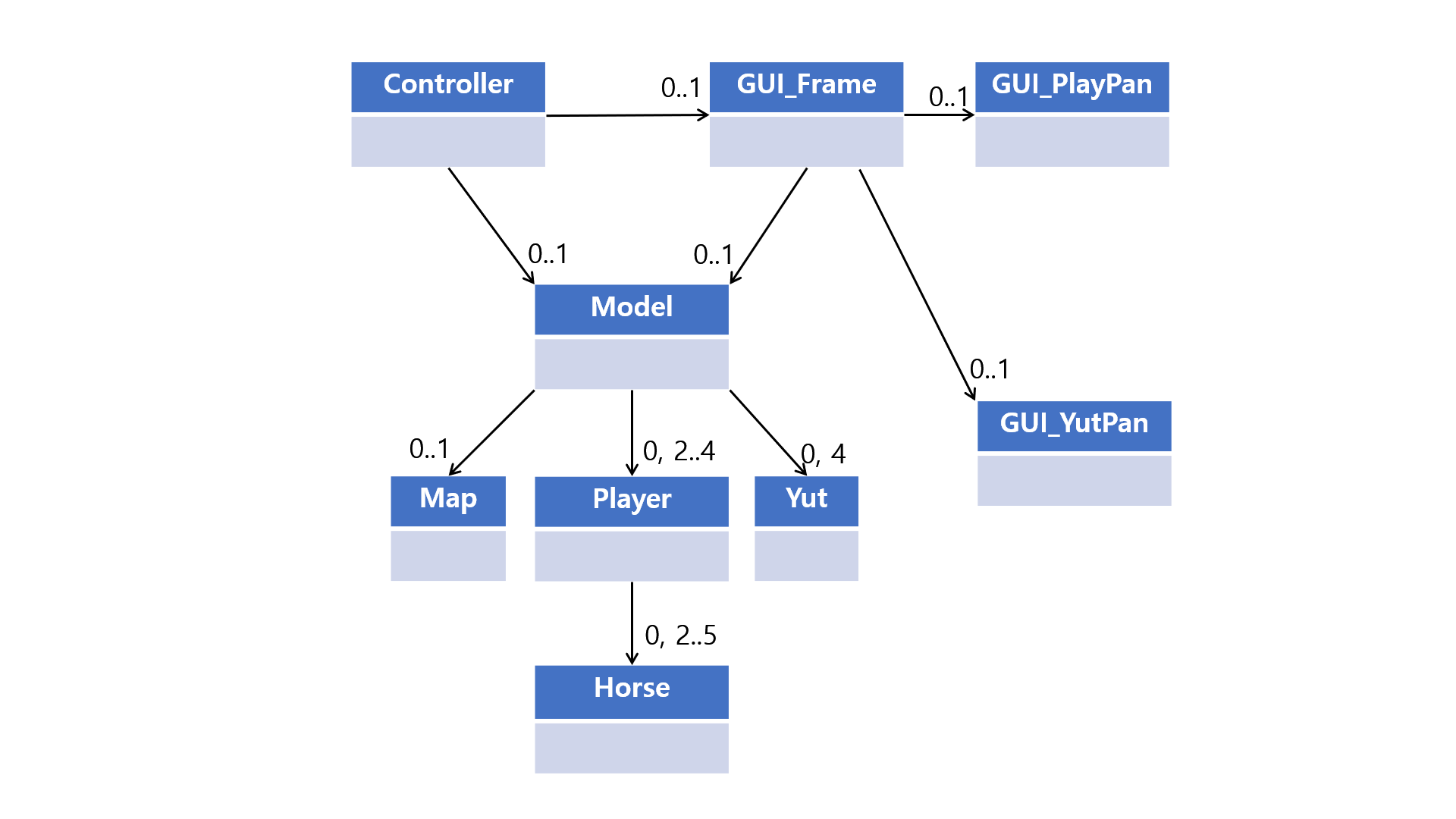
[D] AnalySis & Design, Implementation, and Test Document:

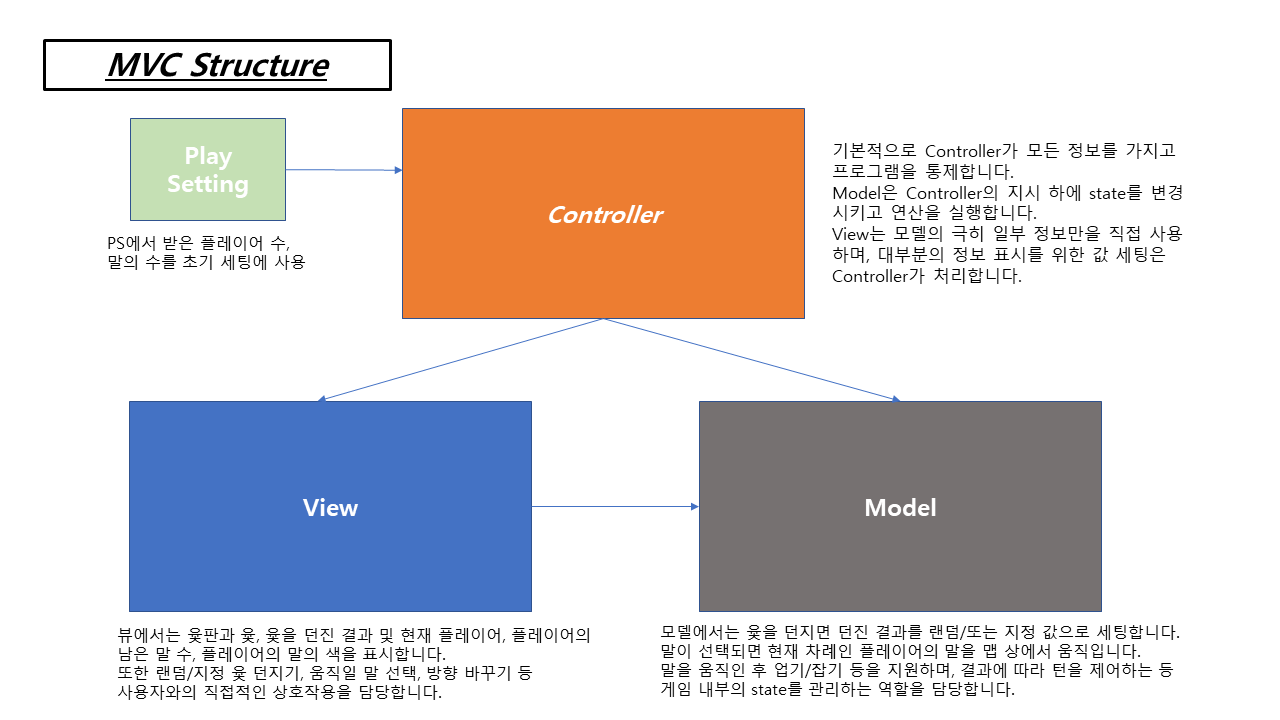
1. Domain Model



본 윷놀이 게임 프로그램의 Domain Model입니다. 저희는 비전 결정 후 프로그램의 설계 중점을 MVC로 놓고 시작하였습니다. 그래서 크게 Model, Controller, View로 나누고 설계를 시작하였습니다. 그에 결과 게임 내 로직 관련 class들(Map, player, Yut, Horse)은 Model이, UI 관련클래스들(GUI\_PlayPan, GUI\_YutPan)은 GUI\_Frame(View)가 처리하고, Controller는 그 둘을 조율해주는 일을 맡도록 설계했습니다.

