ECOLE POLYTECHNIQUE DE BRUXELLES

INFO-H-2001

Nom du cours

Projet informatique : Black Jack

MEMON Salman

MOLI David

REKIEK Saad

BERSINI Hugues

2022

Table des matières

1 Introduction

Dans le cadre du cours d'informatique, il était demandé de réaliser un jeu android à l'aide du language de programmation *Kotlin* et de l'environnement de développement *Android Studio*. Le choix du jeu à réaliser était libre. Le choix s'est porté sur un *Black Jack*. Le rapport se divise en 3 parties. La première partie traitera du jeu qui a été codé, les différentes règles relatives à ce dernier et comment y jouer. La deuxième partie reviendra sur la description du code kotlin et de sa création. Finalement, la dernière partie sera dédiée aux codes UML (diagrammes de classes et séquences).

2

Le *Black Jack*

Le *Black Jack* est un jeu de cartes qui se joue individuellement contre un croupier. Chaque joueur [1] dispose d'une main de n cartes (avec $n \geq 2$) . Chaque carte possède une valeur prédéfinie [2] comprise entre 1 et 11. Le score de chaque joueur est défini en sommant les valeurs des cartes de sa main (autrement dit : $Score = \sum_{j=1}^{n} ValCarte_j$). L'objectif est d'obtenir un score supérieur à celui du croupier tout en ne dépassant pas 21. La situation où le score atteint est de 21 est appelée un *Blakc Jack* [3].

Les différentes options lors d'une partie sont les suivantes :

1. Piocher : Consiste à piocher une carte
2. Passer : Consiste à arrêter de jouer afin de dévoiler les cartes
3. Doubler : Consiste à piocher une carte et doubler sa mise de manière simultanée

Ces choix sont totalement indépendants de ceux du croupier.

Les valeurs des cartes sont les suivantes :

— de 2 à 9 : valeur nominale de la carte

— de 10 au roi : 10 points

— As : 1 ou 11 points

NB : Ce jeu étant très populaire dans les casino, il y a aussi possiblité de jouer de l'argent.

1. Incluant le croupier
2. Cas particulier pour l'As qui peut avoir une valeur de 1 ou 11 selon le cas le plus avantageux pour le joueur
3. Un *Black Jack* amène obligatoirement à une victoire ou une égalité

3 Description du code

Le code qui a été créé se découpe en deux parties principales, le code *XML* (relatif à l'interface graphique du jeu) et le code *Kotlin* étant relatif à la programmation du jeu. Le code *XML* ne sera traité que succinctement. La présentation d'illustrations des différentes interfaces du jeu a été privilégiée.

3.1 Code *XML*

L'interface graphique du jeu se découpe en 3 parties : La page d'accueil, l'interface de jeu et les règles du jeu.

La page d'accueil a pour but de regrouper les données du joueur (Prénom, Sexe) ainsi que de lui introduire le jeu à l'aide d'une image interactive indiquant les règles du jeu lorsqu'elle est cliquée. L'interface de début contient un EditText afin que le joueur y insère son prénom. Il y a également un RadioGroup qui permet au joueur de se définir comme un homme ou une femme. Un bouton en bas à droite permet au joueur d'accéder aux règles du jeu avant de commencer celui-ci. Finalement, le bouton commencer envoie le joueur dans le jeu (voir interface de début 3.1).Pour l'interface règles, les différentes règles relatives du jeu y sont inscrites grâce à un TextView (voir interface règles 3.2). La partie se déroule dans l'interface de jeu, celle-ci contient tous les boutons interactifs afin d'y jouer, elle contient aussi une jauge afin de miser une somme comprise entre 0 et 1000 au commencement. Les différentes options de jeu sont celles évoquées au chapitre (**Le *Black Jack***). L'option piocher s'active lorsque l'on clique sur l'ImageView de la carte blanche que tient le croupier sur la figure ci-dessous (voir interface jeu 3.3.

