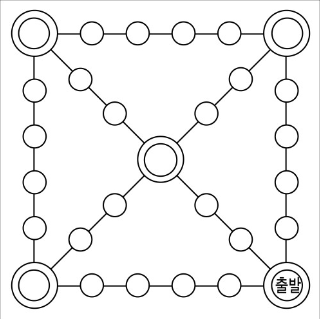
**1. 프로젝트 개요**

* 윷놀이 게임을 OOAD를 적용하여 소프트웨어로 개발한다.
* 게임 시작 시 참여자의 명수(최소 2명, 최대 4명)와 게임 말 갯수(최소 2개, 최대 5개)를 지정할 수 있다.
* 표준 윷놀이 판은 다음 그림과 같게 하며, 각 참여자의 말의 현재 위치가 표시되어야 한다.
* 각 턴의 진행을 위해 <랜덤 윷 던지기> 버튼과 <지정 윷 던지기> 버튼이 표시된다.
  + <지정 윷 던지기> 버튼을 누르면 빽도, 도, 개, 걸, 윷, 모 중 선택한 결과로 진행된다. (테스트용)
  + <랜덤 윷 던지기> 버튼을 누르면 랜덤한 윷의 결과가 나타난다.
* 사용자는 윷 던지기 결과를 적용할 게임 말을 선택할 수 있으며, 그에 따라 진행이 자동으로 되어야 한다.
* 게임 말을 업는(grouping) 기능을 지원해야 한다.
* 다른 사용자의 말을 잡는 기능을 지원해야 한다.
* 게임 말들이 출발에서 시작해서 먼저 모든 말을 내보내는 팀이 게임에 승리하며, 이때 어느 팀이 승리했는지 표시한다.
* 한 게임이 끝났을 때 게임 재시작 혹은 종료가 가능해야 한다

**주요 사항:**

* 본 프로젝트를 수행하면서 수업 시간에 다룬 OOAD 기법을 적극적으로 사용하며, 그 결과를 문서화해야 한다.
* MVC 아키텍처 패턴을 사용하여 UI와 모델을 분리하여 구현해야 한다.
* Extra point: 테스트 용이한 설계를 하여 JUnit으로 모델 테스트를 수행한다.

**2. 소개**

**게임이 시작되면 플레이어는 시작화면에서 게임에 참여할 플레이어의 수와 말의 수를 선택한다. 이후 윷놀이 게임을 플레이할 수 있는 게임 보드 화면으로 진입한다. 각 플레이어는 자신의 순서에 윷던지기 버튼을 클릭하여 게임을 진행하게 된다. 윷 셈 규칙에 따라 해당 값만큼 말을 좌표를 변경한다. 이동할 자리에 상대방의 말이 있을 경우 상대방의 말을 잡을 수 있다. 자신의 말이 있다면 말을 업는다. 플레이어의 모든 말이 시작점으로 들어오면 승리 메시지와 함께 게임이 종료된다. 이후 게임을 종료하거나 재시작 할 수 있다.**

**3. 유스케이스 모델**

**1) 유스 케이스 설명**

**윷 셈 규칙에 따라 말을 움직이는 윷놀이 게임을 구현한다.**

**2) 선행 조건**

**플레이어의 수가 정해져 있다.**

**말의 수가 정해져 있다.**

**3) 종료 조건**

**플레이어의 모든 말이 시작점으로 들어온다.**

**4) 유스 케이스**

1. **윷놀이를 플레이할 유저와 말의 수를 정한다.**
2. **게임을 시작한다.**
3. **윷놀이 판을 생성한다.**
4. **턴을 가지고 있는 플레이어는 윷을 던진다.**

