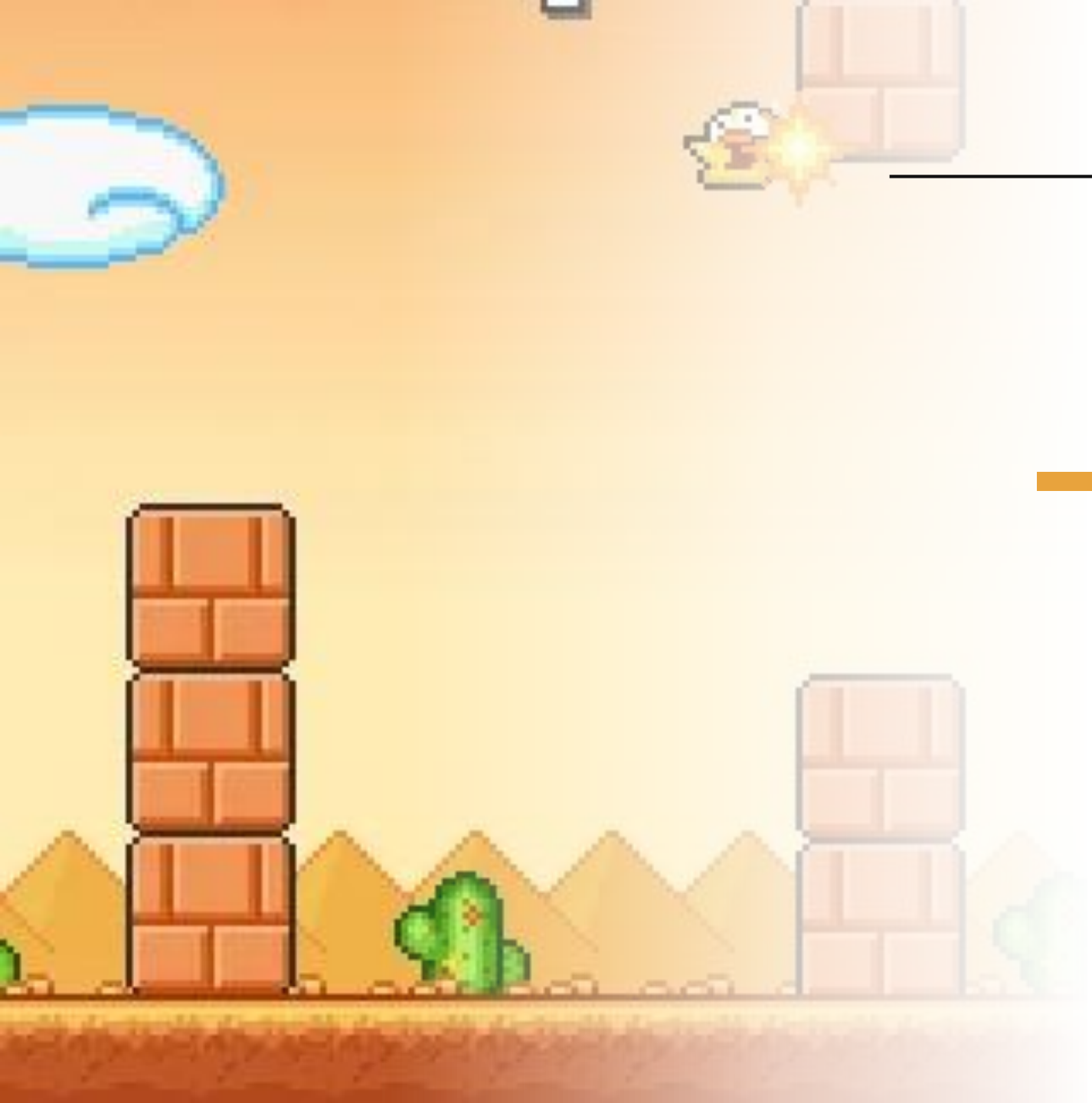


Unity/PlayMaker 학습 과제

FlappyBird (학습) 메인 기획서

핵심 메카닉스와 C#/FSM 연결

문서 최초 작성	문서 최종 수정
20210402	20210412
박도원	박도원

목차

개요 – 2P

1. 목차
2. 로그
3. 문서 계층

설계 목표 – 5P

1. 컨셉 인수
2. 완료 기준

핵심 메카닉스 – 7P

1. 행동과 보상
2. 새 물리 계약
3. 상태 계약
4. 파이프 생성 계약
5. 판정 우선순위

구현 구조 – 12P

1. 책임 분리
2. 이벤트 계약
3. 필요 요소

리소스 – 15P

1. 공통 팩 수량
2. 규격과 생산 경로

작업 계획 – 17P

1. 구현과 검수
2. 세부 이관 조건

개요 – 로그

순번	작성일	작성자	작성 내용	비고
1	2021-04-02	박도원	핵심 규칙 초안	컨셉 목표를 물리/판정 계약으로 전환함
2	2021-04-08	박도원	생성/이벤트 명세	중복 판정과 속도 변수 검수 조건을 보강함
3	2021-04-12	박도원	핵심 구현 완료 기준	한 판 연결 검수 후 세부 작업에 착수할 예정임
4				
5				
6				

개요 – 문서 계층



컨셉 → 메인 → 세부 순으로 상위 문서의 결정을 목표로 받아 구체화함.
강조 표시 = 이 문서

설계 목표 – 컨셉 인수

컨셉의 학습 목표를 구현 단위로 바꿈

확정 범위

단일 씬, 원터치, 기록 재도전을 인수함

구현 대상

물리, 생성, 충돌, 점수 계약을 만들 예정임

후속 범위

화면, 연출, 저장 검수는 세부로 이관함

설계 목표 – 완료 기준

4개 메카닉을 연결해 통과와 실패가 분명한 한 판을 만들 예정임

8	1회	1회