(**NB : Le joueur débute la partie avec un montant prédéfinit de 1000$ et une somme minimale de 5$ doit être placée afin de débuter la partie**)

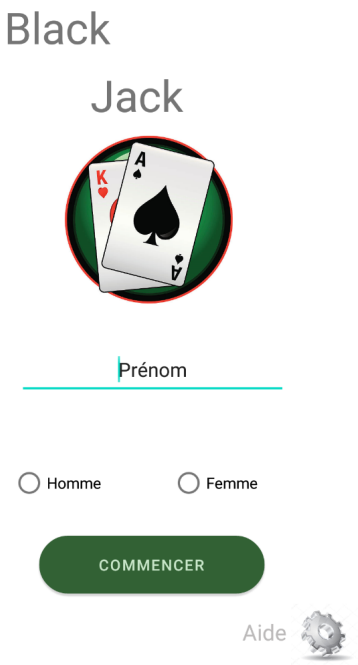

FIGURE 3.1 – Interface de début

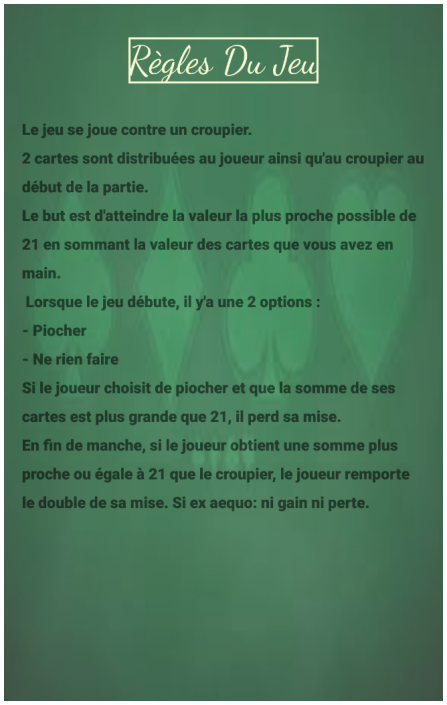

FIGURE 3.2 – Interface relative aux règles du jeu

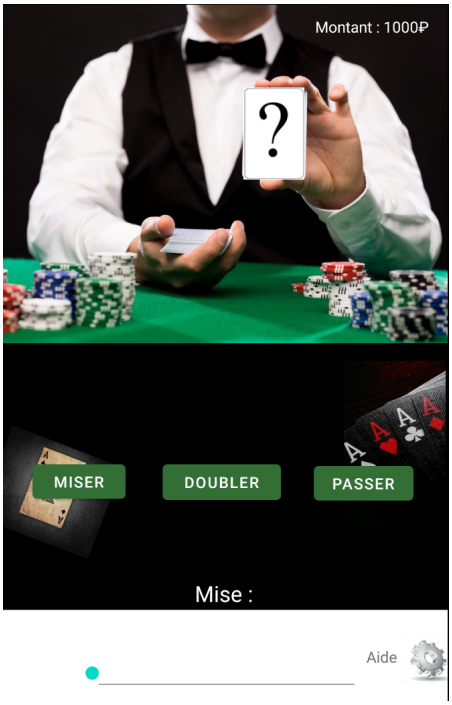

FIGURE 3.3 – Interface de jeu

3.2 Code *Kotlin*

Le code Kotlin a, lui aussi, été divisé en plusieurs parties. Nous avions d'une part les codes relatifs aux interfaces *XML* et d'autre part les codes relatifs à nos classes. Ces derniers ont permis respectivement de rendre capable l'interaction du joueur avec l'interface graphique et de modéliser le jeu.
Les principaux objets et personnages du *Black Jack* sont :

— Les acteurs : Joueur et Croupier

— Le deck : Lot de 52 cartes

— Les cartes

Une classe relative à chaque objet a alors été crée (Personnages, Joueur, Croupier, Deck, Cartes). En plus de celles-ci, la classe hand a été rajoutée incluant la main de chaque personnage afin de centraliser le grand nombre de méthodes relatives à cet objet. Les personnages se sont vus assigner un certain nombre de méthodes (Piocher, Doubler, ...). De cette façon, l'objet *Joueur* peut être capable de piocher par exemple. Il faut tenir compte de l'obligation de la mise, sans cette étape, l'utilisateur ne peut donc rien faire et un toast apparaît en signalant au joueur de miser d'abord (3.4).