**4-1. 윷이나 모가 나온 경우 4번을 다시 수행한다.**

**4-2. 뒷도, 도, 개, 걸이 나오면 5번으로 이동한다.**

1. **말을 움직인다.**

**5-1. 움직인 위치에 자신의 말이 있을 경우 자신의 말을 업는다**

**5-2. 움직인 위치에 상대방의 말이 있을 경우 상대방의 말을 잡고 4번을 다시 수행한다.**

**5-3. 윷놀이 판의 시작점을 지난 말은 진행을 완료한다.**

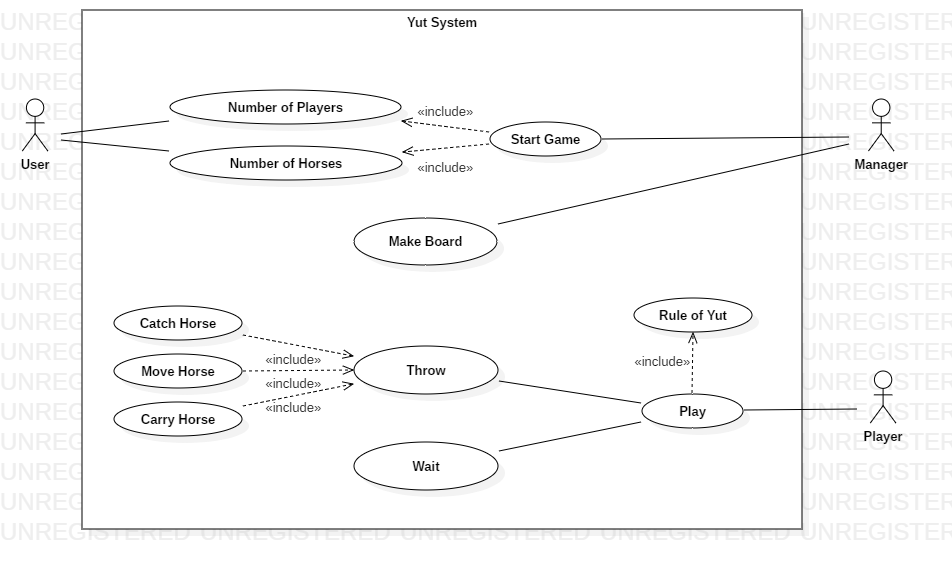
**5-4. 뒷도가 나왔는데 말이 없다면 도착장소로 이동한다..**

1. **턴을 가지고 있는 플레이어가 가진 모든 말이 시작점으로 들어오면 승리한다.**

**6-1. 게임을 종료하고 1번으로 돌아간다.**

**6-2. 승리 조건을 만족하지 못하면 현재 플레이어는 다음 플레이어에게 턴을 넘기고 다음 턴까지 기다린다. 다음 플레이어는 4번으로 돌아가 수행한다.**

**<Usecase Diagram>**



**<System sequence diagram>**



**4. 도메인 모델 설계**

**<도메인 모델>**

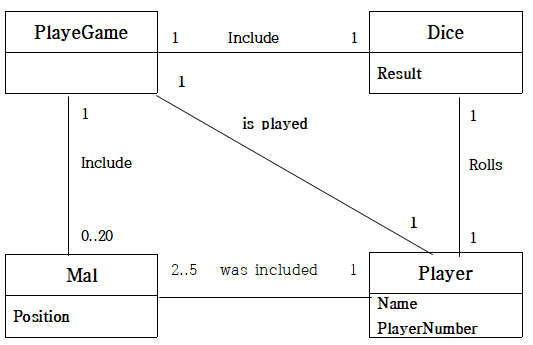
**1) Conceptual classes 찾기**

**- Mal(윷 말), Team(플레이어), Dice(윷)**

**2) Conceptual classes에 associations, attributes추가**

|  |  |  |
| --- | --- | --- |
| **구 분** | **Attributes** | **Associations** |
| **Mal(윷 말)** | **Position (위치)** | **플레이어는 최소 2개에서 최대 5개의 말을 가진다** |
| **Team(플레이어)** | **Name (플레이어 이름)** |
| **PlayerNumber (인원 수)** | **플레이어는 1개 윷 결과를 가진다** |
| **Dice(윷)** | **Result (윷의 결과 값)** |
| **PlayGame(게임진행)** |  | **윷과 말을 포함하고 플레이어가 게임하는 것** |

**3) UML 그리기**



**<소프트웨어 아키텍쳐와 디자인 모델>**

**1) 주요 디자인 결정**

**- Player(플레이어)와 Dice(윷), Mal(말)을 Play game (전체적인 게임 진행 상황)클래스로 설정, JAVA FX를 사용해서 FXML로 플레이어 수, 게임 말 수 선택화면, 윷놀이의 게임을 진행하는 화면으로 2개를 구성, 그리고 2개 FXML의 컨트롤러로 2개의 컨트롤러 클래스** **(Selectcontroller, Boardgame)으로 구성**

**2) 구현한 디자인의 MVC패턴 적용**

**① 사용자와 View와의 상호작용: JAVA FX를 사용해서 View부분을 FXML로 분리, 플레이어 수와 말의 수 선택화면(Select.fxml), 윷놀이 게임이 진행하는 동안의 화면(boardgame.fxml)**