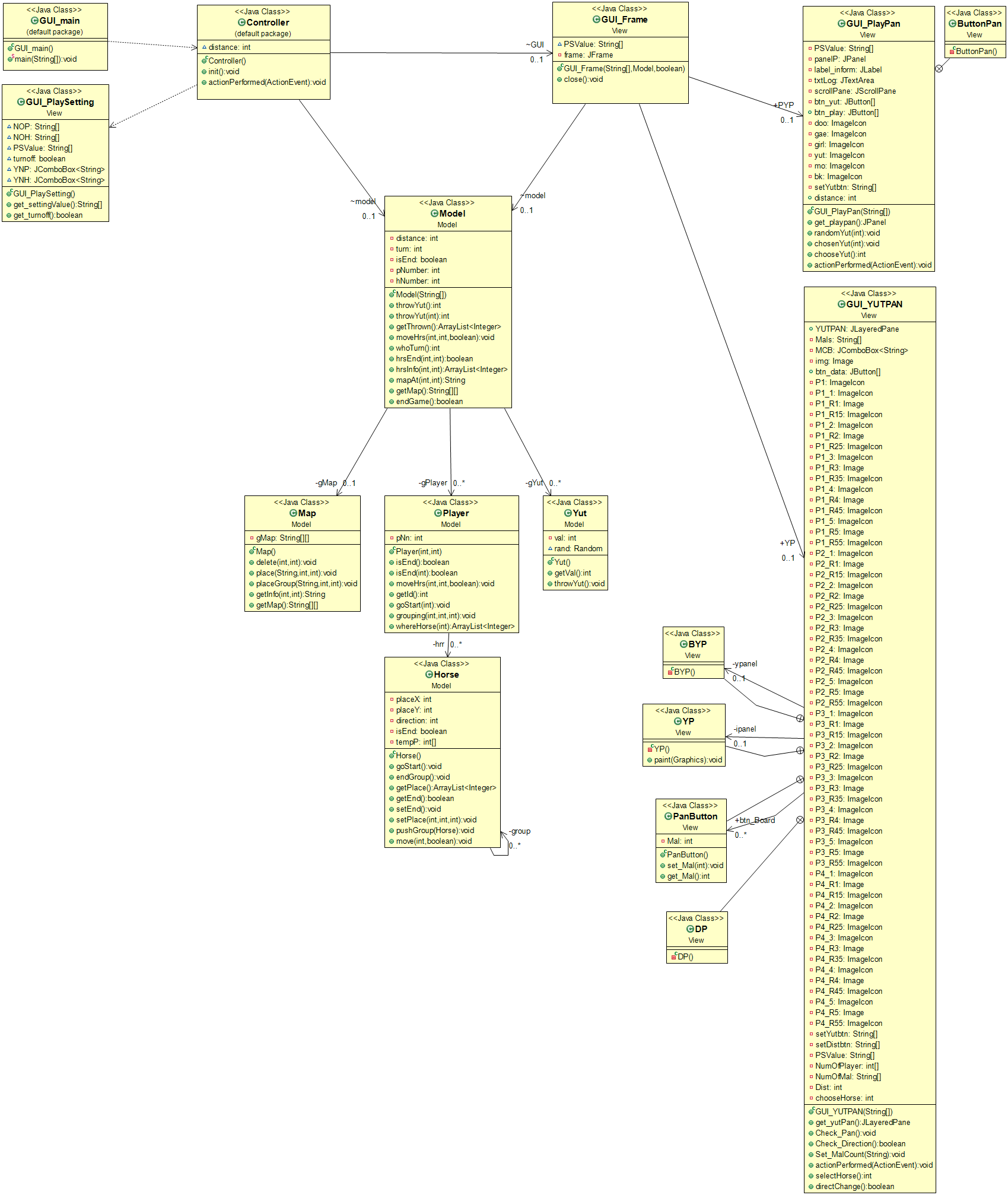
각 클래스 간의 관계에 기본적으로 0..1인 이유는 게임 실행 자체를 취소하는 경우 객체가 생성되지 않기 때문입니다. 단, Player와 Horse는 예외적으로 최소, 최대가 정해져 있으므로 따로 2..4와 2..5로 설정했습니다.

2. Software Architecture + Design Model

<< Software Architecture >>

본 윷놀이 게임 프로그램을 제작함에 있어서 MVC Structure를 적용하였습니다. Controller의 통제 하에 View는 화면 표시 및 사용자와의 상호작용만을 담당하며, Model은 그 외 윷놀이를 플레이함에 있어서 핵심이 되는 윷 던지기, 말 움직이기, 말의 잡기 및 업기와 관련된 기능들을 수행하도록 분리하였습니다. 단, 처음 시작 시 설정하는 플레이어 수와 말의 개수의 경우에는 게임이 시작되면 저장된 값만으로 사용이 가능하기 때문에 따로 인스턴스로 만들어 최초 1회만 사용하는 방식을 사용하도록 하였습니다.

