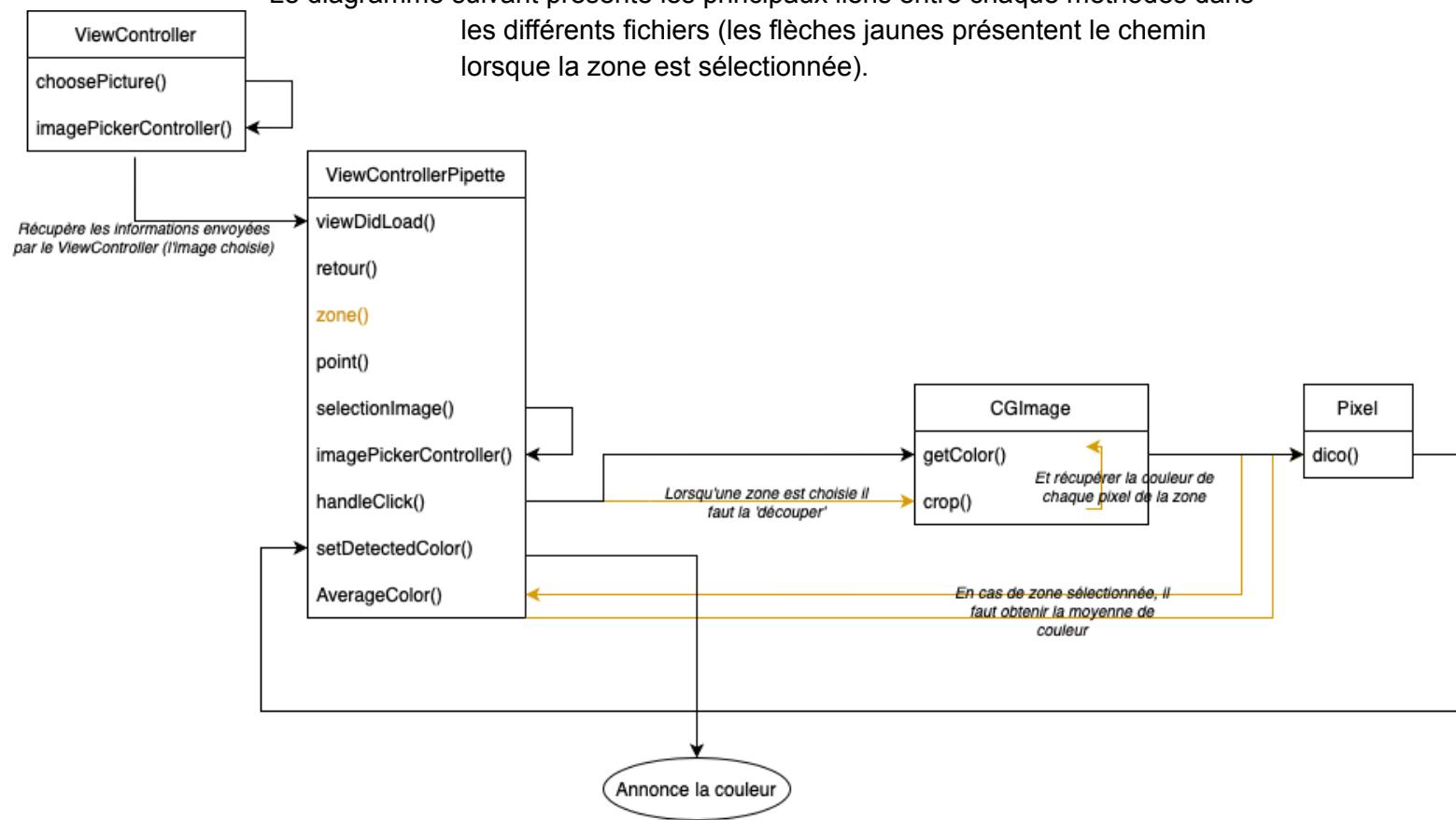
Dans le cadre du cours de programmation IOS nous devions développer une application mobile. L'idée Daltopipette nous est venue avec l'idée de découvrir les possibilités et fonctionnalités de l'appareil. Ainsi, nous avons choisi de développer cette application mobile destinée aux personnes ayant des déficiences visuelles, pour permettre de différencier les couleurs.

Le principe étant, grâce à une image prise sur la galerie, l'utilisateur peut sélectionner un point ou une zone dont il veut déterminer la couleur, le nom de celle-ci est donné par le haut-parleur et s'affiche en haut de l'écran.



Notre projet contient donc différents fichiers, dont l'utilité et les fonctionnalités principales sont détaillées ci-dessous, cela vient en complément de notre code pour plus de compréhension.

Le diagramme suivant présente les principaux liens entre chaque méthodes dans les différents fichiers (les flèches jaunes présentent le chemin lorsque la zone est sélectionnée).