**② 컨트롤러가 사용자의 입력을 처리: 플레이어 수와 말의 수 선택화면의 입력을 처리하는 컨트롤러 클래스(Selectcontroller), 윷놀이 게임이 진행하는 동안 화면에 입력을 처리해주는 컨트롤러 클래스(Boardgame)**

**③ 컨트롤러는 모델을 업데이트 시킴: 컨트롤러 클래스가 기능 별 클래스들(Mal, Dice, Player, PlayGame)에 값을 전달해서 업데이트**

**④ 모델은 변환된 값을 알려 준다: 기능 별 클래스들(Mal, Dice, Player, PlayGame)의 변환된 값을 리턴**

**⑤ View는 사용자와 상호작용: 리턴받은 값을 통해서 View를 업데이트하며 사용자와 상호작용**

**3) OOAD 적용**

**① OOA**

**- 개념식별(Object) : Player(플레이어), Mal(말),Dice(윷)**

**② OOD**

**- Responsibility 할당**

|  |  |  |
| --- | --- | --- |
| **객체** | **속성** | **메소드** |
| **Player** | name(플레이어 이름) | group(말 업기) |
| move(말의 이동) |
| malnumber(말 개수) | capture(다른 플레이어의 말 잡기) |
| win(우승여부 판단) |
| **Mal** | x, y (x, y좌표) | move(말의 좌표 변경) |
| isStart, isFinish(출발 or 도착 구분) | setReady(잡힌 말의 좌표를 시작점으로) |
| **Dice** | dice(윷) | dice(윷 던지기) |
| setdice(지정 윷던지기) |

※ **관심사(SOC) 분리를 하면서 자연스럽게 SOLID구현**

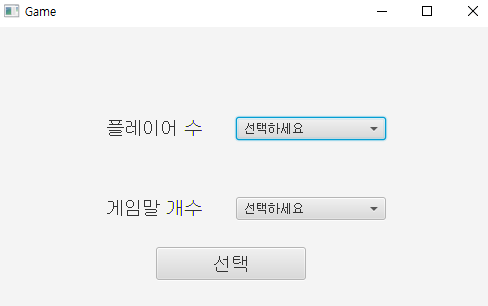
**- 객체들의 관계 및 진행**

|  |  |  |
| --- | --- | --- |
| **호 출 순 서** | **메소드** | **내용** |
| **1** | dice or setdice | 윷 또는 지정된 윷 던지기 |
| **2** | Mal.move | 윷의 값만 큼 좌표변경 |
| **3** | Player.move | 변경된 좌표만큼 말 이동 |
| **3-1** | capture | 다른 플레이어 말 잡기 |
| **3-2** | setReady | 잡힌 말을 시작점으로 변경 |
| **4** | win | 플레이어의 우승여부판단 |

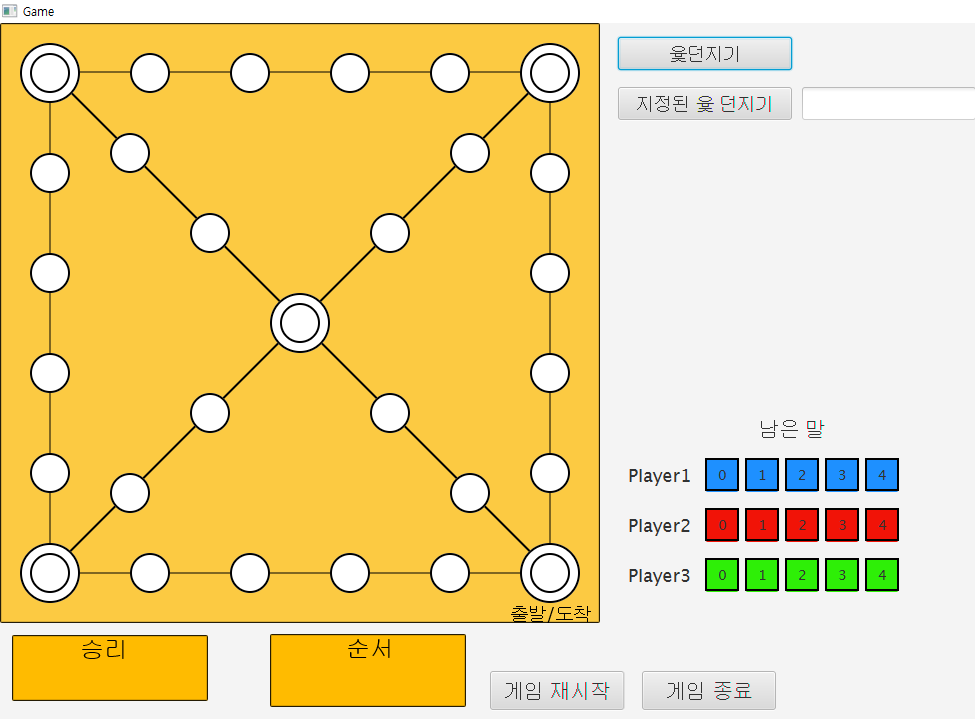
※3-1, 3-2 수행했을 시에 같은 플레이어가 1번으로 돌아가서 다시 실행

