



## Manuel développeur

**ELUECQUE Anthony**

**GINIAUX Anatole**

**LABIT Evan**

**DOURNEL Frédéric**

**BUT2 INFO**

**IUT de Calais**





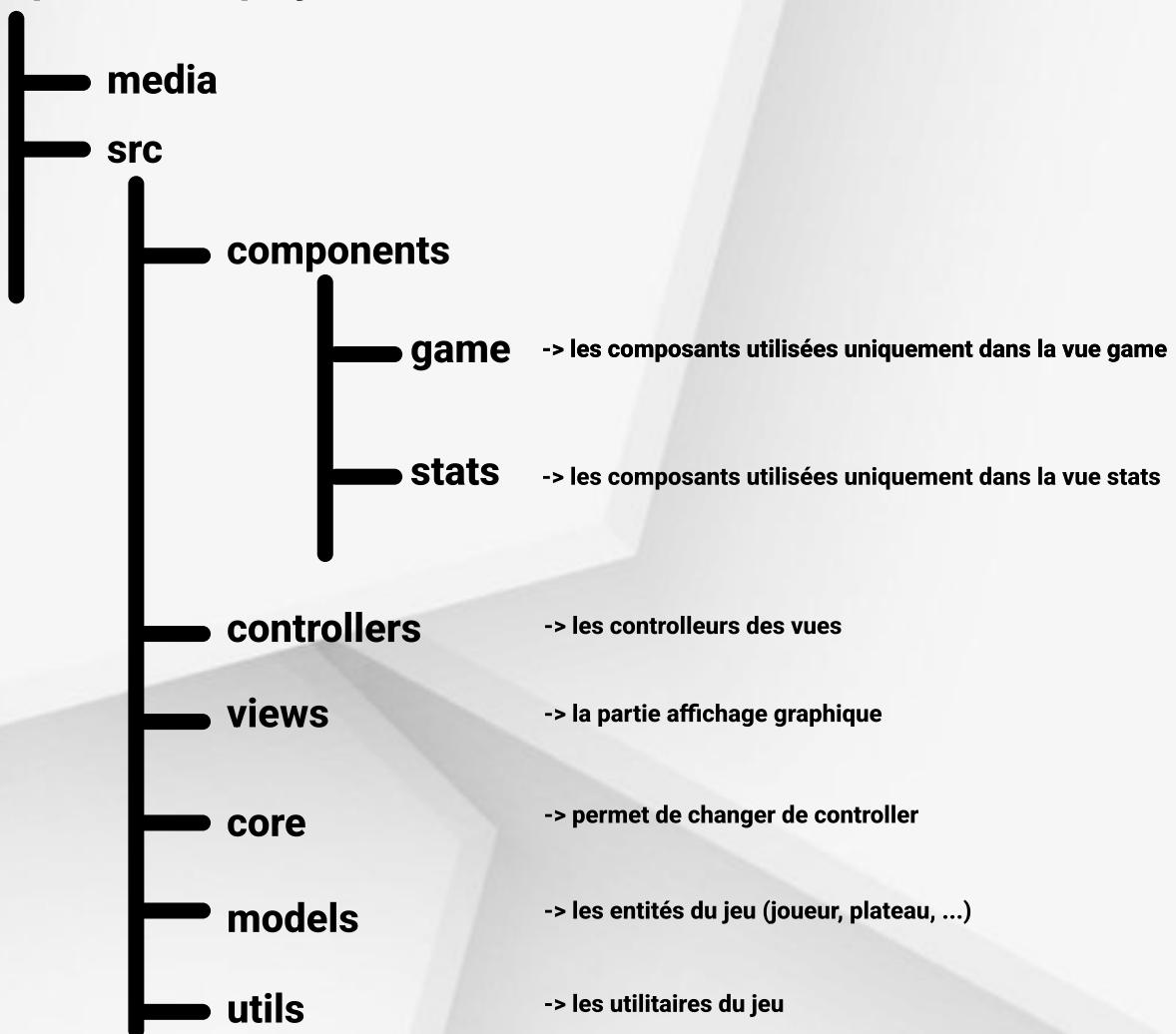
## Sommaire

I - Arborescence du projet .....	3
II - Modules du projet .....	4
III - Schéma réseau .....	8



# I - Arborescence du projet

## Répertoire du projet





## II - Modules du projet

Avant toute chose, il est important de préciser que nous suivons dans ce projet un modèle MVC sous le langage python.