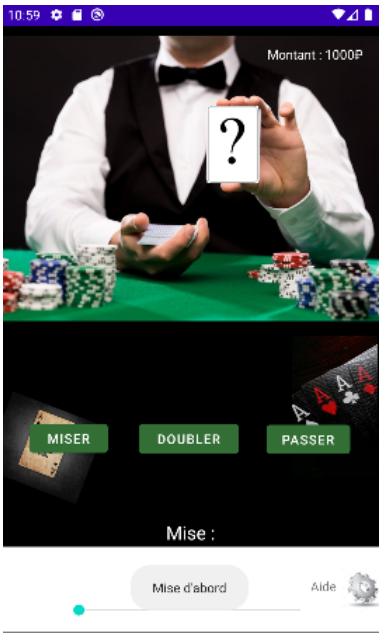

FIGURE 3.4 – Interface de jeu lorsque l'utilisateur essaye de piocher, doubler ou passer avant d'avoir miser

Les interfaces dans lesquelles les objets évoluaient et interragisseaient se sont aussi vu attribuées des méthodes. Ainsi, l'interface *Jeu* intègre la méthode gameOver afin de déterminer si le

jeu était fini ou non par exemple. Les différents objets et attributs de ces classes seront d'avantage détaillées dans la chapitre (***UML***).

Le principe général du code est le suivant, les informations (attributs) des personnages sont stockées dans l'interface *MainActivity* dans laquelle le joueur peut renseigner son nom par exemple. Les personnages sont alors créés à ce même emplacement en se voyant instancier ces attributs. Se lance par la suite l'interface *Jeu* dans laquelle sont récupérées les données de nos personnages. Le joueur doit ensuite, en premier lieu, définir sa mise. Après cela, le jeu se lance immédiatement. Selon les choix du joueur, différentes méthodes seront appelées jusqu'à ce que la fonction gameOver se lance. Auquel cas, le jeu s'arrête et la partie est terminée. Lors de cette dernière étape, plusieurs scénarios sont possibles, la victoire du joueur, la défaite du joueur ou égalité.

FIGURE 3.5 – Interface de jeu lors d'une défaite de l'utilisateur

FIGURE 3.6 – Interface de jeu lors d'une victoire de l'utilisateur

FIGURE 3.7 – Interface de jeu lors d'une égalité de l'utilisateur avec croupier

4 UML

4.1 Diagramme de classes

Le diagramme de classe permet de bien visualiser la hiérarchie du code. En effet, différents liens présents entre les classes relatives au code sont notifiables. Ces liens sont, l'héritage, les compositions, les associations et les aggrégations. Il y'a, en premier lieu, la classe Personnages de laquelle héritent deux classes : Joueur et Croupier. De plus, la classe Joueur et la classe Croupier sont liés entre eux par une relation d'association. Ensuite, ces 2 classes ont un lien d'aggrégation avec la classe Hand (chaque personnage ayant une main) ayant elle même un lien d'aggrégation avec la classe carte (Etant donné qu'une main est composée de n cartes, $n = 0, 1, ... \infty$). De plus, il y'a également un lien de composition entre la classe Joueur et la classe Jeu ainsi qu'entre la classe Croupier et la classe Jeu. Ces liens s'expliquent par le fait que si le joueur et/ou le croupier sont absents, il n'y a pas de jeu, de partie. Il en est de même pour la classe Deck.

4.2 Diagramme de séquence

Le diagramme de séquence revient sur les différentes classes et objets créés et les différentes interactions entre ceux-ci. L'acteur principal est donc le joueur qui est positionné tout à gauche. Plusieurs interactions se feront régulièrement entre le joueur et le jeu. Tout ce qui est en rapport avec le jeu interagit logiquement avec les mains du joueur et croupier. Ensuite il y a des conditions quant au choix du joueur dans le jeu, selon son choix le programme partira une direction différente. Puis une boucle pour le croupier est également prévue afin d'automatiser son tour. Finalement, vient une dernière boucle qui vérifie la valeur de la carte as (1 ou 11 voir 2) puis calcule les scores des deux joueurs dans le jeu afin de retourner au joueur le résultat du match.