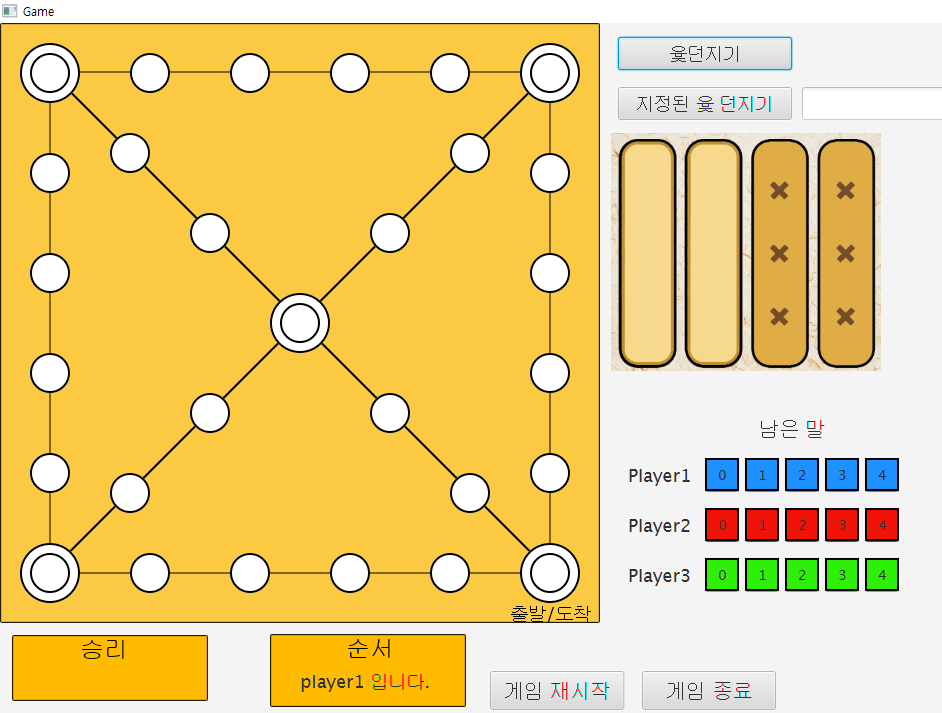
아닐 경우 다음플레이어가 1번부터 실행

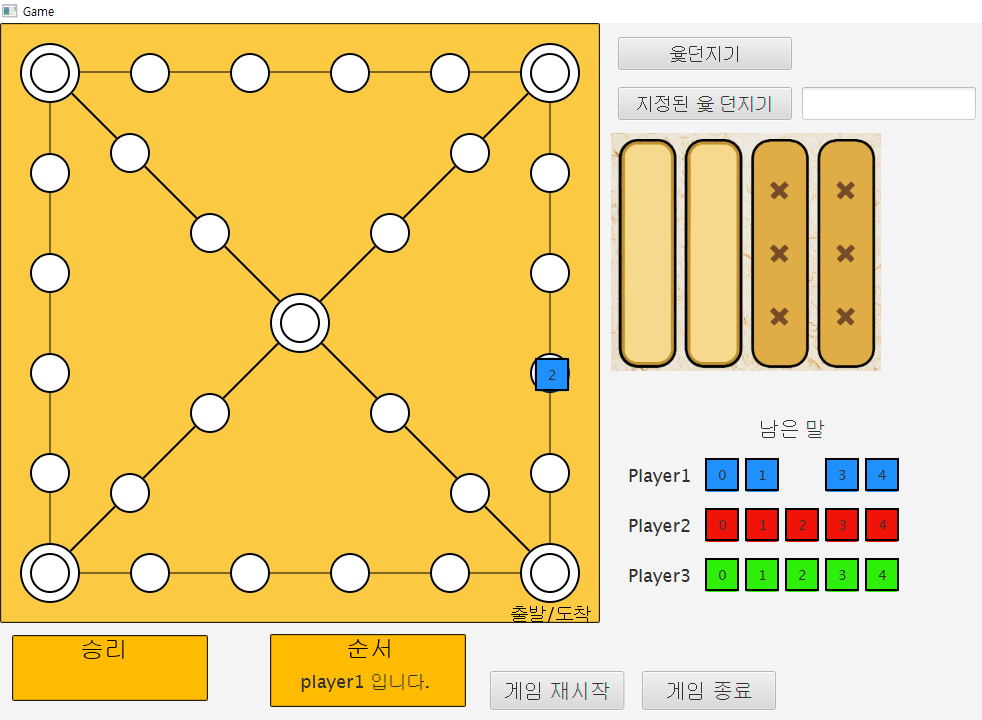
