```
    GameController *-- Board
    GameController *-- Rule
    GameController *-- Die

    Board *-- Cell
```

# Better Ludo Class Diagram

```
classDiagram
    namespace GameFramework{

        class ISceneManager{
            <<interface>>
            + void StageSceneExit()
            + void StageSceneInsert(IScene s)
            + void CommitScene()
            + void GetCurrentScene()
        }
        class GameEngine{
            - Stack~IScene~ _sceneQueue
            + void Run()
            - void Loop()
        }
        class IScene{
            <<interface>>
            + void Update()
        }

        class IContextManager_T_{
            <<interface>>
            + T GetContext()
        }
    }

    ISceneManager -- IScene
    GameEngine --|> ISceneManager
    GameEngine o-- IScene
    namespace GameObject{
        class Player{
            <<interface>>
            + ID : readonly
        }
        class PlayerWithAction{
            <<interface>>
            + ID : readonly
            + Actionable GetActionable()
        }
        class Actionable{
            + Step()
        }
    }
    Player <|-- PlayerWithAction
    PlayerWithAction -- Actionable

    namespace LudoGame{
        class IContextManager_LudoContext_{
            <<interface>>
        }

        class LudoContext{
            + List~LudoPlayer~ players
            + Board board
        }

        class LudoGameScene{
            - ISceneManager _sceneManager
            - LudoContext _context
            - void NextTurn()
        }

        class LudoRule{
            - IContextManager _contextManager
            * bool Check(LudoActionable)
        }
        class LudoActionable{
            - int power
        }
        class SingleTotemActionable{
            - Totem bindedTotem
            * void Bind(totem)
        }
        class LudoTotemStart{
        }
        class LudoTotemMove{
        }
        class LudoTotemMoveTogether{
            - Totem
            + void Bind(Totem, Totem)
        }

        class LudoDice{
            +Roll()
        }
    }
    SingleTotemActionable <|-- LudoTotemStart
    SingleTotemActionable <|-- LudoTotemMove
    Actionable <|-- LudoActionable
    LudoActionable <|-- SingleTotemActionable
    LudoActionable <|-- LudoTotemMoveTogether
    LudoRule -- LudoActionable
    LudoGameScene -- LudoRule

    IScene <|-- LudoGameScene
    IContextManager_T_ <|.. IContextManager_LudoContext_ : bind T as LudoContext
    LudoGameScene -- LudoContext
    LudoGameScene --|> IContextManager_LudoContext_



    namespace LudoObjects{
        class LudoPlayer{
            List~Totem~ totems
        }



        class Board{
            +List~Cell~ Cells : readonly
            +List~List~int~~ Paths : readonly

        }

        class Cell{
            + CellType type : readonly
            - List~Totem~ Occupants
            + AddTotem(Totem)
            + KickTotem(Totem)
            + GetOwnership()
        }

        class CellType{
            <<enumeration>>
            Normal
            Safe
        }

        class Totem{
            + mathvector position : public get
            + mathvector homePosition : public get
            - List~int~ path
            + AdvanceOnce()
            + GoHome()
        }

    }
    Cell --* Board
    Totem --* LudoPlayer
    PlayerWithAction <|-- LudoPlayer
    LudoContext *-- Board
    LudoContext *-- LudoPlayer

    Totem -- LudoTotemMoveTogether
    Totem -- SingleTotemActionable

    Totem -- Cell
```