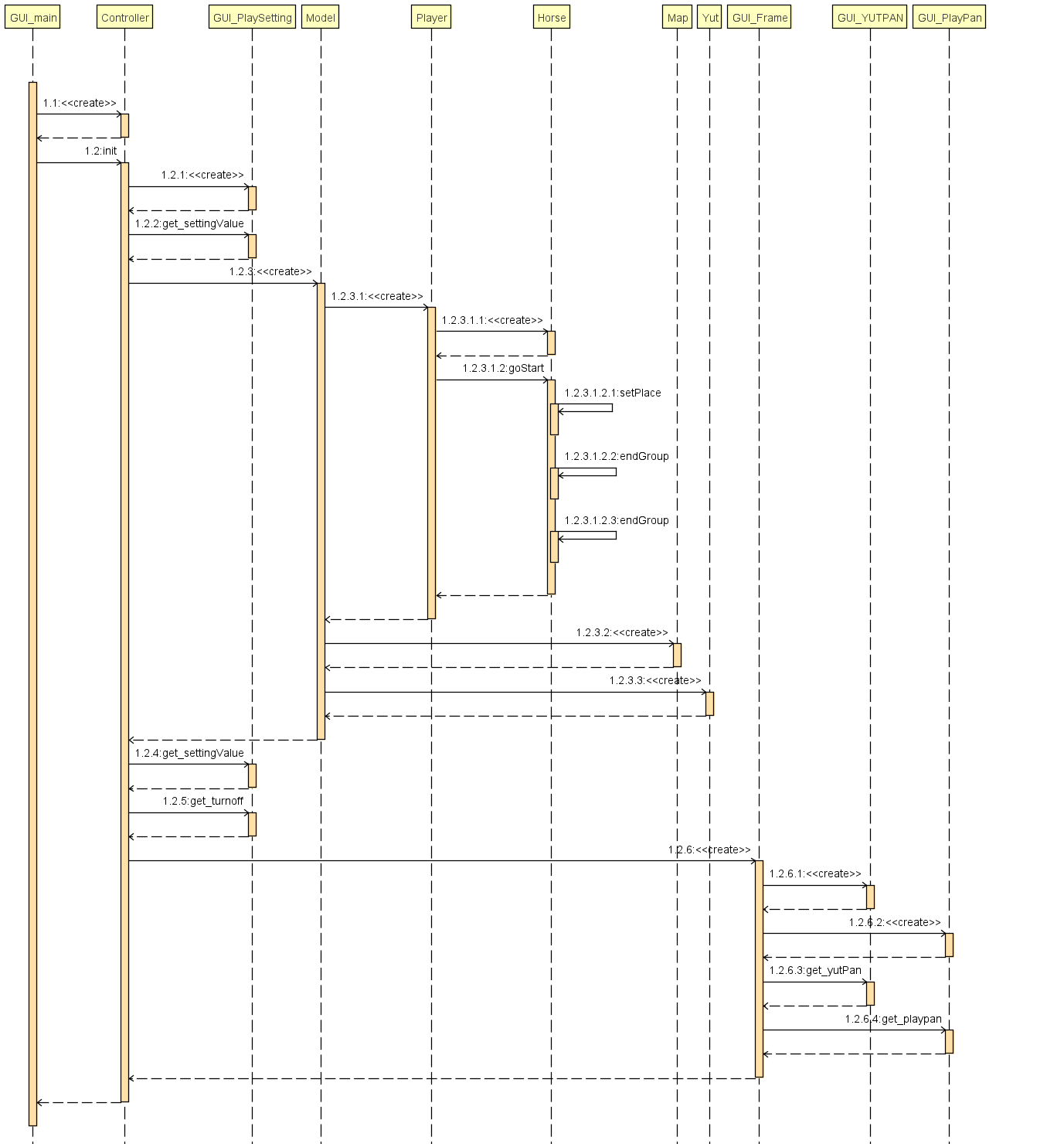
위와 같은 구조에서 Controller는 Model과 View에 대한 정보를 모두 가지고 있으며, View의 각 버튼 및 말 등의 표시에 필요한 값을 Model에서 받아와 설정할 수 있고, 이벤트 리스너를 가지고 있습니다. View의 랜덤/지정 윷 던지기 버튼과 말 선택 및 방향 선택에서 발생하는 사용자의 입력은 Controller가 알맞게 정제하여 Model의 메소드를 실행하는 매개 변수로 사용합니다. Controller의 호출에 따라 Model은 윷 던지기, 말 움직이기 등의 메소드를 실행하며, 이후 변화한 state값을 Controller가 받아 View의 값을 갱신할 수 있도록 getter를 지원합니다. 이 값으로 Controller는 View의 값들을 세팅한 뒤, 최종적으로 View의 화면을 갱신하여 사용자가 새로운 state를 확인할 수 있도록 합니다.

<<Class Diagram>>

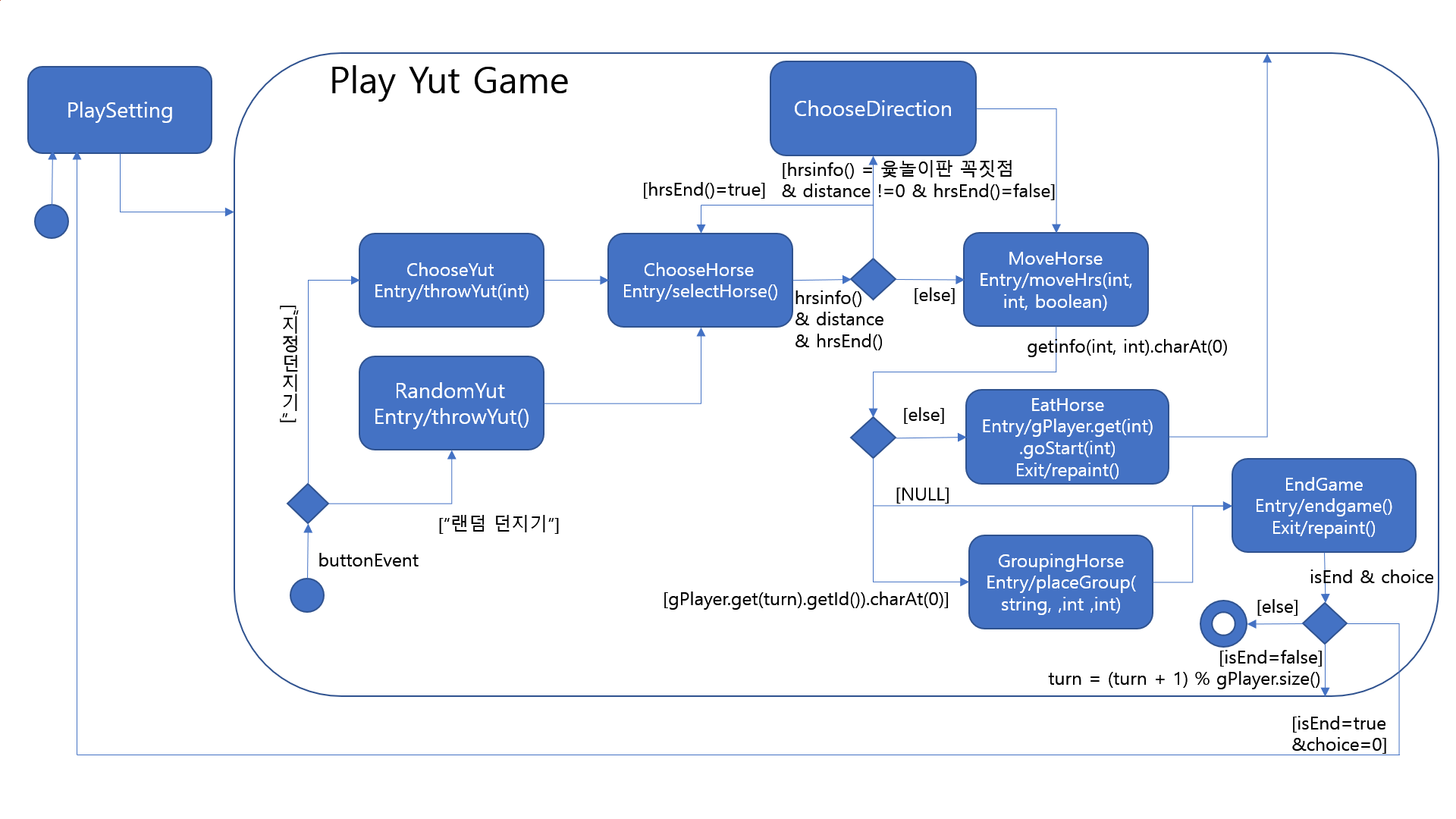
본 윷놀이 게임 프로그램의 클래스 구조는 위와 같습니다. GUI\_main 클래스가 윷놀이 게임 전체의 main클래스로써 동작합니다. GUI\_main에서 Controller의 인스턴스를 생성하고 Controller.init()을 실행하면 바로 PlaySetting이 실행됩니다. PlaySetting에서는 게임을 플레이할 인원 수와 말의 개수를 설정할 수 있으며, 취소를 누르게 되면 프로그램이 종료됩니다. PlaySetting의 설정 값을 기반으로 Controller가 Model과 View의 인스턴스를 생성합니다.

