2022. 작품발표

God of JUMP

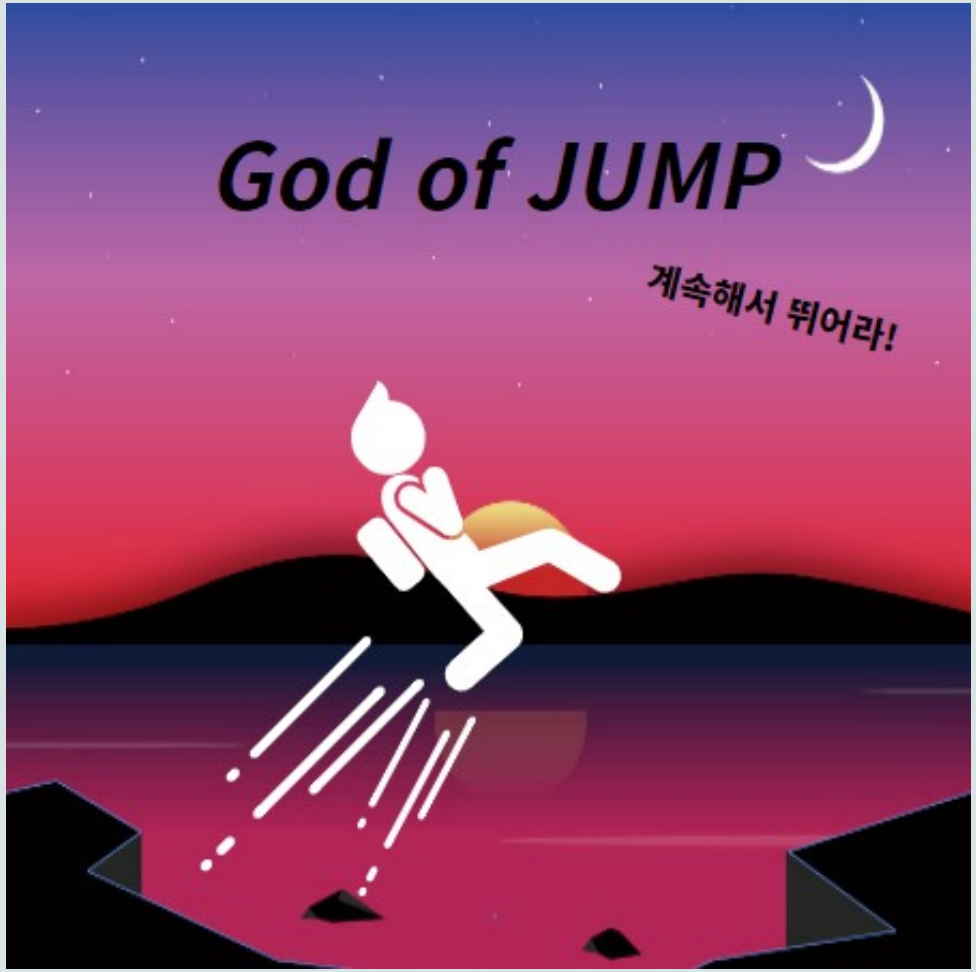
게임공학과

1888037

한 충 헌

목차

게임 소개



GOD of JUMP

- 장르 : 점프 액션 게임
- 플랫폼 : Window
- 개발언어 : Unity / C#

게임 컨셉 & 특징

- 메인컨셉 : 점프

  앞에 펼쳐진 플랫폼을 향해ㅅ

  계속해서 발판을 옮겨가며

  목적지를 향해 올라가게 됩니ㄷ .

게임 컨셉 & 특징

- 서브컨셉 : 발판

  공중에 놓여진 발판을

  이동하며 스테이지를 헤쳐나갇

  .

- 서브컨셉 : 기록

  게임클리어시간에 더히 ,

  점프횟수도 기록해 얼마나 정확힌

  점프로 게임을 클리어했는지 체크 .

- 서브컨셉 : 착지

  착지점을 불안하게 하여 ,

  긴장감을 주며 , 실수를 유발함 .

- 서브컨셉 : 쉬운 조작

  간편한 조작으로 쉽게 입문가능하여

  첫 진입이 굉장히 쉽다 .

