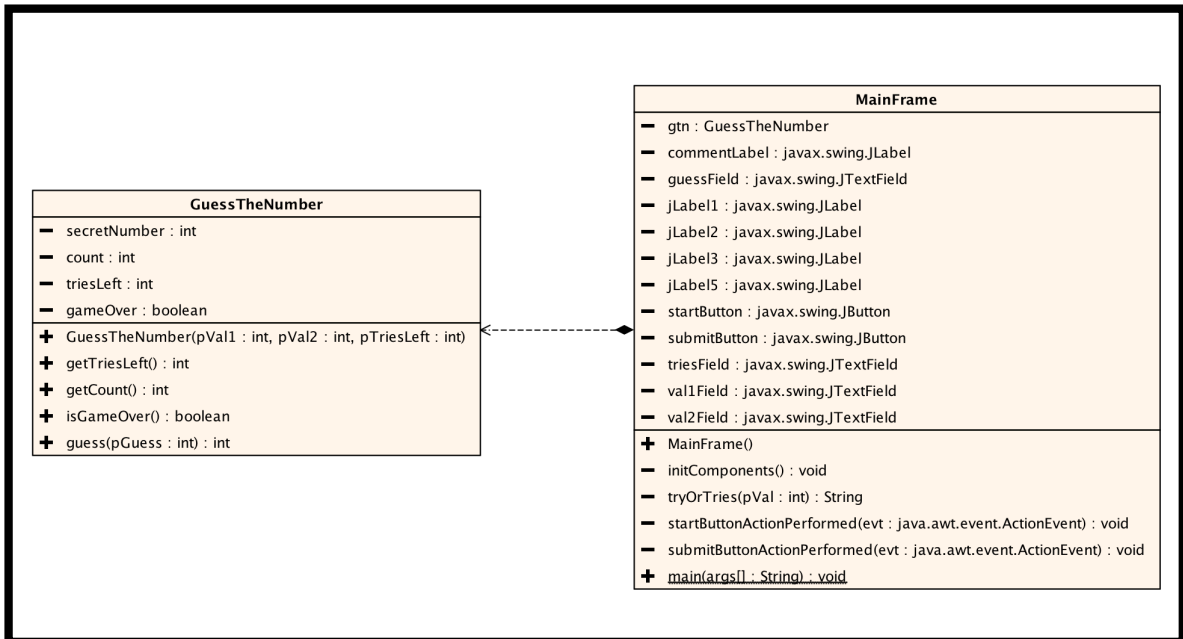


# Exercice D – Guess the Number

Le but de cet exercice est de créer un jeu qui consiste à deviner un nombre aléatoire généré par l'ordinateur.



## Création du modèle

1. Créez un projet **ExerciceD** dans Unimozzer
2. Créez la classe `GuessTheNumber`
3. Ajoutez les attributs suivants :
  7. `secretNumber` à valeur entière qui contient le nombre que l'utilisateur doit deviner
  8. `count` à valeur entière qui compte le nombre d'essais de l'utilisateur
  9. `triesLeft` à valeur entière qui stocke le nombre d'essais restants
  10. `gameOver` à valeur booléenne qui indique si le jeu est terminé
4. Ajoutez des accesseurs `getCount`, `getTriesLeft` et `isGameOver` pour les attributs correspondants. Il n'y a pas d'accesseur pour le nombre secret.
5. Ajoutez une méthode `guess` qui prend un nombre entier `pGuess` en paramètre et le compare au nombre secret si le jeu n'est pas déjà terminé.
  7. Si le jeu est terminé, la méthode retourne le nombre -2.
  8. Si le jeu n'est pas encore terminé, la méthode compare `pGuess` et `secretNumber` et retourne ...
    - i. 0 si les nombres sont égaux
    - ii. -1 si `pGuess` est plus petit que le nombre secret
    - iii. 1 si `pGuess` est plus grand que le nombre secret



*N'oubliez pas d'adapter les valeurs des attributs `count`, `triesLeft` et `gameOver` au cas échéant*



## Création de l'interface graphique

1. Créez une classe *MainFrame.java* et construisez l'interface graphique suivante par glisser-déposer :

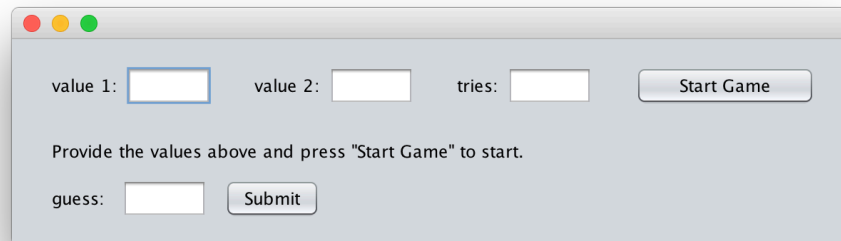


Figure 1 - Interface au démarrage du programme

2. Renommez toutes les variables de tous les éléments graphiques que vous devez accéder au code.
3. Procédez aux changements nécessaires pour que la classe *MainFrame* puisse communiquer avec la classe modèle.
4. Pourquoi n'est-il pas possible cette fois-ci de déclarer et d'initialiser la classe modèle en même temps ?
5. Comment faut-il initialiser la classe modèle cette fois-ci ?
6. Définissez le comportement du bouton **Start Game**. Quand le bouton *Start Game* est appuyé les valeurs des trois champs suivants sont reprises pour initialiser la classe modèle. Aussi l'étiquette du milieu de la fenêtre confirme l'initialisation du jeu en indiquant un message correspondant :