View에서 전체 컨테이너를 담당하는 클래스는 GUI\_Frame이며, 내부에 변수로 가지고 있는 JFrame에 모든 View가 표시됩니다. GUI\_PlayPan은 랜덤 던지기 버튼과 지정 던지기 버튼, 던진 윷에 대한 이미지, 현재 던진 말에 대한 정보를 화면에 출력합니다. GUI\_YutPan은 현재 윷 보드, 플레이어 번호, 플레이어의 남은 말, 플레이어의 색깔을 표시해줍니다.

Model은 맵, 윷, 플레이어 관련 객체 Map, Yut, Player를 가지며, Player는 각각 말 관련 객체 Horse를 가지고 있습니다. Controller로부터 지시를 받으면 거기에 해당하는 메소드를 실행하여 말을 옮기고 맵 정보를 갱신하는 일련의 과정을 처리합니다..

<<Sequence Diagram>>

<<StateChart>>

순서대로 StateChart를 표현하면

1. 맨 처음 프로그램 시작 시 PlaySetting을 실행, 플레이어 수와 말 개수를 확인 후 Play Yut Game State로 넘어 갑니다.

2. Play Yut Game state에 진입 후 button Event 발생시 button Event가 무엇이었는가에 따라 ChooseYut state나 RandomYut으로 이동합니다.

3. 각 state진입 시 해당하는 윷 던지기 메소드를 실행 후 ChooseHorse state로 이동합니다.

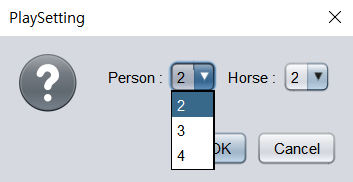
4. ChooseHorse state 진입 시 말을 고르는 selectHorse()를 실행 후 해당 말의 위치, 이동해야 할 거리, 말의 완주 여부에 따라 다시 말을 고르는 ChooseHorse로 가거나, 방향 이동여부를 결정하는 Choose Direction, 아니면 말을 옮기는 MoveHorse로 이동합니다.

5. MoveHorse 진입 시 말 번호, 이동거리, 방향 전환 여부를 가지고 MoveHrs를 실행합니다. 그 후 말을 두고자 하는 위치의 정보를 getinfo(int,int),charAt(0)으로 읽어 값에 따라 말을 잡는 EatHorse, 말을 업는 Grouping Horse, 둘다 아니면 EndGame으로 이동합니다.

5.1 EatHorse State의 경우 해당 플레이어가 다시 말을 던질 수 있으므로, 화면 출력을 갱신하는 repaint()를 실행하며 나가고, Play Yut State의 처음으로 돌아갑니다.

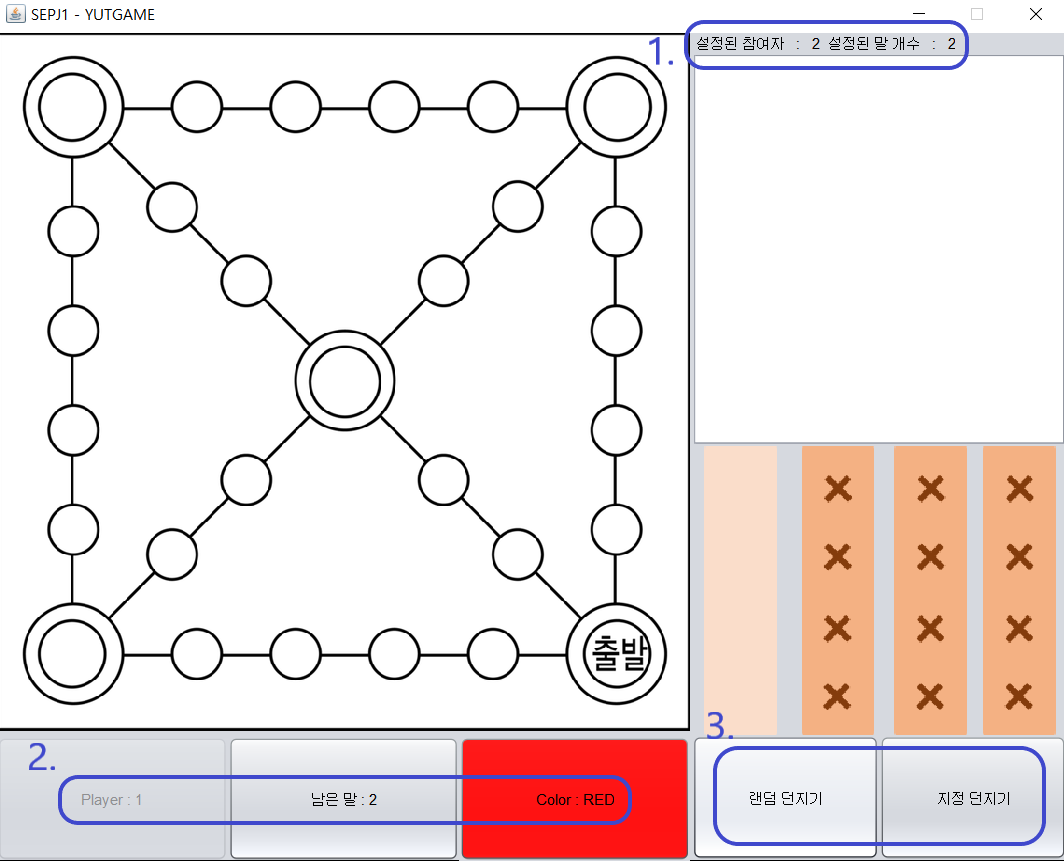
6. EndGame state에서는 endgame()으로 게임종료 여부를 확인하고, 나갈 때 repaint()로 화면 출력을 갱신합니다. 그 후 게임 종료여부인 isEnd와 재시작 여부인 choice에 따라 프로그램을 완전히 종료하거나 게임을 새로 시작하거나 플레이어 턴 변경 후 다시 Play Yut Game state로 돌아가 게임이 끝날 때까지 일련의 행동을 반복합니다.