**3. 프로그램 사용법과 결과화면**

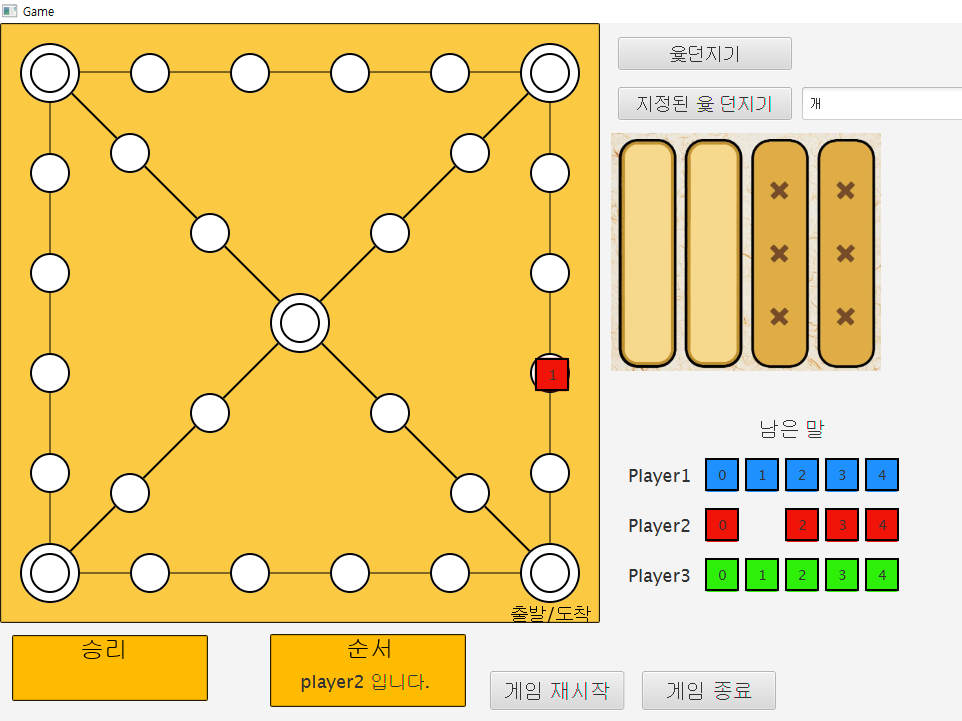
**1) 프로그램을 처음시작 할 때 플레이어 수와 말의 개수 선택 후 버튼 클릭**