게임 컨셉 & 특징

- 대표이미지

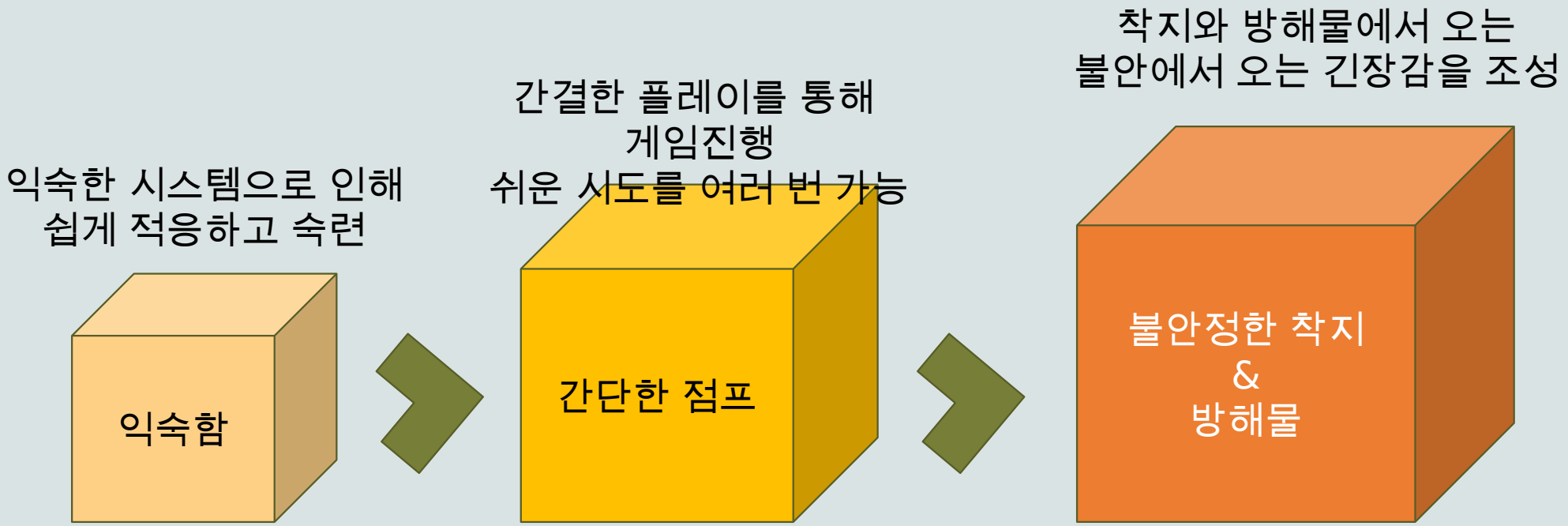
- 관련 이미지

게임 컨셉 & 특징

- 게임 특징

  > 눈앞에 놓여진 부조리한 장애물을 돌파하며 도전욕구를 자극한다.

  > 몇번의 실수로 재도전을 하면서 깨고자 하는 욕구를 자극한다.