3. usage of program and the screen shots



1. 프로그램을 첫 실행하면 PlaySetting이 뜨며 Player의 수와 Horse의 개수를 정할 수 있습니다.

그 후 OK를 누르면 게임이 시작됩니다.

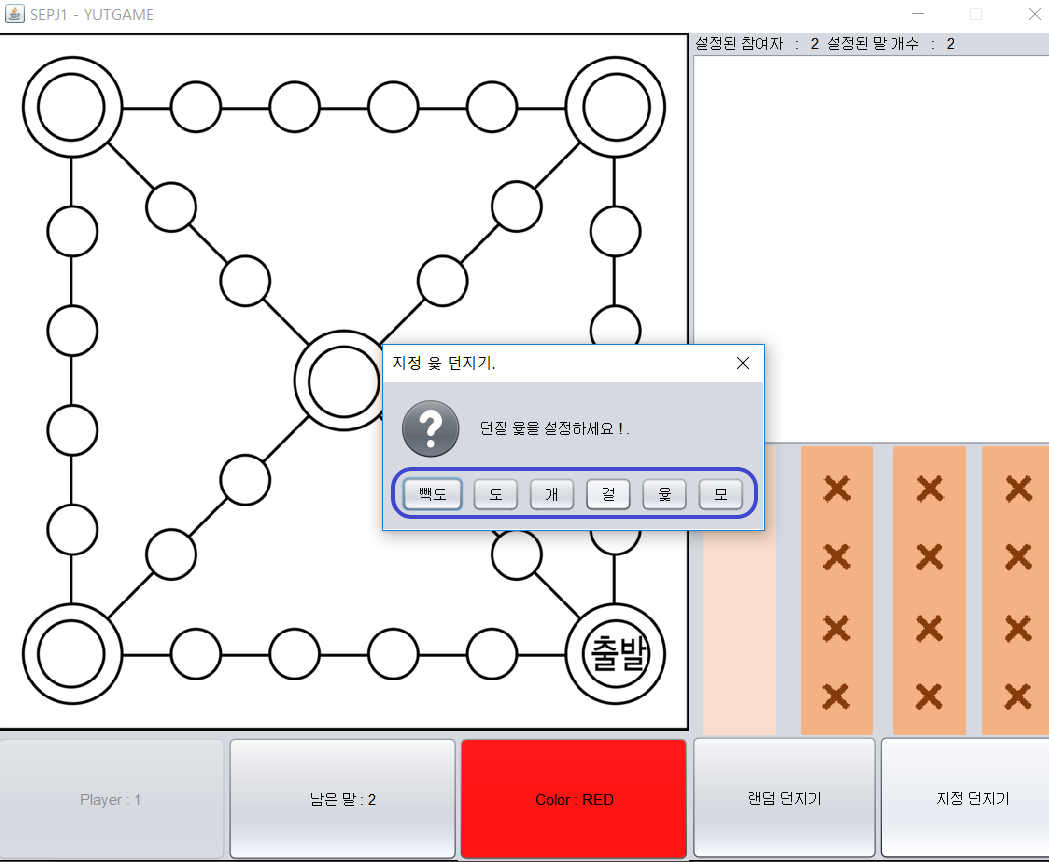


2. 전체적인 윷놀이 게임 UI이며 한눈에 현재 정보를 알 수 있습니다.

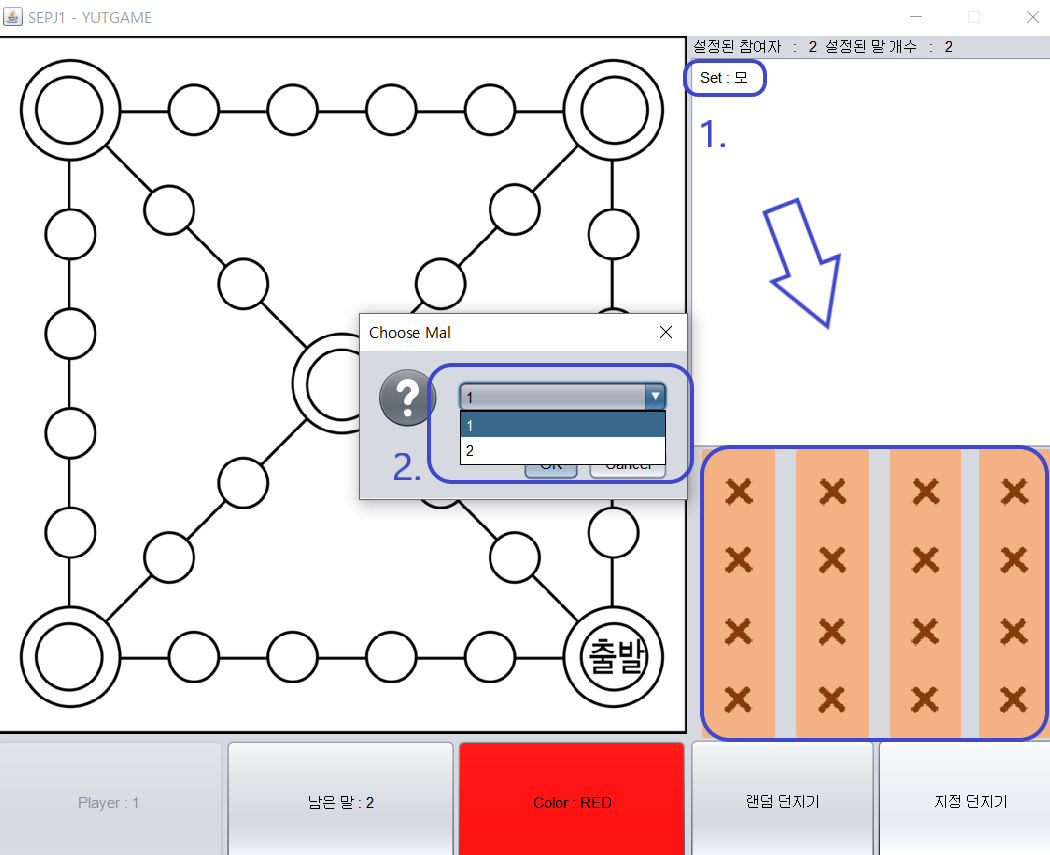
1) 현재 설정된 플레이어의 수와 말 개수를 표시해줍니다.