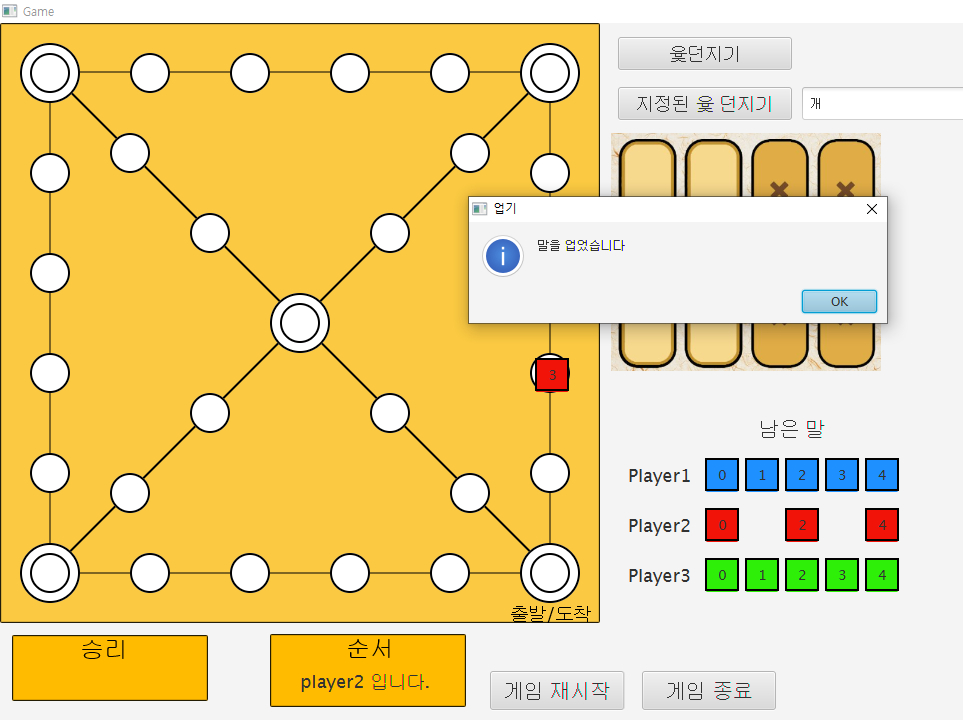
**2) 윷놀이 게임화면**

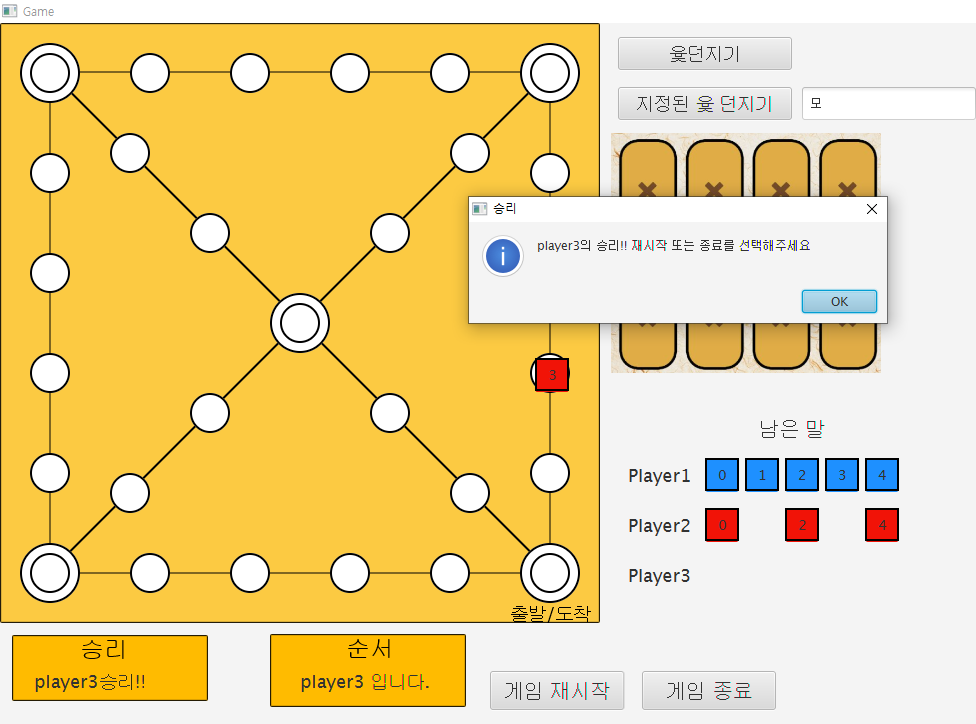


**3) 윷 던지기 or 지정된 윷**

**4) 이동하고 싶은 말을 클릭하면 자동 이동**

**5) 말을 잡을 경우**

**6) 말을 업을 경우**

**7) 플레이어 순서대로 하다가 승부가 결정 났을 경우**

**8) 승패 결정 후 게임 재 시작 또는 게임종료 버튼을 클릭(재시작일 경우 처음 플레이어 수와 말 수 선택화면으로 돌아갑니다)**