탭 직후 Y속도	파이프별 득점	판당 사망 이벤트

의도 상위 목표를 검수 가능한 조건으로 제한함

근거 컨셉 완결 루프; C 원본 씬과 코드 수치

핵심 메카닉스 – 행동과 보상

짧은 입력 판단에 보상을 연결함

1

탭

상승 속도를 덮어써 반응을 일정하게 함

2

틈 정렬

낙하와 재입력으로 높이를 조정함

3

통과

파이프별 +1로 성공을 알림

4

충돌

비행을 종료하고 재도전을 준비함

핵심 메카닉스 – 새 물리 계약

연타에 의한 속도 누적을 막음

구분	설계/검수 기준
점프	FixedUpdate에서 velocity = up × 8로 교체함
중력	새 Rigidbody2D gravityScale 3.5를 사용함
기울기	탭 시 30도, 진행 중 물리 틱마다 1.5도 감소함
주의	동일한 초기 속도이며 탭 간격별 최고 높이는 달라짐

핵심 메카닉스 – 상태 계약

FSM과 물리의 상태 불일치를 막음

구분	설계/검수 기준
대기	isRunning=false / isFinished=false; 중력 0
플레이	true / false; 원래 중력 복원 후 생성 타이머 진행함
사망	false / true; 이동, 입력, 추가 득점을 차단할 예정임
재로드	플래그, 타이머, 점프 예약을 초기화할 예정임

핵심 메카닉스 – 파이프 생성 계약

실제 이동과 생성 밀도를 분리해 조정함

구분	설계/검수 기준
주기	첫 간격 1초, 생성할 때마다 0.01초 감소함
높이	기준 위치에 -1, 0, 1, 2 중 정수 오프셋을 더함
이동	PipeController.moveSpeed 기본 5로 좌측 이동함
정리	Destroyer 진입 시 해당 파이프를 제거함
미연결 변수	speed 200과 speedPace 5는 이동 가속으로 간주하지 않음