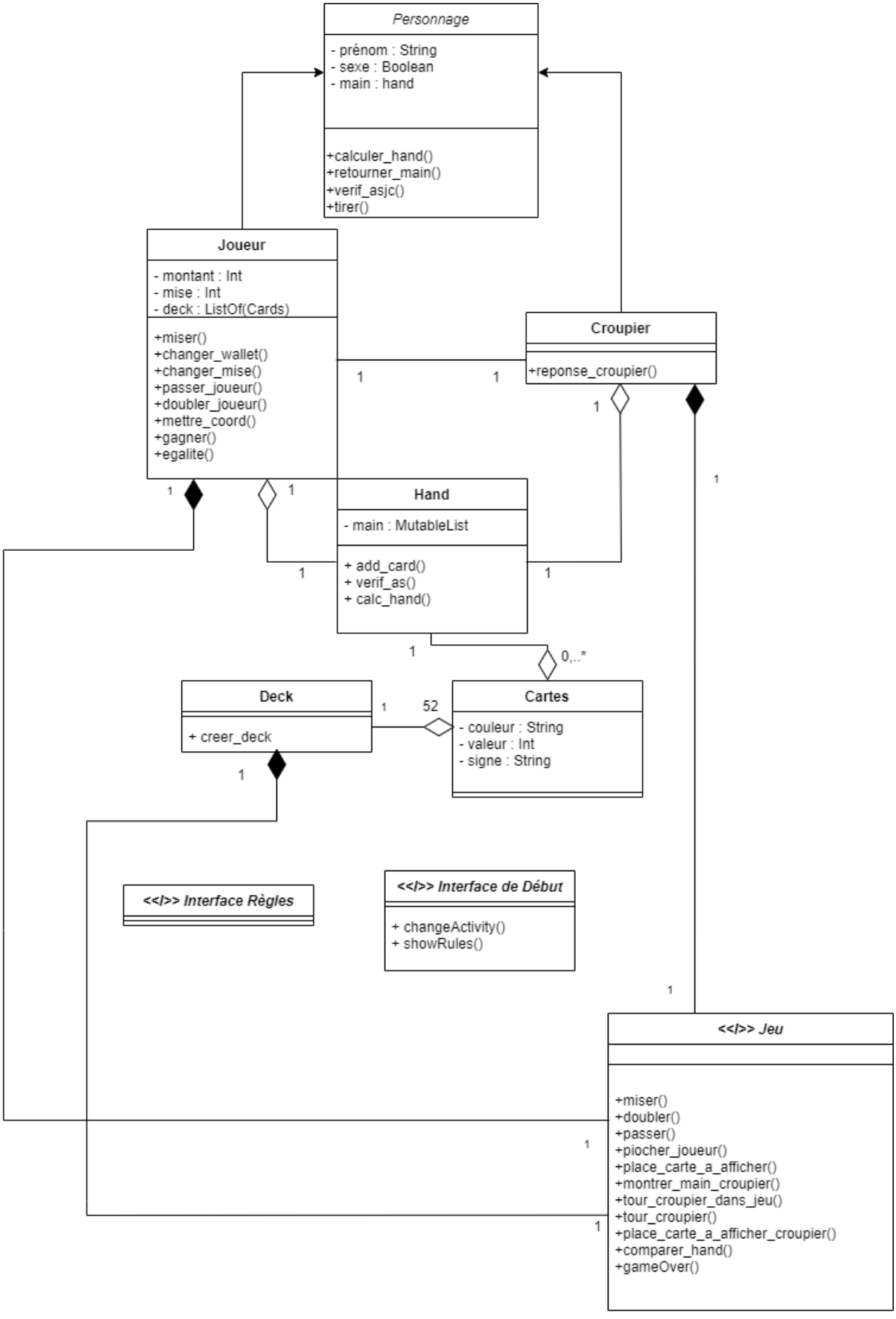

FIGURE 4.1 – Diagramme de classes du Black Jack

