모두의 마블 2.0 개발 지침

**VibeCoding Prompt — “모두의 마블 2D 콘솔 버전 (Java)”**

**목표**

“모두의 마블” 게임 규칙을 단순화한 **2D 콘솔 버전**을 Java로 개발한다.

GUI는 사용하지 않으며 **JavaFX 제외**, 기본 콘솔 입출력(Text 기반)만으로 구현한다.

**프로젝트 구조**

com.marblegame

├─ core/ ← 게임 로직 (룰, 엔진)

├─ model/ ← 데이터 모델 (Player, Tile, Board 등)

├─ cli/ ← 콘솔 입출력 및 렌더링

└─ resources/

└─ rules.json ← 규칙 설정파일

**rules.json (설정값 예시)**

{

"salary": 300000,

"bail": 200000,

"islandMaxTurns": 3,

"tollMultiplierByLevel": [1.0, 1.5, 2.5, 4.0],

"colorMonopolyMultiplier": 2.0,

"doubleExtraTurn": true,

"tripleDoubleSendsToIsland": true

}

**핵심 규칙 (MVP 버전)**

확실한 규칙만 반영

* 주사위 2D6
* 출발지(START) 통과 시 월급 지급
* 토지 매입, 업그레이드, 통행료 지불
* 더블 → 한 턴 추가
* 3연속 더블 → 무인도 이동
* 무인도 3턴 이동 불가 (더블/보석금/카드 탈출 가능)
* 컬러 독점 시 통행료 ×2
* 승리 조건: **마지막 1인 생존 (파산 제외)**

확실하지 않은 부분(금액, 업그레이드 비용 등)은 rules.json으로 분리 관리.

**개발 단계별 진행 (Step-by-Step)**

**Step 1. 기본 모델 구현**

* Dice : 2D6 주사위, 더블 판정 포함
* Player : 이름, 현금, 위치, 무인도턴, 파산 여부
* Tile : 기본 필드 (id, name, type, colorGroup)
* City : Tile 상속 (price, baseToll, owner, level)
* Board : List<Tile> 순환형 보드

**Step 2. RuleEngine 구축**

* 통행료 계산 (level, color 독점 반영)
* 매입, 업그레이드, 파산 로직
* 무인도/더블/3연속 더블 처리

**Step 3. Game 루프**

1. 플레이어 순서대로 턴 진행
2. 무인도 여부 확인
3. 주사위 굴리기 → 이동
4. 출발지 통과 시 월급 지급
5. 도착한 칸 처리 (CITY/ISLAND/CHANCE/START)
6. 파산 여부 체크
7. 더블 시 한 턴 추가 / 3연속 더블은 무인도 이동

**Step 4. 콘솔 렌더러 (2D 텍스트 기반)**

* 보드는 28칸 (정사각형 테두리 느낌)
* 각 칸의 id 또는 이니셜 표시
* 플레이어는 A, B 등 문자로 표현
* 매 턴마다 주사위, 이동, 매입, 통행료 로그 출력

**Step 5. 승리 조건**

* 파산하지 않은 플레이어 1명만 남으면 종료
* 이후 Winner 출력 후 게임 종료

**예시 클래스 스켈레톤**

// Dice.java

public class Dice {

private int d1, d2;

public void roll() {

d1 = 1 + (int)(Math.random() \* 6);

d2 = 1 + (int)(Math.random() \* 6);

}

public int sum() { return d1 + d2; }

public boolean isDouble() { return d1 == d2; }

}

// Player.java

public class Player {

public final String name;

public int cash, pos = 0, jailTurns = 0;

public boolean bankrupt = false;

public Player(String name, int cash) { this.name = name; this.cash = cash; }

}

// City.java

public class City extends Tile {

public final int price, baseToll;

public int level = 0;

public Integer owner = null;

public City(int id, String name, String color, int price, int baseToll) {

super(id, name, Type.CITY, color);

this.price = price; this.baseToll = baseToll;

}

}

**입력 / 출력 설계**

* 사용자 입력: Scanner (Y/N, 1~n)
* 출력 로그:

[턴 3] PlayerA 주사위: 8 → 서울 도착

PlayerA가 서울을 300,000원에 매입했습니다.

PlayerB가 제주도에 도착, 통행료 100,000원 지불.

PlayerB 파산!

→ 승자: PlayerA 🎉

**테스트 체크리스트**

* 더블 3회 → 무인도 이동
* 무인도 3턴 후 자동 해제
* 컬러 독점 시 통행료 2배
* START 통과 시 월급 지급
* 파산 시 즉시 탈락
* 마지막 1명 생존 시 게임 종료

**추가 확장 (선택)**

* CHANCE 카드 3~5종 추가 (±현금, 이동 등)
* 업그레이드 비용/확률형 이벤트
* Swing 간단 UI (사각형 보드 + 말 표시)

**바이브코딩 가이드라인**

1. 한 단계씩 작성 후 **코드 바로 실행 & 콘솔 출력 확인**
2. 매 기능 추가 시 "#region" 주석으로 구역 분리
3. 테스트는 JUnit보다 직접 콘솔 테스트 우선
4. 게임 흐름이 정상 동작하면 rules.json으로 수치 조정

**요약**

본 프로젝트는 “모두의 마블”의 핵심 메커니즘(매입, 통행료, 무인도, 더블)을

**단일 스레드 콘솔 시뮬레이션**으로 구현하는 학습용 게임입니다.

초점은 \*\*“룰엔진 구현 + 게임 루프 설계 능력 향상”\*\*에 있습니다.