ViewController.swift (Accueil)

Ce fichier définit le cycle de vie de la page d'accueil. Sa principale fonction est de permettre à l'utilisateur d'accéder à la galerie de son téléphone pour sélectionner l'image où il veut connaître une couleur. La méthode choosePicture permet donc l'accès à la galerie tandis que imagePickerController récupère l'image sélectionnée et passe à la page pipette.

ViewControllerPipette.swift

Ce fichier définit le cycle de vie de la seconde page de notre application, la page dite "pipette". Cette page récupère l'image sélectionnée précédemment et l'affiche au milieu de l'écran, grâce à la fonction selectionImage (qui est identique à choosePicture de Viewcontroller) et à la méthode imagePickerController.

Quatre boutons sont présents sur la page, le bouton retour, le bouton zone, le bouton point et le bouton selectionImage :

Le bouton retour renvoie à la page d'accueil, le bouton selectionImage permet de sélectionner une nouvelle image dans la galerie.

Les boutons point et zone, eux, permettent d'accéder à la fonctionnalité de l'application. En appuyant sur l'image le bouton point activé, l'application indique la couleur à l'endroit sur l'image, en l'affichant directement en haut de l'écran et via le haut-parleur. Pour le bouton zone, il faut appuyer à deux endroits différents sur l'image, l'application se chargera de former un rectangle entre ces deux points et renvoyer la couleur moyenne de ce rectangle à l'utilisateur.

CGImage.swift

Ce fichier contient une extension de CGImage qui permet donc de récupérer les couleurs de l'image. Les fonctions getColorAt retournent un dictionnaire de couleurs contenant les taux de chaque couleur primaire.

Pixel.swift

Permet de définir une couleur pour un dictionnaire de couleur donné. La fonction dico permet de retourner le nom de la couleur donnée lors de l'initialisation de l'objet.

II. Utilisation des fonctionnalités de l'appareil

Pour ce projet nous avons donc utilisé plusieurs fonctionnalités de l'appareil ; l'accès à la galerie pour récupérer l'image qui va être traitée par l'application. Et également, le haut-parleur via la méthode text To Speech qui va donc lire le texte retourné par la méthode dico de la classe Pixel.

III. Respect du cahier des charges

Nous constatons certaines différences entre les patrons de nos deux vues présentées dans le cahier des charges et nos vues effectives. En effet, hormis les quelques différences d'ordre esthétique et les ajouts de fonctionnalités comme la possibilité de changer d'image depuis la même page, l'application est relativement similaire à ce que l'on avait prévu. L'utilisation du haut-parleur évoquée comme amélioration possible dans le cahier des charges a même pu être rajoutée.

	Maquette du cahier des charges	Capture d'écran de l'application fonctionnelle
Page d'accueil	<p>Daltopipette</p> <p>Sélectionner une image dans la galerie</p> <p>Daltopipette est une application conçue pour l'iphone permettant aux daltoniens et aux déficients visuels de reconnaître les couleurs. En choisissant une image dans votre galerie vous serez capable de déterminer la couleur d'un point ou d'une zone.</p>	<p>Carrier 4:18 PM</p> <p>Daltopipette</p> <p>Sélectionner une image dans la galerie</p> <p>Daltopipette est une application conçue pour l'iphone permettant aux daltoniens et aux déficients visuels de reconnaître les couleurs. En choisissant une image dans votre galerie vous serez capable de déterminer la couleur d'un point ou d'une zone.</p>
Page pipette	<p>jaune</p>  <p>undo icon, crop icon, edit icon</p>	<p>vert kaki</p>  <p>undo icon, crop icon, edit icon, image icon</p>

Nous pouvons donc considérer que le cahier des charges a été respecté.

IV. Les difficultés rencontrées

La majeure difficulté rencontrée était vers la fin du projet. Lorsque l'utilisateur clique il était censé pouvoir lire la couleur de là où avait cliqué en haut de l'écran mais la couleur était mauvaise. On a compris que le problème venait du ratio de la taille de l'image par rapport à celle de l'écran et de son centrage qui n'était pas pris en compte. Ce bug n'a d'ailleurs toujours pas été résolu

Vers le début du projet nous avons eu aussi quelques soucis avec la disposition des widgets sur l'écran pour rendre le tout responsive.

Nous avons aussi eu des difficultés pour récupérer une image depuis la galerie.

V. Ce que nous avons appris

Durant ce cours nous avons appris à coder en Swift et à créer des applications pour Iphone sur Xcode. Nous avons aussi appris plein de noms de couleur.

Nous avons également appris à faire parler le téléphone avec l'accent français, à faire naviguer un utilisateur vers diverses pages, à récupérer une image dans la galerie et récupérer un clic à l'écran et à rendre une application un peu responsive.

VI. Conclusion

Le développement est terminé même s'il persiste quelques bugs comme le fait que la couleur que l'application détecte est mauvaise, si l'image est trop grande alors l'application plantera pour des raisons inconnues...

La vidéo de présentation ne dispose pas du son mais comme vous avez pu l'entendre au cours des derniers cours il fonctionne très bien ;).

Nous avons beaucoup appris et nous sommes maintenant capables de développer des applications pour Iphone.