**Les contrôleurs sont des classes ne gérant que deux éléments :**

### 1. Les callbacks

En effet, ceux-ci sont importants puisqu'il permettent de renvoyer l'information au dossier ./core (cerveau de notre application) qui va permettre de passer d'une vue à l'autre.

**Important : une vue ne peut être ouverte sans son contrôleur**

### 2. Envoyer des informations à leur vue associée

```
class RulesController(Controller):
    """
    Contrôleur gérant la partie des règles héritant
    de la classe Controller ainsi que de sa méthode
    abstraite main()
    """

    def __init__(self, window: CTk):
        self.window = window
        self.rulesView = self.loadView("Rules", self.window)
        self.core: Core = Core()

    def btn_clear(self):
        self.rulesView.close()
        c = Core.openController("home", self.window)
        c.main()

    def main(self):
        self.rulesView.main()
```

Exemple avec le contrôleur des règles, qui connaît sa vue

Utilisation du Core pour l'ouverture d'une Vue



La classe Core permet au Main de ne pas connaître le jeu, respectant ainsi les principes SOLID.

De plus, cette classe core ne possède qu'un méthode statique et vérifie si le contrôleur appelé existe

On peut ainsi ouvrir la classe correspondante (ce qui est notamment intéressant pour la maintenance du code)

```
class Core:
    """
    Classe centrale du jeu , le Main ne connaîtra que cette classe et pas le reste du jeu.
    Permet de respecter les principes SOLID.
    """

    @staticmethod
    def openController(controller,window):
        """Fonction permettant de charger le controller.

        Args:
            controller (str): le nom du controller (sans "Controller" à la fin)
            >>> Core.openController("Game",window)
            window (CTk): La fenêtre de jeu

        Returns:
            response : la classe correspondants
        """
        response = None

        controller = controller[0].upper()+controller[1:]
        controllerName = controller+"Controller"

        if os.path.exists(APP_PATH+"/controllers/"+controllerName+".py"):
            module = importlib.import_module("controllers."+controllerName)
            class_ = getattr(module, controllerName)
            response = class_(window)

    return response
```

```
class Main:
    @staticmethod
    def run():
        try:
            window = CTk()
            _resizeWindow(window, 700, 700)
            window.title("Blokus")
            window.iconbitmap(APP_PATH + r'..\media\Icon\icon.ico')
            app = Core.openController("Game", window)
            app.main()
            window.mainloop()
        except Exception as e:
            print(str(e))

if __name__ == '__main__':
    Main.run()
```

On peut changer l'appelle de "Game" par un contrôleur existant ("Stats", "Game", ...)



**Comment est appelé la vue correspondante au contrôleur :**

Il existe une méthode loadView utilisé par tous les contrôleurs (hérité par la classe contrôleur générale)

On appelle donc la vue correspondante au contrôleur de cette façon :

```
self.homeView = self.loadView("Home", window)
```

La class abstraite contrôleur :



## UML Simplifié pour la compréhension

Voici la forme de notre architecture du jeu, il faut savoir que celle-ci s'adapte parfaitement si vous souhaitez reprendre ce projet et ajouter des fonctionnalités, **il n'est pas nécessaire de modifier l'existant**