핵심 메카닉스 – 판정 우선순위

동시 접촉과 중복 콜백의 보상 오류를 막음

1

접촉 확인

Obsticles 접촉이면 사망을 우선 처리함

2

사망 고정

종료 플래그로 중복 stopGame을 막음

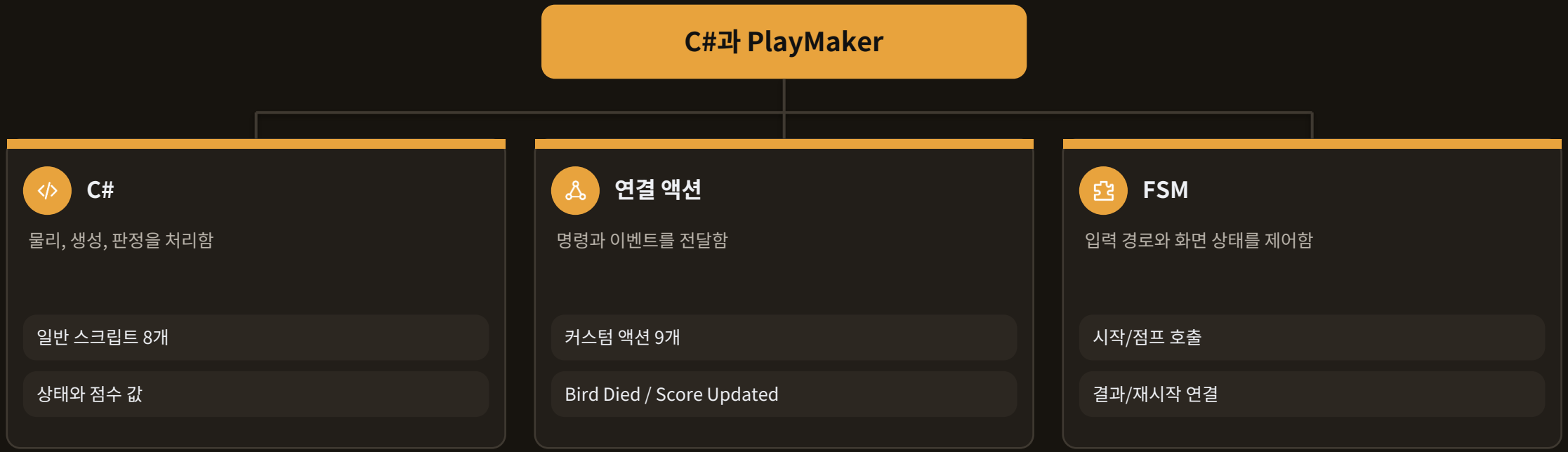
3

통과 승인

진행 중이며 미득점 파이프만 +1 처리함

구현 구조 – 책임 분리

도구 학습의 책임 경계를 명확히 함



구현 구조 – 이벤트 계약

재시작할 때 연결이 누적되지 않게 함

구분	설계/검수 기준
FSM → C#	FlappyBirdStartGame / Jump / RestartGame 액션을 호출함
C# → FSM	DiedBird → Bird Died; UpdatedScore → Score Updated
구독	연결 액션을 단일 수신자로 두고 상태 종료 시 정리함
전역 중력	대기 중 재로드에서도 기준 중력 0 저장을 방지할 예정임

구현 구조 – 필요 요소

참조 누락으로 인한 실행 중단을 줄임

핵심 구현 준비

씬

한 판 실행 오브젝트를 배치함

- 새 Rigidbody2D
- GameController
- Destroyer

프리팹

환경 참조를 연결함

- bg / floor
- ground / pipes

도구/인력

개인 제작 환경을 고정함

- Unity 2019.4.22f1
- PlayMaker
- 박도원 1인

리소스 – 공통 팩 수량

보유 수량과 실제 채택 범위를 분리함



배경	30장	3테마 × 10장; 한 테마부터 적용함
GUI	18장	필수 안내, 결과, 버튼을 선별함
새/효과	10장	새 9장 50×50; hitEffect 34×34
효과음	6개	WAV, mono, 44.1kHz, 32bit float

리소스 – 규격과 생산 경로

외부 리소스를 같은 기준으로 조립함

구분	설계/검수 기준
배경 PNG	sky 40×640, floor 40×190, blockTop/Down 50×48
UI PNG	Ready/Over 188×44, score창 252×126, play 116×64
명명	scene_01_*, player_fly_f01~03, Flap.wav 원명을 유지함
제작	팩 임포트 → Sprite 설정 → 프리팹 연결 → 경계 검수
사본 확장	D 사본의 색종이 84장, 폰트, 추가 음원은 세부 3에서 선별함

작업 계획 – 구현과 검수

단계마다 관찰 가능한 조건으로 다음 작업을 열음

1

새 물리

탭 직후 속도 8과 첫 입력 시작을 확인함

2

환경

높이 4종과 화면 밖 파이프 제거를 확인함

3

판정

통과 +1/충돌 1회/종료 후 0점을 확인함

4

FSM

10회 재시작의 중력과 구독 상태를 확인함

작업 계획 – 세부 이관 조건

핵심 구현 이후 누락 영역을 닫음

구분	설계/검수 기준
화면/입력	상태별 조작, 온보딩, 재도전 UI를 확정해야 함
맵/난이도	최소 생성 간격과 실제 통과 가능성을 검수해야 함
연출/리소스	아트, 사운드, 애니메이션 연결을 확정해야 함
점수/저장	최고점 비교와 재실행 보존을 검수해야 함