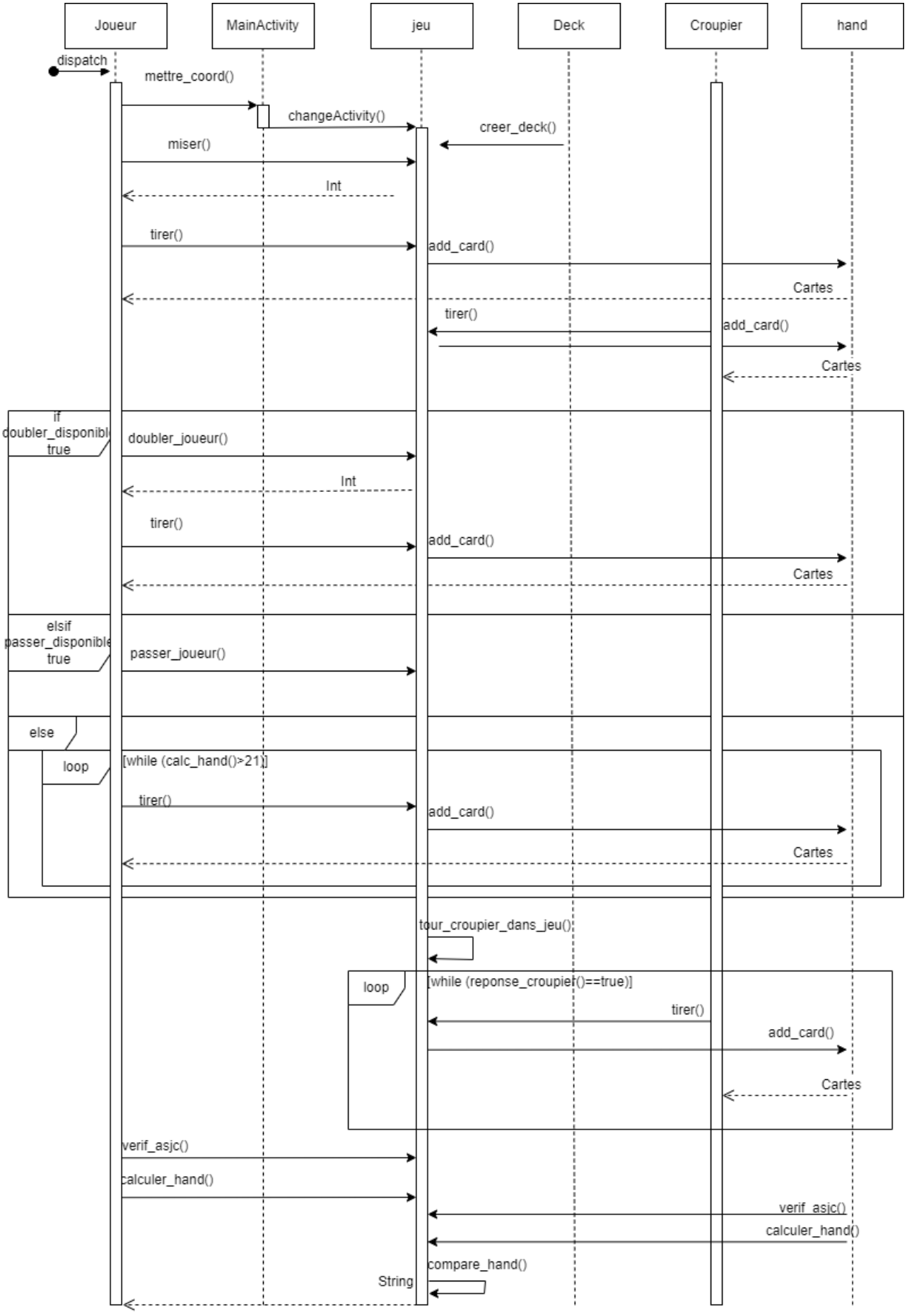

FIGURE 4.2 – Diagramme de séquence du Black Jack