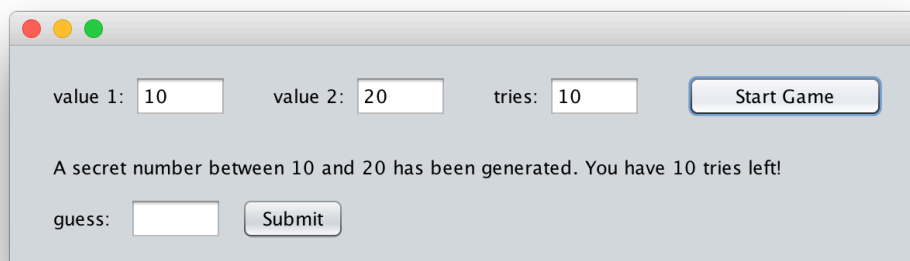


Figure 2 - Un message confirme l'initialisation du jeu avec les paramètres saisis

7. Avant l'initialisation du jeu, le fait d'appuyer sur le bouton **Submit** n'a aucun effet. Seulement après l'initialisation du jeu, il est possible de deviner le nombre secret à travers le champ *guess* et le bouton **Submit**.

Après chaque tentative de l'utilisateur l'étiquette au-dessus du champ *guess* indique si le nombre saisi est égal au nombre secret ou s'il est plus petit ou plus grand que le nombre secret.

Si le nombre saisi par l'utilisateur n'est pas le bon nombre, le nombre de tentatives restantes est affiché aussi.

Finalement, si le jeu est terminé, soit parce que le nombre secret a été deviné, soit parce que le nombre de tentatives restantes vaut zéro, le message *Game Over* est affiché aussi.

Si maintenant l'utilisateur appuie à nouveau sur le bouton *Submit*, un message lui rappelle que le jeu est terminé. Il peut cependant démarrer un nouveau jeu en appuyant sur *Start Game*. Il est possible de garder les valeurs actuelles ou de les changer avant de commencer un nouveau jeu.

Regardez les images des pages qui suivent pour mieux comprendre le déroulement du jeu.

**Ajoutez la méthode suivante à votre classe `MainFrame` :**

```
private String tryOrTries(int pVal){  
    if (pVal == 1){  
        return "try";  
    }else{  
        return "tries";  
    }  
}
```



Elle vous sera utile pour créer des phrases grammaticalement correctes : ..., 2 tries left, 1 **try** left, ..., You guessed right after 4 tries, ... You guessed right after 1 **try**

A screenshot of a game window with a light gray background and a standard macOS-style title bar with red, yellow, and green buttons. The window contains three input fields at the top: "value 1:" with "10", "value 2:" with "20", and "tries:" with "10". To the right of these is a "Start Game" button. Below the input fields, a message reads "15 is smaller than the secret number. You have 9 tries left!". At the bottom, there is a "guess:" label followed by an input field containing "15" and a "Submit" button.

Figure 3 – Premier essai : Le nombre saisi est plus petit que le nombre secret. Il reste 9 essais.

A screenshot of the same game window. The "value 1:" and "value 2:" fields remain "10" and "20" respectively, and the "tries:" field remains "10". The "Start Game" button is still present. The feedback message now reads "18 is greater than the secret number. You have 8 tries left!". The "guess:" input field now contains "18" and is highlighted with a blue border, and the "Submit" button is also highlighted with a blue border.

Figure 4 – 2ème essai : Le nombre saisi est plus grand que le nombre secret. Il reste 8 essais.

A screenshot of the game window. The "value 1:" and "value 2:" fields remain "10" and "20", and the "tries:" field remains "10". The "Start Game" button is still present. The feedback message now reads "17 is greater than the secret number. You have 7 tries left!". The "guess:" input field now contains "17" and is highlighted with a blue border, and the "Submit" button is also highlighted with a blue border.

Figure 5 - 3ème essai : Le nombre saisi est toujours trop grand. Il reste 7 essais.

value 1:  value 2:  tries:

16 is the correct answer. You guessed right after 4 tries. Game over.

guess:

Figure 6 - 4ème essai : Le nombre saisi est égal au nombre secret. Le jeu est terminé.

value 1:  value 2:  tries:

The game is over.

guess:

Figure 7 - Un message rappelle l'utilisateur que le jeu est terminé, si celui-ci appuie à nouveau sur le bouton *Submit*