En langage Swift, différents types de classes sont disponibles pour représenter des éléments à l'écran dans le développement de l'interface utilisateur. Par exemple, l'objet TabView{} permet de créer des boutons situés en bas de l'écran. Ces boutons peuvent être associés à des fonctions et des icônes, permettant ainsi de créer un menu de navigation (appelé Hub dans les chapitres précédents). Cependant, une vue ou un affichage en Swift n'est pas en soi un objet. Lorsque j'ai essayé de coder ces vues comme des objets, j'ai constaté que Swift ne proposait pas de mise à jour en temps réel. Les vues étaient calculées à l'avance et ne pouvaient pas être actualisées ultérieurement.

J'ai donc décidé de changer de paradigme et de passer à une approche plus procédurale. En utilisant des variables globales plutôt que des objets, et en adoptant un style de codage plus procédural, j'ai pu rendre mes vues plus dynamiques.

Par exemple voici un exemple de ce qu'on pourrait apparenter à un objet :

TabView{

ViewChrono(record:record)

.tabItem {

Image(systemName:"timer")

Text("Chrono")

}

ViewSync(record:record)

.tabItem{

Image(systemName:"icloud")

Text("Sync")

}

ViewRecord(records:record)

.tabItem{

Image(systemName:"folder")

Text("Record")

}

ViewHelp()

.tabItem{

Image(systemName:"questionmark.circle")

Text("Help")

}

}

Cet exemple illustre l'objet "TabView{}" qui m'a permit de réaliser le menu de navigation de l'application. On peut remarquer que chaque élément à l'intérieur correspond à un bouton que l'utilisateur peut interagir afin de naviguer entre les différentes vues. (Dans cet exemple respectivement les vues : Chrono, Synchronisation, Enregistrement et aide).