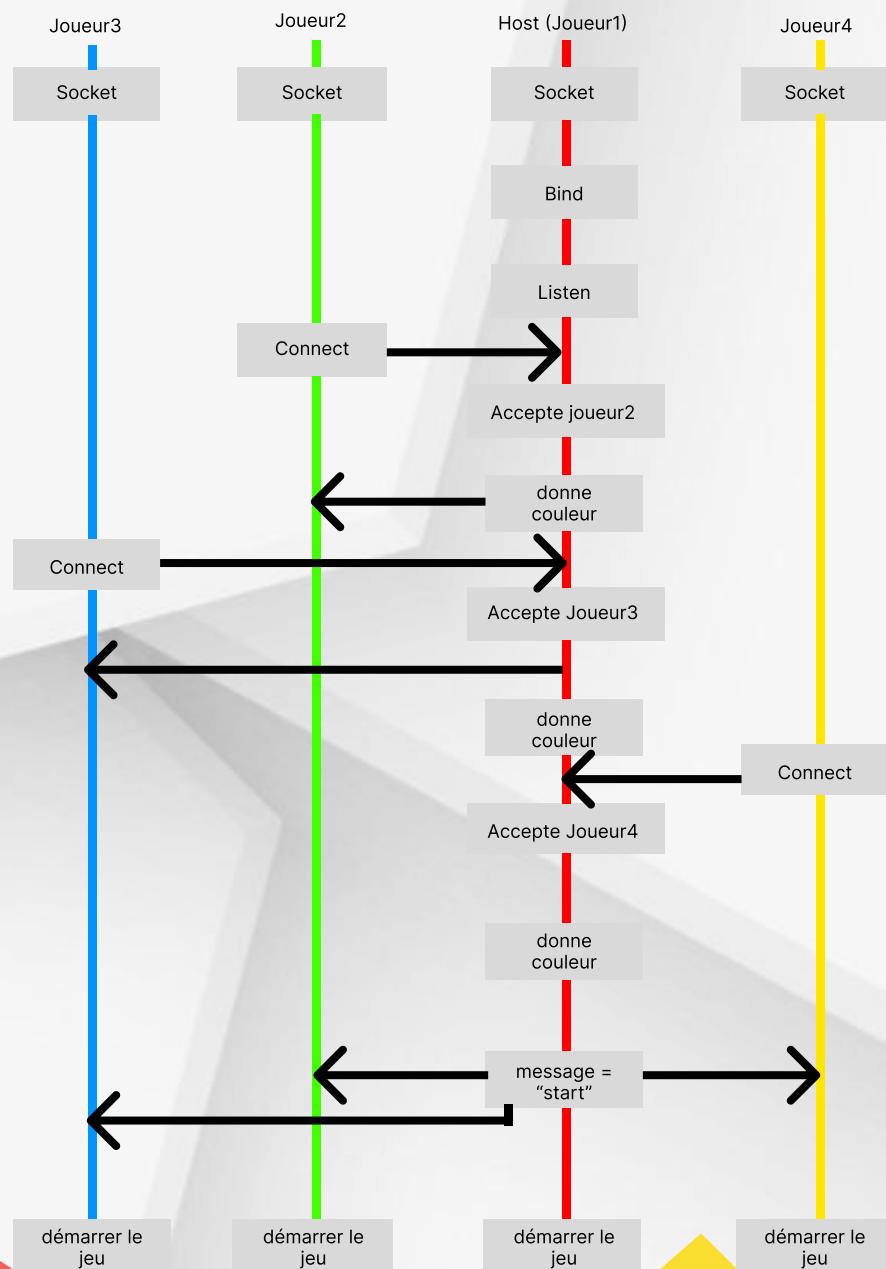
**Uml du 3ème semestre disponible à cette adresse :**

[https://github.com/Antorakk/blokus-game/blob/main/  
UML.PDF](https://github.com/Antorakk/blokus-game/blob/main/UML.PDF)



## III - Schéma réseau

Schéma démarrage partie





### III - Schéma réseau

Schéma pendant la partie