2) 현재 플레이어 번호와 해당 플레이어의 남은 말 개수, 플레이어 색깔을 표시해줍니다.

3) 윷을 던지는 버튼으로 랜덤 던지기는 임의의 윷이, 지정 던지기는 말을 고르는 창이 뜹니다.



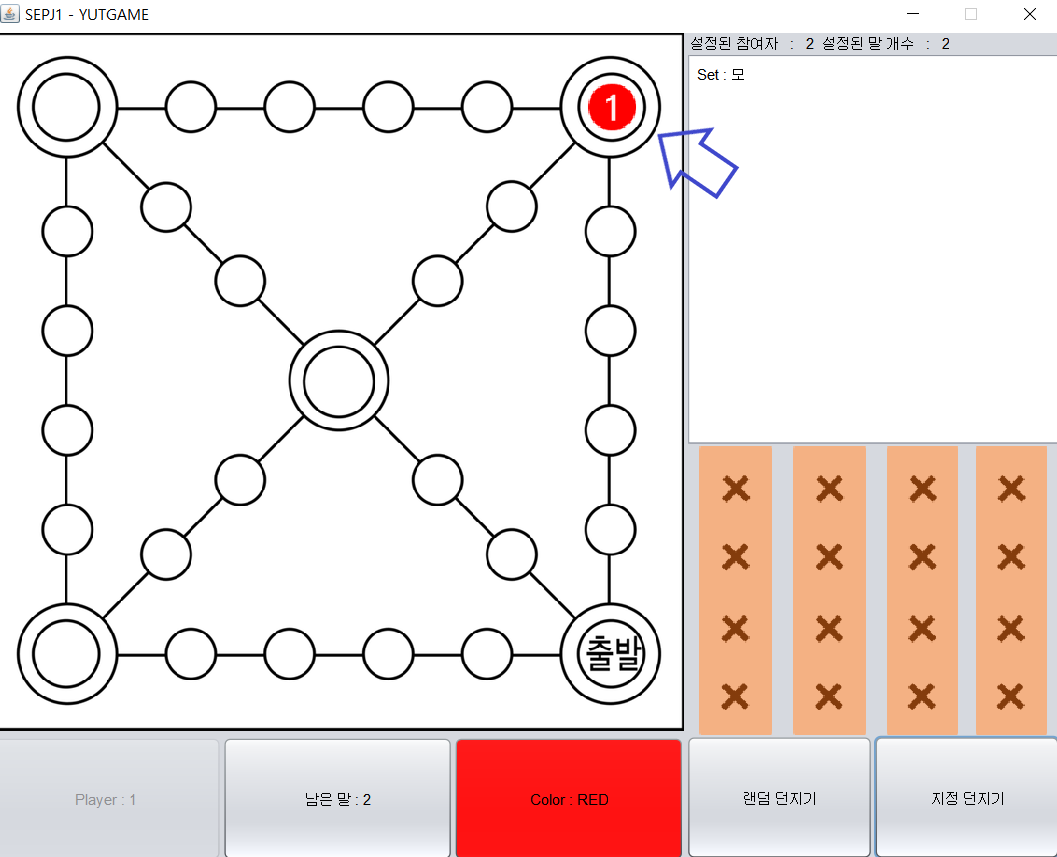
3. 지정 던지기를 눌렀을 때의 팝업 창이며 해당 6개의 윷 중 원하는 윷을 던질 수 있습니다.



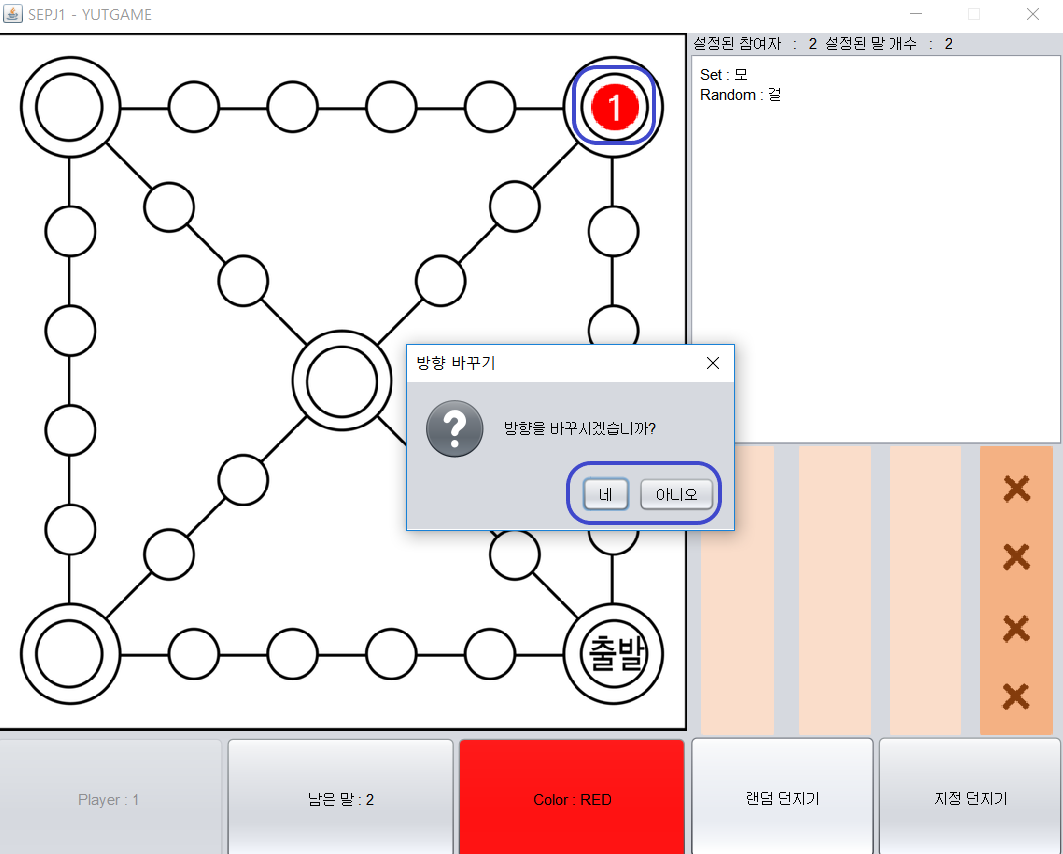
4. 윷 던지기를 실행 후의 화면입니다.