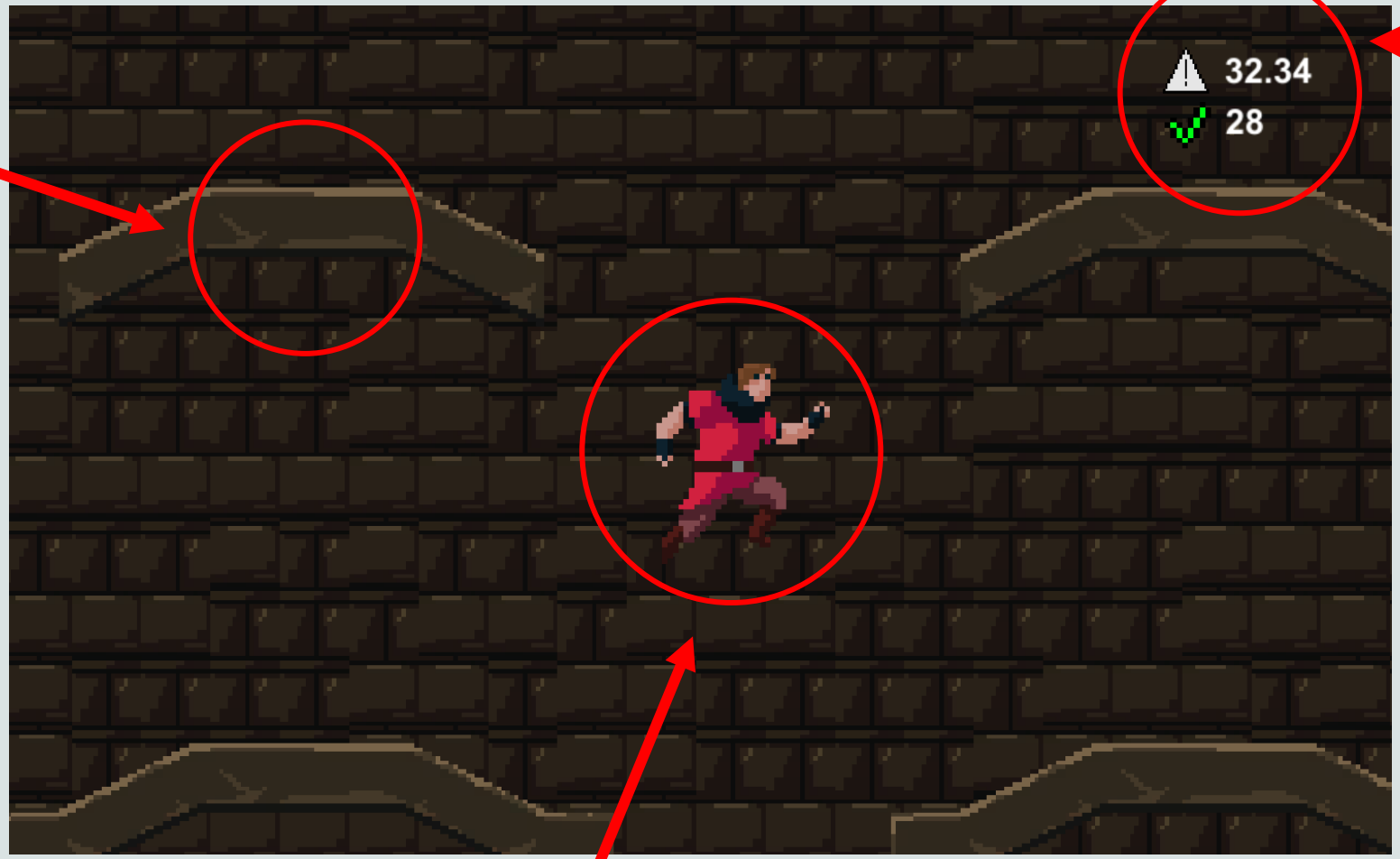
시놉시스

일단 달려라 ! 일단 뛰어넘어라 !

위험한 고성과 몬스터를 돌파하여

용사를 가두어 놓은 폐성에서 탈출하고 자유를 찾아라 !

화면구성 & 조작방법



발판

플레이타임
&
점프횟수

플레이어

스페이스바를
통해 점프

SPACE

게임 규칙

> 게임 규칙

- 방향키로 이동
- 스페이스 바를 통해 점프
- 게임시작과 동시에 게임시간이 흘러감
.
- 점프를 할때마다 점프카운트가 올라감
.
- 방해물에 닿으면 일정거리만큼
튕겨나감 .

> 엔딩

제일 최상층에 있는
골인지점에 도달하면 클리어 .

점프를 실수하거나 , 방해물에
닿아서 밀려나면 바닥으로
떨어져서 거의 이전단계부터
다시 시작 .

주요 코드

```
private void FixedUpdate()
{
    rigid.velocity = new Vector2(nextMove, rigid.velocity.y);

    Vector2 frontVec = new Vector2(rigid.position.x + nextMove * 0.2f, rigid.position.y);
    Debug.DrawRay(frontVec, Vector3.down, new Color(0, 1, 0));

    RaycastHit2D rayHit = Physics2D.Raycast(frontVec, Vector3.down, 1, LayerMask.GetMask("Platform"));

    if (rayHit.collider == null)
    {
        //Debug.Log("cliff");

        Turn();
    }
}
```

발판사이를 이동하는 게임이므로 , 방해물이 정해진 위치를 벗어나 발판밖으로 나가면 안되기 때문에 ,

오브젝트 아래로 Raycast 를 하도록하여 , 플랫폼의 유무를 판단 .

만약 Ray 가 무언가에 충돌하지 않을 경우 , 방향을 전환하는 함수를 호출 .

주요 코드

```
void Check()
{
    nextMove = Random.Range(-1, 2);

    anim.SetInteger("NowWalking", nextMove);

    if (nextMove != 0)
    {
        spriteRenderer.flipX = (nextMove == -1);
    }

    float nextThinkTime = Random.Range(2f, 5f);

    Invoke("Check", nextThinkTime);

}

void Turn()
{
    nextMove = nextMove * (-1);
    spriteRenderer.flipX = (nextMove == -1);

    CancelInvoke();
    Invoke("Check", 2);
}


void OnCollisionEnter2D(Collision2D collision)
{
    if (collision.gameObject.tag == "Player")
    {
        CancelInvoke();
        anim.SetTrigger("doAttack");
        Invoke("Check", 4);
    }
```

Check() 함수는 현재 움직이고 있는지를 체크하여 파라미터를 조절하는 역활을 함 .

Turn() 함수는 다음 이동방향을 반대로 전환함 .
그에 따라서 오브젝트 애니메이션 방향도 바꿔줌 .

플레이어와 충돌했을때 , 공격애니메이션을 출력하도록 함 .

후기

간단한 조작으로 힘들게 하는 게임을
만들고 싶었습니다 .

기간동안 외적으로 여러가지 사안이
발생해서

고된 시간이 있었습니다만 ,

그래도 무언가 만들고나니 기분은
좋습니다 .

Q & A

감사합니다

1888037 한충헌