1) 방금 던진 윷의 정보와 해당 윷 이미지가 표시됩니다.

2) 어떤 말을 옮길지 선택할 수 있습니다.



5. 플레이어가 선택한 대로 말이 옮겨진 모습입니다.

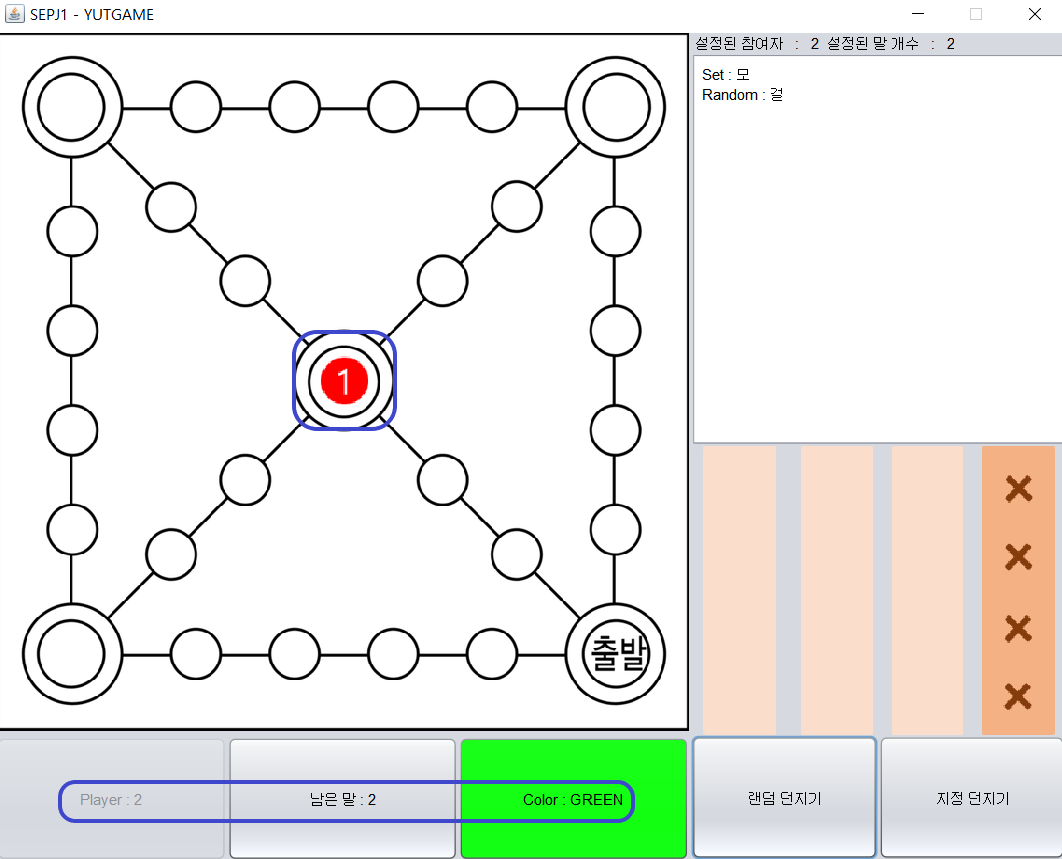


6. 모를 던졌기에 턴이 넘어가지 않고 한 번 더 윷을 던지는 상황입니다.

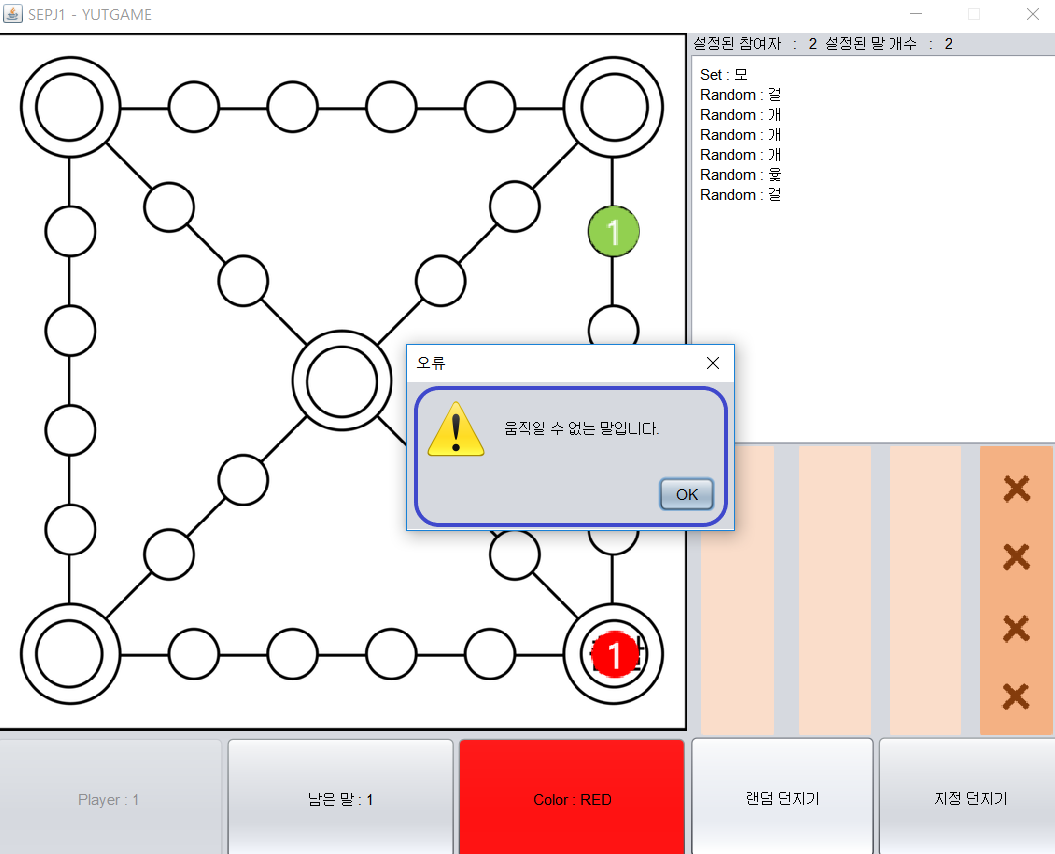
(모 외에도 윷을 던지거나, 상대방의 말을 잡은 경우에도 한번 더 던질 수 있습니다.)

방향을 결정할 수 있는 위치에 있기에 방향결정 여부를 묻는 창이 뜬 상황입니다.

여기서 ‘예’를 누를 경우 대각선으로 ‘아니오’를 누르면 왼쪽으로 이동합니다.

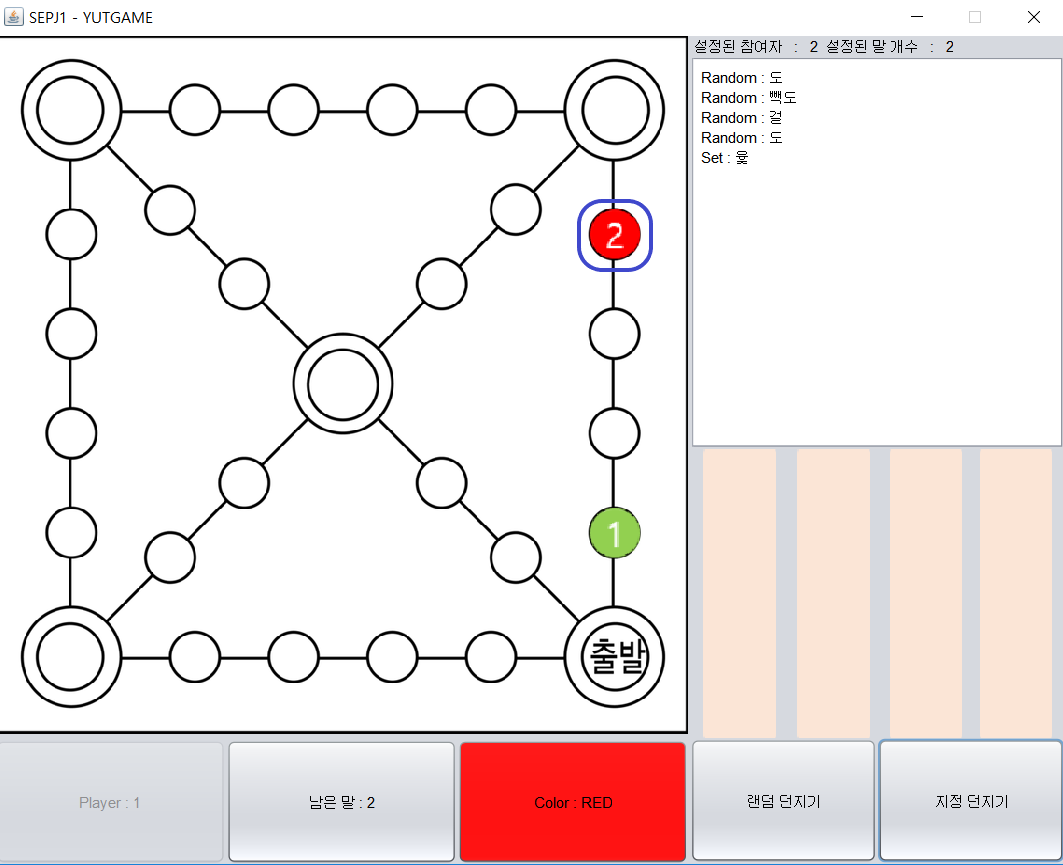


7. 6에서 예를 눌렀을 때의 화면이고, 현재 플레이어 턴이 다음 플레이어로 넘어간 모습입니다.

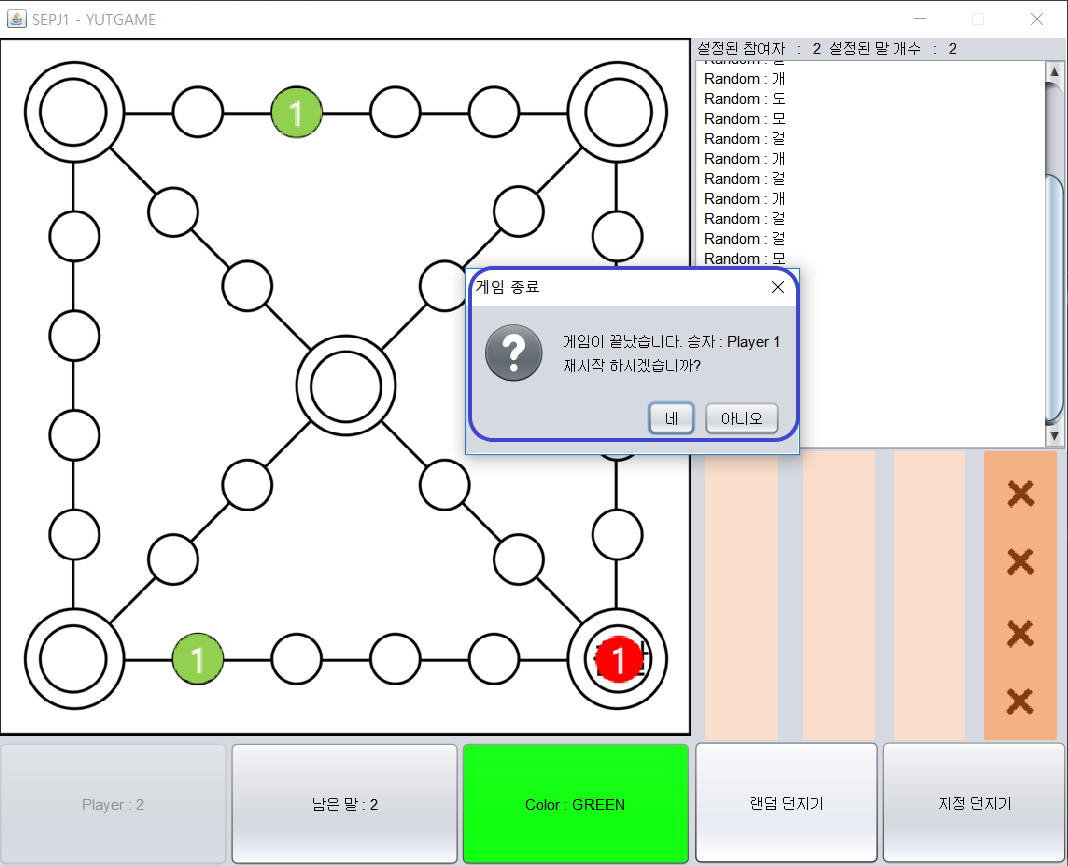


8. 만약 플레이어의 1번 말이 골인한 후 1번 말을 옮기려 할 경우

위와 같은 경고창이 뜨며 다시 말을 고르게 합니다.



9. 플레이어가 자신의 말을 업은 경우 위와 같이 업힌 말의 수가 변경 됩니다.



10. 특정 플레이어가 모든 말을 골인 시킨 경우 위와 같은 메시지가 출력되며

‘네’를 누를 경우 1번으로 돌아가며, ‘아니오’를 누를 경우 프로그램이 종료됩니다.